

## Analyst Base

Comprehensive Project Report

Author: Abhishek Chauhan

## Project Files

### **.claude/commands/fix-flakey-test.md**

Fix flakey test \$ARGUMENTS

### **.claude/commands/fix-issue.md**

Fix issue \$ARGUMENTS

### **.claude/commands/fix-pr.md**

Fetch unresolved comments for this branch's PR, then fix them

### **.claude/commands/repro-issue.md**

Repro issue \$ARGUMENTS in a failing test

### **.clj-kondo/README.md**

Update `clj-kondo` configs for libraries using

```
```sh
clj-kondo --copy-configs --dependencies --lint "$(clojure -Spath -A:dev)"
```
```

### **.devcontainer/Dockerfile**

FROM mcr.microsoft.com/vscode/devcontainers/java:21

# Set up nodesource, install node, yarn, fontconfig for static viz, rlwrap for dev ergonomics

RUN ( curl -fsSL...

### **.dockerignore**

.babel\_cache  
docs/\*  
target/\*  
.circleci  
.cpcache  
.devcontainer  
.github  
.husky  
.lsp  
.shadow-cljs  
.github  
.vscode

## Analyst Base

hooks/\*  
test/\*  
test\_config/\*  
test\_modules/\*  
test\_resources/\*  
node\_modules  
\*\*metabase.jar  
...

### **.editorconfig**

Metabase EditorConfig settings  
<http://editorconfig.org/>  
use the top most editor config file

### **.eslintignore**

frontend/src/cljs  
frontend/src/cljs\_release  
e2e/support/cypress\_sample\_database.js  
e2e/support/cypress\_sample\_instance\_data.js

### **.eslintrc.js**

eslint-disable import/no-commonjs  
eslint-disable no-undef

### **.gitignore**

\*.class  
\*. \*.rej  
\*.iml  
\*.jar  
\*.log  
\*.po~  
\*.sqlite  
\*.trace.db  
\*.trace.db.old  
\*.xcworkspacedata  
.DS\_Store  
.\#\*  
.cpcache/  
.eastwood  
.idea/  
.nrepl-port  
.portal  
.vscode  
/\* .h2.db

## Analyst Base

/\*.lock.db

/\*.mv.db

/\*.trace...

### **.lycheeignore**

http://localhost\*

https://admin.google.com

### **.nvmrc**

v22.13.1

### **.prettierignore**

frontend/src/cljs

docs/configuring-metabase/environment-variables.md

docs/api/\*

docs/api-documentation.md

.storybook-sdk/msw-public/mockServiceWorker.js

### **.prettierrc**

```
{  
  "trailingComma": "all"  
}
```

### **.yarnrc**

network-timeout 600000

## **Dockerfile**

#####

STAGE 1: builder

#####

## **LICENSE-AGPL.txt**

GNU AFFERO GENERAL PUBLIC LICENSE

Version 3, 19 November 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim...

## **LICENSE-EMBEDDING.txt**

METABASE APP-EMBED.JS

SOFTWARE LICENSE AGREEMENT

PLEASE READ THE FOLLOWING TERMS AND CONDITIONS CAREFULLY BEFORE DOWNLOADING, INSTALLING OR USING THE APP-EMBED.JS SOFTWARE OR ANY ACCOMPANYING...

## **LICENSE-MCL.txt**

## Analyst Base

Usage of files in the top-level "/enterprise" directory and subdirectories thereof, and of Metabase Enterprise Edition features, is subject to the Metabase Commercial License...

### **LICENSE.txt**

Source code in this repository is variously licensed under the GNU Affero General Public License (AGPL), or the Metabase Commercial License (<https://www.metabase.com/license/commercial>).

\* Outside...

### **README.md**

analystBase\_qorix\_incubDeployment

AnalystBase: A Modern Business Intelligence Frontend

### **bad-blobs.txt**

10c975ab6e6f4de9d0f77cec9a3dc793267a8e16

05b89437a9a43be2d23c25a9ae006ec237240434

### **bin/README.md**

# Downloading Jars

### **bin/backward-compatibility-test.js**

!/usr/bin/env node

eslint-disable no-console

global process

This workflow is experimental POC, non-required, and is owned by team Embedding. For the time being, unless you think you did a breaking...

### **bin/build-drivers.md**

Build-drivers scripts

### **bin/build-for-test**

!/usr/bin/env bash

### **bin/build-mb.md**

# Build Metabase Tooling

### **bin/claude-print**

!/bin/bash

### **bin/debug-proxy**

!/usr/bin/env node

This proxy is useful for debugging frontend issues on deployed instances of Metabase. You can use the frontend of a local instance in conjunction with them backend of a deployed...

### **bin/docker/Dockerfile**

## Analyst Base

FROM eclipse-temurin:21-jre-alpine

ENV FC\_LANG=en-US LC\_CTYPE=en\_US.UTF-8

ARG GIT\_COMMIT\_SHA

ENV GIT\_COMMIT\_SHA=\${GIT\_COMMIT\_SHA}

# dependencies

RUN apk add -U bash fontconfig curl font-noto...

### **bin/docker/Dockerfile\_ubuntu**

FROM eclipse-temurin:21-jre-noble as runner

ENV FC\_LANG=en-US LC\_CTYPE=en\_US.UTF-8

ARG GIT\_COMMIT\_SHA

ENV GIT\_COMMIT\_SHA=\${GIT\_COMMIT\_SHA}

# Dependencies

RUN apt-get update && \  
apt-get upgrade...

### **bin/embed-sign**

!/usr/bin/env node

eslint-env node

eslint-disable import/no-commonjs

### **bin/embedding-sdk/fixup-types-after-compilation.js**

!/usr/bin/env node

eslint-env node

eslint-disable import/no-commonjs, import/order, no-console

### **bin/embedding-sdk/generate-sdk-package-files.js**

!/usr/bin/env node

eslint-env node

eslint-disable import/no-commonjs, import/order, no-console

### **bin/find-never-defined-css-variables/find-never-defined-css-variables.ts**

!/usr/bin/env tsx

eslint-disable no-console

### **bin/find-never-defined-css-variables/find-never-defined-css-variables.unit.spec.ts**

import {

## Analyst Base

```
extractVariableDefinitionsFromFileContent,  
extractVariableUsagesFromFileContent,  
} from "./find-never-defined-css-variables";
```

```
describe("extractVariableDefinitionsFromFileContent", () =>...
```

### **bin/i18n/README.md**

```
# i18n info
```

```
### Building the backend pot file
```

### **bin/i18n/build-translation-resources**

```
! /usr/bin/env bash
```

### **bin/i18n/merge-translations**

```
!/usr/bin/env bash
```

This script merges translations from the given branch into the current branch.

It's useful to run this after backporting a translation PR, since that PR may remove translations...

### **bin/i18n/update-translation-template**

```
! /usr/bin/env bash
```

### **bin/lint-migrations-file/README.md**

Linters that validate the Liquibase migrations file against a spec. This lets us check and enforce additional constraints for the `000\_migrations.yaml` file (e.g. make sure you're not using duplicate...

### **bin/mage**

```
!/usr/bin/env bash
```

### **bin/mb-download**

```
!/usr/bin/env bb
```

### **bin/release-list/README.md**

Release list

### **bin/release-list/resources/releases-template.md**

```
---
```

```
title: Metabase versions
```

```
---
```

```
# Metabase versions
```

Below are links to releases for:

- [Metabase Enterprise Edition](#metabase-enterprise-edition-releases). This edition is used in the...

### bin/release.md

# Metabase Release Script 3.0

### Prereqs

### bin/templates/country-codes-template.md

---

title: {{title}}

---

# {{title}}

This reference lists all country codes and their corresponding country names used in Metabase's default world map visualizations. The data comes from the...

### bin/test-cljs.js

!node

### bin/verify-doc-links

!/usr/bin/env node

### blob\_ids.txt

10c975ab6e6f4de9d0f77cec9a3dc793267a8e16

05b89437a9a43be2d23c25a9ae006ec237240434

### dev/src/dev/readme.md

# Render png

### docs/CONTRIBUTING.md

---

title: Contributing to Metabase

redirect\_from:

- /docs/latest/developers-guide/contributing

---

# Contributing to Metabase

## Thank you

First off, thanks for your interest in Metabase and...

### docs/README.md

---

title: Metabase documentation

redirect\_from:

## Analyst Base

- /docs/latest/enterprise-guide
- /docs/latest/users-guide
- /docs/latest/administration-guide
- /docs/latest/operations-guide
- ...

### **docs/actions/basic.md**

---

title: Basic actions

---

#### # Basic actions

Basic actions are "implicit" [actions](./introduction.md) that do things that people typically want to do when interacting with a database: Create,...

### **docs/actions/custom.md**

---

title: Custom actions

---

#### # Custom actions

Write SQL to update records in your databases.

![Custom action](./images/custom-action.png)

#### ## Creating a custom action

> You must be in a group...

### **docs/actions/introduction.md**

---

title: Introduction to actions

---

#### # Introduction to actions

> For now, actions are only available for PostgreSQL and MySQL.

![Example action](./images/example-action.png)

#### ## What are...



### **docs/actions/start.md**

---

title: Actions overview

redirect\_from:

- /docs/latest/actions

---

# Actions overview

![An action updating a plan on a dashboard](./images/dashboard-action.gif)

Actions let you write...

### **docs/cloud/accounts-and-billing.md**

---

title: "Accounts and billing"

version: latest

has\_magic\_breadcrumbs: true

show\_category\_breadcrumb: true

show\_title\_breadcrumb: true

category: "Cloud"

layout: new-docs

---

# Accounts and...

### **docs/cloud/change-region.md**

---

title: "Changing which region your Metabase is hosted in"

version: latest

has\_magic\_breadcrumbs: true

show\_category\_breadcrumb: true

show\_title\_breadcrumb: true

category: "Cloud"

layout:...

### **docs/cloud/cloud-vs-self-hosting.md**

---

title: "Metabase Cloud versus self-hosting"

version: latest

has\_magic\_breadcrumbs: true

show\_category\_breadcrumb: true

show\_title\_breadcrumb: true

category: "Cloud"  
layout: new-docs  
---

#...

## **docs/cloud/custom-domain.md**

---  
title: "Changing your domain name"  
version: latest  
has\_magic\_breadcrumbs: true  
show\_category\_breadcrumb: true  
show\_title\_breadcrumb: true  
category: "Cloud"  
layout: new-docs  
redirect\_from:  
-...

## **docs/cloud/google-sheets.md**

---  
title: Sync Google Sheets with Metabase  
version: latest  
has\_magic\_breadcrumbs: true  
show\_category\_breadcrumb: true  
show\_title\_breadcrumb: true  
category: "Cloud"  
layout: new-docs  
---

# Sync...

## **docs/cloud/how-billing-works.md**

---  
title: "How Metabase billing works"  
version: latest  
has\_magic\_breadcrumbs: true  
show\_category\_breadcrumb: true  
show\_title\_breadcrumb: true  
category: "Cloud"  
layout: new-docs  
redirect\_from:  
-...

## **docs/cloud/ip-addresses-to-whitelist.md**

---

title: "IP addresses to whitelist"

version: latest

has\_magic\_breadcrumbs: true

show\_category\_breadcrumb: true

show\_title\_breadcrumb: true

category: "Cloud"

layout: new-docs

redirect\_from:

~...

### **docs/cloud/limitations.md**

---

title: "Limitations of Metabase Cloud"

version: latest

has\_magic\_breadcrumbs: true

show\_category\_breadcrumb: true

show\_title\_breadcrumb: true

category: "Cloud"

layout: new-docs

redirect\_from:

~...

### **docs/cloud/migrate/cloud-to-self-hosted.md**

---

title: "Migrate from Metabase Cloud to a self-hosted Metabase"

version: latest

has\_magic\_breadcrumbs: true

show\_category\_breadcrumb: true

show\_title\_breadcrumb: true

category: "Cloud"

layout:...

### **docs/cloud/migrate/guide-pre-50.md**

---

title: "Migrate to Metabase Cloud - Metabase 49 or lower"

version: latest

has\_magic\_breadcrumbs: true

show\_category\_breadcrumb: true

show\_title\_breadcrumb: true

category: "Cloud"

layout:...

### **docs/cloud/migrate/guide.md**

---

title: "Migrate to Metabase Cloud"  
version: latest  
has\_magic\_breadcrumbs: true  
show\_category\_breadcrumb: true  
show\_title\_breadcrumb: true  
category: "Cloud"  
layout: new-docs  
section:...

### **docs/cloud/migrate/heroku.md**

---

title: "Migrating from Heroku to Metabase Cloud"  
version: latest  
has\_magic\_breadcrumbs: true  
show\_category\_breadcrumb: true  
show\_title\_breadcrumb: true  
category: "Cloud"  
layout:...

### **docs/cloud/start.md**

---

title: Metabase Cloud  
version: latest  
has\_magic\_breadcrumbs: true  
show\_category\_breadcrumb: true  
show\_title\_breadcrumb: false  
category: "Cloud"  
redirect\_from:  
- /cloud/docs

---

# Metabase...

### **docs/cloud/storage.md**

---

title: Metabase Cloud Storage  
version: latest  
has\_magic\_breadcrumbs: true  
show\_category\_breadcrumb: true  
show\_title\_breadcrumb: true  
category: "Cloud"  
layout: new-docs

---

# Metabase Cloud...

### **docs/configuring-metabase/appearance.md**

---

title: Appearance

redirect\_from:

- /docs/latest/administration-guide/whitelabeling
- /docs/latest/enterprise-guide/whitelabeling
- /docs/latest/enterprise-guide/customize-embeds

^...

### **docs/configuring-metabase/caching.md**

---

title: Caching query results

redirect\_from:

- /docs/latest/administration-guide/14-caching
- /docs/latest/enterprise-guide/cache

---

# Caching query results

If your question results don't...

### **docs/configuring-metabase/config-file.md**

---

title: "Configuration file"

---

# Configuration file

{% include plans-blockquote.html feature="Loading from a configuration file" self-hosted-only="true" %}

On self-hosted...

### **docs/configuring-metabase/config-template.md**

---

title: "Metabase config file template"

---

# Metabase config file template

You can generate the following docs by changing into the top-level Metabase directory and running:

...

clojure...

### **docs/configuring-metabase/custom-maps.md**

---

title: Custom maps

redirect\_from:

- /docs/latest/administration-guide/20-custom-maps

---

#### # Custom maps

By default, Metabase uses OpenStreetMaps for map visualizations, but there are a few...

### **docs/configuring-metabase/customizing-jetty-webserver.md**

---

title: Customizing the Metabase Jetty webserver

redirect\_from:

- /docs/latest/operations-guide/customizing-jetty-webserver

---

#### # Customizing the Metabase Jetty webserver

In most cases there...

### **docs/configuring-metabase/email.md**

---

title: Set up email

summary: Learn how to set up email in Metabase to enable dashboard subscriptions and alerts.

redirect\_from:

- /docs/latest/administration-guide/02-setting-up-email

---

#...

### **docs/configuring-metabase/environment-variables.md**

---

title: Environment variables

redirect\_from:

- /docs/latest/operations-guide/environment-variables

---

#### # Environment variables

\_This documentation was generated from source by...

### **docs/configuring-metabase/fonts.md**

---

title: Fonts

redirect\_from:

- /docs/latest/enterprise-guide/fonts
- /docs/latest/embedding/fonts

---

#### # Fonts

{% include plans-blockquote.html feature="Customizable font" %}

On Pro and...

### **docs/configuring-metabase/localization.md**

---

title: Languages and localization

redirect\_from:

- /docs/latest/administration-guide/localization

---

#### # Languages and localization

Admins can update the localization settings for the...

### **docs/configuring-metabase/log-configuration.md**

---

title: Metabase logs

redirect\_from:

- /docs/latest/operations-guide/log-configuration

summary: Configure how much information Metabase displays in its logs.

---

#### # Metabase logs

Metabase logs...

### **docs/configuring-metabase/setting-up-metabase.md**

---

title: Setting up Metabase

redirect\_from:

- /docs/latest/setting-up-metabase

## Analyst Base

---

### # Setting up Metabase

This guide will help you set up Metabase once you've gotten it installed. If you haven't...

#### **docs/configuring-metabase/settings.md**

---

title: General settings

redirect\_from:

- /docs/latest/administration-guide/08-configuration-settings

---

### # General settings

This section contains settings for your whole instance, like its...

#### **docs/configuring-metabase/slack.md**

---

title: Set up Slack

redirect\_from:

- /docs/latest/administration-guide/09-setting-up-slack

---

### # Set up Slack

If you want to have your [dashboard...

#### **docs/configuring-metabase/start.md**

---

title: "Configuration overview"

redirect\_from:

- /docs/latest/configuring-metabase

---

### # Configuration overview

#### **## [Setting up Metabase](../setting-up-metabase.md)**

A walkthrough of when you...

#### **docs/configuring-metabase/timezones.md**

---

title: Timezones



## Analyst Base

redirect\_from:

- /docs/latest/operations-guide/handling-timezones

---

# Timezones

Metabase does its best to ensure proper and accurate reporting in whatever timezone you...

### **docs/configuring-metabase/webhooks.md**

---

title: Webhooks

---

# Webhooks

Admins can set up webhooks so that people can send [alerts](../questions/alerts.md) to a particular URL. Which means you can set up an alert to send the results...

### **docs/dashboards/actions.md**

---

title: Actions on dashboards

---

# Actions on dashboards

![Dashboard with filter, action button, and detail card view](../images/dashboard-filter-action.png)

To put...

### **docs/dashboards/filters.md**

---

title: Dashboard filters

summary: Make your dashboards do more with filters and parameters. Instead of creating a bunch of similar dashboards, just add widgets to change...

### **docs/dashboards/interactive.md**

---

title: Dashboard interactivity

redirect\_from:

- /docs/latest/users-guide/interactive-dashboards

---

# Dashboard interactivity

## Analyst Base

You can customize what happens when people click on questions in...

### **docs/dashboards/introduction.md**

---

title: Introduction to dashboards

redirect\_from:

- /docs/latest/users-guide/07-dashboards

---

# Introduction to dashboards

![Interactive dashboard](./images/interactive-dashboard.png)

##...

### **docs/dashboards/linked-filters.md**

---

title: Linked filters

---

# Linked filters

You can **link filters** on a dashboard so that a child filter limits its values based on the value(s) applied by a parent filter.

For example, let's...

### **docs/dashboards/multiple-series.md**

---

title: Charts with multiple series

redirect\_from:

- /docs/latest/users-guide/09-multi-series-charting

---

# Charts with multiple series

One of the best ways to add context and clarity when...

### **docs/dashboards/start.md**

---

title: Dashboards overview

redirect\_from:

- /docs/latest/dashboards

---

## Analyst Base

# Dashboards overview

![Example dashboard](./images/dashboard.png)

## [Introduction to...

### **docs/dashboards/subscriptions.md**

---

title: Dashboard subscriptions

redirect\_from:

- /docs/latest/users-guide/dashboard-subscriptions
- /docs/latest/enterprise-guide/dashboards-subscriptions

---

# Dashboard...

### **docs/data-modeling/formatting.md**

---

title: Formatting defaults

summary: Configure how dates, numbers, currencies, and text display in Metabase at global, field, and question levels.

redirect\_from:

-...

### **docs/data-modeling/json-unfolding.md**

---

title: Working with JSON

summary: Learn how to unfold JSON columns into separate fields that you can filter on in the query builder.

---

# Working with JSON

## Filtering JSON

In the [query...

### **docs/data-modeling/legacy-metrics.md**

---

title: Legacy metrics

---

# Legacy metrics

## Analyst Base

Metabase upgraded [metrics](./metrics.md) starting in version 51.

If you're upgrading from Metabase 50 or earlier, all of your existing metrics will...

### **docs/data-modeling/metadata-editing.md**

---

title: "Table metadata admin settings"

redirect\_from:

- /docs/latest/administration-guide/03-metadata-editing

---

# Table metadata admin settings

\_Admin settings > Table metadata\_

!Table...

### **docs/data-modeling/metrics.md**

---

title: Metrics

redirect\_from:

- /docs/latest/administration-guide/07-segments-and-metrics
- /docs/latest/data-modeling/segments-and-metrics

---

# Metrics

Create metrics to define the...

### **docs/data-modeling/model-persistence.md**

---

title: Model persistence

---

# Model persistence

> Currently available for PostgreSQL, MySQL, and Redshift.

Metabase can persist the results of your models so that your models (and the...

### **docs/data-modeling/models.md**

---

title: "Models"

redirect\_from:

## Analyst Base

- /docs/latest/users-guide/models

---

### # Models

Models are a fundamental building block in Metabase. Models curate data from another table or tables from the...

## docs/data-modeling/segments.md

---

title: "Segments"

---

### # Segments

Metabase allows admins to create segments so people can quickly and easily reference them in the query builder.

To manage segments:

1. Click the **gear** icon...

## docs/data-modeling/semantic-types.md

---

title: "Data types and semantic types"

redirect\_from:

- /docs/latest/users-guide/field-types
- /docs/latest/data-modeling/field-types

summary: "Metabase uses both data and semantic types to..."

## docs/data-modeling/start.md

---

title: "Data modeling overview"

redirect\_from:

- /docs/latest/data-modeling

---

### # Data modeling overview

Metabase provides tools for organizing your data and making it easier for people to...

## docs/databases/connecting.md

---

title: Adding and managing databases

## Analyst Base

redirect\_from:

- /docs/latest/administration-guide/01-managing-databases
- /docs/latest/databases/connections/sql-server
- ...

### **docs/databases/connections/athena.md**

---

title: Amazon Athena

---

# Amazon Athena

To add a database connection, click on the **gear** icon in the top right, and navigate to **Admin settings** > **Databases** > **Add a database**.

##...

### **docs/databases/connections/aws-rds.md**

---

title: "Connecting to AWS's Relational Database Service (RDS)"

redirect\_from:

- /docs/latest/administration-guide/databases/aws-rds

---

# Connecting to AWS's Relational Database Service...

### **docs/databases/connections/bigquery.md**

---

title: Google BigQuery

redirect\_from:

- /docs/latest/administration-guide/databases/bigquery

---

# Google BigQuery

To add a database connection, click on the **gear** icon in the top right,...

### **docs/databases/connections/clickhouse.md**

---

title: ClickHouse

description: Learn how to connect Metabase to your ClickHouse database, including connection settings, database selection, and SSL configuration.

---

## Analyst Base

# ClickHouse

To add a...

### **docs/databases/connections/databricks.md**

---

title: Databricks

---

# Databricks

To add a database connection, click on the **gear** icon in the top right, and navigate to **Admin settings** > **Databases** > **Add a database**. Then...

### **docs/databases/connections/druid.md**

---

title: Druid

---

# Druid

> Prefer the Druid JDBC connection.

To add a database connection, click on the **gear** icon in the top right, and navigate to **Admin settings** > **Databases** >...

### **docs/databases/connections/mariadb.md**

---

title: MariaDB

---

# MariaDB

> This page covers connecting to MariaDB as a `_data warehouse_`. For using MariaDB as Metabase's `_application database_`, see [Configuring the Metabase application...]

### **docs/databases/connections/mongodb.md**

---

title: MongoDB

redirect\_from:

- /docs/latest/administration-guide/databases/mongodb

---

# MongoDB

## Analyst Base

To add a database connection, click on the **gear** icon in the top right, and navigate to...

### **docs/databases/connections/mysql.md**

---

title: MySQL

redirect\_from:

- /docs/latest/administration-guide/databases/mysql

---

# MySQL

> This page covers connecting to MySQL as a **\_data warehouse\_**. For using MySQL as Metabase's...

### **docs/databases/connections/oracle.md**

---

title: Oracle

redirect\_from:

- /docs/latest/administration-guide/databases/oracle

---

# Oracle

To add a database connection, click on the **gear** icon in the top right, and navigate to...

### **docs/databases/connections/postgresql.md**

---

title: PostgreSQL

redirect\_from:

- /docs/latest/administration-guide/databases/postgresql

---

# PostgreSQL

> This page covers connecting to PostgreSQL as a **\_data warehouse\_**. For using...

### **docs/databases/connections/presto.md**

---

title: Presto

---

# Presto

To add a database connection, click on the **gear** icon in the top right, and navigate to **Admin settings** >



## Analyst Base

**\*\*Databases\*\*** > **\*\*Add a database\*\***.

Fill out the...

### **docs/databases/connections/redshift.md**

---

title: Amazon Redshift

description: Connect Metabase to Amazon Redshift data warehouses to run queries and create dashboards.

redirect\_from:

-...

### **docs/databases/connections/snowflake.md**

---

title: Snowflake

redirect\_from:

- /docs/latest/administration-guide/databases/snowflake

---

# Snowflake

To add a database connection, click on the **\*\*gear\*\*** icon in the top right, and navigate...

### **docs/databases/connections/sparksql.md**

---

title: SparkSQL

---

# SparkSQL

To add a database connection, click on the **\*\*gear\*\*** icon in the top right, and navigate to **\*\*Admin settings\*\*** > **\*\*Databases\*\*** > **\*\*Add a database\*\***.

##...

### **docs/databases/connections/sql-server.md**

---

title: SQL Server

---

# SQL Server

To add a database connection, click on the **\*\*gear\*\*** icon in the top right, and navigate to **\*\*Admin settings\*\*** > **\*\*Databases\*\*** > **\*\*Add a database\*\***.

## Analyst Base

##...

### **docs/databases/connections/sqlite.md**

---

title: SQLite

---

# SQLite

> SQLite isn't available for [Metabase Cloud](https://www.metabase.com/cloud/).

To add a database connection, click on the **gear** icon in the top right, and...

### **docs/databases/connections/starburst.md**

---

title: Starburst

description: Learn how to connect Metabase to your Starburst or Trino database, including connection settings, SSL configuration, and database sync options.

---

# Starburst

>...

### **docs/databases/connections/vertica.md**

---

title: Working with Vertica in Metabase

redirect\_from:

- /docs/latest/administration-guide/databases/vertica

---

# Working with Vertica in Metabase

Starting in v0.20.0, Metabase provides a...

### **docs/databases/danger-zone.md**

---

title: Danger zone

description: The Danger Zone section of database connections is where you can discard field values or remove database connections and all of their related content.

---

#...

### **docs/databases/encrypting-details-at-rest.md**

## Analyst Base

---

title: Encrypting your database connection  
redirect\_from:  
- /docs/latest/operations-guide/encrypting-database-details-at-rest

---

# Encrypting your database connection

Metabase stores...

**docs/databases/ssh-tunnel.md**

---

title: SSH tunneling  
redirect\_from:  
- /docs/latest/administration-guide/ssh-tunnel-for-database-connections

---

# SSH tunneling

Metabase can connect to some databases by first establishing a...

**docs/databases/ssl-certificates.md**

---

title: SSL certificate  
redirect\_from:  
- /docs/latest/administration-guide/secure-database-connections-with-ssl-certificates

---

# SSL certificate

If you'd like to connect your Metabase Cloud...

**docs/databases/start.md**

---

title: Databases overview  
redirect\_from:  
- /docs/latest/databases

---

# Databases overview

## [Adding and managing databases](./connecting.md)

Connect to and manage your databases.

##...

### **docs/databases/sync-scan.md**

---

title: Syncing and scanning databases

summary: Learn how Metabase stays in sync with your database by running periodic queries to update metadata, sample field values, and compute stats.

---

#...

### **docs/databases/uploads.md**

---

title: Setting up data uploads

---

# Setting up data uploads

This page covers how admins can set up data uploads so people can upload CSVs to your Metabase. For `_how_` to upload data once this...

### **docs/databases/users-roles-privileges.md**

---

title: Database users, roles, and privileges

---

# Database users, roles, and privileges

We recommend creating a ``metabase`` database user with the following database roles:

- `[`analytics` for...`

### **docs/developers-guide/api-changelog.md**

---

title: API changelog

---

# Breaking changes to the API interface

## Metabase 0.55.0

- ``POST /api/card/from-csv`` has been renamed to ``POST /api/upload/csv``.

## Analyst Base

- `GET /api/util/stats` has been...

### **docs/developers-guide/build.md**

---

title: Building Metabase

---

#### # Building Metabase

This doc will show you how you can build and run Metabase on your own computer so you can play around with it or test features in development....

### **docs/developers-guide/clojure.md**

---

title: Working with Clojure

---

#### # Working with Clojure

Check out [Clojure for the Brave and True](https://www.braveclojure.com/clojure-for-the-brave-and-true/). It's free online.

If you don't...

### **docs/developers-guide/code-reviews.md**

---

title: "Code reviews"

redirect\_from:

- /docs/latest/code-reviews

---

#### # Code reviews

The overall goal of a code review is to serve as a safety net for other people on our team and help them...

### **docs/developers-guide/community-drivers.md**

---

title: Community drivers

redirect\_from:

- /docs/latest/developers-guide-drivers
- /docs/latest/developers-guide/partner-and-community-drivers

---

#### # Community drivers

> Community drivers are...

### **docs/developers-guide/dev-branch-docker.md**

---

title: How to run a development branch of Metabase using Docker

---

# How to run a development branch of Metabase using Docker

If you want to run a branch of Metabase that's currently in...

### **docs/developers-guide/devenv.md**

---

title: Development environment

---

# Development environment

The Metabase application has two basic components:

1. A backend written in Clojure which contains a REST API as well as all the...

### **docs/developers-guide/docs.md**

---

title: Developing Metabase documentation

---

# Developing Metabase documentation

Notes on writing docs for Metabase.

## Linting markdown links

You can check for broken links in the...

### **docs/developers-guide/driver-changelog.md**

---

title: Driver interface changelog

---

# Driver Interface Changelog

## Metabase 0.56.0

## Analyst Base

- Add the testing multi-method ``tx/track-dataset`` for shared cloud dbs to track loaded datasets for more...

### **docs/developers-guide/drivers/basics.md**

---

title: Database driver basics

---

# Database driver basics

A Metabase driver:

- **\*\*Provides Metabase with basic information for the database\*\*** such as the database's capabilities, connection...

### **docs/developers-guide/drivers/driver-tests.md**

---

title: Submitting a PR for a new driver

---

# Submitting a PR for a new driver

If you want to submit a PR to add a driver plugin to the [Metabase repo](https://github.com/metabase/metabase) (as...

### **docs/developers-guide/drivers/multimethods.md**

---

title: Implementing multimethods for your driver

---

# Implementing multimethods for your driver

Implementing multimethods lets you take advantage of Metabase's existing driver code by...

### **docs/developers-guide/drivers/plugins.md**

---

title: Plugin manifests

---

# Plugin manifests

Metabase plugin JARs contain a `_plugin manifest_` -- a top-level file named ``metabase-plugin.yaml``. When Metabase launches, it iterates over every...

### **docs/developers-guide/drivers/start.md**

## Analyst Base

---

title: Guide to writing a Metabase driver

---

# Guide to writing a Metabase driver

So here's the scenario: you love Metabase. It's changed your life. But you have some data in a Visual Fox Pro...

### **docs/developers-guide/e2e-tests.md**

---

title: End-to-end tests with Cypress

---

# End-to-end tests with Cypress

Metabase uses Cypress for "end-to-end testing", that is, tests that are executed against the application as a whole,...

### **docs/developers-guide/emacs.md**

---

title: Developing Metabase with Emacs

---

# Developing Metabase with Emacs

`.dir-locals.el` contains some Emacs Lisp that tells `clojure-mode` how to indent Metabase macros and which arguments...

### **docs/developers-guide/frontend.md**

---

title: Frontend

---

# Frontend

## Entity Loaders

If you're developing a new feature or just generally need to get at some of the application data on the frontend, Entity Loaders are going to...

### **docs/developers-guide/internationalization.md**

---

title: Internationalization



## Analyst Base

---

### # Internationalization

We are an application with lots of users all over the world. To help them use Metabase in their own language, we mark all of our strings...

### **docs/developers-guide/mage.md**

MAGE - Development Automation

### **docs/developers-guide/mbql-library-changelog.md**

---

title: MBQL Library changelog

---

### # MBQL Library Changelog

Changes made to the library API for manipulating MBQL queries, found in `metabase.lib.js`. The latest API documentation can be found...

### **docs/developers-guide/security-token-scanner.md**

Security Token Scanner

### **docs/developers-guide/start.md**

---

title: "Developer Guide"

redirect\_from:

- /docs/latest/developers-guide

---

### # Developer Guide

This guide contains detailed information on how to work on Metabase codebase.

### ## Contributing

^-...

### **docs/developers-guide/versioning.md**

---

title: Metabase release versioning

---

## Analyst Base

# Metabase release versioning

We follow our own flavor of the [semantic versioning guidelines](https://semver.org/) in order to distinguish the...

### **docs/developers-guide/visual-studio-code.md**

---

title: Developing with Visual Studio Code

---

# Developing with Visual Studio Code

## Debugging

First, install the following extension:

- [Debugger for...

### **docs/developers-guide/visual-tests.md**

---

title: Visual Tests

---

# Visual Tests

We use [Loki](https://loki.js.org/) with Storybook for visual tests. Loki captures snapshots from selected stories and creates PNG references in...

### **docs/embedding/embedded-analytics-js.md**

---

title: Embedded Analytics JS

summary: Getting started with Embedded Analytics JS for embedding Metabase entities into external applications

---

# Embedded Analytics JS

Embedded analytics JS...

### **docs/embedding/interactive-embedding-quick-start-guide.md**

---

title: "Interactive embedding quickstart"

---

## Analyst Base

# Interactive embedding quickstart

You'll embed the full Metabase application in your app. Once logged in, people can view a Metabase dashboard in...

### **docs/embedding/interactive-embedding.md**

---

title: Interactive embedding

redirect\_from:

- /docs/latest/enterprise-guide/full-app-embedding
- /docs/latest/embedding/full-app-embedding

---

# Interactive embedding

{% include...

### **docs/embedding/interactive-ui-components.md**

---

title: Interactive embedding UI components

description: Customize the UI components in your interactive Metabase embed by adding parameters to the embedding URL.

---

# Interactive embedding UI...

### **docs/embedding/introduction.md**

---

title: Embedding introduction

redirect\_from:

- /docs/latest/administration-guide/13-embedding

---

# Embedding introduction

You can embed Metabase tables, charts, and dashboards-even...

### **docs/embedding/public-links.md**

---

title: Public sharing

redirect\_from:

- /docs/latest/administration-guide/12-public-links
- /docs/latest/embedding/12-public-links

---

## Analyst Base

# Public sharing

> Only admins can create public links...

### **docs/embedding/sdk/appearance.md**

---

title: "Embedded analytics SDK - appearance"

---

# Embedded analytics SDK - appearance

{% include plans-blockquote.html feature="Embedded analytics SDK" sdk=true %}

You can style your embedded...

### **docs/embedding/sdk/authentication.md**

---

title: Embedded analytics SDK - authentication

---

# Embedded analytics SDK - authentication

{% include plans-blockquote.html feature="Embedded analytics SDK" sdk=true %}

For using the SDK in...

### **docs/embedding/sdk/collections.md**

---

title: Embedded analytics SDK - collections

---

# Embedded analytics SDK - collections

{% include plans-blockquote.html feature="Embedded analytics SDK" sdk=true %}

## Embedding a collection...

### **docs/embedding/sdk/config.md**

---

title: Embedded analytics SDK - config

---

# Embedded analytics SDK - config

## Analyst Base

```
{% include plans-blockquote.html feature="Embedded analytics SDK" sdk=true %}
```

## Passing a configuration object to...

### **docs/embedding/sdk/dashboards.md**

---

title: "Embedded analytics SDK - dashboards"

---

# Embedded analytics SDK - dashboards

```
{% include plans-blockquote.html feature="Embedded analytics SDK" sdk=true %}
```

You can embed an...

### **docs/embedding/sdk/introduction.md**

---

title: Embedded analytics SDK

redirect\_from:

- /docs/latest/embedding/sdk

---

# Embedded analytics SDK

```
{% include plans-blockquote.html feature="Embedded analytics SDK" sdk=true %}
```

With the...

### **docs/embedding/sdk/next-js.md**

---

title: Embedded analytics SDK - Using the SDK with Next.js

---

# Embedded analytics SDK - Using the SDK with Next.js

```
{% include plans-blockquote.html feature="Embedded analytics SDK" sdk=true...
```

### **docs/embedding/sdk/plugins.md**

---

title: Embedded analytics SDK - plugins

---

# Embedded analytics SDK - plugins

## Analyst Base

```
{% include plans-blockquote.html feature="Embedded analytics SDK" sdk=true %}
```

The Metabase Embedded analytics SDK...

### **docs/embedding/sdk/questions.md**

---

title: "Embedded analytics SDK - questions"

description: How to embed charts in your app with the Embedded analytics SDK.

---

# Embedded analytics SDK - questions

```
{% include...
```

### **docs/embedding/sdk/quickstart-cli.md**

---

title: Embedded analytics SDK - CLI quickstart

---

# Embedded analytics SDK - CLI quickstart

```
{% include plans-blockquote.html feature="Embedded analytics SDK" sdk=true...
```

### **docs/embedding/sdk/quickstart-with-sample-app.md**

---

title: Embedded analytics SDK - quickstart with sample app

---

# Embedded analytics SDK - quickstart with sample app

```
{% include plans-blockquote.html feature="Embedded analytics SDK" sdk=true...
```

### **docs/embedding/sdk/quickstart.md**

---

title: Embedded analytics SDK - quickstart

description: "This guide walks you through how to set up the Embedded analytics SDK in your application with your Metabase."

---

# Embedded analytics...

### **docs/embedding/sdk/snippets/appearance/theme.ts**

```
import { defineMetabaseTheme } from "@metabase/embedding-sdk-react";
```

## Analyst Base

```
const theme = defineMetabaseTheme({  
  // Specify a font to use from the set of fonts supported by Metabase.  
  // You can set the...
```

### **docs/embedding/sdk/snippets/authentication/express-server-cors.ts**

```
import cors from "cors";  
import express from "express";  
import session from "express-session";  
  
const PORT = 3000;  
const SESSION_SECRET = "SECRET";
```

```
const app = express();  
const metabaseAuthHandler =...
```

### **docs/embedding/sdk/snippets/authentication/express-server-interactive-and-sd k.ts**

```
import express from "express";  
import jwt from "jsonwebtoken";  
  
// Replace this with your Metabase URL  
const METABASE_INSTANCE_URL = "YOUR_METABASE_URL_HERE";  
// Replace this with the JWT signing...
```

### **docs/embedding/sdk/snippets/authentication/express-server.ts**

```
import express from "express";  
import cors from "cors";  
import session from "express-session";  
import jwt from "jsonwebtoken";  
import fetch from "node-fetch";
```

```
// Replace this with your Metabase...
```

### **docs/embedding/sdk/snippets/next-js/app-router-authentication-api-route.ts**

```
[<snippet imports>]
```

### **docs/embedding/sdk/snippets/next-js/declarations.d.ts**

Required to properly resolve the ``import { StaticQuestion } from "@metabase";`` in ``manual-wrapping-usage.tsx`` snippet

### **docs/embedding/sdk/snippets/next-js/pages-router-authentication-api-route.ts**

```
[<snippet imports>]
```

### **docs/embedding/sdk/upgrade.md**

## Analyst Base

---

title: Upgrading Metabase and the Embedded analytics SDK

summary: How to upgrade your Metabase and Embedded analytics SDK versions, test the changes, and check for breaking changes that might...

**docs/embedding/sdk/version.md**

---

title: Embedded analytics SDK - versions

---

# Embedded analytics SDK - versions

{% include plans-blockquote.html feature="Embedded analytics SDK" sdk=true %}

The SDK stable version tracks...

**docs/embedding/securing-embeds.md**

---

title: Securing embedded Metabase

summary: How to hide and protect sensitive data in different types of embeds.

---

# Securing embedded Metabase

## Securing embeds with authentication and...

**docs/embedding/snippets/interactive-embedding-quick-start-guide/sso-with-jwt.ts**

```
import express from "express";
```

```
// [<snippet jsonwebtoken-import>]
```

```
import jwt from "jsonwebtoken";
```

```
// [<endsnippet jsonwebtoken-import>]
```

```
const app = express();
```

```
const METABASE_INSTANCE_URL =...
```

**docs/embedding/start.md**

---

title: Embedding overview

redirect\_from:

- /docs/latest/embedding

---



## Analyst Base

# Embedding overview

## [Introduction](./introduction.md)

What is embedding, and how does it work?

## [Interactive...

### **docs/embedding/static-embedding-parameters.md**

---

title: Parameters for static embeds

summary: Using parameters to customize embedded static questions and dashboards.

redirect\_from:

- /docs/latest/embedding/signed-embedding-parameters

---

#...

### **docs/embedding/static-embedding.md**

---

title: Static embedding

redirect\_from:

- /docs/latest/embedding/signed-embedding

---

# Static embedding

Also known as: standalone embedding, or signed embedding.

{% include...

### **docs/embedding/translations.md**

---

title: Translate embedded dashboards and questions

summary: Upload a translation dictionary to translate questions and dashboards into different languages.

Only available for static...

### **docs/exploration-and-organization/collections.md**

---

title: Collections

redirect\_from:

- /docs/latest/users-guide/collections

---

## Analyst Base

### # Collections

Collections are the main way to organize [questions](../questions/introduction.md),...

### **docs/exploration-and-organization/content-verification.md**

---

title: Content verification

---

### # Content verification

{% include plans-blockquote.html feature="Content verification" %}

![Verified icon](../images/verified-icon.png)

Admins can verify items...

### **docs/exploration-and-organization/data-model-reference.md**

---

title: Data reference

redirect\_from:

- /docs/latest/users-guide/12-data-model-reference
- /docs/latest/questions/native-editor/data-model-reference

---

### # Data reference

You can open the...

### **docs/exploration-and-organization/delete-and-restore.md**

---

title: Delete and restore

---

### # Delete and restore

Sometimes your questions, dashboards, models, or collections outlive their usefulness. You can send outdated items to **Trash**.

![Move to...

### **docs/exploration-and-organization/events-and-timelines.md**

---

title: Events and timelines

## Analyst Base

redirect\_from:

- /docs/latest/users-guide/events-and-timelines

---

# Events and timelines

A lot of discussions around data have a moment when someone asks a...

**docs/exploration-and-organization/exploration.md**

---

title: Basic exploration

redirect\_from:

- /docs/latest/users-guide/03-basic-exploration

---

# Basic exploration

## See what your teammates have made

As long as you're not the very first user...

**docs/exploration-and-organization/history.md**

---

title: History

---

# History

For questions, dashboards, and models, Metabase keeps a version history for the previous fifteen versions of that item. You can view changes, and revert to previous...

**docs/exploration-and-organization/keyboard-shortcuts.md**

---

title: Keyboard shortcuts

summary: A handy list of keyboard shortcuts to help you zip around Metabase.

---

# Keyboard shortcuts

A handy list of keyboard shortcuts to help you zip around...

**docs/exploration-and-organization/start.md**

---

title: "Organization overview"

## Analyst Base

redirect\_from:

- /docs/latest/exploration-and-organization

---

# Organization overview

Tools for finding things and keeping your Metabase organized.

## [Basic...

### **docs/exploration-and-organization/uploads.md**

---

title: Uploading data

---

# Uploading data

If an admin has [set up uploads](../databases/uploads.md), you can upload CSV data by clicking on the **Upload** icon in the top right of the...

### **docs/exploration-and-organization/x-rays.md**

---

title: X-rays

redirect\_from:

- /docs/latest/users-guide/14-x-rays

---

# X-rays

X-rays are a way to get automatic insights and explorations of your data.

## Get automatic insights when...

### **docs/installation-and-operation/accessibility.md**

---

title: Accessibility in Metabase

redirect\_from:

- /docs/latest/accessibility
- /docs/latest/people-and-groups/accessibility

---

# Accessibility in Metabase

While we're working to make a...

## **docs/installation-and-operation/activating-the-enterprise-edition.md**

---

title: Activating your Metabase commercial license

redirect\_from:

- /docs/latest/enterprise-guide/activating-the-enterprise-edition

---

## **docs/installation-and-operation/backing-up-metabase-application-data.md**

---

title: Backing up Metabase

redirect\_from:

- /docs/latest/operations-guide/backing-up-metabase-application-data

---

### **# Backing up Metabase**

Avoid losing your application data (all of your...

## **docs/installation-and-operation/commands.md**

---

title: Metabase CLI

---

### **# Metabase CLI**

Metabase ships with some handy CLI commands. To view a list of commands, run the Metabase jar followed by `help`.

---

java --add-opens...

## **docs/installation-and-operation/configuring-application-database.md**

---

title: Configuring the Metabase application database

redirect\_from:

- /docs/latest/operations-guide/configuring-application-database

---

### **# Configuring the Metabase application database**

The...

## **docs/installation-and-operation/creating-RDS-database-on-AWS.md**

## Analyst Base

---

title: "Creating an RDS database on AWS"

redirect\_from:

- /docs/latest/operations-guide/creating-RDS-database-on-AWS

---

# Creating an RDS database on AWS

If you want to move from using...

**docs/installation-and-operation/development-instance.md**

---

title: Development instances

summary: Create development instances of Metabase for testing without paying per user. Perfect for trying out changes before pushing them to production.

---

#...

**docs/installation-and-operation/information-collection.md**

---

title: About the anonymous usage data we collect

redirect\_from:

- /docs/latest/information-collection

---

# About the anonymous usage data we collect

If you're self-hosting Metabase and...

**docs/installation-and-operation/installing-metabase.md**

---

title: Installing Metabase

redirect\_from:

- /docs/latest/operations-guide/installing-metabase

---

# Installing Metabase

Metabase is built and packaged as a Java JAR file and can be run...

**docs/installation-and-operation/migrating-from-h2.md**

---

title: Migrating to a production application database

## Analyst Base

redirect\_from:

- /docs/latest/operations-guide/migrating-from-h2
- /docs/latest/operations-guide/running-migrations-manually

---

#...

### **docs/installation-and-operation/monitoring-metabase.md**

---

title: Monitoring your Metabase

redirect\_from:

- /docs/latest/operations-guide/jmx-monitoring
- /docs/latest/operations-guide/enable-jmx

---

#### # Monitoring Your Metabase

Diagnosing...

### **docs/installation-and-operation/observability-with-prometheus.md**

---

title: Observability with Prometheus

---

#### # Observability with Prometheus

You can export metrics in [Prometheus](https://prometheus.io/) format from your Metabase.

#### ## Running Metabase and...

### **docs/installation-and-operation/privacy.md**

---

title: Privacy

redirect\_from:

- /docs/latest/privacy

---

#### # Privacy

#### ## Do you need a Data Processing Agreement with Metabase to comply with GDPR?

#### ### Self-hosted

If you self-host your...

## **docs/installation-and-operation/running-metabase-on-azure.md**

---

title: Running Metabase on Microsoft Azure

redirect\_from:

- /docs/latest/operations-guide/running-metabase-on-azure

---

# Running Metabase on Microsoft Azure

This guide covers the basics for...

## **docs/installation-and-operation/running-metabase-on-debian.md**

---

title: Running Metabase on Debian as a service with nginx

redirect\_from:

- /docs/latest/operations-guide/running-metabase-on-debian

---

# Running Metabase on Debian as a service with...

## **docs/installation-and-operation/running-metabase-on-docker.md**

---

title: Running Metabase on Docker

redirect\_from:

- /docs/latest/operations-guide/running-metabase-on-docker

---

# Running Metabase on Docker

> To get fast, reliable, and secure deployment...

## **docs/installation-and-operation/running-metabase-on-elastic-beanstalk.md**

---

title: Running Metabase on AWS Elastic Beanstalk

redirect\_from:

- /docs/latest/operations-guide/running-metabase-on-elastic-beanstalk

^...

## **docs/installation-and-operation/running-metabase-on-podman.md**

---

title: Running Metabase on Podman

redirect\_from:

- /docs/latest/operations-guide/running-metabase-on-podman



## Analyst Base

---

# Running Metabase on Podman

Our official Metabase Docker image is compatible...

**docs/installation-and-operation/running-the-metabase-jar-file.md**

---

title: Running the Metabase JAR file

redirect\_from:

- /docs/latest/operations-guide/running-the-metabase-jar-file
- /docs/installation-and-operation/java-versions

---

# Running the Metabase...

**docs/installation-and-operation/serialization.md**

---

title: "Serialization"

summary: How to export and import Metabase content between instances using serialization. Useful for version control, staging environments, and duplicating...

**docs/installation-and-operation/start.md**

---

title: "Installation and operation overview"

---

# Installation and operation overview

The birth, care, and feeding of your Metabase.

## [Installing...

**docs/installation-and-operation/supported-browsers.md**

---

title: Supported browsers

redirect\_from:

- /docs/latest/administration-guide/supported-browsers

---

# Supported browsers

We try our best to make sure Metabase works in as many browsers as...

**docs/installation-and-operation/upgrading-metabase.md**

## Analyst Base

---

title: Upgrading Metabase

redirect\_from:

- /docs/latest/operations-guide/upgrading-metabase

---

### # Upgrading Metabase

This page covers how to upgrade to a new Metabase release.

^...

## **docs/people-and-groups/account-settings.md**

---

title: Account settings

redirect\_from:

- /docs/latest/users-guide/account-settings

---

### # Account settings

You can view your account settings by going to the top right of the screen and...

## **docs/people-and-groups/api-keys.md**

---

title: API keys

---

### # API keys

Metabase can create API keys to authenticate programmatic requests to the API. To set the permissions for an API key, you can assign the key to a...

## **docs/people-and-groups/authenticating-with-jwt.md**

---

title: JWT-based authentication

description: How to set up JWT-based authentication in Metabase to connect with your identity provider and manage user access.

redirect\_from:

^...

## **docs/people-and-groups/authenticating-with-saml.md**

---

title: SAML-based authentication

## Analyst Base

redirect\_from:

- /docs/latest/enterprise-guide/authenticating-with-saml

---

# SAML-based authentication

{% include plans-blockquote.html feature="SAML..."

### **docs/people-and-groups/changing-password-complexity.md**

---

title: Passwords

redirect\_from:

- /docs/latest/operations-guide/changing-password-complexity

---

# Passwords

Metabase can allow authentication via email and password.

## Password...

### **docs/people-and-groups/changing-session-expiration.md**

---

title: Session expiration

redirect\_from:

- /docs/latest/operations-guide/changing-session-expiration

---

# Session expiration

By default, Metabase sessions are valid for two weeks after a...

### **docs/people-and-groups/google-sign-in.md**

---

title: Google Sign-In

redirect\_from:

- /docs/latest/administration-guide/10-single-sign-on
- /docs/latest/people-and-groups/google-and-ldap

---

# Google Sign-In

Enabling [Google...

## docs/people-and-groups/ldap.md

---

title: LDAP

---

### # LDAP

Metabase supports authentication with Lightweight Directory Access Protocol (LDAP).

You can find SSO options under **Admin settings** > **Settings** >...

## docs/people-and-groups/managing.md

---

title: People and groups

redirect\_from:

- /docs/latest/administration-guide/04-managing-users

---

### # People and groups

People can have [accounts](#creating-an-account) in Metabase, and those...

## docs/people-and-groups/saml-auth0.md

---

title: SAML with Auth0

redirect\_from:

- /docs/latest/enterprise-guide/saml-auth0

---

### # SAML with Auth0

{% include plans-blockquote.html feature="SAML authentication" %}

1. [Configure SAML in...

## docs/people-and-groups/saml-azure.md

---

title: SAML with Microsoft Entra ID

redirect\_from:

- /docs/latest/enterprise-guide/authenticating-with-saml-azure-ad

---

## Analyst Base

# SAML with Microsoft Entra ID

{% include plans-blockquote.html...

### **docs/people-and-groups/saml-google.md**

---

title: SAML with Google

redirect\_from:

- /docs/latest/enterprise-guide/saml-google

---

# SAML with Google

{% include plans-blockquote.html feature="Google SAML authentication" %}

1. Set up a...

### **docs/people-and-groups/saml-keycloak.md**

---

title: SAML with Keycloak

redirect\_from:

- /docs/latest/enterprise-guide/saml-keycloak

---

# SAML with Keycloak

Keycloak is an open source platform that can be used as a user directory to...

### **docs/people-and-groups/saml-okta.md**

---

title: SAML with Okta

---

# SAML with Okta

{% include plans-blockquote.html feature="Okta SAML authentication" %}

1. [Turn on SAML-based SSO in...

### **docs/people-and-groups/start.md**

---

title: "People overview"

redirect\_from:

- /docs/latest/administration-guide/sso

## Analyst Base

- /docs/latest/people-and-groups

---

### # People overview

User accounts, groups, and authentication. For...

## **docs/people-and-groups/user-provisioning.md**

---

title: User provisioning with SCIM

---

### # User provisioning with SCIM

{% include plans-blockquote.html feature="User provisioning with SCIM" %}

Metabase supports user provisioning via the...

## **docs/permissions/application.md**

---

title: Application permissions

redirect\_from:

- /docs/latest/administration-guide/application-permissions

---

### # Application permissions

{% include plans-blockquote.html feature="Application..." %}

## **docs/permissions/collections.md**

---

title: Collection permissions

redirect\_from:

- /docs/latest/administration-guide/06-collections

---

### # Collection permissions

![Collection detail](./images/collection-detail.png)

You can use...

## **docs/permissions/data.md**

---

## Analyst Base

title: Data permissions

redirect\_from:

- /docs/latest/administration-guide/data-permissions

---

# Data permissions

This page covers permissions for databases and tables. If you haven't...

### **docs/permissions/database-routing.md**

---

title: Database routing

summary: Route queries to different databases based on who's viewing them. Great for multi-tenant setups where each customer has their own database.

---

# Database...

### **docs/permissions/embedding.md**

---

title: Configuring permissions for embedding

summary: Learn about which permissions tooling you should use depending on whether your customer data is in one database or split across multiple...

### **docs/permissions/impersonation.md**

---

title: Impersonation access

---

# Impersonation permissions

{% include plans-blockquote.html feature="Impersonation access" %}

> For now, impersonation access is only available for ClickHouse,...

### **docs/permissions/introduction.md**

---

title: Permissions introduction

redirect\_from:

- /docs/latest/administration-guide/05-setting-permissions

---

# Permissions introduction

## Analyst Base

There are always going to be sensitive bits of...

### **docs/permissions/no-self-service-deprecation.md**

---

title: Migrating from legacy permissions

---

# Migrating from legacy permissions

> In Metabase 56, we renamed Data sandboxing to Row and column security. The functionality is the same. Some...

### **docs/permissions/notifications.md**

---

title: Notification permissions

summary: Learn who can create and edit alerts and dashboard subscriptions, and what data recipients can see in their notifications.

---

# Notification...

### **docs/permissions/row-and-column-security-examples.md**

---

title: Row and column security examples

redirect\_from:

- /docs/latest/permissions/row-and-column-security-examples

---

# Row and column security examples

{% include plans-blockquote.html...

### **docs/permissions/row-and-column-security.md**

---

title: Row and column security

redirect\_from:

- /docs/latest/enterprise-guide/data-sandboxes
- /docs/latest/permissions/data-sandboxes

---

# Row and column security

{% include...

### **docs/permissions/snippets.md**



## Analyst Base

---

title: Snippet folder permissions

redirect\_from:

- /docs/latest/enterprise-guide/sql-snippets

---

# Snippet folder permissions

{% include plans-blockquote.html feature="Snippet controls"...

### **docs/permissions/start.md**

---

title: Permissions overview

redirect\_from:

- /docs/latest/permissions

---

# Permissions overview

Metabase has a simple, but powerful permissions system.

## [Permissions...

### **docs/questions/alerts.md**

---

title: Alerts

summary: Set up alerts to get notified via email, Slack, or webhooks when your questions return interesting results.

redirect\_from:

- /docs/latest/users-guide/15-alerts

-...

### **docs/questions/exporting-results.md**

---

title: Exporting results

summary: Exporting your Metabase questions and dashboards as CSV, Excel, JSON, PDF, or PNG files.

redirect\_from:

-...

### **docs/questions/introduction.md**

---

title: Questions

redirect\_from:

- /docs/latest/users-guide/04-asking-questions

## Analyst Base

- /docs/latest/users-guide/06-sharing-answers

---

# Questions

![[Metabase...

### **docs/questions/native-editor/basic-sql-parameters.md**

---

title: Basic SQL parameters

summary: Text, number, and date variables let you plug basic values into your SQL code.

---

# Basic SQL parameters

> If you want to filter on a database field in...

### **docs/questions/native-editor/field-filters.md**

---

title: Field filters

summary: Field filters let you create smart filter widgets for your SQL questions by connecting variables to database fields. Where possible, prefer field filters to basic...

### **docs/questions/native-editor/filter-widgets.md**

---

title: Filter and parameter widgets for native code

---

# Filter and parameter widgets for native code

When you add a [SQL variable or parameter](./sql-parameters.md) to your native/SQL query,...

### **docs/questions/native-editor/optional-variables.md**

---

title: Optional variables

summary: Make parts of your SQL query optional by wrapping clauses in double brackets.

---

# Optional variables

You can make a clause optional in a query. For example,...

### **docs/questions/native-editor/referencing-saved-questions-in-queries.md**

---

## Analyst Base

title: Referencing models and saved questions

redirect\_from:

- /docs/latest/users-guide/referencing-saved-questions-in-queries

---

# Referencing models and saved questions

With SQL databases,...

### **docs/questions/native-editor/snippets.md**

---

title: Snippets

redirect\_from:

- /docs/latest/users-guide/sql-snippets
- /docs/latest/questions/native-editor/sql-snippets

---

# Snippets

![SQL...

### **docs/questions/native-editor/sql-parameters.md**

---

title: SQL parameters

summary: Create SQL templates by adding filters and parameters to your SQL questions in the native code editor.

redirect\_from:

^...

### **docs/questions/native-editor/time-grouping-parameters.md**

---

title: Time grouping parameter

---

# Time grouping parameter

You can add a parameter to SQL questions to change how results are grouped by time: by day, week, month, and so on.

To add a time...

### **docs/questions/native-editor/writing-sql.md**

---

title: SQL editor

## Analyst Base

redirect-from:

- /docs/latest/users-guide/writing-sql
- /docs/latest/questions/native-editor

---

# SQL editor

If you ever need to ask questions that can't be expressed...

### **docs/questions/query-builder/editor.md**

---

title: The query builder

redirect\_from:

- /docs/latest/questions/query-builder/introduction
- /docs/latest/questions/query-builder

---

# The query builder

Metabase includes a graphical...

### **docs/questions/query-builder/expressions/case.md**

---

title: Case

---

# Case

`case` (alias `if`) checks if a value matches a list of conditions, and returns some output based on the first condition that's met. `if` and `case` work exactly the...

### **docs/questions/query-builder/expressions/coalesce.md**

---

title: Coalesce

---

# Coalesce

`coalesce` looks at the values in a list (in order), and returns the first non-null value.

This function is useful when you want to:

- [fill in missing...

## **docs/questions/query-builder/expressions/concat.md**

```
---
title: Concat
---

# Concat

`concat` concatenates data from two or more columns or values, and returns a string.

## Syntax

...

concat(value1, value2, ...)
...

`value1`, `value2` ... can be...
```

## **docs/questions/query-builder/expressions/converttimezone.md**

```
---
title: ConvertTimezone
---

# ConvertTimezone

`convertTimezone` shifts a timestamp into a specified time zone by adding or subtracting the right interval from the timestamp.

| Syntax      ...
```

## **docs/questions/query-builder/expressions/countif.md**

```
---
title: CountIf
---

# CountIf

`CountIf` counts the total number of rows in a table that match a condition. `CountIf` counts every row, not just unique rows.

Syntax:...
```

## **docs/questions/query-builder/expressions/cumulative.md**

```
---
```

## Analyst Base

title: Cumulative count and sum

redirect\_from:

- /docs/latest/questions/expressions/cumulativesum
- /docs/latest/questions/expressions/cumulativecount

---

# Cumulative count and...

**docs/questions/query-builder/expressions/datetimeadd.md**

---

title: DatetimeAdd

---

# DatetimeAdd

`datetimeAdd` takes a datetime value and adds some unit of time to it. This function is useful when you're working with time series data that's marked by a...

**docs/questions/query-builder/expressions/datetimediff.md**

---

title: DatetimeDiff

---

# DatetimeDiff

`datetimeDiff` gets the amount of time between two datetime values, using the specified unit of time. Note that the difference is calculated in `_whole_...`

**docs/questions/query-builder/expressions/datetimesubtract.md**

---

title: DatetimeSubtract

---

# DatetimeSubtract

`datetimeSubtract` takes a datetime value and subtracts some unit of time from it. You might want to use this function when working with time...

**docs/questions/query-builder/expressions/in.md**

---

title: In

---

# In

## Analyst Base

``in`` compares values and returns true if ``value1`` equals ``value2`` (OR ``value3``, etc., if specified).

### ## Syntax

---

`in(value1, value2, ...)`

---

``value1`` is the column or...

**[docs/questions/query-builder/expressions/isempty.md](#)**

---

title: Iempty

---

### # Iempty

``isEmpty`` checks whether a value in a **string column** is an empty string (``""``) or null. Calling ``isEmpty`` on a non-string column would cause an error.

##...

**[docs/questions/query-builder/expressions/isnull.md](#)**

---

title: Isnull

---

### # Isnull

``isNull`` checks if a value is a ``null``, a special kind of placeholder that's used by a database when something is missing or unknown.

### ## Syntax

---

`isNull(text...`

**[docs/questions/query-builder/expressions/now.md](#)**

---

title: Now

---

### # Now

## Analyst Base

``now`` returns the current datetime using your Metabase [report timezone](../../configuring-metabase/localization.md#report-timezone).

## Creating conditional logic...

### **docs/questions/query-builder/expressions/offset.md**

---

title: Offset

---

# Offset

> The ``Offset`` function is currently unavailable for MySQL/MariaDB, ClickHouse, MongoDB, and Druid.

The ``Offset`` function returns the value of an expression in a...

### **docs/questions/query-builder/expressions/regexextract.md**

---

title: RegexExtract

---

# RegexExtract

> ``regexExtract`` is unavailable for MongoDB, SQLite, and SQL Server. For Druid, ``regexExtract`` is only available for the Druid-JDBC...

### **docs/questions/query-builder/expressions/substring.md**

---

title: Substring

---

# Substring

``substring`` extracts part of some text. This function is useful for cleaning up text (or any value with a [string data...

### **docs/questions/query-builder/expressions/sumif.md**

---

title: SumIf

---

# SumIf



## Analyst Base

`SumIf` adds up the values in a column based on a condition.

Syntax: `SumIf(column, condition)`.

Example: in the table below, `SumIf([Payment], [Plan] = "Basic")` ...

### **docs/questions/query-builder/expressions/week.md**

---

title: Week of year

description: In Metabase, you can group by week of year in the query builder, or extract the week of year from a date column using a custom expression. Metabase supports...

### **docs/questions/query-builder/expressions-list.md**

---

title: List of expressions

redirect\_from:

- /docs/latest/users-guide/expressions-list

---

# List of expressions

For an introduction to expressions, check out the [overview of custom...

### **docs/questions/query-builder/expressions.md**

---

title: Custom expressions

redirect\_from:

- /docs/latest/users-guide/expressions

---

# Custom expressions

![Custom expression editor](../images/custom-expression-editor.png)

[Custom...

### **docs/questions/query-builder/filters.md**

---

title: Filtering

---

# Filtering

Filtering just means narrowing things down based on certain criteria. You're probably already familiar with

## Analyst Base

filtering when looking for something online, like...

### **docs/questions/query-builder/join.md**

---

title: Joining data

summary: Learn how to combine data from different tables using joins. We'll show you how to pick tables, match columns, and choose the right join type.

redirect\_from:

-...

### **docs/questions/query-builder/summarizing-and-grouping.md**

---

title: Summarizing and grouping

summary: Learn how to summarize and group your data in Metabase's query builder to calculate metrics like counts, sums, and averages across different...

### **docs/questions/start.md**

---

title: Questions overview

redirect\_from:

- /docs/latest/questions

---

#### # Questions overview

Questions are queries plus their visualization. You can ask questions using Metabase's graphical...

### **docs/questions/visualizations/combo-chart.md**

---

title: Combo charts

redirect\_from:

- /docs/latest/questions/sharing/visualizations/combo-chart

---

#### # Combo charts

Combo charts let you combine bars and lines (or areas) on the same...

### **docs/questions/visualizations/country-codes.md**

---

title: Country codes

---

#### # Country codes

## Analyst Base

This reference lists all country codes and their corresponding country names used in Metabase's default world map visualizations. The data comes from...

### **docs/questions/visualizations/detail.md**

---

title: Detail

redirect\_from:

- /docs/latest/questions/sharing/visualizations/detail

---

# Detail

The **Detail** visualization shows a single result record (row) in an easy-to-read, two-column...

### **docs/questions/visualizations/funnel.md**

---

title: Funnel charts

redirect\_from:

- /docs/latest/questions/sharing/visualizations/funnel

description: Funnel charts visualize how a value is broken out by a series of steps, and the percent...

### **docs/questions/visualizations/gauge.md**

---

title: Gauge chart

redirect\_from:

- /docs/latest/questions/sharing/visualizations/gauge

---

# Gauge chart

Ah, **gauges**: you either love 'em or you hate 'em. ...Or you feel "meh" about them,...

### **docs/questions/visualizations/line-bar-and-area-charts.md**

---

title: Line charts, bar charts, and area charts

redirect\_from:

- /docs/latest/questions/visualizations/line-bar-and-area-charts

---

# Line charts, bar charts, and area charts

They're pretty...

### **docs/questions/visualizations/map.md**

## Analyst Base

---

title: Maps

redirect\_from:

- /docs/latest/questions/sharing/visualizations/maps

description: "Maps in Metabase allow you to visualize geographical data either using coordinates or by region...."

### **docs/questions/visualizations/numbers.md**

---

title: Numbers

redirect\_from:

- /docs/latest/questions/sharing/visualizations/numbers

---

# Numbers

The **Numbers** option is for displaying a single number, nice and big. The options for...

### **docs/questions/visualizations/pie-or-donut-chart.md**

---

title: Pie and sunburst charts

redirect\_from:

- /docs/latest/questions/sharing/visualizations/pie-or-donut-chart

---

# Pie and sunburst charts

A **pie chart** can be used when breaking out a...

### **docs/questions/visualizations/pivot-table.md**

---

title: Pivot tables

redirect\_from:

- /docs/latest/questions/sharing/visualizations/pivot-table

---

# Pivot tables

> Pivot tables are currently only supported for questions built in the [query...

### **docs/questions/visualizations/progress-bar.md**

---

title: Progress bars

redirect\_from:

## Analyst Base

- /docs/latest/questions/sharing/visualizations/progress-bar

---

### # Progress bars

**\*\*Progress bars\*\*** are for comparing a single number to a goal value that...

## **docs/questions/visualizations/sankey.md**

---

title: Sankey charts

redirect\_from:

- /docs/latest/questions/sharing/visualizations/sankey

---

### # Sankey charts

Sankey charts show how data flows through multi-dimensional steps. They're...

## **docs/questions/visualizations/scatterplot-or-bubble-chart.md**

---

title: Scatterplots and bubble charts

redirect\_from:

- /docs/latest/questions/sharing/visualizations/scatterplot-or-bubble-chart

---

### # Scatterplots and bubble charts

**\*\*Scatterplots\*\*** are...

## **docs/questions/visualizations/table.md**

---

title: Tables

redirect\_from:

- /docs/latest/questions/sharing/visualizations/table

---

### # Tables

![A table in Metabase](../images/table.png)

Tables are data's natural habitat.

##...

## docs/questions/visualizations/tooltips.md

---

title: Tooltips

redirect\_from:

- /docs/latest/questions/sharing/visualizations/tooltips

---

### # Tooltips

When you hover over a data point on a chart in Metabase, you'll see a tooltip about that...

## docs/questions/visualizations/trend.md

---

title: Trend

redirect\_from:

- /docs/latest/questions/sharing/visualizations/trend

---

### # Trend

![Trend settings](../images/trend-settings.png)

The **Trend** visualization is great for...

## docs/questions/visualizations/visualizing-results.md

---

title: Visualization overview

redirect\_from:

- /docs/latest/users-guide/05-visualizing-results
- /docs/latest/questions/sharing/visualizing-results
- ...

## docs/questions/visualizations/waterfall-chart.md

---

title: Waterfall charts

redirect\_from:

- /docs/latest/questions/sharing/visualizations/waterfall-chart

---

### # Waterfall charts

Waterfall charts are a kind of bar chart useful for visualizing...

## docs/troubleshooting-guide/bigquery-drive.md

## Analyst Base

---

title: Troubleshooting BigQuery and Google Drive connections in Metabase

---

# Troubleshooting BigQuery and Google Drive connections in Metabase

[This...

**docs/troubleshooting-guide/bugs.md**

---

title: Reporting a bug

---

# Reporting a bug

If you come across something that looks like a bug, please start by searching our [GitHub issues][metabase-issues] to see if it has already been...

**docs/troubleshooting-guide/cant-log-in.md**

---

title: People can't log in to Metabase

---

# People can't log in to Metabase

## Reset password

To reset a password for your Metabase instance, see:

- [Reset a user's...

**docs/troubleshooting-guide/cant-see-tables.md**

---

title: I can't see my tables

---

# I can't see my tables

You've connected Metabase to a database, but:

- you don't see the tables in the [Table Metadata](../data-modeling/metadata-editing.md)...

**docs/troubleshooting-guide/cant-send-email.md**

---

## Analyst Base

title: Metabase isn't sending email

---

# Metabase isn't sending email

You have told Metabase to send email notifications, but:

- the notifications aren't arriving.

Before any other...

**docs/troubleshooting-guide/cant-view-or-edit.md**

---

title: Can't view or edit

---

# Can't view or edit

1. Clear your browser cache.
2. Check if a browser extension or plugin is interfering with Metabase:
  - Disable all extensions and plugins,

...

**docs/troubleshooting-guide/create-har-file.md**

---

title: Creating a HAR file for troubleshooting

---

# Creating a HAR file for troubleshooting

HAR (short for \_HTTP Archive\_) files record network requests generated by your browser. HAR files...

**docs/troubleshooting-guide/data-permissions.md**

---

title: Troubleshooting data permissions

---

# Troubleshooting data permissions

If a person has the wrong level of access to the data that's returned by a question or query, you'll need to...

**docs/troubleshooting-guide/db-connection.md**

---

title: Troubleshooting database connections



## Analyst Base

redirect\_from:

- /docs/latest/troubleshooting-guide/datawarehouse

---

# Troubleshooting database connections

If you can't connect to your database,...

**docs/troubleshooting-guide/db-performance.md**

---

title: Troubleshooting database performance

---

# Troubleshooting database performance

This guide deals with databases or data warehouses that are [connected to...

**docs/troubleshooting-guide/diagnostic-info.md**

---

title: Diagnostic information for troubleshooting

---

# Diagnostic information for troubleshooting

To download diagnostic information, hit `Cmd + F1` on Macs, `Ctrl + F1` on PCs. Hit Cmd/Ctrl +...

**docs/troubleshooting-guide/docker.md**

---

title: Troubleshooting Metabase on Docker

---

# Troubleshooting Metabase on Docker

Docker simplifies many aspects of running Metabase, but there are some pitfalls to keep in mind. If you have...

**docs/troubleshooting-guide/error-message.md**

---

title: Troubleshooting error messages

---

# Troubleshooting error messages

An error message can help you find the right troubleshooting guide. The exact wording depends on your

database and...

### **docs/troubleshooting-guide/filters.md**

---

title: Troubleshooting filters

---

# Troubleshooting filters

It's always a good idea to start with a quick sanity check:

1. Clear your browser cache.
2. Refresh the page.
- 3.

### **docs/troubleshooting-guide/index.md**

---

title: Troubleshooting guides

---

# Troubleshooting guides

Problems, their causes, how to detect them, and how to fix them.

## Getting diagnostic info

- [Download diagnostic...

### **docs/troubleshooting-guide/known-issues.md**

---

title: How to find a known bug or limitation

---

# How to find a known bug or limitation

If you can't find or solve your problem using the [troubleshooting guides](./index.md), you may be...

### **docs/troubleshooting-guide/ldap.md**

---

title: Troubleshooting LDAP

---

# Troubleshooting LDAP

## Analyst Base

Metabase can use LDAP for authentication. [This article](../people-and-groups/ldap.md) explains how to set it up, and the guide below will...

### **docs/troubleshooting-guide/linked-filters.md**

---

title: My linked filters don't work

---

# My linked filters don't work

You have created a [linked filter][linked-filter-gloss] so that (for example) if a dashboard contains both a "State" and a...

### **docs/troubleshooting-guide/loading-from-h2.md**

---

title: Using or migrating from an H2 application database

---

# Using or migrating from an H2 application database

You've installed Metabase, but:

- You're trying to migrate the application...

### **docs/troubleshooting-guide/models.md**

---

title: Troubleshooting models

---

# Troubleshooting models

What kind of problem are you having with your [model][model-docs]?

## Can't create a model

If you don't see [the model]...

### **docs/troubleshooting-guide/my-dashboard-is-slow.md**

---

title: My dashboard is slow

---

# My dashboard is slow

## Analyst Base

First, you'll want to make sure your browser is on friendly terms with Metabase:

- Clear your browser cache and disable all extensions...

### **docs/troubleshooting-guide/notifications.md**

---

title: Troubleshooting notifications

---

#### # Troubleshooting notifications

Metabase is failing to send notifications like alerts or dashboard subscriptions.

**\*\*Root cause:\*\***

When long running...

### **docs/troubleshooting-guide/permissions.md**

---

title: Troubleshooting permissions

---

#### # Troubleshooting permissions

If someone has the wrong level of access to a dashboard or a question, the problem may be coming from group settings,...

### **docs/troubleshooting-guide/proxies.md**

---

title: Can't save questions or dashboards, or getting a blank page

---

#### # Can't save questions or dashboards, or getting a blank page

If attempting to save a question or dashboard sometimes...

### **docs/troubleshooting-guide/requesting-new-features.md**

---

title: How to request new features

---

#### # How to request new features

1. Check out the [issues in the github repo][github-issues] to make sure someone hasn't already requested

the feature.

2. If...

### **docs/troubleshooting-guide/row-and-column-security.md**

---

title: Troubleshooting row and column security access

redirect\_from:

- /docs/latest/troubleshooting-guide/sandboxing

---

# Troubleshooting row and column security

[Row and column...

### **docs/troubleshooting-guide/running.md**

---

title: Troubleshooting memory and JVM issues

---

# Troubleshooting memory and JVM issues

Metabase runs on the Java Virtual Machine (JVM), and depending on how it's configured, it may use the...

### **docs/troubleshooting-guide/saml.md**

---

title: Troubleshooting SAML authentication setup

---

# Troubleshooting SAML authentication setup

{% include plans-blockquote.html feature="SAML authentication" %}

Some common problems when...

### **docs/troubleshooting-guide/server-logs.md**

---

title: How to read the server logs

---

# How to read the server logs

Here's an example log from running a query:

...

## Analyst Base

2021-07-07 15:53:18,560 DEBUG middleware.log :: POST /api/dataset 202...

### **docs/troubleshooting-guide/sql.md**

---

title: Troubleshooting SQL questions

---

# Troubleshooting SQL questions

## Incorrect results

- [Aggregations (counts, sums, etc.) are...

### **docs/troubleshooting-guide/sync-fingerprint-scan.md**

---

title: Troubleshooting syncs, scans, and fingerprinting

---

# Troubleshooting syncs, scans, and fingerprinting

First, check if your data is outdated because of browser caching:

1. Clear your...

### **docs/troubleshooting-guide/timeout.md**

---

title: Troubleshooting connection timeouts

---

# Troubleshooting connection timeouts

If your queries are hanging or timing out, the problem could be coming from your:

- [Database...

### **docs/troubleshooting-guide/timezones.md**

---

title: The dates and times in my questions and charts are wrong

---

# The dates and times in my questions and charts are wrong

You are doing calculations with dates and times, or displaying...

### **docs/troubleshooting-guide/visualization.md**

## Analyst Base

---

title: Troubleshooting question and dashboard visualizations

---

# Troubleshooting question and dashboard visualizations

To start, check if your current browser settings are compatible with...

**docs/usage-and-performance-tools/audit.md**

---

title: Auditing tools

redirect\_from:

- /docs/latest/enterprise-guide/audit

---

# Auditing tools

> Auditing tools are deprecated. Instead, check out the (much better) [Usage...

**docs/usage-and-performance-tools/start.md**

---

title: "Tools overview"

redirect\_from:

- /docs/latest/usage-and-performance-tools

---

# Tools overview

Administration tools for managing your Metabase.

## [Usage...

**docs/usage-and-performance-tools/tools.md**

---

title: Admin tools

redirect\_from:

- /docs/latest/enterprise-guide/tools

---

# Admin tools

The Admin **Tools** tab contains features for troubleshooting. To get to the Admin tools sections, go...

**docs/usage-and-performance-tools/usage-analytics.md**

## Analyst Base

```
---
title: Usage analytics
---

# Usage analytics

{% include plans-blockquote.html feature="Usage analytics" %}

The Usage analytics collection is a special collection that contains view-only...
```

### docs/util/README.md

Doc tools

### docs/util/generate-shortcuts-docs.ts

```
import fs from 'fs';
import path from 'path';
import prettier from 'prettier';

import { shortcuts } from '../../../frontend/src/metabase/palette/shortcuts';

interface Shortcut {
  name: string;
  ...
```

### docs/util/resources/introduction.md

```
---
title: Keyboard shortcuts
summary: A handy list of keyboard shortcuts to help you zip around Metabase.
---

# Keyboard shortcuts
```

A handy list of keyboard shortcuts to help you zip around...

### e2e/eslintrc

```
{
  "rules": {
    "no-unscooped-text-selectors": 2,
    "import/no-commonjs": 0,
    "no-color-literals": 0,
    "no-console": 0,
    "@typescript-eslint/no-namespace": "off",
    ...
```

### e2e/embedding-sdk-host-apps/eslintrc



## Analyst Base

```
{
  "rules": {
    "import/no-default-export": "off",
    "import/no-unresolved": "off"
  }
}
```

**e2e/embedding-sdk-host-apps/angular-20-host-app/.gitignore**

dependencies

**e2e/embedding-sdk-host-apps/angular-20-host-app/README.md**

Angular@20 Host App

**e2e/embedding-sdk-host-apps/angular-20-host-app/src/app/app.config.ts**

```
import {
  type ApplicationConfig,
  provideZonelessChangeDetection,
} from "@angular/core";
import { provideRouter } from "@angular/router";
import { defineMetabaseAuthConfig } from...
```

**e2e/embedding-sdk-host-apps/angular-20-host-app/src/app/app.routes.ts**

```
import type { Routes } from "@angular/router";
```

```
import { InteractiveDashboardPageComponent } from "../interactive-dashboard-page.component";
import { InteractiveQuestionPageComponent } from...
```

**e2e/embedding-sdk-host-apps/angular-20-host-app/src/app/reactmount.directive.ts**

```
import {
  type AfterViewInit,
  Directive,
  ElementRef,
  Inject,
  Input,
  NgZone,
  type OnChanges,
  type OnDestroy,
  type SimpleChanges,
} from "@angular/core";
import type { ReactElement }...
```

**e2e/embedding-sdk-host-apps/angular-20-host-app/src/main.ts**

```
import { bootstrapApplication } from "@angular/platform-browser";
```

## Analyst Base

```
import { AppComponent } from "../app/app.component";  
import { appConfig } from...
```

### **e2e/embedding-sdk-host-apps/next-15-app-router-host-app/.gitignore**

dependencies

### **e2e/embedding-sdk-host-apps/next-15-pages-router-host-app/.gitignore**

dependencies

### **e2e/embedding-sdk-host-apps/vite-6-host-app/.gitignore**

dependencies

### **e2e/embedding-sdk-host-apps/vite-6-host-app/vite.config.js**

```
import react from "@vitejs/plugin-react";  
import { defineConfig, loadEnv } from "vite";
```

```
export default defineConfig(({ mode }) => {  
  const env = loadEnv(mode, process.cwd(), "");  
  const clientPort...
```

### **e2e/runner/constants/backend-port.js**

```
export const BACKEND_PORT = process.env.BACKEND_PORT ?? 4000;
```

### **e2e/runner/constants/exit-code.js**

```
export const SUCCESS_EXIT_CODE = 0;  
export const FAILURE_EXIT_CODE = 1;
```

### **e2e/runner/constants/paths.js**

```
import path from "path";
```

```
export const ROOT_FOLDER_PATH = path.resolve(__dirname, "../..");  
export const E2E_FOLDER_PATH = path.join(ROOT_FOLDER_PATH, "e2e");
```

### **e2e/runner/cypress-runner-backend.js**

```
#!/usr/bin/env node
```

### **e2e/runner/cypress-runner-run-tests.js**

```
const cypress = require("cypress");
```

```
const { BACKEND_PORT } = require("../constants/backend-port");  
const { FAILURE_EXIT_CODE } = require("../constants/exit-code");  
const { parseArguments, args } =...
```

### **e2e/runner/cypress-runner-utils.js**

```
const { execSync, spawn } = require("child_process");
```

## Analyst Base

```
const arg = require("arg");
const chalk = require("chalk");
const cypress = require("cypress");
```

```
function printBold(message) {
```

```
...
```

### **e2e/runner/embedding-sdk/host-apps/constants/host-app-folder-path.js**

```
import path from "path";
```

```
import { E2E_FOLDER_PATH } from "../../constants/paths";
```

```
export const HOST_APP_FOLDER_PATH = path.join(
  E2E_FOLDER_PATH,
  "embedding-sdk-host-apps",
);
```

### **e2e/runner/embedding-sdk/host-apps/constants/host-app-setup-configs.js**

```
import { BACKEND_PORT } from "../../constants/backend-port.js";
```

```
const BASE_ENV = {
  WATCH: process.env.HOST_APP_ENVIRONMENT === "development" ? "true" : "false",
  MB_PORT: BACKEND_PORT,
  ...
}
```

### **e2e/runner/embedding-sdk/host-apps/helpers/start-app.ts**

```
import { shell } from "../../cypress-runner-utils";
import { waitForHealth } from "../../shared/helpers/wait-for-health";
```

```
export async function startApp({
  cwd,
  env,
  appRunCommand,
  ...
}) {
```

### **e2e/runner/embedding-sdk/host-apps/start-ci.ts**

```
import { startHostAppContainers } from "./start-host-app-containers";
import type { HostAppTestSuiteName } from "./types";
```

```
const testSuite = process.argv?.[2]?.trim() as HostAppTestSuiteName;
```

```
if...
```

### **e2e/runner/embedding-sdk/host-apps/start-host-app-containers.ts**

## Analyst Base

```
import { FAILURE_EXIT_CODE } from "../../constants/exit-code";
import { printBold } from "../../cypress-runner-utils";
import { setupAppCleanup } from "../shared/helpers/setup-app-cleanup";
```

```
import {...
```

### **e2e/runner/embedding-sdk/host-apps/types.ts**

```
export type HostAppTestSuiteName =
  | "vite-6-host-app-e2e"
  | "next-15-app-router-host-app-e2e"
  | "next-15-pages-router-host-app-e2e"
  | "angular-20-host-app-e2e";
```

### **e2e/runner/embedding-sdk/sample-apps/constants/e2e-tmp-folder-path.js**

```
import path from "path";
```

```
import { E2E_FOLDER_PATH } from "../../../constants/paths";
```

```
export const E2E_TMP_FOLDER_PATH = path.join(E2E_FOLDER_PATH, "tmp");
```

### **e2e/runner/embedding-sdk/sample-apps/constants/sample-app-setup-configs.js**

```
const BRANCH_NAME = "main"; // Affects the `local` testing only. On CI is passed as an ENV variable.
```

```
const BASE_ENV = {
```

```
  WATCH:
```

```
    process.env.SAMPLE_APP_ENVIRONMENT === "development" ? "true" :...
```

### **e2e/runner/embedding-sdk/sample-apps/helpers/copy-shoppy-metabase-app-d b-dump.ts**

```
import fs from "fs";
```

```
import * as path from "path";
```

```
import { E2E_TMP_FOLDER_PATH } from "../../../constants/e2e-tmp-folder-path";
```

```
const LOCAL_DIST_PATH = "./local-dist";
```

```
// See...
```

### **e2e/runner/embedding-sdk/sample-apps/helpers/fetch-app.ts**

```
import fs from "fs";
```

```
import { shell } from "../../../cypress-runner-utils";
```

```
import { E2E_TMP_FOLDER_PATH } from "../../../constants/e2e-tmp-folder-path";
```

## Analyst Base

```
export function fetchApp({
  appName,
  ...
```

### **e2e/runner/embedding-sdk/sample-apps/helpers/prepare-app.ts**

```
import fs from "fs";
import * as path from "path";

import { ROOT_FOLDER_PATH } from "../../constants/paths";

const METABASE_JAR_DIST_PATH = path.join(ROOT_FOLDER_PATH, "target/uberjar");

const...
```

### **e2e/runner/embedding-sdk/sample-apps/helpers/start-containers.ts**

```
import { shell } from "../../cypress-runner-utils";
import { waitForHealth } from "../../shared/helpers/wait-for-health";

export async function startContainers({
  cwd,
  env,
  dockerUpCommand,
  ...
```

### **e2e/runner/embedding-sdk/sample-apps/start-ci.ts**

```
import { startSampleAppContainers } from "./start-sample-app-containers";
import type { SampleAppTestSuiteName } from "./types";

const testSuite = process.argv?.[2]?.trim() as...
```

### **e2e/runner/embedding-sdk/sample-apps/start-sample-app-containers.ts**

```
import { FAILURE_EXIT_CODE } from "../../constants/exit-code";
import { printBold } from "../../cypress-runner-utils";
import { setupAppCleanup } from "../../shared/helpers/setup-app-cleanup";

import {...
```

### **e2e/runner/embedding-sdk/sample-apps/types.ts**

```
export type SampleAppTestSuiteName =
  | "metabase-nodejs-react-sdk-embedding-sample-e2e"
  | "metabase-nextjs-sdk-embedding-sample-e2e"
  | "shoppy-e2e";

export type EmbeddingSdkVersion = "local" |...
```

## Analyst Base

### **e2e/runner/embedding-sdk/shared/helpers/setup-app-cleanup.ts**

```
import fs from "fs";

import { shell } from "../../cypress-runner-utils";

export function setupAppCleanup({
  rootPath,
  env,
  appDownCommand,
  cleanupAppDir,
}): {
  rootPath: string;
  env:...
```

### **e2e/runner/embedding-sdk/shared/helpers/wait-for-health.ts**

```
import { delay } from "../../cypress-runner-utils";

const HEALTH_CHECK_ATTEMPTS_COUNT = 60 * 5;
const HEALTH_CHECK_WAIT_TIME_MS = 2000;

export async function waitForHealth(url: string,...
```

### **e2e/runner/run\_cypress\_ci.js**

```
const { FAILURE_EXIT_CODE } = require("./constants/exit-code");
const CypressBackend = require("./cypress-runner-backend");
const runCypress = require("./cypress-runner-run-tests");
const { printBold...
```

### **e2e/runner/run\_cypress\_local.ts**

```
import { FAILURE_EXIT_CODE, SUCCESS_EXIT_CODE } from "./constants/exit-code";
import CypressBackend from "./cypress-runner-backend";
import runCypress from "./cypress-runner-run-tests";
import {
  ...
```

### **e2e/snapshot-creators/default.cy.snap.js**

```
import _ from "underscore";

import { loginCache } from "e2e/support/commands/user/authentication";
import {
  METABASE_SECRET_KEY,
  SAMPLE_DB_ID,
  SAMPLE_DB_TABLES,
```

## Analyst Base

```
USERS,  
USER_GROUPS,  
} from...
```

### **e2e/snapshot-creators/qa-db.cy.snap.js**

```
import {  
  addMongoDatabase,  
  addMySQLDatabase,  
  addPostgresDatabase,  
  restore,  
  setupWritableDB,  
  snapshot,  
} from "e2e/support/helpers";
```

```
describe("qa databases snapshots", { tags: "@external"...
```

### **e2e/support/ci\_tasks.ts**

```
import fetch from "node-fetch"; // must be node-fetch v2 because it's non-esm
```

```
const {  
  GITHUB_RUN_ID,  
  GITHUB_TOKEN,  
  HASH,  
  PR_NUMBER,  
  GITHUB_REPOSITORY,  
  JOB_NAME,  
} = process.env;
```

```
const...
```

### **e2e/support/collectFailedTests.js**

```
const fs = require("fs");  
const path = require("path");
```

```
/**
```

```
 * Collects failed test specs from the most recent Cypress test run  
 *
```

```
 * After each run, a file will store failed spec paths as a...
```

### **e2e/support/commands/component.ts**

We need to use the `mount` function from `@cypress/react` to allow running SDK component tests on multiple React versions

### **e2e/support/commands/database/addSQLiteDatabase.js**

## Analyst Base

```
Cypress.Commands.add(
  "addSQLiteDatabase",
  ({ name = "sqlite", auto_run_queries = true, is_full_sync = true } = {}) => {
    cy.log(`Add SQLite database DB called "${name}"`);
    ...
  }
);
```

### **e2e/support/commands/downloads/downloadUtils-untyped.js**

```
const fs = require("fs");
const path = require("path");

const removeDirectory = (path) => {
  try {
    if (fs.existsSync(path)) {
      fs.rmdirSync(path, { maxRetries: 10, recursive: true });
    }
    ...
  }
};
```

### **e2e/support/commands/downloads/downloadUtils.ts**

```
export * from "../downloadUtils-untyped";

/**
 * NOTE: Cypress commands are deprecated in favor of custom functions, as they
 * are easier to type. These commands rely on cypress tasks and were not...
```

### **e2e/support/commands/overwrites/log.js**

```
Cypress.Commands.overwrite("log", (originalFn, text) => {
  const logConfig = {
    displayName: `${window.logCalls}. ${text}`,
    name: "log",
    message: "",
  };
  ...
});
```

### **e2e/support/commands/permissions/sandboxTable.js**

```
import { SAMPLE_DB_TABLES, USER_GROUPS } from "e2e/support/cypress_data";

const { STATIC_ORDERS_ID } = SAMPLE_DB_TABLES;

const { COLLECTION_GROUP } = USER_GROUPS;

Cypress.Commands.add(
  ...
);
```

### **e2e/support/commands/permissions/updatePermissions.ts**



## Analyst Base

```
import type {
  CollectionPermissions,
  CollectionPermissionsGraph,
  GroupsPermissions,
  Impersonation,
  PermissionsGraph,
} from "metabase-types/api";
```

```
declare global {
  namespace Cypress {
    ...
```

### **e2e/support/commands/ui/button.ts**

```
declare global {
  namespace Cypress {
    interface Chainable {
      /**
       * Get a button either unscoped, or chained to a previously yielded subject.
       * Uses `findByRole` under the...
```

### **e2e/support/commands/ui/icon.ts**

```
import type { IconName } from "metabase/ui";

declare global {
  namespace Cypress {
    interface Chainable {
      /**
       * Get an icon either unscoped, or chained to a previously yielded...
```

### **e2e/support/commands/ui/paste.ts**

```
Cypress.Commands.add(
  "paste",
  { prevSubject: "element" },
  (subject, text: string) => {
    cy.wrap(subject).then(($element) => {
      const clipboardData = new DataTransfer();
      ...
```

### **e2e/support/commands/user/authentication.ts**

```
import { USERS } from "e2e/support/cypress_data";
declare global {
  namespace Cypress {
    interface Chainable {
      signIn: (user?: keyof typeof USERS, options?: LoginOptions) => void;
```

...

### **e2e/support/commands/user/createUser.js**

```
import { USERS } from "e2e/support/cypress_data";
```

```
Cypress.Commands.add("createUserFromRawData", (user) => {  
  return cy.request("POST", "/api/user", user).then(({ body: user }) => {  
    // Dismiss...
```

### **e2e/support/commands/visibility/findByTextEnsureVisible.ts**

```
declare global {  
  namespace Cypress {  
    interface Chainable {  
      findByTextEnsureVisible(  
        text: string,  
      ): Cypress.Chainable<jQuery<HTMLElement>>;  
    }  
  }  
}
```

...

### **e2e/support/commands/visibility/isRenderedWithinViewport.js**

```
Cypress.Commands.add(  
  "isRenderedWithinViewport",  
  {  
    prevSubject: true,  
  },  
  (subject) => {  
    const viewportTop = 0;  
    const viewportBottom = Cypress.$(cy.state("window")).height();  
    ...
```

...

### **e2e/support/commands/visibility/isVisibleInPopover.js**

```
import { POPOVER_ELEMENT } from "e2e/support/helpers";
```

```
Cypress.Commands.add(  
  "isVisibleInPopover",  
  {  
    prevSubject: true,  
  },  
  (subject) => {  
    cy.wrap(subject)  
    ...
```

### **e2e/support/commands.js**

this is the only place we allow direct helper import

## Analyst Base

eslint-disable-next-line no-direct-helper-import

### **e2e/support/component-cypress.js**

```
import "./cypress";
import { renameConflictingCljsGlobals } from "metabase/embedding-sdk/test/rename-conflicting-cljs-globals";

beforeEach(() => {
  renameConflictingCljsGlobals();
});
```

### **e2e/support/component-webpack.config.js**

```
const fs = require("fs");
const path = require("path");

const webpack = require("webpack");

const mainConfig = require("../rspack.main.config");

const SDK_PACKAGE_NAME = ...
```

### **e2e/support/config.js**

```
import fs from "node:fs";
import path from "node:path";

import installLogsPrinter from "cypress-terminal-report/src/installLogsPrinter";

import * as ciTasks from "./ci_tasks";
import {...
```

### **e2e/support/cypress-embedding-sdk-component-test.config.js**

```
const { embeddingSdkComponentTestConfig } = require("./config");

module.exports = {
  component: embeddingSdkComponentTestConfig,
};
```

### **e2e/support/cypress-snapshots.config.js**

```
const { defineConfig } = require("cypress");

const { snapshotsConfig } = require("./config");

module.exports = defineConfig({ e2e: snapshotsConfig });
```

### **e2e/support/cypress-stress-test.config.js**

```
const { defineConfig } = require("cypress");
```

## Analyst Base

```
const { stressTestConfig } = require("./config");
```

```
module.exports = defineConfig({ e2e: stressTestConfig });
```

### **e2e/support/cypress.config.js**

```
const { defineConfig } = require("cypress");
```

```
const { mainConfig } = require("./config");
```

```
module.exports = defineConfig({ e2e: mainConfig });
```

### **e2e/support/cypress.js**

```
import registerCypressGrep from "@cypress/grep"; // eslint-disable-line import/order
registerCypressGrep();
```

```
import "@cypress/skip-test/support";
```

```
import...
```

### **e2e/support/cypress\_data.js**

```
import { ORDERS_PRODUCTS_ACCESS } from "../test_roles";
```

```
/**
 * We are keeping the references to most commonly used ids and objects in this file.
 *
 * Please note that these ids are hard coded and...
```

### **e2e/support/cypress\_sample\_database.js**

```
*
```

### **e2e/support/cypress\_sample\_instance\_data.js**

```
*
```

### **e2e/support/db\_tasks.js**

```
import Knex from "knex";
```

```
import { QA_DB_CONFIG, WRITABLE_DB_CONFIG } from "../cypress_data";
```

```
import { Roles } from "../test_roles";
```

```
import * as testTables from "../test_tables";
```

```
const dbClients =...
```

### **e2e/support/external/e2e-jwt-sign.js**

```
#!/usr/bin/env node
```

### **e2e/support/helpers/api/addOrUpdateDashboardCard.ts**

## Analyst Base

```
import type {
  CardId,
  Dashboard,
  DashboardCard,
  DashboardId,
} from "metabase-types/api";

import { DEFAULT_CARD } from "../updateDashboardCards";

export function addOrUpdateDashboardCard({
  ...
```

### **e2e/support/helpers/api/addQuestionToDashboard.ts**

```
import type { CardId, DashboardCard, DashboardId } from "metabase-types/api";

export const addQuestionToDashboard = ({
  dashboardId,
  cardId,
}): {
  dashboardId: DashboardId;
  cardId: CardId;
} {...
```

### **e2e/support/helpers/api/archiveCollection.ts**

```
import type { CollectionId } from "metabase-types/api";

export const archiveCollection = (id: CollectionId) => {
  cy.log(` Archiving a collection with id: ${id}`);

  return cy.request("PUT", ...
```

### **e2e/support/helpers/api/archiveDashboard.ts**

```
import type { Dashboard, DashboardId } from "metabase-types/api";

export const archiveDashboard = (
  id: DashboardId,
): Cypress.Chainable<Cypress.Response<Dashboard>> => {
  cy.log(` Archiving a...
```

### **e2e/support/helpers/api/archiveQuestion.ts**

```
import type { Card } from "metabase-types/api";

export const archiveQuestion = (
  id: Card["id"],
```

## Analyst Base

```
): Cypress.Chainable<Cypress.Response<Card>> => {  
  cy.log(` Archiving a question with id:...
```

### **e2e/support/helpers/api/createApiKey.ts**

```
export const createApiKey = (name: string, group_id: number) => {  
  return cy.request("POST", "/api/api-key", {  
    name,  
    group_id,  
  });  
};
```

### **e2e/support/helpers/api/createCollection.ts**

```
import type { Collection, RegularCollectionId } from "metabase-types/api";  
  
export const createCollection = ({  
  name,  
  description = null,  
  parent_id = null,  
  authority_level = null,  
  alias,  
}: ...
```

### **e2e/support/helpers/api/createDashboard.ts**

```
import type {  
  CreateDashboardRequest,  
  Dashboard,  
  DashboardCard,  
} from "metabase-types/api";  
  
export interface DashboardDetails extends Omit<CreateDashboardRequest, "name"> {  
  name?: string;  
  ...
```

### **e2e/support/helpers/api/createDashboardWithQuestions.ts**

```
import type { Card, Dashboard, DashboardCard } from "metabase-types/api";  
  
import { cypressWaitAll } from "../e2e-misc-helpers";  
  
import { type DashboardDetails, createDashboard } from...
```

### **e2e/support/helpers/api/createDashboardWithTabs.ts**

```
import type { Dashboard } from "metabase-types/api";  
  
import { type DashboardDetails, createDashboard } from "../createDashboard";
```

## Analyst Base

```
export function createDashboardWithTabs({
  dashcards = [],
  tabs,
  ...
```

### **e2e/support/helpers/api/createModerationReview.ts**

```
import type { ModerationReview } from "metabase-types/api";
```

```
export const createModerationReview = ({
  status,
  moderated_item_type,
  moderated_item_id,
}): {
  status: "verified" | null;
  ...
```

### **e2e/support/helpers/api/createNativeQuestion.ts**

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import type { Card, DatasetQuery, NativeQuery } from "metabase-types/api";
```

```
import {
  type Options,
  type QuestionDetails,
  logAction,
  ...
```

### **e2e/support/helpers/api/createNativeQuestionAndDashboard.ts**

```
import { createNativeQuestion } from "e2e/support/helpers";
```

```
import type {
  CardId,
  Dashboard,
  DashboardCard,
  DashboardId,
} from "metabase-types/api";
```

```
import { type DashboardDetails, ...
```

### **e2e/support/helpers/api/createNotification.ts**

```
import type {
  CardId,
  CreateAlertNotificationRequest,
  Notification,
  NotificationCardSendCondition,
```

## Analyst Base

```
NotificationHandler,  
  UserId,  
} from "metabase-types/api";
```

```
export const...
```

### **e2e/support/helpers/api/createPulse.ts**

```
import type {  
  CreateSubscriptionRequest,  
  DashboardSubscription,  
} from "metabase-types/api";
```

```
export const createPulse = ({  
  name = "Pulse",  
  cards = [],  
  channels = [],  
  dashboard_id,  
}:...
```

### **e2e/support/helpers/api/createQuestion.ts**

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import type {  
  Card,  
  DatasetQuery,  
  NativeQuery,  
  StructuredQuery,  
} from "metabase-types/api";
```

```
import { visitMetric, visitModel,...
```

### **e2e/support/helpers/api/createQuestionAndAddToDashboard.ts**

```
import type { Card, DashboardCard, DashboardId } from "metabase-types/api";
```

```
import {  
  type NativeQuestionDetails,  
  createNativeQuestion,  
} from "../createNativeQuestion";  
import {  
  type...
```

### **e2e/support/helpers/api/createQuestionAndDashboard.ts**

```
import type { CardId, Dashboard, DashboardCard } from "metabase-types/api";
```

```
import { type DashboardDetails, createDashboard } from "../createDashboard";
```



## Analyst Base

```
import {  
  type StructuredQuestionDetails,  
  ...  
}
```

### **e2e/support/helpers/api/createSnippet.ts**

```
export function createSnippet({  
  name,  
  content,  
}): {  
  name: string;  
  content: string;  
}: Cypress.Chainable<Cypress.Response<NativeQuerySnippet>> {  
  cy.request("POST",...
```

### **e2e/support/helpers/api/createTimeline.ts**

```
import type { CreateTimelineRequest, Timeline } from "metabase-types/api";
```

```
export const createTimeline = ({  
  name = "Timeline",  
  icon = "star",  
  default: isDefault = false,  
  archived = false,  
  ...  
}) => {
```

### **e2e/support/helpers/api/createTimelineEvent.ts**

```
import type {  
  CreateTimelineEventRequest,  
  TimelineEvent,  
} from "metabase-types/api";
```

```
export const createTimelineEvent = ({  
  name = "Event",  
  icon = "star",  
  timestamp = ...  
}) => {
```

### **e2e/support/helpers/api/createTimelineWithEvents.ts**

```
import type {  
  CreateTimelineEventRequest,  
  CreateTimelineRequest,  
  Timeline,  
  TimelineEvent,  
} from "metabase-types/api";
```

```
import { cypressWaitAll } from "../e2e-misc-helpers";
```

## Analyst Base

```
import {...
```

### **e2e/support/helpers/api/editDashboardCard.ts**

```
import _ from "underscore";
```

```
import type { Dashboard, DashboardCard } from "metabase-types/api";
```

```
export const editDashboardCard = (  
  dashboardCard: DashboardCard,  
  updatedProperties:...
```

### **e2e/support/helpers/api/getCurrentUser.ts**

```
import type { User } from "metabase-types/api";
```

```
export const getCurrentUser = (): Cypress.Chainable<Cypress.Response<User>> => {  
  return cy.request<User>("GET", "/api/user/current");  
};
```

### **e2e/support/helpers/api/index.ts**

```
export { addOrUpdateDashboardCard } from "./addOrUpdateDashboardCard";  
export { addQuestionToDashboard } from "./addQuestionToDashboard";  
export { archiveCollection } from...
```

### **e2e/support/helpers/api/remapDisplayValueToFK.ts**

```
*
```

### **e2e/support/helpers/api/updateDashboardCards.ts**

```
import type { Dashboard, DashboardCard, DashboardId } from "metabase-types/api";
```

```
export const DEFAULT_CARD = {  
  id: -1,  
  row: 0,  
  col: 0,  
  size_x: 11,  
  size_y: 8,  
  visualization_settings: {},  
  ...
```

### **e2e/support/helpers/api/updateSetting.ts**

```
import type { Settings } from "metabase-types/api";
```

```
export const updateSetting = <  
  TKey extends keyof Settings,  
  TValue extends Settings[TKey],
```

## Analyst Base

```
>(
  setting: TKey,
  value: TValue,
):...
```

### **e2e/support/helpers/e2e-action-helpers.js**

```
import { capitalize } from "inflection";
```

```
import {
  commandPalette,
  commandPaletteButton,
  commandPaletteInput,
} from "../e2e-command-palette-helpers";
import { NativeEditor } from...
```

### **e2e/support/helpers/e2e-ad-hoc-question-helpers.js**

```
import { SAMPLE_DB_ID, SAMPLE_DB_TABLES } from "e2e/support/cypress_data";
```

```
import { runNativeQuery } from "../e2e-misc-helpers";
import { NativeEditor } from "../e2e-native-editor-helpers";
```

```
const {
  ...
```

### **e2e/support/helpers/e2e-api-key-helpers.ts**

```
eslint-disable-next-line no-direct-helper-import
```

### **e2e/support/helpers/e2e-bi-basics-helpers.js**

```
import { queryBuilderMain, tableHeaderColumn } from "e2e/support/helpers";
```

```
/**
 * Initiate Summarize action
 *
 * @param {Object} options
 * @param {"notebook"|undefined} options.mode
 */
export...
```

### **e2e/support/helpers/e2e-boolean-helpers.js**

```
import { createNativeQuestion } from "../api";
```

```
// until we have a test dataset that includes boolean data, we can use this questions to test booleans
export function setupBooleanQuery(questionName =...
```

### **e2e/support/helpers/e2e-browser-helpers.ts**

\*

### **e2e/support/helpers/e2e-cloud-helpers.js**

```
import { updateSetting } from "../api";

export const setupMetabaseCloud = () => {
  updateSetting("site-url", "https://CYPRESSTESTENVIRONMENT.metabaseapp.com");
};
```

### **e2e/support/helpers/e2e-collection-helpers.ts**

```
import {
  entityPickerModal,
  entityPickerModalLevel,
  entityPickerModalTab,
  getFullName,
  navigationSidebar,
  popover,
} from "e2e/support/helpers";
import type { CollectionId } from...
```

### **e2e/support/helpers/e2e-command-palette-helpers.js**

```
export const commandPalette = () => cy.findByTestId("command-palette");
export const shortcutModal = () =>
  cy.findByRole("dialog", { name: "Shortcuts" });

export const openCommandPalette = () =>...
```

### **e2e/support/helpers/e2e-custom-column-helpers.ts**

```
import { popover } from "e2e/support/helpers/e2e-ui-elements-helpers";

export function expressionEditorWidget() {
  return cy.findByTestId("expression-editor");
}
```

export function...

### **e2e/support/helpers/e2e-dashboard-helpers.ts**

```
import type {
  DashCardId,
  DashboardCard,
  DashboardId,
  DashboardTab,
  VirtualDashboardCard,
  WritebackActionId,
```

## Analyst Base

```
} from "metabase-types/api";
```

```
import { visitDashboard } from...
```

### **e2e/support/helpers/e2e-dashboard-visualizer-helpers.ts**

```
import type { VisualizationDisplay } from "metabase-types/api";
```

```
import {  
  getDashboardCard,  
  showDashboardCardActions,  
} from "../e2e-dashboard-helpers";  
import { modal, sidebar } from...
```

### **e2e/support/helpers/e2e-database-metadata-helpers.ts**

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import type {  
  Database,  
  Databaseld,  
  FieldId,  
  TableId,  
} from "metabase-types/api";
```

```
type FieldsMap = Record<string, FieldId>;  
type...
```

### **e2e/support/helpers/e2e-datamodel-helpers.ts**

```
import type {  
  Databaseld,  
  FieldId,  
  Schemald,  
  TableId,  
} from "metabase-types/api";
```

```
export const DataModel = {  
  visit,  
  get: getDataModel,  
  TablePicker: {  
    get: getTablePicker,  
    ...  
  }  
};
```

### **e2e/support/helpers/e2e-dimension-list-helpers.js**

```
export function getDimensions(isSelected) {  
  if (typeof isSelected === "undefined") {  
    return cy.findAllByTestId("dimension-list-item");  
  }  
}
```

## Analyst Base

```
}
```

```
return cy.get(  
  ...
```

### **e2e/support/helpers/e2e-downloads-helpers.ts**

```
import xlsx, { type Sheet } from "xlsx";
```

```
import {  
  getDashboardCard,  
  getDashboardCardMenu,  
  getEmbeddedDashboardCardMenu,  
} from "../e2e-dashboard-helpers";  
import { popover } from...
```

### **e2e/support/helpers/e2e-dragndrop-helpers.js**

Rely on native drag events, rather than on the coordinates  
We have 3 "drag-handles" in this test. Their indexes are 0-based.

### **e2e/support/helpers/e2e-element-visibility-helpers.ts**

```
*
```

### **e2e/support/helpers/e2e-email-helpers.js**

```
import { openSharingMenu, sidebar } from "e2e/support/helpers";
```

```
import { WEBMAIL_CONFIG } from "../cypress_data";
```

```
const INBOX_TIMEOUT = 5000;  
const INBOX_INTERVAL = 100;
```

```
const { WEB_PORT,...
```

### **e2e/support/helpers/e2e-embedding-helpers.js**

```
import { METABASE_SECRET_KEY } from "e2e/support/cypress_data";  
import { modal, popover } from "e2e/support/helpers/e2e-ui-elements-helpers";
```

```
import { openSharingMenu } from...
```

### **e2e/support/helpers/e2e-embedding-iframe-sdk-helpers.ts**

```
import { match } from "ts-pattern";
```

```
import type { MetabaseTheme } from "metabase/embedding-sdk/theme/MetabaseTheme";  
import type { CreateApiKeyResponse } from "metabase-types/api";
```

```
import {...
```

## Analyst Base

### **e2e/support/helpers/e2e-embedding-sdk-assertion-helpers.ts**

```
import { popover, tableInteractive } from "../e2e-ui-elements-helpers";

export function assertSdkNotebookEditorUsable(root: Cypress.Chainable = cy) {
  cy.findByText("Orders").should("be.visible");
}
```

...

### **e2e/support/helpers/e2e-embedding-sdk-helpers.ts**

```
export const EMBEDDING_SDK_STORY_HOST = "http://localhost:6006/iframe.html";

export const getSdkRoot = () => cy.get("[data-cy-root]").should("be.visible");

export const getStorybookSdkRoot = () =>
```

...

### **e2e/support/helpers/e2e-enterprise-helpers.js**

```
export const mockSessionPropertiesTokenFeatures = (features) => {
  cy.intercept({ method: "GET", url: "/api/session/properties" }, (request) => {
    request.on("response", (response) => {
      if...
```

### **e2e/support/helpers/e2e-filter-helpers.js**

```
import {
  entityPickerModal,
  modal,
  popover,
} from "e2e/support/helpers/e2e-ui-elements-helpers";
```

```
export function setDropdownFilterType() {
  cy.findByText("Dropdown list").click();
}
```

export...

### **e2e/support/helpers/e2e-jwt-helpers.ts**

```
import { AUTH_PROVIDER_URL } from "../embedding-sdk-helpers/constants";
```

```
export const JWT_SHARED_SECRET = "0".repeat(64);
```

```
export const enableJwtAuth = () => {
  cy.request("PUT", "/api/setting", {
    ...
```

### **e2e/support/helpers/e2e-jwt-tasks.ts**

```
import jwt from "jsonwebtoken";

export function signJwt({
  payload,
  secret,
}): {
  payload: Record<string, string | number>;
  secret: string;
}): string {
  return jwt.sign(payload, secret);
}
```

### **e2e/support/helpers/e2e-ldap-helpers.js**

\*

### **e2e/support/helpers/e2e-metabot-helpers.ts**

```
import type { StaticResponse } from "cypress/types/net-stubbing";

import {
  commandPaletteAction,
  openCommandPalette,
} from "../e2e-command-palette-helpers";
import { appBar } from...
```

### **e2e/support/helpers/e2e-misc-helpers.js**

```
import { pickEntity } from "../e2e-collection-helpers";
import { modal } from "../e2e-ui-elements-helpers";
```

// Find a text field by label text, type it in, then blur the field.

// Commonly used in our...

### **e2e/support/helpers/e2e-mock-app-settings-helpers.js**

```
function mockProperty(propertyOrObject, value, url) {
  cy.intercept("GET", url, (req) => {
    req.reply((res) => {
      if (typeof propertyOrObject === "object") {
        Object.assign(res.body, ...
```

### **e2e/support/helpers/e2e-model-index-helper.js**

```
export function createModelIndex({ modelId, pkName, valueName }) {
  // since field ids are non-deterministic, we need to get them from the api
  cy.request("GET", ...
```



## Analyst Base

### **e2e/support/helpers/e2e-models-metadata-helpers.js**

```
import {
  interceptIfNotPreviouslyDefined,
  popover,
  tableInteractive,
} from "e2e/support/helpers";

export function datasetEditBar() {
  return cy.findByTestId("dataset-edit-bar");
}
```

export...

### **e2e/support/helpers/e2e-native-editor-helpers.ts**

```
import { popover } from "../e2e-ui-elements-helpers";

function nativeEditor() {
  cy.findAllByTestId("loading-indicator").should("not.exist");
  return cy.get("[data-testid=native-query-editor]");
}
```

### **e2e/support/helpers/e2e-notebook-helpers.ts**

```
import type { CyHttpMessages } from "cypress/types/net-stubbing";

import {
  entityPickerModal,
  entityPickerModalTab,
  popover,
  shouldDisplayTabs,
} from "e2e/support/helpers";
```

### **e2e/support/helpers/e2e-notification-helpers.ts**

```
import type { NotificationChannel } from "metabase-types/api/notification-channels";

export const getAlertChannel = (name: string) => {
  cy.findByRole("listitem", {
    name,
  });
};

export const ...
```

### **e2e/support/helpers/e2e-permissions-helpers.js**

```
import _ from "underscore";
```

## Analyst Base

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { modal, popover } from "e2e/support/helpers";
```

```
export function selectSidebarItem(item) {
  ...
}
```

### **e2e/support/helpers/e2e-qa-databases-helpers.js**

```
import {
  QA_DB_CONFIG,
  QA_DB_CREDENTIALS,
  QA_MONGO_PORT,
  QA_MYSQL_PORT,
  QA_POSTGRES_PORT,
  WRITABLE_DB_CONFIG,
  WRITABLE_DB_ID,
} from "e2e/support/cypress_data";
```

```
import { createQuestion...
```

### **e2e/support/helpers/e2e-relative-date-picker-helpers.js**

Units have different labels depending on the value and time direction (past/future)

Example: "1 day", "2 days" for interval value "1 day ago" and "2 days from now" for offset value

### **e2e/support/helpers/e2e-request-helpers.js**

\*

### **e2e/support/helpers/e2e-search-helpers.js**

```
export function getSearchBar() {
  return cy.findByPlaceholderText("Search...");
}
```

```
/**
 * Checks the search results against expectedSearchValues, including descriptions,
 * collection names, and...
```

### **e2e/support/helpers/e2e-setup-helpers.js**

```
import { resetWritableDb } from "../e2e-qa-databases-helpers";
```

```
export function snapshot(name) {
  cy.request("POST", `/api/testing/snapshot/${name}`);
}
```

```
/**
 *
```

## Analyst Base

```
* @param { |  
* "blank" |  
* "setup"...
```

### **e2e/support/helpers/e2e-sharing-helpers.ts**

```
import { modal, popover } from "e2e/support/helpers/e2e-ui-elements-helpers";
```

```
export const sharingMenuButton = () => cy.findByTestId("sharing-menu-button");  
export const sharingMenu = () =>...
```

### **e2e/support/helpers/e2e-slack-helpers.js**

Since there is no testing API token for Slack, it's easier to mock its configuration

### **e2e/support/helpers/e2e-snowplow-helpers.js**

```
import { updateSetting } from "e2e/support/helpers";
```

```
const { IS_ENTERPRISE } = Cypress.env();  
const HAS_SNOWPLOW = Cypress.env("HAS_SNOWPLOW_MICRO");  
const SNOWPLOW_URL =...
```

### **e2e/support/helpers/e2e-table-metadata-helpers.js**

```
*
```

### **e2e/support/helpers/e2e-token-helpers.ts**

```
import { match } from "ts-pattern";
```

```
/**  
* Just because an instance is using an Enterprise artifact (jar or Docker image),  
* doesn't mean that the token is active or that it has all feature flags...
```

### **e2e/support/helpers/e2e-ui-elements-helpers.js**

Functions that get key elements in the app

### **e2e/support/helpers/e2e-ui-elements-overflow-helpers.js**

```
export const isScrollableHorizontally = (element) => {  
  const { clientHeight, offsetHeight } = element;  
  const style = window.getComputedStyle(element);  
  const borderTopWidth =...
```

### **e2e/support/helpers/e2e-upload-helpers.js**

```
import { WRITABLE_DB_ID } from "../cypress_data";
```

```
import { modal } from "../e2e-ui-elements-helpers";
```

```
export const FIXTURE_PATH = "../e2e/support/assets";
```

## Analyst Base

```
export const VALID_CSV_FILES = [
  {
    ...
```

### **e2e/support/helpers/e2e-users-helpers.ts**

```
interface User {
  email: string;
  first_name?: string | null;
  last_name?: string | null;
}
/**
 * Get user's full name, or an email address if name is not available.
 */
export function...
```

### **e2e/support/helpers/e2e-visual-tests-helpers.js**

```
import { popover } from "e2e/support/helpers/e2e-ui-elements-helpers";
import { color as getColor } from "metabase/lib/colors";
import { Icons } from "metabase/ui";
import { GOAL_LINE_DASH } from...
```

### **e2e/support/helpers/e2e-viz-settings-helpers.js**

```
export function openSeriesSettings(field, isBreakout = false) {
  if (isBreakout) {
    cy.get("[data-testid^=draggable-item]")
      .contains(field)
      .closest("[data-testid^=draggable-item]")
  }
  ...
```

### **e2e/support/helpers/embedding-sdk-component-testing/index.ts**

```
export * from "../component-embedding-sdk-helpers";
export * from "../component-embedding-sdk-question-helpers";
export * from "../component-embedding-sdk-console-helpers";
```

### **e2e/support/helpers/embedding-sdk-helpers/constants.ts**

```
export const METABASE_INSTANCE_URL = "http://localhost:4000";
export const AUTH_PROVIDER_URL = "http://auth-provider/sso";
export const JWT_SHARED_SECRET =
  ...
```

### **e2e/support/helpers/embedding-sdk-testing/embedding-sdk-helpers.ts**

```
import * as jose from "jose";

import { loginCache } from "e2e/support/commands/user/authentication";
```

## Analyst Base

```
import { USERS } from "e2e/support/cypress_data";
import {
  AUTH_PROVIDER_URL,
  ...
```

### **e2e/support/helpers/embedding-sdk-testing/index.ts**

```
export * from "../embedding-sdk-helpers";
```

### **e2e/support/helpers/index.ts**

```
export * from "../api";
export * from "../e2e-action-helpers";
export * from "../e2e-ad-hoc-question-helpers";
export * from "../e2e-api-key-helpers";
export * from "../e2e-bi-basics-helpers";
export *...
```

### **e2e/support/index.ts**

H is for helpers

### **e2e/support/integration/visit-dashboard.cy.spec.js**

```
const { H } = cy;
import { USERS } from "e2e/support/cypress_data";
```

```
import { setup } from "../visit-dashboard";
```

```
describe("visitDashboard e2e helper", () => {
  Object.keys(Cypress._.omit(USERS,...
```

### **e2e/support/integration/visit-dashboard.js**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  addOrUpdateDashboardCard,
  createDashboard,
  createNativeQuestion,
  createNativeQuestionAndDashboard,
  ...
```

### **e2e/support/test-visualizer-data.ts**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import type {
  NativeQuestionDetails,
  StructuredQuestionDetails,
} from "e2e/support/helpers/api";
import...
```

## Analyst Base

### **e2e/support/test\_roles.js**

```
export const ORDERS_PRODUCTS_ACCESS = "orders_products_access";

export const getCreatePostgresRoleIfNotExistSql = (roleName, grantSql) => {
  return `
    DO
    $do$
    BEGIN
      IF NOT EXISTS (...
```

### **e2e/support/test\_tables.js**

define test schema such that they can be used in multiple SQL dialects using  
knex's schema builder <https://knexjs.org/guide/schema-builder.html>  
we cannot use knex to define multi-dialect schemas...

### **e2e/support/test\_tables\_data.js**

```
export const many_data_types_rows = [
  {
    uuid: "a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11",
    integer: 1,
    integerUnsigned: 2,
    tinyint: -128,
    tinyint1: 1,
    smallint: 100,
    ...
```

### **e2e/test/scenarios/actions/actions-in-object-detail-view.cy.spec.js**

```
import dayjs from "dayjs";

const { H } = cy;
import { USER_GROUPS, WRITABLE_DB_ID } from "e2e/support/cypress_data";

const WRITABLE_TEST_TABLE = "scoreboard_actions";
const FIRST_SCORE_ROW_ID = ...
```

### **e2e/test/scenarios/actions/actions-on-dashboards.cy.spec.js**

```
import { assocIn } from "icepick";

const { H } = cy;
import { WRITABLE_DB_ID } from "e2e/support/cypress_data";
import { many_data_types_rows } from "e2e/support/test_tables_data";
import {...
```

### **e2e/test/scenarios/actions/actions-reproductions.cy.spec.js**

## Analyst Base

```
const { H } = cy;
import { SAMPLE_DB_ID, WRITABLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ORDERS_DASHBOARD_ID,
  ...
```

### **e2e/test/scenarios/actions/model-actions.cy.spec.js**

```
import { assocIn } from "icepick";
```

```
const { H } = cy;
import {
  SAMPLE_DB_ID,
  USER_GROUPS,
  WRITABLE_DB_ID,
} from "e2e/support/cypress_data";
import { IMPERSONATED_USER_ID } from...
```

### **e2e/test/scenarios/admin/admin-reproductions.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID, WRITABLE_DB_ID } from "e2e/support/cypress_data";

describe("issue 26470", { tags: "@external" }, () => {
  beforeEach(() => {
    H.restore("postgres_12");
    ...
```

### **e2e/test/scenarios/admin/database-connection-strings.cy.spec.ts**

```
import { QA_MYSQL_PORT, QA_POSTGRES_PORT } from "e2e/support/cypress_data";

import { waitForDbSync } from "../helpers/e2e-database-helpers";

beforeEach(() => {
  cy.H.restore();
  ...
```

### **e2e/test/scenarios/admin/database-routing/database-routing-admin.cy.spec.ts**

```
import _ from "underscore";

import {
  QA_POSTGRES_PORT,
  SAMPLE_DB_ID,
  USER_GROUPS,
} from "e2e/support/cypress_data";
```

## Analyst Base

```
import { interceptPerformanceRoutes } from...
```

### **e2e/test/scenarios/admin/database-routing/database-routing-usage.cy.spec.ts**

```
const { H } = cy;
```

```
import _ from "underscore";
```

```
import { USER_GROUPS } from "e2e/support/cypress_data";
```

```
import { DataPermissionValue } from "metabase/admin/permissions/types";
```

```
import {...
```

### **e2e/test/scenarios/admin/database-routing/helpers/e2e-database-routing-helper.s.ts**

```
const { H } = cy;
```

```
import {
```

```
  QA_DB_CREDENTIALS,
```

```
  QA_POSTGRES_PORT,
```

```
  USER_GROUPS,
```

```
} from "e2e/support/cypress_data";
```

```
import type { DatabaseData, User } from "metabase-types/api";
```

```
const {...
```

### **e2e/test/scenarios/admin/databases.cy.spec.js**

```
import {
```

```
  QA_MONGO_PORT,
```

```
  QA_MYSQL_PORT,
```

```
  QA_POSTGRES_PORT,
```

```
  SAMPLE_DB_ID,
```

```
  WRITABLE_DB_CONFIG,
```

```
  WRITABLE_DB_ID,
```

```
} from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from...
```

### **e2e/test/scenarios/admin/datamodel/datamodel.cy.spec.ts**

```
import {
```

```
  SAMPLE_DB_ID,
```

```
  SAMPLE_DB_SCHEMA_ID,
```

```
  USER_GROUPS,
```

```
  WRITABLE_DB_ID,
```

```
} from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import {...
```



## Analyst Base

### **e2e/test/scenarios/admin/datamodel/reproductions.cy.spec.ts**

```
const { H } = cy;
import {
  SAMPLE_DB_ID,
  SAMPLE_DB_SCHEMA_ID,
  WRITABLE_DB_ID,
} from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const {...
```

### **e2e/test/scenarios/admin/datamodel/segments.cy.spec.ts**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { H } = cy;
const { ORDERS, ORDERS_ID } = SAMPLE_DATABASE;

describe("scenarios > admin > datamodel > segments", () => {
  ...
```

### **e2e/test/scenarios/admin/helpers/e2e-database-helpers.js**

```
*
```

### **e2e/test/scenarios/admin/i18n/content-translation/constants.ts**

```
import type { NonEmpty } from "metabase/i18n/types";
import type { DictionaryArray } from "metabase-types/api";

export const germanFieldNames: NonEmpty<DictionaryArray> = [
  { locale: "de", msgid:...
```

### **e2e/test/scenarios/admin/i18n/content-translation/dashboards.cy.spec.ts**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  NORMAL_USER_ID,
  ORDERS_DASHBOARD_ID,
} from "e2e/support/cypress_sample_instance_data";
import type {...
```

### **e2e/test/scenarios/admin/i18n/content-translation/helpers/e2e-content-translation-helpers.ts**

```
import { parse } from "csv-parse/browser/esm/sync";

import { METABASE_SECRET_KEY } from "e2e/support/cypress_data";
import type { DictionaryArray, DictionaryResponse } from...
```

## Analyst Base

### **e2e/test/scenarios/admin/i18n/content-translation/questions.cy.spec.ts**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { NORMAL_USER_ID } from "e2e/support/cypress_sample_instance_data";
```

```
import { germanFieldNames } from...
```

### **e2e/test/scenarios/admin/i18n/content-translation/upload-and-download.cy.spec.ts**

```
import path from "path";
```

```
import {
  germanFieldNames,
  invalidLocaleXX,
  multipleInvalidLocales,
  nonAsciiFieldNames,
  portugueseFieldNames,
  stringTranslatedTwice,
} from "../constants";
import...
```

### **e2e/test/scenarios/admin/performance/clock.cy.spec.ts**

```
import dayjs from "dayjs";
import timezone from "dayjs/plugin/timezone";
```

```
const { H } = cy;
dayjs.extend(timezone);
```

```
import {
  freezeServerTime,
  resetServerTime,
} from...
```

### **e2e/test/scenarios/admin/performance/dashboardsAndQuestions.cy.spec.ts**

```
const { H } = cy;
```

```
import {
  TEST_TABLE,
  instanceDefault,
  sampleAdaptiveStrategy,
  sampleDashboard,
  sampleDatabase,
  sampleDurationStrategy,
  sampleQuestion,
```

## Analyst Base

```
} from...
```

### **e2e/test/scenarios/admin/performance/helpers/constants.ts**

```
import { WRITABLE_DB_ID } from "e2e/support/cypress_data";
import type { NativeQuestionDetails } from "e2e/support/helpers";
import {
  type AdaptiveStrategy,
  CacheDurationUnit,
  type...
```

### **e2e/test/scenarios/admin/performance/helpers/e2e-performance-helpers.ts**

```
import { WRITABLE_DB_ID } from "e2e/support/cypress_data";
import type { NativeQuestionDetails } from "e2e/support/helpers";
import {
  createNativeQuestion,
  createNativeQuestionAndDashboard,
  ...
```

### **e2e/test/scenarios/admin/performance/helpers/e2e-strategy-form-helpers.ts**

```
import { match } from "ts-pattern";

import { popover } from "e2e/support/helpers";
import {
  type ScheduleComponentType,
  getScheduleComponentLabel,
} from...
```

### **e2e/test/scenarios/admin/performance/helpers/types.ts**

```
import type { CacheStrategy, CacheableModel } from "metabase-types/api";

export type CacheTestParameters = {
  description: string;
  strategy: CacheStrategy;
  /** Item whose cache we are testing...
```

### **e2e/test/scenarios/admin/performance/preemptiveCaching.cy.spec.ts**

```
const { H } = cy;

import { WRITABLE_DB_ID } from "e2e/support/cypress_data";

import {
  advanceServerClockBy,
  interceptPerformanceRoutes,
  resetServerTime,
```

```
} from...
```

### **e2e/test/scenarios/admin/performance/schedule.cy.spec.ts**

```
import type { ScheduleComponentType } from "metabase/common/components/Schedule/strings";
import { checkNotNull } from "metabase/lib/types";
import type { CacheableModel } from...
```

### **e2e/test/scenarios/admin/performance/strategyForm.cy.spec.ts**

```
const { H } = cy;
import {
  ORDERS_COUNT_QUESTION_ID,
  ORDERS_QUESTION_ID,
} from "e2e/support/cypress_sample_instance_data";
```

```
import {
  goToPerformancePage,
  interceptPerformanceRoutes,
} from...
```

### **e2e/test/scenarios/admin/troubleshooting.cy.spec.js**

```
import dayjs from "dayjs";

import { createMockTask } from "metabase-types/api/mocks";

const { H } = cy;

describe("scenarios > admin > tools > help", { tags: "@OSS" }, () => {
  beforeEach(() => {
    ...
  })
})
```

### **e2e/test/scenarios/admin-2/api-keys.cy.spec.ts**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ADMINISTRATORS_GROUP_ID,
  ...
}
```

### **e2e/test/scenarios/admin-2/authentication.cy.spec.ts**

```
const { H } = cy;

import { setupSaml } from "../sso/shared/helpers.js";

describe("scenarios > admin > settings > user provisioning", () => {
```

## Analyst Base

```
beforeEach(() => {  
  H.restore();  
  ...  
})
```

### **e2e/test/scenarios/admin-2/error-reporting.cy.spec.ts**

```
const { H } = cy;  
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";  
  
const downloadsFolder = Cypress.config("downloadsFolder");  
  
describe("error reporting modal", () =>...
```

### **e2e/test/scenarios/admin-2/people.cy.spec.js**

```
import _ from "underscore";  
  
const { H } = cy;  
import { USERS, USER_GROUPS } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ...  
}
```

### **e2e/test/scenarios/admin-2/settings.cy.spec.js**

```
const { H } = cy;  
import {  
  SAMPLE_DB_ID,  
  SAMPLE_DB_SCHEMA_ID,  
  WEBMAIL_CONFIG,  
} from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {...
```

### **e2e/test/scenarios/admin-2/sso/google.cy.spec.js**

```
const { H } = cy;  
  
const CLIENT_ID_SUFFIX = "apps.googleusercontent.com";  
  
describe("scenarios > admin > settings > SSO > Google", () => {  
  beforeEach(() => {  
    H.restore();  
    ...  
  })  
})
```

### **e2e/test/scenarios/admin-2/sso/jwt.cy.spec.js**

```
const { H } = cy;  
import { enableJwtAuth } from "e2e/support/helpers/e2e-jwt-helpers";
```

## Analyst Base

```
import {
  checkGroupConsistencyAfterDeletingMappings,
  crudGroupMappingsWidget,
} from...
```

### **e2e/test/scenarios/admin-2/sso/ldap.cy.spec.js**

```
const { H } = cy;
```

```
import {
  checkGroupConsistencyAfterDeletingMappings,
  crudGroupMappingsWidget,
} from "../shared/group-mappings-widget";
```

```
describe(
  "scenarios > admin > settings > SSO >..."
```

### **e2e/test/scenarios/admin-2/sso/saml.cy.spec.js**

```
const { H } = cy;
```

```
import {
  checkGroupConsistencyAfterDeletingMappings,
  crudGroupMappingsWidget,
} from "../shared/group-mappings-widget";
import { getSamlCertificate, setupSaml } from...
```

### **e2e/test/scenarios/admin-2/sso/shared/group-mappings-widget.js**

```
import { popover } from "e2e/support/helpers";
```

```
export function crudGroupMappingsWidget(authenticationMethod) {
  cy.visit("/admin/settings/authentication/" + authenticationMethod);
  ...
}
```

### **e2e/test/scenarios/admin-2/sso/shared/helpers.js**

```
export function getSuccessUi() {
  return cy.findByTestId("admin-layout-content").findByText("Success");
}
```

```
export const getSamlCertificate = () => {
  return...
```

### **e2e/test/scenarios/admin-2/whitelabel.cy.spec.js**

```
const { H } = cy;
import { ORDERS_QUESTION_ID } from "e2e/support/cypress_sample_instance_data";
```

## Analyst Base

```
function checkFavicon(url) {  
  cy.request("/api/setting/application-favicon-url")  
    .its("body")  
  ...  
}
```

### **e2e/test/scenarios/binning/binning-options.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS_ID, ORDERS, PEOPLE_ID, PEOPLE,...
```

### **e2e/test/scenarios/binning/binning-reproductions.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID } =...
```

### **e2e/test/scenarios/binning/correctness/longitude.cy.spec.js**

```
const { H } = cy;  
  
import { LONGITUDE_OPTIONS } from "../shared/constants";  
  
describe("scenarios > binning > correctness > longitude", () => {  
  beforeEach(() => {  
    H.restore();  
    ...  
  })  
})
```

### **e2e/test/scenarios/binning/correctness/shared/constants.js**

```
export const TIME_OPTIONS = {  
  Minute: {  
    selected: "by minute",  
    representativeValues: ["April 30, 2022, 6:56 PM", "May 10, 2022, 9:38 AM"],  
  },  
  Hour: {  
    selected: "by hour",  
    ...  
  }  
}
```

### **e2e/test/scenarios/binning/correctness/time-series.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

## Analyst Base

```
import { TIME_OPTIONS } from "../shared/constants";
```

```
const { ORDERS_ID } = SAMPLE_DATABASE;
```

```
const...
```

### **e2e/test/scenarios/binning/qb-explicit-joins.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS_ID, ORDERS, PEOPLE_ID, PEOPLE, PRODUCTS_ID, PRODUCTS } =  
  SAMPLE_DATABASE;
```

```
/**
```

```
 * The list...
```

### **e2e/test/scenarios/binning/qb-implicit-joins.cy.spec.js**

```
const { H } = cy;
```

```
import { ORDERS_QUESTION_ID } from "e2e/support/cypress_sample_instance_data";
```

```
/**
```

```
 * The list of issues this spec covers:
```

```
 * - metabase#15648
```

```
 * -
```

```
 */
```

```
describe("scenarios >...
```

### **e2e/test/scenarios/binning/qb-regular-table.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS_ID, PEOPLE_ID } = SAMPLE_DATABASE;
```

```
describe("scenarios > binning > binning options", () => {
```

```
...
```

### **e2e/test/scenarios/binning/reproductions/23851-drill-temporal-extraction.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID } = SAMPLE_DATABASE;
```



## Analyst Base

```
const CREATED_AT_BREAKOUT = [  
  "field",  
  ...
```

### **e2e/test/scenarios/binning/reproductions/34688-34690-time-series-footer.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS_ID, PRODUCTS } = ...
```

### **e2e/test/scenarios/binning/sql.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
const questionDetails = {  
  name: "SQL Binning",  
  native: {  
    query:  
      "SELECT ORDERS.CREATED_AT, ORDERS.TOTAL,..."
```

### **e2e/test/scenarios/collections/cleanup.cy.spec.js**

```
import dayjs from "dayjs";  
import { assocIn } from "icepick";  
import { P, isMatching } from "ts-pattern";
```

```
const { H } = cy;  
import { SAMPLE_DB_TABLES } from "e2e/support/cypress_data";  
import {  
  ...
```

### **e2e/test/scenarios/collections/collection-pinned-overview.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ORDERS_COUNT_QUESTION_ID,  
  ORDERS_DASHBOARD_ID,  
  ORDERS_QUESTION_ID,  
} from ...
```

### **e2e/test/scenarios/collections/collections-reproductions.cy.spec.js**

```
const { H } = cy;  
import {
```

## Analyst Base

```
ADMIN_PERSONAL_COLLECTION_ID,  
FIRST_COLLECTION_ID,  
ORDERS_COUNT_QUESTION_ID,  
ORDERS_QUESTION_ID,  
} from "e2e/support/cypress_sample_instance_data";
```

```
describe("issue...
```

### **e2e/test/scenarios/collections/collections.cy.spec.js**

```
import { assocIn } from "icepick";  
import _ from "underscore";
```

```
const { H } = cy;  
import { SAMPLE_DB_ID, USERS, USER_GROUPS } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from...
```

### **e2e/test/scenarios/collections/helpers/e2e-collections-sidebar.js**

```
export function displaySidebarChildOf(collectionName) {  
  cy.findByText(collectionName)  
    .parentsUntil("[data-testid=sidebar-collection-link-root]")  
    .find(".icon-chevronright")  
    .eq(0) //...
```

### **e2e/test/scenarios/collections/instance-analytics.cy.spec.js**

```
const { H } = cy;  
import {  
  ORDERS_DASHBOARD_ID,  
  ORDERS_QUESTION_ID,  
} from "e2e/support/cypress_sample_instance_data";
```

```
const ANALYTICS_COLLECTION_NAME = "Usage analytics";  
const...
```

### **e2e/test/scenarios/collections/permissions.cy.spec.js**

```
import { onlyOn } from "@cypress/skip-test";
```

```
const { H } = cy;  
import { USERS } from "e2e/support/cypress_data";  
import { FIRST_COLLECTION_ID } from...
```

### **e2e/test/scenarios/collections/personal-collections.cy.spec.js**

```
const { H } = cy;  
import { USERS } from "e2e/support/cypress_data";  
import {
```

## Analyst Base

```
ADMIN_PERSONAL_COLLECTION_ID,  
NORMAL_USER_ID,  
NO_DATA_PERSONAL_COLLECTION_ID,  
} from...
```

### **e2e/test/scenarios/collections/revision-history.cy.spec.js**

```
import { onlyOn } from "@cypress/skip-test";  
  
const { H } = cy;  
import {  
  ORDERS_DASHBOARD_ID,  
  ORDERS_QUESTION_ID,  
} from "e2e/support/cypress_sample_instance_data";
```

```
const PERMISSIONS = {  
  ...
```

### **e2e/test/scenarios/collections/trash.cy.spec.js**

```
import { P, isMatching } from "ts-pattern";  
  
const { H } = cy;  
import {  
  FIRST_COLLECTION_ID,  
  ORDERS_COUNT_QUESTION_ID,  
  ORDERS_QUESTION_ID,  
  READ_ONLY_PERSONAL_COLLECTION_ID,  
} from...
```

### **e2e/test/scenarios/collections/uploads.cy.spec.js**

```
const { H } = cy;  
import { USER_GROUPS, WRITABLE_DB_ID } from "e2e/support/cypress_data";  
import { FIRST_COLLECTION_ID } from "e2e/support/cypress_sample_instance_data";  
import { FIXTURE_PATH,...
```

### **e2e/test/scenarios/cross-version/helpers/cross-version-helpers.js**

```
export function parseVersionString(versionString) {  
  if (typeof versionString === "undefined") {  
    return []; // Return empty array if versionString is undefined  
  }  
}
```

```
const segments =...
```

### **e2e/test/scenarios/cross-version/source/00-setup.cy.spec.js**

```
import {
```

## Analyst Base

```
    setupInstance,  
    setupLanguage,  
  } from "e2e/test/scenarios/cross-version/helpers/cross-version-helpers.js";
```

```
import { version } from...
```

### **e2e/test/scenarios/cross-version/source/01-generate-metadata.cy.spec.js**

```
const { H } = cy;
```

```
it("should generate metadata", () => {  
  cy.signInAsAdmin();
```

```
  H.withSampleDatabase((SAMPLE_DATABASE) => {  
    cy.writeFile("e2e/support/cypress_sample_database.json",...
```

### **e2e/test/scenarios/cross-version/source/02-datamodel.cy.spec.js**

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  createMetric,  
  createSegment,  
} from...
```

### **e2e/test/scenarios/cross-version/source/03-questions.cy.spec.js**

```
const { H } = cy;  
import {  
  fillAreaUnderLineChart,  
  newQuestion,  
  saveQuestion,  
} from "e2e/test/scenarios/cross-version/helpers/cross-version-helpers.js";
```

```
import { version } from...
```

### **e2e/test/scenarios/cross-version/source/helpers/cross-version-source-helpers.js**

```
import { parseVersionString } from "e2e/test/scenarios/cross-version/helpers/cross-version-helpers.js";
```

```
export const version = parseVersionString(Cypress.env("SOURCE_VERSION"));
```

### **e2e/test/scenarios/cross-version/source/shared/cross-version-source.config.js**

```
const { defineConfig } = require("cypress");
```

```
const { crossVersionSourceConfig } = require("e2e/support/config");
```

## Analyst Base

```
module.exports = defineConfig({ e2e: crossVersionSourceConfig });
```

### **e2e/test/scenarios/cross-version/target/helpers/cross-version-target-helpers.js**

```
import { parseVersionString } from "e2e/test/scenarios/cross-version/helpers/cross-version-helpers.js";
```

```
export const version = parseVersionString(Cypress.env("TARGET_VERSION"));
```

### **e2e/test/scenarios/cross-version/target/shared/cross-version-target.config.js**

```
const { defineConfig } = require("cypress");
```

```
const { crossVersionTargetConfig } = require("e2e/support/config");
```

```
module.exports = defineConfig({ e2e: crossVersionTargetConfig });
```

### **e2e/test/scenarios/cross-version/target/smoke.cy.spec.js**

```
import {
  assertTimelineData,
  dismissOkToPlayWithQuestionsModal,
} from "e2e/test/scenarios/cross-version/helpers/cross-version-helpers";
import { version } from...
```

### **e2e/test/scenarios/custom-column/cc-boolean-functions.cy.spec.ts**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import type {
  DashboardDetails,
  StructuredQuestionDetails,
} from "e2e/support/helpers";
import type {...
```

### **e2e/test/scenarios/custom-column/cc-cast-functions.cy.spec.ts**

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
const { H } = cy;
```

```
type CastTestCase = {
  name: string;
  expression: string;
  filterOperator: string;
  filterValue: string;
  ...
```

### **e2e/test/scenarios/custom-column/cc-fields.cy.spec.ts**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

## Analyst Base

```
import type { StructuredQuestionDetails } from "e2e/support/helpers";
```

```
const { ORDERS_ID, ORDERS } =...
```

### **e2e/test/scenarios/custom-column/cc-literals.cy.spec.ts**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS, PRODUCTS_ID } =...
```

### **e2e/test/scenarios/custom-column/cc-shortcuts-combine.cy.spec.ts**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
describe("scenarios > question > custom column > expression shortcuts > combine", () => {  
  beforeEach(() => {  
    ...
```

### **e2e/test/scenarios/custom-column/cc-shortcuts.cy.spec.ts**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS_ID, ORDERS } =...
```

### **e2e/test/scenarios/custom-column/cc-typing-suggestion.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID } = SAMPLE_DATABASE;
```

```
describe("scenarios > question > custom column > typing...
```

### **e2e/test/scenarios/custom-column/custom-column-reproductions.cy.spec.js**

```
const { H } = cy;
```

```
import { dedent } from "ts-dedent";
```

```
import { SAMPLE_DB_ID, WRITABLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from...
```

### **e2e/test/scenarios/custom-column/custom-column.cy.spec.js**

```
const { H } = cy;
```

```
import { dedent } from "ts-dedent";
```

## Analyst Base

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS,...
```

### **e2e/test/scenarios/dashboard/dashboard-back-navigation.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ADMIN_PERSONAL_COLLECTION_ID,
  ...
```

### **e2e/test/scenarios/dashboard/dashboard-management.cy.spec.js**

```
import { onlyOn } from "@cypress/skip-test";
```

```
const { H } = cy;
import { USERS } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {...
```

### **e2e/test/scenarios/dashboard/dashboard-questions.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import * as S from "e2e/support/cypress_sample_instance_data";
import { createMockDashboardCard } from...
```

### **e2e/test/scenarios/dashboard/dashboard-reproductions.cy.spec.js**

```
import { assoc } from "icepick";
import _ from "underscore";
```

```
const { H } = cy;
import { SAMPLE_DB_ID, USERS, USER_GROUPS } from "e2e/support/cypress_data";
import {
  ORDERS_COUNT_QUESTION_ID,
  ...
```

### **e2e/test/scenarios/dashboard/dashboard.cy.spec.js**

```
import { assoc } from "icepick";
import _ from "underscore";
```

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

## Analyst Base

```
import { SAMPLE_DATABASE } from...
```

### **e2e/test/scenarios/dashboard/tabs.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ADMIN_PERSONAL_COLLECTION_ID,
  NORMAL_PERSONAL_COLLECTION_ID,
  ORDERS_BY_YEAR_QUESTION_ID,
  ...
```

### **e2e/test/scenarios/dashboard/text-cards.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
import { createMockParameter }...
```

### **e2e/test/scenarios/dashboard/title-drill.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { ORDERS, ORDERS_ID, PEOPLE, PEOPLE_ID, PRODUCTS, PRODUCTS_ID } =
  ...
```

### **e2e/test/scenarios/dashboard/visualizer/basics.cy.spec.ts**

```
const { H } = cy;

import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
import {
  ORDERS_COUNT_BY_CREATED_AT,
  ...
```

### **e2e/test/scenarios/dashboard/visualizer/cartesian.cy.spec.ts**

```
const { H } = cy;

import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
import {
  ORDERS_COUNT_BY_CREATED_AT,
  ORDERS_COUNT_BY_CREATED_AT_AND_PRODUCT_CATEGORY,
  ...
```

### **e2e/test/scenarios/dashboard/visualizer/columns-mapping.cy.spec.ts**

```
const { H } = cy;
```



## Analyst Base

```
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
import {
  ACCOUNTS_COUNT_BY_COUNTRY,
  COUNTRY_CODES,
  ORDERS_COUNT_BY_PRODUCT_CATEGORY,
} from...
```

### **e2e/test/scenarios/dashboard/visualizer/drillthrough.cy.spec.ts**

```
const { H } = cy;
```

```
import {
  ORDERS_COUNT_BY_CREATED_AT,
  ORDERS_COUNT_BY_PRODUCT_CATEGORY,
  PRODUCTS_COUNT_BY_CATEGORY,
  PRODUCTS_COUNT_BY_CATEGORY_PIE,
  PRODUCTS_COUNT_BY_CREATED_AT,
  ...
```

### **e2e/test/scenarios/dashboard/visualizer/filters.cy.spec.ts**

```
const { H } = cy;
```

```
import {
  PRODUCTS_AVERAGE_BY_CATEGORY,
  PRODUCTS_COUNT_BY_CATEGORY,
} from "e2e/support/test-visualizer-data";
```

```
describe("scenarios > dashboard > visualizer > filters", () => {
```

```
...
```

### **e2e/test/scenarios/dashboard/visualizer/funnels.cy.spec.ts**

```
const { H } = cy;
```

```
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
import {
  ORDERS_COUNT_BY_CREATED_AT,
  ORDERS_COUNT_BY_PRODUCT_CATEGORY,
  ...
```

### **e2e/test/scenarios/dashboard/visualizer/pie.cy.spec.ts**

```
const { H } = cy;
```

```
import {
  ORDERS_COUNT_BY_CREATED_AT,
  ORDERS_COUNT_BY_PRODUCT_CATEGORY,
```

## Analyst Base

```
PRODUCTS_COUNT_BY_CATEGORY,  
PRODUCTS_COUNT_BY_CATEGORY_PIE,  
PRODUCTS_COUNT_BY_CREATED_AT,  
...
```

### **e2e/test/scenarios/dashboard/visualizer/snowplow-tracking.cy.spec.ts**

```
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";  
import {  
  ACCOUNTS_COUNT_BY_CREATED_AT,  
  ORDERS_COUNT_BY_CREATED_AT,  
  ORDERS_COUNT_BY_PRODUCT_CATEGORY,  
  ...  
}
```

### **e2e/test/scenarios/dashboard/x-rays.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import { ORDERS_BY_YEAR_QUESTION_ID } from...
```

### **e2e/test/scenarios/dashboard-cards/click-behavior.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  NORMAL_USER_ID,  
  ...  
}
```

### **e2e/test/scenarios/dashboard-cards/dashboard-card-fetching.cy.spec.js**

```
const { H } = cy;  
import {  
  ORDERS_BY_YEAR_QUESTION_ID,  
  ORDERS_COUNT_QUESTION_ID,  
} from "e2e/support/cypress_sample_instance_data";
```

```
const cards = [  
  {  
    card_id: ORDERS_COUNT_QUESTION_ID,  
    ...  
  },  
  ...  
]
```

### **e2e/test/scenarios/dashboard-cards/dashboard-card-reproductions.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, WRITABLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import { ORDERS_QUESTION_ID } from...
```

## Analyst Base

### **e2e/test/scenarios/dashboard-cards/dashboard-card-resizing.cy.spec.js**

```
import _ from "underscore";
```

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import { GRID_WIDTH }...
```

### **e2e/test/scenarios/dashboard-cards/dashboard-card-undo.cy.spec.js**

```
const { H } = cy;
```

```
describe("scenarios > dashboard cards > undo", () => {
```

```
  beforeEach(() => {
```

```
    H.restore();
```

```
    cy.signInAsAdmin();
```

```
  });
```

```
  it(
```

```
    "when undoing a dashcard removal or dashcard..."
```

### **e2e/test/scenarios/dashboard-cards/dashboard-drill.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import {
```

```
  ORDERS_DASHBOARD_DASHCARD_ID,
```

```
  ...
```

### **e2e/test/scenarios/dashboard-cards/dashboard-sections.cy.spec.js**

```
import {
```

```
  ORDERS_DASHBOARD_ID,
```

```
  READ_ONLY_PERSONAL_COLLECTION_ID,
```

```
} from "e2e/support/cypress_sample_instance_data";
```

```
import * as H from "e2e/support/helpers";
```

```
import { createMockParameter } from...
```

### **e2e/test/scenarios/dashboard-cards/dashcard-replace-question.cy.spec.js**

```
const { H } = cy;
```

```
import { USER_GROUPS } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import {
```

```
  FIRST_COLLECTION_ID,
```

```
  ...
```

## Analyst Base

### **e2e/test/scenarios/dashboard-cards/duplicate-dashcards-tabs.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  createMockDashboardCard,
  createMockParameter,
} from "metabase-types/api/mocks";
```

```
const {...
```

### **e2e/test/scenarios/dashboard-cards/visualization-options.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
```

```
describe("scenarios >...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-chained-filters.cy.spec.js**

```
const { H } = cy;
import { WRITABLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ORDERS_DASHBOARD_ID } from...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filter-data-permissions.cy.spec.js**

```
const { H } = cy;
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
```

```
function filterDashboard(suggests = true) {
  H.visitDashboard(ORDERS_DASHBOARD_ID);
  ...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filter-defaults.cy.spec.ts**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import type { StructuredQuestionDetails } from "e2e/support/helpers";
```

```
const { PRODUCTS_ID, PRODUCTS } =...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-auto-apply.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS, PRODUCTS_ID } = SAMPLE_DATABASE;
```

## Analyst Base

```
const FILTER = {  
  name: "Category",  
  slug: "category",  
  ...  
}
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-auto-wiring.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ORDERS_BY_YEAR_QUESTION_ID,  
  ORDERS_COUNT_QUESTION_ID,  
} from...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-boolean.cy.spec.ts**

```
const { H } = cy;  
  
import { WRITABLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import type {  
  DashboardDetails,  
  ...  
}
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-clear-and-restore.cy.spec.ts**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
  
const { ORDERS_ID } = SAMPLE_DATABASE;
```

```
describe("dashboard filters values source config clearing and...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-date.cy.spec.js**

```
const { H } = cy;  
import {  
  ORDERS_DASHBOARD_DASHCARD_ID,  
  ORDERS_DASHBOARD_ID,  
} from "e2e/support/cypress_sample_instance_data";
```

```
import * as DateFilter from...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-explicit-join.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

## Analyst Base

```
const { ORDERS, ORDERS_ID, PRODUCTS, PRODUCTS_ID } = SAMPLE_DATABASE;
```

```
const questionDetails = {  
  name:...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-id.cy.spec.js**

```
const { H } = cy;  
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
```

```
import { addWidgetStringFilter } from...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-location.cy.spec.js**

```
const { H } = cy;  
import {  
  ORDERS_DASHBOARD_DASHCARD_ID,  
  ORDERS_DASHBOARD_ID,  
} from "e2e/support/cypress_sample_instance_data";
```

```
import {  
  addWidgetStringFilter,  
  selectFilterValueFromList,  
}...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-management.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import { createMockParameter } from "metabase-types/api/mocks";
```

```
const { PEOPLE, PEOPLE_ID, ORDERS_ID } =...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-nested.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS_ID } = SAMPLE_DATABASE;
```

```
describe("scenarios > dashboard > filters > nested questions", ()...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-number-source.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
```

## Analyst Base

```
const { ACCOUNTS, ORDERS_ID }...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-number.cy.spec.js**

```
const { H } = cy;
```

```
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
```

```
import { addWidgetNumberFilter } from "../native-filters/helpers/e2e-field-filter-helpers";
```

```
import...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-remapping.cy.spec.ts**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import type {
```

```
  DashboardDetails,
```

```
  NativeQuestionDetails,
```

```
  StructuredQuestionDetails,
```

```
} from...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-reset-clear.cy.spec.ts**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import {
```

```
  ORDERS_COUNT_QUESTION_ID,
```

```
  ORDERS_QUESTION_ID,
```

```
} from...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-source.cy.spec.js**

```
const { H } = cy;
```

```
import { USER_GROUPS, WRITABLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS, PRODUCTS_ID } =...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-sql-date.cy.spec.js**

```
const { H } = cy;
```

```
import * as DateFilter from "../native-filters/helpers/e2e-date-filter-helpers";
```

```
import {
```

```
  DASHBOARD_SQL_DATE_FILTERS,
```

```
  questionDetails,
```

## Analyst Base

```
} from...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-sql-id.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import { addWidgetStringFilter } from "../native-filters/helpers/e2e-field-filter-helpers";
```

```
const { ORDERS }...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-sql-location.cy.spec.js**

```
const { H } = cy;
```

```
import {  
  addWidgetStringFilter,  
  applyFilterByType,  
  selectFilterValueFromList,  
} from "../native-filters/helpers/e2e-field-filter-helpers";
```

```
import {  
  ...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-sql-management.cy.spec.js**

```
const { H } = cy;
```

```
describe("scenarios > dashboard > filters > SQL > management", () => {  
  beforeEach(() => {  
    H.restore();  
    cy.signInAsAdmin();  
  });
```

```
  describe("number filter", () => {  
    ...
```

### **e2e/test/scenarios/dashboard-filters/dashboard-filters-sql-number.cy.spec.js**

```
const { H } = cy;
```

```
import { addWidgetNumberFilter } from "../native-filters/helpers/e2e-field-filter-helpers";
```

```
import {  
  DASHBOARD_SQL_NUMBER_FILTERS,  
  questionDetails,  
} from...
```



## **e2e/test/scenarios/dashboard-filters/dashboard-filters-sql-required-field-filter.cy.spec.js**

```
import { produce } from "immer";
```

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS } = SAMPLE_DATABASE;
```

```
const questionDetails = {  
  name:...
```

## **e2e/test/scenarios/dashboard-filters/dashboard-filters-sql-required-simple-filter.cy.spec.js**

```
const { H } = cy;
```

```
const questionDetails = {  
  name: "Return input value",  
  native: {  
    query: "select {{filter}}",  
    "template-tags": {  
      filter: {  
        id:...
```

## **e2e/test/scenarios/dashboard-filters/dashboard-filters-sql-text-category.cy.spec.js**

```
const { H } = cy;
```

```
import { applyFilterByType } from "../native-filters/helpers/e2e-field-filter-helpers";
```

```
import {  
  DASHBOARD_SQL_TEXT_FILTERS,  
  questionDetails,  
} from...
```

## **e2e/test/scenarios/dashboard-filters/dashboard-filters-text-category.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import {  
  applyFilterByType,  
  selectDefaultValueFromPopover,  
} from...
```

## Analyst Base

### **e2e/test/scenarios/dashboard-filters/old-parameters.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { PEOPLE, PEOPLE_ID, PRODUCTS, PRODUCTS_ID } = SAMPLE_DATABASE;

// the dashboard parameters used in...
```

### **e2e/test/scenarios/dashboard-filters/parameters.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ORDERS_BY_YEAR_QUESTION_ID,
  ...
```

### **e2e/test/scenarios/dashboard-filters/shared/dashboard-filters-date.js**

```
export const DASHBOARD_DATE_FILTERS = {
  "Month and Year": {
    value: {
      month: "Nov",
      year: "2022",
    },
    representativeResult: "85.88",
  },
  "Quarter and Year": {
    value: {
      ...
```

### **e2e/test/scenarios/dashboard-filters/shared/dashboard-filters-location.js**

```
export const DASHBOARD_LOCATION_FILTERS = {
  Is: {
    value: "Abbeville",
    representativeResult: "1510",
  },
  "Is not": {
    value: "Abbeville",
    representativeResult: "37.65",
  },
  ...
```

### **e2e/test/scenarios/dashboard-filters/shared/dashboard-filters-number.js**

```
export const DASHBOARD_NUMBER_FILTERS = [
```

## Analyst Base

```
{
  operator: "Equal to",
  value: "2.07",
  representativeResult: "37.65",
},
{
  operator: "Equal to",
  value: "2.07",
  ...
}
```

### **e2e/test/scenarios/dashboard-filters/shared/dashboard-filters-sql-date.js**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { PEOPLE } = SAMPLE_DATABASE;

export const DASHBOARD_SQL_DATE_FILTERS = {
  "Month and Year": {
    sqlFilter:...
```

### **e2e/test/scenarios/dashboard-filters/shared/dashboard-filters-sql-location.js**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { PEOPLE } = SAMPLE_DATABASE;

export const questionDetails = {
  name: "SQL with Field Filter",
  native: {
    query:
  ...
}
```

### **e2e/test/scenarios/dashboard-filters/shared/dashboard-filters-sql-number.js**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { PRODUCTS } = SAMPLE_DATABASE;

export const DASHBOARD_SQL_NUMBER_FILTERS = {
  "Equal to": {
    sqlFilter:...
```

### **e2e/test/scenarios/dashboard-filters/shared/dashboard-filters-sql-text-category.js**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { PRODUCTS } = SAMPLE_DATABASE;
```

## Analyst Base

```
export const DASHBOARD_SQL_TEXT_FILTERS = {
  ls: {
    sqlFilter: "string/=",
    ...
  }
}
```

### **e2e/test/scenarios/dashboard-filters/shared/dashboard-filters-text-category.js**

```
export const DASHBOARD_TEXT_FILTERS = [
  {
    operator: "Is",
    single: true,
    value: "Organic",
    representativeResult: "39.58",
  },
  {
    operator: "Is not",
    single: true,
    ...
  }
]
```

### **e2e/test/scenarios/dashboard-filters/temporal-unit-parameters.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { ORDERS, ORDERS_ID, PRODUCTS } = SAMPLE_DATABASE;

const dashboardDetails = {
  name: "Test..."
}
```

### **e2e/test/scenarios/dashboard-filters-2/dashboard-filters-1-stage.cy.spec.ts**

```
const { H } = cy;

import * as QSHelpers from "../shared/dashboard-filters-query-stages";

/**
 * Abbreviations used for card aliases in this test suite:
 * qbq = question-based question
 * qbm =...
```

### **e2e/test/scenarios/dashboard-filters-2/dashboard-filters-2-stage-1.cy.spec.ts**

```
const { H } = cy;

import * as QSHelpers from "../shared/dashboard-filters-query-stages";

/**
```

## Analyst Base

- \* Abbreviations used for card aliases in this test suite:
- \* qbq = question-based question
- \* qbm =...

### **e2e/test/scenarios/dashboard-filters-2/dashboard-filters-2-stage-2.cy.spec.ts**

```
const { H } = cy;
```

```
import * as QSHelpers from "../shared/dashboard-filters-query-stages";
```

```
/**
```

- \* Abbreviations used for card aliases in this test suite:
- \* qbq = question-based question
- \* qbm =...

### **e2e/test/scenarios/dashboard-filters-2/dashboard-filters-2-stage-3.cy.spec.ts**

```
const { H } = cy;
```

```
import * as QSHelpers from "../shared/dashboard-filters-query-stages";
```

```
/**
```

- \* Abbreviations used for card aliases in this test suite:
- \* qbq = question-based question
- \* qbm =...

### **e2e/test/scenarios/dashboard-filters-2/dashboard-filters-2-stage-4.cy.spec.ts**

```
const { H } = cy;
```

```
import * as QSHelpers from "../shared/dashboard-filters-query-stages";
```

```
/**
```

- \* Abbreviations used for card aliases in this test suite:
- \* qbq = question-based question
- \* qbm =...

### **e2e/test/scenarios/dashboard-filters-2/dashboard-filters-3-stage.cy.spec.ts**

```
const { H } = cy;
```

```
import * as QSHelpers from "../shared/dashboard-filters-query-stages";
```

```
/**
```

- \* Abbreviations used for card aliases in this test suite:
- \* qbq = question-based question
- \* qbm =...

### **e2e/test/scenarios/dashboard-filters-2/dashboard-filters-misc.cy.spec.ts**

## Analyst Base

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import type { Card, StructuredQuery } from "metabase-types/api";
```

```
import * as QSHelpers from...
```

### **e2e/test/scenarios/dashboard-filters-2/shared/dashboard-filters-query-stages.ts**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import type {
  Card,
  ConcreteFieldReference,
  StructuredQuery,
} from "metabase-types/api";
```

```
const {...
```

### **e2e/test/scenarios/dashboard-filters-matrix/helpers/matrix-helpers.ts**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import type { ValuesQueryType } from "metabase-types/api";
```

```
import { matrix } from "../matrix";
```

```
export type...
```

### **e2e/test/scenarios/dashboard-filters-matrix/helpers/matrix.ts**

```
import type { TestCase } from "../matrix-helpers";
```

```
export const matrix: TestCase[] = [
  {
    arity: "single",
    type: "search",
    adminType: "search",
    operator: "Is",
    source:...
```

### **e2e/test/scenarios/dashboard-filters-matrix/page-0.cy.spec.ts**

```
import { runPage } from "../helpers/matrix-helpers";
```

```
describe("scenarios > dashboard > parameters > matrix", () => {
  runPage(0);
});
```

### **e2e/test/scenarios/dashboard-filters-matrix/page-1.cy.spec.ts**

## Analyst Base

```
import { runPage } from "../helpers/matrix-helpers";
```

```
describe("scenarios > dashboard > parameters > matrix", () => {  
  runPage(1);  
});
```

### **e2e/test/scenarios/dashboard-filters-matrix/page-10.cy.spec.ts**

```
import { runPage } from "../helpers/matrix-helpers";
```

```
describe("scenarios > dashboard > parameters > matrix", () => {  
  runPage(10);  
});
```

### **e2e/test/scenarios/dashboard-filters-matrix/page-2.cy.spec.ts**

```
import { runPage } from "../helpers/matrix-helpers";
```

```
describe("scenarios > dashboard > parameters > matrix", () => {  
  runPage(2);  
});
```

### **e2e/test/scenarios/dashboard-filters-matrix/page-3.cy.spec.ts**

```
import { runPage } from "../helpers/matrix-helpers";
```

```
describe("scenarios > dashboard > parameters > matrix", () => {  
  runPage(3);  
});
```

### **e2e/test/scenarios/dashboard-filters-matrix/page-4.cy.spec.ts**

```
import { runPage } from "../helpers/matrix-helpers";
```

```
describe("scenarios > dashboard > parameters > matrix", () => {  
  runPage(4);  
});
```

### **e2e/test/scenarios/dashboard-filters-matrix/page-5.cy.spec.ts**

```
import { runPage } from "../helpers/matrix-helpers";
```

```
describe("scenarios > dashboard > parameters > matrix", () => {  
  runPage(5);  
});
```

### **e2e/test/scenarios/dashboard-filters-matrix/page-6.cy.spec.ts**

```
import { runPage } from "../helpers/matrix-helpers";
```

```
describe("scenarios > dashboard > parameters > matrix", () => {
```

## Analyst Base

```
runPage(6);  
});
```

### **e2e/test/scenarios/dashboard-filters-matrix/page-7.cy.spec.ts**

```
import { runPage } from "../helpers/matrix-helpers";  
  
describe("scenarios > dashboard > parameters > matrix", () => {  
  runPage(7);  
});
```

### **e2e/test/scenarios/dashboard-filters-matrix/page-8.cy.spec.ts**

```
import { runPage } from "../helpers/matrix-helpers";  
  
describe("scenarios > dashboard > parameters > matrix", () => {  
  runPage(8);  
});
```

### **e2e/test/scenarios/dashboard-filters-matrix/page-9.cy.spec.ts**

```
import { runPage } from "../helpers/matrix-helpers";  
  
describe("scenarios > dashboard > parameters > matrix", () => {  
  runPage(9);  
});
```

### **e2e/test/scenarios/embedding/embed-resource-downloads.cy.spec.ts**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ORDERS_BY_YEAR_QUESTION_ID,  
  ORDERS_DASHBOARD_ID,  
} from...
```

### **e2e/test/scenarios/embedding/embedding-dashboard.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";  
import { createMockParameter }...
```

### **e2e/test/scenarios/embedding/embedding-linked-filters.cy.spec.js**

```
const { H } = cy;  
  
import {  
  guiDashboard,  
  guiQuestion,  
  mapGUIDashboardParameters,
```



## Analyst Base

```
mapNativeDashboardParameters,  
nativeDashboardDetails,  
nativeQuestionDetails,  
} from...
```

### **e2e/test/scenarios/embedding/embedding-native.cy.spec.js**

```
const { H } = cy;
```

```
import * as SQLFilter from "../native-filters/helpers/e2e-sql-filter-helpers";
```

```
import {  
  questionDetails as questionDetails2,  
  questionDetailsWithDefaults,  
} from...
```

### **e2e/test/scenarios/embedding/embedding-questions.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ORDERS_DASHBOARD_ID,  
  ORDERS_QUESTION_ID,  
} from...
```

### **e2e/test/scenarios/embedding/embedding-reproductions.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";  
import { defer } from...
```

### **e2e/test/scenarios/embedding/embedding-smoketests.cy.spec.js**

```
const { H } = cy;  
import { METABASE_SECRET_KEY } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ORDERS_DASHBOARD_ID,  
  ...
```

### **e2e/test/scenarios/embedding/embedding-snippets.cy.spec.js**

```
const { H } = cy;  
import {  
  ORDERS_DASHBOARD_ID,  
  ORDERS_QUESTION_ID,  
} from "e2e/support/cypress_sample_instance_data";
```

## Analyst Base

```
import { IFRAME_CODE, getEmbeddingJsCode } from...
```

### **e2e/test/scenarios/embedding/interactive-embedding.cy.spec.js**

```
const { H } = cy;
import { USERS, USER_GROUPS, WRITABLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ...
```

### **e2e/test/scenarios/embedding/sdk-iframe-embedding/authentication.cy.spec.ts**

```
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
import {
  mockAuthSsoEndpointForSamlAuthProvider,
  stubWindowOpenForSamlPopup,
} from...
```

### **e2e/test/scenarios/embedding/sdk-iframe-embedding/custom-elements-api.cy.spec.ts**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ORDERS_COUNT_QUESTION_ID,
  ORDERS_DASHBOARD_ID,
  ORDERS_QUESTION_ID,
  THIRD_COLLECTION_ID,
} from...
```

### **e2e/test/scenarios/embedding/sdk-iframe-embedding/embed-options.cy.spec.ts**

```
import {
  FIRST_COLLECTION_ID,
  ORDERS_DASHBOARD_ID,
  ORDERS_QUESTION_ID,
} from "e2e/support/cypress_sample_instance_data";

const { H } = cy;
```

```
describe("scenarios > embedding > sdk iframe embed...
```

### **e2e/test/scenarios/embedding/sdk-iframe-embedding/missing-tokens.cy.spec.ts**

```
const { H } = cy;

describe("scenarios > embedding > sdk iframe embedding > without token features", () => {
  beforeEach(() => {
    H.prepareSdkIframeEmbedTest({
      withTokenFeatures: false,
```

...

### **e2e/test/scenarios/embedding/sdk-iframe-embedding/sdk-iframe-embedding.cy.spec.ts**

```
import {
  ORDERS_COUNT_QUESTION_ID,
  ORDERS_DASHBOARD_ENTITY_ID,
  ORDERS_DASHBOARD_ID,
  ORDERS_QUESTION_ENTITY_ID,
  ORDERS_QUESTION_ID,
} from "e2e/support/cypress_sample_instance_data";
```

const...

### **e2e/test/scenarios/embedding/sdk-iframe-embedding/theming.cy.spec.ts**

```
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
import type { MetabaseTheme } from "metabase/embedding-sdk/theme/MetabaseTheme";
```

```
const { H } = cy;
```

const LIGHT\_THEME...

### **e2e/test/scenarios/embedding/sdk-iframe-embedding-setup/auto-enable-embedding.cy.spec.ts**

```
const { H } = cy;
```

```
const suiteTitle =
```

```
"scenarios > embedding > sdk iframe embed setup > auto enable embedding settings";
```

```
describe(suiteTitle, () => {
```

```
  beforeEach(() => {
```

```
    H.restore();
```

...

### **e2e/test/scenarios/embedding/sdk-iframe-embedding-setup/embed-parameters.cy.spec.ts**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import {
```

```
  codeBlock,
```

```
  getEmbedSidebar,
```

```
  navigateToEmbedOptionsStep,
```

## Analyst Base

```
navigateToEntitySelectionStep,  
} from...
```

### **e2e/test/scenarios/embedding/sdk-iframe-embedding-setup/get-code.cy.spec.ts**

```
import {  
  ORDERS_COUNT_QUESTION_ID,  
  ORDERS_DASHBOARD_ID,  
} from "e2e/support/cypress_sample_instance_data";  
import { enableJwtAuth } from "e2e/support/helpers/e2e-jwt-helpers";  
import {...
```

### **e2e/test/scenarios/embedding/sdk-iframe-embedding-setup/helpers/index.ts**

```
import { match } from "ts-pattern";  
  
import { entityPickerModal } from "e2e/support/helpers";  
import type { Dashboard, RecentItem } from "metabase-types/api";  
  
type RecentActivityIntercept = {  
  ...
```

### **e2e/test/scenarios/embedding/sdk-iframe-embedding-setup/select-embed-entity .cy.spec.ts**

```
import {  
  ORDERS_BY_YEAR_QUESTION_ID,  
  ORDERS_COUNT_QUESTION_ID,  
  ORDERS_DASHBOARD_ID,  
} from "e2e/support/cypress_sample_instance_data";  
import { mockEmbedJsToDevServer } from...
```

### **e2e/test/scenarios/embedding/sdk-iframe-embedding-setup/select-embed-exper ience.cy.spec.ts**

```
import { ORDERS_COUNT_QUESTION_ID } from "e2e/support/cypress_sample_instance_data";  
  
import {  
  assertDashboard,  
  assertRecentItemName,  
  getEmbedSidebar,  
  visitNewEmbedPage,  
} from...
```

### **e2e/test/scenarios/embedding/sdk-iframe-embedding-setup/select-embed-optio ns.cy.spec.ts**

```
import { mockEmbedJsToDevServer } from "e2e/support/helpers";
```

## Analyst Base

```
import {
  codeBlock,
  getEmbedSidebar,
  navigateToEmbedOptionsStep,
} from "../helpers";
```

```
const { H } = cy;
```

```
const DASHBOARD_NAME =...
```

### **e2e/test/scenarios/embedding/sdk-iframe-embedding-setup/user-settings-persistence.cy.spec.ts**

```
import { enableJwtAuth } from "e2e/support/helpers/e2e-jwt-helpers";
```

```
import {
  codeBlock,
  getEmbedSidebar,
  navigateToEmbedOptionsStep,
  navigateToGetCodeStep,
} from "../helpers";
```

```
const { H }...
```

### **e2e/test/scenarios/embedding/shared/embedding-dashboard.js**

```
import { produce } from "immer";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, PEOPLE } = SAMPLE_DATABASE;
```

```
export const questionDetails = {
  native: {
    ...
  }
}
```

### **e2e/test/scenarios/embedding/shared/embedding-linked-filters.js**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PEOPLE, PRODUCTS, PRODUCTS_ID } = SAMPLE_DATABASE;
```

```
export const nativeQuestionDetails = {
  name: "Count of People by..."
}
```

### **e2e/test/scenarios/embedding/shared/embedding-native.js**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

## Analyst Base

```
const { ORDERS, PEOPLE } = SAMPLE_DATABASE;
```

```
const query = `
SELECT orders.id, orders.product_id, orders.created_at AS...
```

### **e2e/test/scenarios/embedding/shared/embedding-questions.js**

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, PEOPLE, PEOPLE_ID } = SAMPLE_DATABASE;
```

```
export const regularQuestion = {
  name: "Orders4t#7 t3",
  ...
```

### **e2e/test/scenarios/embedding/shared/embedding-snippets.js**

```
export const getEmbeddingJsCode = ({
  type,
  id,
  downloads,
  theme,
  // Match the actual default value (metabase#43838)
  background = true,
}) => {
  return new RegExp(
    `// you will need...
```

### **e2e/test/scenarios/embedding-sdk/static-dashboard-cors.cy.spec.js**

```
const { H } = cy;
import { USERS } from "e2e/support/cypress_data";
import {
  ORDERS_DASHBOARD_DASHCARD_ID,
  ORDERS_QUESTION_ID,
} from "e2e/support/cypress_sample_instance_data";
import {...
```

### **e2e/test/scenarios/filters/filter-bigint.cy.spec.ts**

```
import type { Sheet } from "xlsx";

import { WRITABLE_DB_ID } from "e2e/support/cypress_data";
import type {
  DashboardDetails,
  NativeQuestionDetails,
  StructuredQuestionDetails,
```

## Analyst Base

```
} from...
```

### **e2e/test/scenarios/filters/filter-bulk.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS_ID, ORDERS, PEOPLE_ID, PRODUCTS_ID,...
```

### **e2e/test/scenarios/filters/filter-sources.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS_ID, PRODUCTS, ORDERS_ID, ORDERS } =...
```

### **e2e/test/scenarios/filters/filter-types.cy.spec.js**

```
const { H } = cy;

const STRING_CASES = [
  {
    title: "is",
    columnName: "Category",
    operator: "Is",
    options: ["Widget"],
    expectedDisplayName: "Category is Widget",
    ...
  }
]
```

### **e2e/test/scenarios/filters/filter.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, PRODUCTS, PRODUCTS_ID,...
```

### **e2e/test/scenarios/filters/operators.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS_ID, PEOPLE_ID } = SAMPLE_DATABASE;
```

```
describe("operators in questions", () => {
```

```
...
```

### **e2e/test/scenarios/filters/relative-datetime.cy.spec.js**

## Analyst Base

```
import dayjs from "dayjs";
```

```
import "metabase/lib/dayjs";
```

```
const { H } = cy;
```

```
const STARTING_FROM_UNITS = [  
  "minutes",  
  "hours",  
  "days",  
  "weeks",  
  "months",  
  "quarters",  
  ...  
];
```

### **e2e/test/scenarios/filters/time-series-chrome.cy.spec.ts**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS, PRODUCTS_ID } = ...
```

### **e2e/test/scenarios/filters/view.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS } = SAMPLE_DATABASE;
```

```
describe("scenarios > question > view", () => {  
  beforeEach(() => {  
    ...  
  })  
})
```

### **e2e/test/scenarios/filters-reproductions/dashboard-filters-reproductions.cy.spec.js**

```
import dayjs from "dayjs";
```

```
const { H } = cy;
```

```
import {  
  SAMPLE_DB_ID,  
  SAMPLE_DB_SCHEMA_ID,  
  USER_GROUPS,  
  WRITABLE_DB_ID,  
} from "e2e/support/cypress_data";
```



## Analyst Base

```
import { SAMPLE_DATABASE } from...
```

### **e2e/test/scenarios/filters-reproductions/dashboard-filters-with-question-revert.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { REVIEWS, REVIEWS_ID } =...
```

### **e2e/test/scenarios/filters-reproductions/filters-reproductions.cy.spec.js**

```
const { H } = cy;
```

```
import {
```

```
  SAMPLE_DB_ID,
```

```
  SAMPLE_DB_SCHEMA_ID,
```

```
  WRITABLE_DB_ID,
```

```
} from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const...
```

### **e2e/test/scenarios/i18n/i18n.cy.spec.ts**

```
const { H } = cy;
```

```
const paths = [
```

```
  "/",
```

```
  "/getting-started",
```

```
  "/collection/root",
```

```
  "/browse/models",
```

```
  "/browse/databases",
```

```
  "/browse/metrics",
```

```
  "/trash",
```

```
  "/admin",
```

```
];
```

```
const locales = [
```

```
  ...
```

### **e2e/test/scenarios/joins/joins-custom-expressions.cy.spec.ts**

```
import { ORDERS_MODEL_ID } from "e2e/support/cypress_sample_instance_data";
```

```
const { H } = cy;
```

## Analyst Base

```
type TestCase = {  
  operator: string;  
  lhsExpression: string;  
  rhsExpression: string;  
  ...  
}
```

### **e2e/test/scenarios/joins/joins-reproductions.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const {  
  ORDERS,  
  ORDERS_ID,  
  PRODUCTS,  
  ...  
}
```

### **e2e/test/scenarios/joins/joins.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, PRODUCTS, PRODUCTS_ID } =...
```

### **e2e/test/scenarios/maildev-keys/README.md**

# Files used by docker-compose maildev-ssl

### **e2e/test/scenarios/metabot/metabot.cy.spec.ts**

```
const { H } = cy;  
  
const loremIpsum =  
  "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer auctor id erat non sollicitudin. ";  
  
describe("Metabot UI", () => {  
  beforeEach(() => {  
    ...  
  })  
})
```

### **e2e/test/scenarios/metrics/browse.cy.spec.ts**

```
const { H } = cy;  
import { USERS } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  FIRST_COLLECTION_ID,  
  ORDERS_MODEL_ID,  
  ...  
}
```

## Analyst Base

```
} from...
```

### **e2e/test/scenarios/metrics/metrics-collection.cy.spec.js**

```
const { H } = cy;
import { USER_GROUPS } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  FIRST_COLLECTION_ID,
  ORDERS_MODEL_ID,
}...
```

### **e2e/test/scenarios/metrics/metrics-dashboard.cy.spec.js**

```
const { H } = cy;
import { USER_GROUPS } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  FIRST_COLLECTION_ID,
  ...
```

### **e2e/test/scenarios/metrics/metrics-editing.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ORDERS_MODEL_ID } from "e2e/support/cypress_sample_instance_data";

const { ORDERS_ID, ORDERS,...
```

### **e2e/test/scenarios/metrics/metrics-question.cy.spec.js**

```
const { H } = cy;
import { USER_GROUPS } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  FIRST_COLLECTION_ID,
  ORDERS_MODEL_ID,
}...
```

### **e2e/test/scenarios/metrics/metrics-search.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { ORDERS_ID } = SAMPLE_DATABASE;

const ORDERS_SCALAR_METRIC = {
  name: "Count of orders",
  type:...
```

## Analyst Base

### **e2e/test/scenarios/metrics/reproductions/metrics-reproductions.cy.spec.ts**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import type { StructuredQuestionDetails } from "e2e/support/helpers";
```

```
const { ORDERS_ID, ORDERS } =...
```

### **e2e/test/scenarios/models/create.cy.spec.js**

```
const { H } = cy;
import { USERS } from "e2e/support/cypress_data";
```

```
describe("scenarios > models > create", () => {
  beforeEach(() => {
    H.restore();
    cy.signInAsAdmin();
    ...
  });
});
```

### **e2e/test/scenarios/models/helpers/e2e-models-helpers.js**

```
import {
  NativeEditor,
  entityPickerModal,
  entityPickerModalTab,
  interceptIfNotPreviouslyDefined,
  modal,
  openQuestionActions,
  popover,
} from "e2e/support/helpers";
```

```
export function...
```

### **e2e/test/scenarios/models/model-indexes.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { createModelIndex } from "e2e/support/helpers/e2e-model-index-helper";
```

```
const { PRODUCTS_ID, PEOPLE_ID...
```

### **e2e/test/scenarios/models/models-metadata.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import { startQuestionFromModel } from "../helpers/e2e-models-helpers";
```

## Analyst Base

```
const { PEOPLE, PRODUCTS,...
```

### **e2e/test/scenarios/models/models-query-editor.cy.spec.js**

```
const { H } = cy;  
import { ORDERS_QUESTION_ID } from "e2e/support/cypress_sample_instance_data";
```

```
import { selectFromDropdown } from "../helpers/e2e-models-helpers";
```

```
describe("scenarios > models...
```

### **e2e/test/scenarios/models/models-revision-history.cy.spec.js**

```
const { H } = cy;  
import { ORDERS_BY_YEAR_QUESTION_ID } from "e2e/support/cypress_sample_instance_data";
```

```
describe("scenarios > models > revision history", () => {  
  beforeEach(() => {  
    ...
```

### **e2e/test/scenarios/models/models-with-aggregation-and-breakout.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import { turnIntoModel } from "../helpers/e2e-models-helpers";  
const { ORDERS, ORDERS_ID } =...
```

### **e2e/test/scenarios/models/models.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ORDERS_BY_YEAR_QUESTION_ID,  
  ...
```

### **e2e/test/scenarios/models/reproductions.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, SAMPLE_DB_SCHEMA_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ORDERS_DASHBOARD_ID,  
  ...
```

### **e2e/test/scenarios/models/reproductions.cy.spec.ts**

```
const { H } = cy;  
  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

## Analyst Base

```
import {
  FIRST_COLLECTION_ID,
  ORDERS_DASHBOARD_ID,
  ORDERS_MODEL_ID,
  ORDERS_QUESTION_ID,
} from...
```

### **e2e/test/scenarios/native/ai-sql-fixer.cy.spec.ts**

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";

const { H } = cy;

const FIX_MESSAGE = "Fixes applied. Run your query to view results.";

describe.skip("scenarios > native > ai sql fixer",...
```

### **e2e/test/scenarios/native/native-database-source.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";

const PG_DB_ID = 2;
const mongoName = "QA Mongo";
const postgresName = "QA Postgres12";
const additionalPG =...
```

### **e2e/test/scenarios/native/native-reproductions.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

import {
  getRunQueryButton,
  ...
```

### **e2e/test/scenarios/native/native-reproductions.cy.spec.ts**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import type {
  NativeQuestionDetails,
  ...
```

### **e2e/test/scenarios/native/native.cy.spec.js**

```
const { H } = cy;
```

## Analyst Base

```
import {
  SAMPLE_DB_ID,
  USER_GROUPS,
  WRITABLE_DB_ID,
} from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {...
```

### **e2e/test/scenarios/native/native\_subquery.cy.spec.js**

```
const { H } = cy;
import {
  ADMIN_PERSONAL_COLLECTION_ID,
  ORDERS_QUESTION_ID,
} from "e2e/support/cypress_sample_instance_data";

describe("scenarios > question > native subquery", () => {
  ...
```

### **e2e/test/scenarios/native/snippets.cy.spec.js**

```
const { H } = cy;
import { USER_GROUPS } from "e2e/support/cypress_data";

const { ALL_USERS_GROUP } = USER_GROUPS;

// HACK which lets us type (even very long words) without losing focus
// this is...
```

### **e2e/test/scenarios/native/suggestions.cy.spec.ts**

```
const { H } = cy;

describe("scenarios > question > native > suggestions", () => {
  beforeEach(() => {
    H.restore();
    cy.signInAsNormalUser();
  });

  it("should show suggestions for tables",...
```

### **e2e/test/scenarios/native-filters/helpers/e2e-date-filter-helpers.js**

```
import { popover, selectDropdown } from "e2e/support/helpers";

const currentYearString = new Date().getFullYear().toString();

export function setMonthAndYear({ month, year } = {}) {
```

...

### **e2e/test/scenarios/native-filters/helpers/e2e-field-filter-data-objects.js**

```
import dayjs from "dayjs";
```

```
export const STRING_FILTER_SUBTYPES = {  
  String: {  
    searchTerm: "Synerg",  
    value: "Synergistic Granite Chair",  
    representativeResult: "Synergistic Granite..."
```

### **e2e/test/scenarios/native-filters/helpers/e2e-field-filter-helpers.js**

```
import { filterWidget, popover, selectDropdown } from "e2e/support/helpers";
```

```
// FILTER WIDGET TYPE
```

```
/**  
 * Sets a field filter widget type. Depends on the field that field filter is mapped to.  
 *  
 * ...
```

### **e2e/test/scenarios/native-filters/helpers/e2e-sql-filter-helpers.js**

```
import {  
  NativeEditor,  
  filterWidget,  
  selectDropdown,  
} from "e2e/support/helpers";
```

```
// FILTER TYPES
```

```
/**  
 * Opens popover with the list of possible SQL filter types to choose from.  
 * It does...
```

### **e2e/test/scenarios/native-filters/native-filters-remapping.cy.spec.ts**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import type { NativeQuestionDetails } from "e2e/support/helpers";  
import type { GetFieldValuesResponse } from...
```

### **e2e/test/scenarios/native-filters/native-filters-reproductions.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";
```



## Analyst Base

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import * as FieldFilter from...
```

### **e2e/test/scenarios/native-filters/sql-field-filter-types.cy.spec.js**

```
const { H } = cy;
```

```
import { WRITABLE_DB_ID } from "e2e/support/cypress_data";
```

```
import * as DateFilter from "../helpers/e2e-date-filter-helpers";
```

```
import {  
  DATE_FILTER_SUBTYPES,  
  ...
```

### **e2e/test/scenarios/native-filters/sql-field-filter.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import * as FieldFilter from "../helpers/e2e-field-filter-helpers";
```

```
import * as SQLFilter from...
```

### **e2e/test/scenarios/native-filters/sql-filters-reset-clear.cy.spec.ts**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import type { TemplateTag } from "metabase-types/api";
```

```
type SectionId =  
  | "no_default_non_required"  
  | ...
```

### **e2e/test/scenarios/native-filters/sql-filters-source.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import * as FieldFilter from...
```

### **e2e/test/scenarios/native-filters/sql-filters.cy.spec.js**

```
const { H } = cy;
```

```
import * as DateFilter from "../helpers/e2e-date-filter-helpers";
```

```
import * as SQLFilter from "../helpers/e2e-sql-filter-helpers";
```

```
describe("scenarios > filters > sql filters > basic...
```

## Analyst Base

### **e2e/test/scenarios/navigation/navbar.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ORDERS_DASHBOARD_ID,
  THIRD_COLLECTION_ID,
} from...
```

### **e2e/test/scenarios/onboarding/about.cy.spec.js**

```
const { H } = cy;

describe("scenarios > about Metabase", () => {
  beforeEach(() => {
    H.restore();
    cy.signInAsAdmin();

    cy.visit("/");
    cy.icon("gear").click();
    //...
```

### **e2e/test/scenarios/onboarding/add-initial-data.cy.spec.ts**

```
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";
import {
  FIRST_COLLECTION_ID,
  SECOND_COLLECTION_ID,
} from "e2e/support/cypress_sample_instance_data";

const { H } =...
```

### **e2e/test/scenarios/onboarding/auth/forgot\_password.cy.spec.js**

```
const { H } = cy;
import { USERS } from "e2e/support/cypress_data";

const { admin } = USERS;

describe("scenarios > auth > password", { tags: "@external" }, () => {
  beforeEach(() => {
    ...
```

### **e2e/test/scenarios/onboarding/auth/signin.cy.spec.js**

```
const { H } = cy;
import { USERS } from "e2e/support/cypress_data";
```

## Analyst Base

```
const sizes = [  
  [1280, 800],  
  [640, 360],  
];  
const { admin } = USERS;
```

```
describe("scenarios > auth > signin", () => {  
  ...
```

### **e2e/test/scenarios/onboarding/auth/sso.cy.spec.js**

```
const { H } = cy;  
import { USERS } from "e2e/support/cypress_data";
```

```
const { admin } = USERS;
```

```
describe("scenarios > auth > signin > SSO", () => {  
  beforeEach(() => {  
    H.restore();  
    ...
```

### **e2e/test/scenarios/onboarding/command-palette.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, USERS } from "e2e/support/cypress_data";  
import {  
  ADMIN_PERSONAL_COLLECTION_ID,  
  ORDERS_BY_YEAR_QUESTION_ID,  
  ORDERS_COUNT_QUESTION_ID,  
  ...
```

### **e2e/test/scenarios/onboarding/embedding-homepage.cy.spec.ts**

```
const { H } = cy;  
  
H.describeWithSnowplow(  
  "scenarios > embedding-homepage > snowplow events",  
  () => {  
    beforeEach(() => {  
      H.restore("default");  
      H.resetSnowplow();  
  
      ...
```

### **e2e/test/scenarios/onboarding/home/browse.cy.spec.ts**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

## Analyst Base

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ORDERS_MODEL_ID,
  ORDERS_QUESTION_ID,
}...
```

### **e2e/test/scenarios/onboarding/home/homepage.cy.spec.js**

```
const { H } = cy;
import { USERS } from "e2e/support/cypress_data";
import {
  ADMIN_PERSONAL_COLLECTION_ID,
  ORDERS_BY_YEAR_QUESTION_ID,
  ORDERS_DASHBOARD_ID,
} from...
```

### **e2e/test/scenarios/onboarding/navbar/new-menu.cy.spec.js**

```
const { H } = cy;

describe("metabase > scenarios > navbar > new menu", () => {
  beforeEach(() => {
    H.restore();
    cy.signInAsAdmin();

    cy.visit("/");
    // eslint-disable-next-line...
```

### **e2e/test/scenarios/onboarding/navbar/whats-new.cy.spec.js**

```
const { H } = cy;
import {
  createMockVersionInfo,
  createMockVersionInfoRecord as mockVersion,
} from "metabase-types/api/mocks";
```

```
describe("nav > what's new notification", () => {
  beforeEach(()...
```

### **e2e/test/scenarios/onboarding/notifications.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { ORDERS_ID } = SAMPLE_DATABASE;

const getQuestionDetails = () => ({
  name: "Question",
```

## Analyst Base

```
query: {
```

```
...
```

### **e2e/test/scenarios/onboarding/onboarding-checklist.cy.spec.ts**

```
const { H } = cy;
```

```
import type { ChecklistItemValue } from "metabase/home/components/Onboarding/types";
```

```
describe("Onboarding checklist page", () => {
```

```
  beforeEach(() => {
```

```
    H.restore();
```

```
...
```

### **e2e/test/scenarios/onboarding/reference/databases.cy.spec.js**

```
const { H } = cy;
```

```
describe("scenarios > reference > databases", () => {
```

```
  beforeEach(() => {
```

```
    H.restore();
```

```
    cy.signInAsAdmin();
```

```
  });
```

```
  it("should see the listing", () => {
```

```
    ...
```

### **e2e/test/scenarios/onboarding/reference/reproductions/5276-remove-field-type.cy.spec.js**

```
const { H } = cy;
```

```
describe("issue 5276", () => {
```

```
  beforeEach(() => {
```

```
    H.restore();
```

```
    cy.signInAsAdmin();
```

```
    cy.intercept("PUT", "/api/field/*").as("updateField");
```

```
  });
```

```
  it("should allow...
```

### **e2e/test/scenarios/onboarding/setup/setup-embedding.cy.spec.ts**

```
const { H } = cy;
```

```
H.describeWithSnowplowEE("scenarios > setup embedding (EMB-477)", () => {
```

```
  beforeEach(() => {
```

```
    H.resetSnowplow();
```

```
    H.restore("blank");
```

## Analyst Base

```
});
```

```
it("should redirect correctly...
```

### **e2e/test/scenarios/onboarding/setup/setup.cy.spec.ts**

```
const { H } = cy;  
const { IS_ENTERPRISE } = Cypress.env();  
import { USERS } from "e2e/support/cypress_data";  
import { SUBSCRIBE_URL } from "metabase/setup/constants";  
  
const { admin } = USERS;
```

```
//...
```

### **e2e/test/scenarios/onboarding/setup/user\_settings.cy.spec.js**

```
const { H } = cy;  
import { USERS } from "e2e/support/cypress_data";  
import { NORMAL_USER_ID } from "e2e/support/cypress_sample_instance_data";  
  
const { normal } = USERS;
```

```
const { first_name,...
```

### **e2e/test/scenarios/onboarding/urls.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, USERS } from "e2e/support/cypress_data";  
import {  
  ADMIN_PERSONAL_COLLECTION_ID,  
  FIRST_COLLECTION_ID,  
  NORMAL_PERSONAL_COLLECTION_ID,  
  ...
```

### **e2e/test/scenarios/organization/bookmarks-collection.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_TABLES, USERS } from "e2e/support/cypress_data";  
import {  
  ADMIN_PERSONAL_COLLECTION_ID,  
  FIRST_COLLECTION_ID,  
} from...
```

### **e2e/test/scenarios/organization/bookmarks-dashboard.cy.spec.js**

```
const { H } = cy;  
import { ORDERS_DASHBOARD_ID } from "e2e/support/cypress_sample_instance_data";
```

## Analyst Base

```
describe("scenarios > dashboard > bookmarks", () => {  
  beforeEach(() => {  
    H.restore();  
    ...  
  })  
})
```

### **e2e/test/scenarios/organization/bookmarks-question.cy.spec.js**

```
const { H } = cy;  
import { ORDERS_QUESTION_ID } from "e2e/support/cypress_sample_instance_data";  
  
import { toggleQuestionBookmarkStatus } from "./helpers/bookmark-helpers";
```

```
describe("scenarios >...
```

### **e2e/test/scenarios/organization/bookmarks-reordering.cy.spec.ts**

```
const { H } = cy;  
import {  
  ORDERS_COUNT_QUESTION_ID,  
  ORDERS_QUESTION_ID,  
} from "e2e/support/cypress_sample_instance_data";
```

```
import {  
  createAndBookmarkQuestion,  
  moveBookmark,  
  ...  
}
```

### **e2e/test/scenarios/organization/content-verification.cy.spec.js**

```
const { H } = cy;  
import {  
  ORDERS_COUNT_QUESTION_ID,  
  ORDERS_DASHBOARD_ID,  
} from "e2e/support/cypress_sample_instance_data";
```

```
describe(  
  "scenarios > premium > content verification",  
  { tags:...
```

### **e2e/test/scenarios/organization/edit-history-metadata.cy.spec.js**

```
const { H } = cy;  
import { USERS } from "e2e/support/cypress_data";  
import {  
  ORDERS_DASHBOARD_ID,  
  ORDERS_QUESTION_ID,  
} from "e2e/support/cypress_sample_instance_data";
```

## Analyst Base

```
describe("scenarios >...
```

### **e2e/test/scenarios/organization/entity-picker.cy.spec.ts**

```
const { H } = cy;  
import { USER_GROUPS, WRITABLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ...
```

### **e2e/test/scenarios/organization/helpers/bookmark-helpers.ts**

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  createQuestion,  
  moveDnDKitElement,  
  navigationSidebar,  
}...
```

### **e2e/test/scenarios/organization/official-collections.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
  
const { ORDERS, ORDERS_ID } = SAMPLE_DATABASE;  
  
const COLLECTION_NAME = "Official Collection Test";  
  
const...
```

### **e2e/test/scenarios/organization/timelines-collection.cy.spec.js**

```
const { H } = cy;  
import { USERS } from "e2e/support/cypress_data";  
import {  
  ADMIN_PERSONAL_COLLECTION_ID,  
  FIRST_COLLECTION_ID,  
} from "e2e/support/cypress_sample_instance_data";  
  
const { admin }...
```

### **e2e/test/scenarios/organization/timelines-question.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import { ORDERS_BY_YEAR_QUESTION_ID } from...
```

### **e2e/test/scenarios/permissions/admin-permissions.cy.spec.js**



## Analyst Base

```
const { H } = cy;
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";
import {
  ORDERS_DASHBOARD_ID,
  ORDERS_QUESTION_ID,
} from "e2e/support/cypress_sample_instance_data";
```

const...

### **e2e/test/scenarios/permissions/application-permissions.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ORDERS_DASHBOARD_ID,
  ORDERS_QUESTION_ID,
} from...
```

### **e2e/test/scenarios/permissions/create-queries.cy.spec.js**

```
const { H } = cy;
import { USER_GROUPS } from "e2e/support/cypress_data";
```

```
const { ALL_USERS_GROUP } = USER_GROUPS;
```

```
const NATIVE_QUERIES_PERMISSION_INDEX = 0;
```

```
describe("scenarios > admin >...
```

### **e2e/test/scenarios/permissions/data-model-permissions.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID, SAMPLE_DB_SCHEMA_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID } =...
```

### **e2e/test/scenarios/permissions/database-details-permissions.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
const DETAILS_PERMISSION_INDEX = 4;
```

```
describe("scenarios > admin > permissions > database details permissions", () => {
```

```
...
```

### **e2e/test/scenarios/permissions/downgrade-ee-to-oss.cy.spec.js**

## Analyst Base

```
const { H } = cy;  
import { USER_GROUPS } from "e2e/support/cypress_data";
```

```
const { ALL_USERS_GROUP } = USER_GROUPS;
```

```
const EE_DATA_ACCESS_PERMISSION_INDEX = 0;
```

```
const...
```

### **e2e/test/scenarios/permissions/download-permissions.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ORDERS_DASHBOARD_ID,  
  ...  
}
```

### **e2e/test/scenarios/permissions/impersonated.cy.spec.js**

```
const { H } = cy;  
import { USER_GROUPS } from "e2e/support/cypress_data";  
import * as PH from "e2e/test/scenarios/admin/performance/helpers/e2e-strategy-form-helpers";
```

```
const { ALL_USERS_GROUP,...
```

### **e2e/test/scenarios/permissions/permissions-baseline.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import {  
  ADMIN_PERSONAL_COLLECTION_ID,  
  ORDERS_DASHBOARD_ID,  
  ORDERS_QUESTION_ID,  
} from...
```

### **e2e/test/scenarios/permissions/permissions-reproductions.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, USERS, USER_GROUPS } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  NODATA_USER_ID,  
  ...  
}
```

### **e2e/test/scenarios/permissions/permissions-reproductions.cy.spec.ts**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

## Analyst Base

```
import type {  
  ConcreteFieldReference,  
  ...  
}
```

### **e2e/test/scenarios/permissions/sandboxing/helpers/e2e-sandboxing-helpers.ts**

```
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import type { StructuredQuestionDetails } from...
```

### **e2e/test/scenarios/permissions/sandboxing/helpers/types.ts**

```
import type { Dataset } from "metabase-types/api";
```

```
export type DatasetResponse = {  
  body: Dataset;  
  url: string;  
  headers: any;  
  statusCode: number;  
};
```

```
export type DashcardQueryResponse = {  
  ...  
}
```

### **e2e/test/scenarios/permissions/sandboxing/sandboxing-misconfiguration.cy.spec.ts**

```
import { USER_GROUPS, WRITABLE_DB_ID } from "e2e/support/cypress_data";
```

```
import {  
  assertResponseFailsClosed,  
  assignAttributeToUser,  
  configureSandboxPolicy,  
  getCardResponses,  
  gizmoViewer,  
  ...  
}
```

### **e2e/test/scenarios/permissions/sandboxing/sandboxing-via-api.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, USERS, USER_GROUPS } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  NORMAL_USER_ID,  
  ...  
}
```

### **e2e/test/scenarios/permissions/sandboxing/sandboxing-via-ui.cy.spec.ts**

```
import { USER_GROUPS } from "e2e/support/cypress_data";
```

## Analyst Base

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { checkNotNull } from "metabase/lib/types";
import type {...
```

### **e2e/test/scenarios/permissions/view-data.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ORDERS_QUESTION_ID } from...
```

### **e2e/test/scenarios/question/caching.cy.spec.js**

```
const { H } = cy;
import { ORDERS_QUESTION_ID } from "e2e/support/cypress_sample_instance_data";
```

```
import { interceptPerformanceRoutes } from...
```

### **e2e/test/scenarios/question/column-compare.cy.spec.ts**

```
import _ from "underscore";

const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import type {...
```

### **e2e/test/scenarios/question/document-title.cy.spec.js**

```
const { H } = cy;

const PG_DB_ID = 2;

describe(
  "question loading changes document title",
  { tags: "@external" },
  () => {
    beforeEach(() => {
      H.restore("postgres-12");
    })
    ...
  })
```

### **e2e/test/scenarios/question/multiple-column-breakouts.cy.spec.ts**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import type {
  DashboardDetails,
  StructuredQuestionDetails,
} from "e2e/support/helpers";
```

## Analyst Base

```
const {...
```

### **e2e/test/scenarios/question/native-query-drill.cy.spec.ts**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import type { NativeQuestionDetails } from "e2e/support/helpers";
```

```
const ordersTableQuestionDetails: NativeQuestionDetails =...
```

### **e2e/test/scenarios/question/nested.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, PRODUCTS, PRODUCTS_ID,...
```

### **e2e/test/scenarios/question/new.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, USERS } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ORDERS_COUNT_QUESTION_ID,  
  ...
```

### **e2e/test/scenarios/question/notebook-data-source.cy.spec.ts**

```
const { H } = cy;  
import { WRITABLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ORDERS_COUNT_QUESTION_ID,  
  ...
```

### **e2e/test/scenarios/question/notebook-link-to-data-source.cy.spec.ts**

```
const { H } = cy;  
import { SAMPLE_DB_ID, USERS } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import {  
  ADMIN_PERSONAL_COLLECTION_ID,  
  ...
```

### **e2e/test/scenarios/question/notebook-native-preview-sidebar.cy.spec.ts**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

## Analyst Base

```
import {  
  ORDERS_COUNT_QUESTION_ID,  
  ...
```

### **e2e/test/scenarios/question/notebook.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, WRITABLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import { ADMIN_USER_ID } from...
```

### **e2e/test/scenarios/question/nulls.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID } = SAMPLE_DATABASE;
```

```
describe("scenarios > question > null", () => {  
  ...
```

### **e2e/test/scenarios/question/offset.cy.spec.ts**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import type { StructuredQuestionDetails } from "e2e/support/helpers";  
import { uuid } from...
```

### **e2e/test/scenarios/question/query-external.cy.spec.js**

```
const { H } = cy;  
import { WRITABLE_DB_ID } from "e2e/support/cypress_data";
```

```
const supportedDatabases = [  
  {  
    database: "Mongo",  
    snapshotName: "mongo-5",  
    dbName: "QA Mongo",  
  },  
  {  
    ...
```

### **e2e/test/scenarios/question/question-analytics.cy.spec.js**

```
const { H } = cy;
```

```
H.describeWithSnowplow("scenarios > question > snowplow", () => {  
  describe("chart_generated", () => {  
    const generateNonTableVisualization = () => {
```

## Analyst Base

```
cy.visit("/");
```

```
...
```

### **e2e/test/scenarios/question/question-management.cy.spec.js**

```
import { onlyOn } from "@cypress/skip-test";
```

```
import { USERS, USER_GROUPS } from "e2e/support/cypress_data";
```

```
import {
```

```
  ORDERS_COUNT_QUESTION_ID,
```

```
  ORDERS_DASHBOARD_ID,
```

```
  ORDERS_QUESTION_ID,
```

```
} from...
```

### **e2e/test/scenarios/question/questions-entity-id.cy.spec.ts**

```
const { H } = cy;
```

```
import {
```

```
  ORDERS_QUESTION_ENTITY_ID,
```

```
  ORDERS_QUESTION_ID,
```

```
} from "e2e/support/cypress_sample_instance_data";
```

```
describe("scenarios > questions > entity id support", () => {
```

```
  ...
```

### **e2e/test/scenarios/question/saved.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import {
```

```
  ORDERS_BY_YEAR_QUESTION_ID,
```

```
  ORDERS_COUNT_QUESTION_ID,
```

```
  ORDERS_DASHBOARD_ID,
```

```
...
```

### **e2e/test/scenarios/question/settings.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, PRODUCTS, PRODUCTS_ID } =...
```

### **e2e/test/scenarios/question/summarization.cy.spec.js**

```
const { H } = cy;
```

```
import { dedent } from "ts-dedent";
```

## Analyst Base

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ORDERS_QUESTION_ID } from...
```

### **e2e/test/scenarios/question-reproductions/reproductions-1.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID, WRITABLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ORDERS_QUESTION_ID } from...
```

### **e2e/test/scenarios/question-reproductions/reproductions-2.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ORDERS_QUESTION_ID } from...
```

### **e2e/test/scenarios/question-reproductions/reproductions-3.cy.spec.js**

```
const { H } = cy;
import {
  SAMPLE_DB_ID,
  SAMPLE_DB_SCHEMA_ID,
  USER_GROUPS,
  WRITABLE_DB_ID,
} from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from...
```

### **e2e/test/scenarios/question-reproductions/reproductions.cy.spec.ts**

```
const { H } = cy;

import { WRITABLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ORDERS_QUESTION_ID } from...
```

### **e2e/test/scenarios/search/recently-viewed.cy.spec.js**

```
const { H } = cy;
import {
  ORDERS_DASHBOARD_ID,
  ORDERS_QUESTION_ID,
} from "e2e/support/cypress_sample_instance_data";
```

```
describe("search > recently viewed", () => {
  beforeEach(() => {
    ...
```

### **e2e/test/scenarios/search/search-filters.cy.spec.js**

```
const { H } = cy;
```



## Analyst Base

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ADMIN_USER_ID,
  FIRST_COLLECTION_ID,
  ...
```

### **e2e/test/scenarios/search/search-pagination.cy.spec.js**

```
import _ from "underscore";

const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { ORDERS, ORDERS_ID } = SAMPLE_DATABASE;

const PAGE_SIZE = 50;
const...
```

### **e2e/test/scenarios/search/search-snowplow.cy.spec.js**

```
const { H } = cy;

import { commandPaletteInput } from "../../support/helpers/e2e-command-palette-helpers";

H.describeWithSnowplow("scenarios > search > snowplow", () => {
  const...
```

### **e2e/test/scenarios/search/search-typeahead.cy.spec.js**

```
const { H } = cy;
import { USERS } from "e2e/support/cypress_data";

["admin", "normal"].forEach((user) => {
  describe(`search > ${user} user`, () => {
    beforeEach(() => {
      H.restore();
    })
    ...
```

### **e2e/test/scenarios/search/search.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ORDERS_DASHBOARD_ID,
  ORDERS_QUESTION_ID,
} from...
```

### **e2e/test/scenarios/sharing/alert/alert-permissions.cy.spec.js**

## Analyst Base

```
const { H } = cy;
import { USERS } from "e2e/support/cypress_data";
import {
  ORDERS_BY_YEAR_QUESTION_ID,
  ORDERS_COUNT_QUESTION_ID,
  ORDERS_QUESTION_ID,
} from...
```

### **e2e/test/scenarios/sharing/alert/alert-types.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ORDERS_BY_YEAR_QUESTION_ID,
  ORDERS_QUESTION_ID,
} from...
```

### **e2e/test/scenarios/sharing/alert/alert.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ORDERS_DASHBOARD_ID,
  ORDERS_MODEL_ID,
  ORDERS_QUESTION_ID,
} from...
```

### **e2e/test/scenarios/sharing/alert/email-alert.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ORDERS_QUESTION_ID } from "e2e/support/cypress_sample_instance_data";

const { PEOPLE_ID } =...
```

### **e2e/test/scenarios/sharing/downloads/downloads.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ORDERS_DASHBOARD_DASHCARD_ID,
  ORDERS_DASHBOARD_ID,
  ORDERS_QUESTION_ID,
} from...
```

### **e2e/test/scenarios/sharing/downloads/sharing-download-reproductions.cy.spec.js**

```
const { H } = cy;
```

## Analyst Base

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, REVIEWS, REVIEWS_ID,...
```

### **e2e/test/scenarios/sharing/public-dashboard.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS, ORDERS_ID } = SAMPLE_DATABASE;
```

```
const questionDetails = {
  name: "sql param",
  native: {
```

```
...
```

### **e2e/test/scenarios/sharing/public-question.cy.spec.js**

```
import xlsx from "xlsx";
```

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PEOPLE } = SAMPLE_DATABASE;
```

```
const questionData = {
  name: "Parameterized...
```

### **e2e/test/scenarios/sharing/public-resource-downloads.cy.spec.ts**

```
const { H } = cy;
import {
  ORDERS_BY_YEAR_QUESTION_ID,
  ORDERS_DASHBOARD_DASHCARD_ID,
  ORDERS_DASHBOARD_ID,
} from "e2e/support/cypress_sample_instance_data";
```

```
/** These tests are about the...
```

### **e2e/test/scenarios/sharing/public-sharing-embed-button-behavior.cy.spec.js**

```
const { H } = cy;
```

```
["dashboard", "question"].forEach((resource) => {
  describe(`embed modal behavior for ${resource}s`, () => {
    beforeEach(() => {
```

## Analyst Base

```
H.restore();
cy.signInAsAdmin();
```

...

### **e2e/test/scenarios/sharing/public-sharing.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { ORDERS_ID } = SAMPLE_DATABASE;
```

const...

### **e2e/test/scenarios/sharing/sharing-reproductions.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID, USERS, WEBMAIL_CONFIG } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ADMIN_USER_ID,
  ...
```

### **e2e/test/scenarios/sharing/subscriptions.cy.spec.js**

```
const { H } = cy;
import { USERS } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ORDERS_DASHBOARD_ID } from...
```

### **e2e/test/scenarios/stats/instance-stats-snowplow.cy.spec.js**

```
const { H } = cy;

H.describeWithSnowplow("scenarios > stats > snowplow", () => {
  beforeEach(() => {
    H.restore();
    H.resetSnowplow();
    cy.signInAsAdmin();
    H.enableTracking();
  });
});
```

...

### **e2e/test/scenarios/visualizations-charts/bar\_chart.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

## Analyst Base

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, PEOPLE, PRODUCTS,...
```

### **e2e/test/scenarios/visualizations-charts/combo.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PRODUCTS, PRODUCTS_ID, ORDERS_ID, ORDERS } =...
```

### **e2e/test/scenarios/visualizations-charts/funnel.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PEOPLE_ID, PEOPLE } =...
```

### **e2e/test/scenarios/visualizations-charts/gauge.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS_ID } = SAMPLE_DATABASE;
```

```
describe("scenarios > visualizations > gauge chart", () => {
```

```
...
```

### **e2e/test/scenarios/visualizations-charts/legend.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, PEOPLE, PRODUCTS } = SAMPLE_DATABASE;
```

```
const ORDERS_CREATED_AT_FIELD_REF = [
```

```
...
```

### **e2e/test/scenarios/visualizations-charts/line-bar-tooltips.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, PRODUCTS } =...
```

### **e2e/test/scenarios/visualizations-charts/line\_chart.cy.spec.js**

```
const { H } = cy;
```

## Analyst Base

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, PRODUCTS, PRODUCTS_ID,...
```

### **e2e/test/scenarios/visualizations-charts/maps.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PEOPLE, PEOPLE_ID } =...
```

### **e2e/test/scenarios/visualizations-charts/pie\_chart.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const {
  ACCOUNTS,
  ACCOUNTS_ID,
  PRODUCTS,
  ...
```

### **e2e/test/scenarios/visualizations-charts/progress-bar.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS_ID } = SAMPLE_DATABASE;
```

```
describe("scenarios > visualizations > progress chart", () => {
```

```
  ...
```

### **e2e/test/scenarios/visualizations-charts/rows.cy.spec.js**

```
const { H } = cy;
```

```
describe("scenarios > visualizations > rows", () => {
  beforeEach(() => {
    H.restore();
    cy.signInAsAdmin();
  });
```

```
// Until we enable multi-browser support, this repro...
```

### **e2e/test/scenarios/visualizations-charts/sankey.cy.spec.js**

## Analyst Base

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
const SANKEY_QUERY = `  
SELECT 'Social Media' AS source, 'Landing Page' AS target, 30000 AS metric  
UNION ALL  
SELECT 'Email...
```

### **e2e/test/scenarios/visualizations-charts/scatter.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, PRODUCTS } = ...
```

### **e2e/test/scenarios/visualizations-charts/trendline.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS_ID, ORDERS, PRODUCTS_ID, PRODUCTS } = SAMPLE_DATABASE;
```

```
describe("scenarios > question >...
```

### **e2e/test/scenarios/visualizations-charts/visualizations-charts-reproductions.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID } = SAMPLE_DATABASE;  
const...
```

### **e2e/test/scenarios/visualizations-charts/visualizations-charts-reproductions.cy.spec.ts**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import type { StructuredQuestionDetails } from...
```

### **e2e/test/scenarios/visualizations-charts/waterfall.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

## Analyst Base

```
const { ORDERS, ORDERS_ID, PRODUCTS } =...
```

### **e2e/test/scenarios/visualizations-tabular/column-shortcuts.cy.spec.ts**

```
import _ from "underscore";
```

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PEOPLE,...
```

### **e2e/test/scenarios/visualizations-tabular/drillthroughs/chart\_drill.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID, PRODUCTS,...
```

### **e2e/test/scenarios/visualizations-tabular/drillthroughs/column\_extract\_drill.cy.spec.js**

```
import _ from "underscore";
```

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import {...
```

### **e2e/test/scenarios/visualizations-tabular/drillthroughs/combine-column.cy.spec.ts**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { PEOPLE, PEOPLE_ID } =...
```

### **e2e/test/scenarios/visualizations-tabular/drillthroughs/dash\_drill.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
import { ORDERS_COUNT_QUESTION_ID } from...
```

### **e2e/test/scenarios/visualizations-tabular/drillthroughs/table\_drills.cy.spec.js**

```
const { H } = cy;
```

```
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
```



## Analyst Base

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const {  
  ORDERS,  
  ORDERS_ID,  
  PRODUCTS,  
  ...
```

### **e2e/test/scenarios/visualizations-tabular/object\_detail.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, WRITABLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const {  
  ORDERS,  
  ORDERS_ID,  
  ...
```

### **e2e/test/scenarios/visualizations-tabular/pivot\_tables.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID, USER_GROUPS } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import { PIVOT_TABLE_BODY_LABEL } from...
```

### **e2e/test/scenarios/visualizations-tabular/scalar.cy.spec.js**

```
const { H } = cy;  
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
```

```
const { ORDERS, ORDERS_ID } =...
```

### **e2e/test/scenarios/visualizations-tabular/smartscale-trend.cy.spec.js**

```
import Color from "color";
```

```
const { H } = cy;  
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";  
import { colors } from "metabase/lib/colors";
```

```
const { ORDERS, ORDERS_ID } =...
```

### **e2e/test/scenarios/visualizations-tabular/table-column-settings.cy.spec.js**

```
import _ from "underscore";
```

```
const { H } = cy;
```

## Analyst Base

```
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";

const { ORDERS_ID, ORDERS, PRODUCTS_ID, PRODUCTS } = SAMPLE_DATABASE;

const...
```

### **e2e/test/scenarios/visualizations-tabular/table.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID, WRITABLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import {
  ...
```

### **e2e/test/scenarios/visualizations-tabular/visualizations-tabular-reproductions.cy.spec.js**

```
const { H } = cy;
import { SAMPLE_DB_ID } from "e2e/support/cypress_data";
import { SAMPLE_DATABASE } from "e2e/support/cypress_sample_database";
import { ADMIN_USER_ID } from...
```

### **e2e/test-host-app/shared/compatibility.cy.spec.js**

```
import {
  mockAuthProviderAndJwtSignIn,
  signInAsAdminAndEnableEmbeddingSdk,
} from "e2e/support/helpers/embedding-sdk-testing";
```

```
const TIMEOUT_MS = 40000;
```

```
const CLIENT_PORT =...
```

### **e2e/validate-e2e-test-files.js**

```
#!/usr/bin/env node
```

### **enterprise/LICENSE.txt**

Usage of files in this directory and its subdirectories, and of Metabase Enterprise Edition features, is subject to the Metabase Commercial License (<https://www.metabase.com/license/commercial>), and...

### **enterprise/README.md**

Metabase Enterprise Edition  
# License

### **enterprise/backend/README.md**

## EE Code Structure Notes

### **enterprise/frontend/.eslintrc**

```
{
  "rules": {
    // Note: adding this rule to a eslint config file in a subfolder will remove
    // *not* carry over the restricted imports from parent folders, you will
    // need to copy them...
```

### **enterprise/frontend/src/embedding/auth-common/.eslintrc**

```
{
  "rules": {
    "no-restricted-imports": [
      "error",
      {
        "patterns": [
          {
            "group": [
              "embedding-sdk/*",
              "metabase/*",
            ],
            ...
```

### **enterprise/frontend/src/embedding/auth-common/README.md**

Common auth functions

### **enterprise/frontend/src/embedding/auth-common/connect-to-instance-auth-sso.ts**

```
import * as MetabaseError from "embedding-sdk/errors";
import type { MetabaseAuthMethod } from "embedding-sdk/types";
```

```
export async function connectToInstanceAuthSso(
  url: string,
  {
    headers,
  }
) {
  ...
```

### **enterprise/frontend/src/embedding/auth-common/index.ts**

```
export { connectToInstanceAuthSso } from "./connect-to-instance-auth-sso";
export { jwtDefaultRefreshTokenFunction } from "./jwt";
export { openSamlLoginPopup } from "./saml";
export { ...
```

### **enterprise/frontend/src/embedding/auth-common/jwt.ts**

```
import * as MetabaseError from "embedding-sdk/errors";

export async function jwtDefaultRefreshTokenFunction(
```

## Analyst Base

```
responseUrl: string,  
instanceUrl: string,  
requestHeaders: Record<string, string>,  
...
```

### **enterprise/frontend/src/embedding/auth-common/saml-token-storage.ts**

```
import type { MetabaseEmbeddingSessionToken } from "embedding-sdk/types/refresh-token";  
  
class TypedStorage<T> {  
  constructor(private key: string) {}  
  
  // Get data with proper typing  
  get(): T |...
```

### **enterprise/frontend/src/embedding/auth-common/saml.ts**

```
import * as MetabaseError from "embedding-sdk/errors";  
import type { MetabaseEmbeddingSessionToken } from "embedding-sdk/types/refresh-token";  
  
/*  
 * For the markup for the popup (nice rhyme), visit  
...  
*/
```

### **enterprise/frontend/src/embedding/auth-common/validate-session-token.ts**

```
import * as MetabaseError from "embedding-sdk/errors";  
  
export function validateSessionToken(session: any) {  
  if (!session || typeof session !== "object") {  
    throw...
```

### **enterprise/frontend/src/embedding/data-picker/DataSelector/index.ts**

```
export { DataSourceSelector } from "./DataSelector";
```

### **enterprise/frontend/src/embedding/data-picker/DataSelector/tests/DataSelector.unit.spec.js**

```
import userEvent from "@testing-library/user-event";  
  
import { createMockMetadata } from "__support__/metadata";  
import { getIcon, render, screen } from "__support__/ui";  
import { delay } from...
```

### **enterprise/frontend/src/embedding/data-picker/DataSelectorDataBucketPicker/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **enterprise/frontend/src/embedding/data-picker/DataSelectorDatabasePicker/index.ts**

## **ex.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **enterprise/frontend/src/embedding/data-picker/DataSelectorDatabaseSchemaPicker/DataSelectorDatabaseSchemaPicker.unit.spec.js**

```
import { createMockEntitiesState } from "__support__/store";
import { render, renderWithProviders, screen } from "__support__/ui";
import { checkNotNull } from "metabase/lib/types";
import {...
```

## **enterprise/frontend/src/embedding/data-picker/DataSelectorDatabaseSchemaPicker/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **enterprise/frontend/src/embedding/data-picker/DataSelectorLoading/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **enterprise/frontend/src/embedding/data-picker/DataSelectorSchemaPicker/DataSelectorSchemaPicker.unit.spec.js**

```
import { render, screen } from "__support__/ui";

import DataSelectorSchemaPicker from "../DataSelectorSchemaPicker";

describe("DataSelectorSchemaPicker", () => {
  it("displays schema name", () =>...
```

## **enterprise/frontend/src/embedding/data-picker/DataSelectorSchemaPicker/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **enterprise/frontend/src/embedding/data-picker/DataSelectorSectionHeader/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **enterprise/frontend/src/embedding/data-picker/DataSelectorTablePicker/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **enterprise/frontend/src/embedding/data-picker/SimpleDataPicker/index.ts**

```
export { SimpleDataPicker } from "../SimpleDataPicker";
export { SimpleDataPickerView } from "../SimpleDataPickerView";
```

## **enterprise/frontend/src/embedding/data-picker/constants.ts**

```
import { t } from "ttag";
```

## Analyst Base

```
import type { DataPickerDataType, DataTypeInfoItem } from "../types";
```

```
export const CONTAINER_WIDTH = 300;
```

```
type DataBucket = "MODELS" | "RAW_DATA" | ...
```

### **enterprise/frontend/src/embedding/data-picker/saved-entity-picker/constants.ts**

```
import { t } from "ttag";
```

```
export const CARD_INFO = {
  question: {
    get title() {
      return t`Saved Questions`;
    },
    model: "card",
    icon: "table2",
  },
  model: {
    get title() {
      ...

```

### **enterprise/frontend/src/embedding/data-picker/saved-entity-picker/utils.js**

```
export const findCollectionById = (collections, collectionId) => {
  if (!collections || collections.length === 0) {
    return null;
  }

```

```
  const collection = collections.find((c) => c.id ===...
```

### **enterprise/frontend/src/embedding/data-picker/types.ts**

```
import type { IconName } from "metabase/ui";
```

```
export type DataPickerDataType = "models" | "raw-data" | "questions";
```

```
export type DataTypeInfoItem = {
  id: DataPickerDataType;
  icon: IconName;
  ...

```

### **enterprise/frontend/src/embedding/data-picker/utils.ts**

```
import {
  MODELS_INFO_ITEM,
  RAW_DATA_INFO_ITEM,

```

## Analyst Base

```
SAVED_QUESTIONS_INFO_ITEM,  
} from "./constants";  
import type { DataTypeInfoItem } from "./types";
```

```
export function getDataTypes({  
  hasModels,  
  ...  
})
```

### **enterprise/frontend/src/embedding/sdk-common/lib/get-sdk-loader-css.ts**

```
export type SdkLoaderData = {  
  className: string;  
  size?: string;  
  color?: string;  
};
```

```
export const getSdkLoaderCss = ({  
  className,  
  size = "1.5rem",  
  // eslint-disable-next-line...
```

### **enterprise/frontend/src/embedding-sdk/.eslintrc.js**

```
eslint-disable no-undef,import/no-commonjs
```

### **enterprise/frontend/src/embedding-sdk/CHANGELOG.md**

```
#  
[0.55.5-metabot](https://github.com/metabase/metabase/compare/embedding-sdk-0.55.4-metabot...embedding-sdk-0.55.5-metabot) (2025-05-07)  
## Bug Fixes
```

### **enterprise/frontend/src/embedding-sdk/README.md**

Metabase embedded analytics SDK

### **enterprise/frontend/src/embedding-sdk/bin/.eslintrc**

```
{  
  "rules": {  
    "no-literal-metabase-strings": "off",  
    "no-console": "off",  
  },  
}
```

### **enterprise/frontend/src/embedding-sdk/bin/generate-cli-snippets-for-testing.ts**

```
#!/usr/bin/env node
```

### **enterprise/frontend/src/embedding-sdk/bin/generate-nextjs-compat.ts**

```
import fs from "fs";
```

## Analyst Base

```
import path from "path";
```

```
import { getPublicComponents } from "embedding-sdk/bin/get-public-components";
```

```
const destinationDir = path.resolve(
  __dirname,
  ...
```

### **enterprise/frontend/src/embedding-sdk/bin/get-public-components.ts**

```
import { type ExportSpecifier, type Node, Project, SyntaxKind } from "ts-morph";
```

```
export type ComponentDefinition = {
  mainComponent: string;
};
```

```
export function getPublicComponents() {
  const...
```

### **enterprise/frontend/src/embedding-sdk/bundle.ts**

```
import { EMBEDDING_SDK_CONFIG } from "metabase/embedding-sdk/config";
```

```
// Enable SDK mode as we are in the SDK bundle
```

```
// This applies to SDK derivatives such as new iframe...
```

### **enterprise/frontend/src/embedding-sdk/cli/.eslintrc**

```
{
  "rules": {
    "no-literal-metabase-strings": "off",
    "no-color-literals": "off",
    "no-console": "off"
  }
}
```

### **enterprise/frontend/src/embedding-sdk/cli/actions/start.ts**

```
import { runCli } from "../run";
```

```
import { printError } from "../utils/print";
```

```
export async function start() {
```

```
  // When the user runs the CLI with npx, there will be some deprecation warnings that...
```

### **enterprise/frontend/src/embedding-sdk/cli/cli.ts**

```
import { Command } from "commander";
```

```
import { start } from "../actions/start";
```



## Analyst Base

```
const program = new Command();
```

```
program
```

```
.name("metabase-embedding-sdk-cli")
```

```
.description("Metabase Embedding SDK...")
```

### **enterprise/frontend/src/embedding-sdk/cli/constants/config.ts**

```
export const IMAGE_NAME = "metabase/metabase-enterprise:latest";
```

```
export const CONTAINER_NAME = "metabase-enterprise-embedding";
```

```
export const SITE_NAME = "Metabase Embedding SDK Demo";
```

```
export const...
```

### **enterprise/frontend/src/embedding-sdk/cli/constants/database.ts**

```
* Show popular database engines in the CLI
```

### **enterprise/frontend/src/embedding-sdk/cli/constants/env.ts**

```
import { SITE_NAME } from "../config";
```

```
/**
```

```
 * Use the same setup token for every demo instance.
```

```
 * This makes it easy to configure across runs.
```

```
 */
```

```
export const EMBEDDING_DEMO_SETUP_TOKEN =
```

```
...
```

### **enterprise/frontend/src/embedding-sdk/cli/constants/hardcoded-users.ts**

```
import { SANDBOXED_GROUP_NAMES } from "../config";
```

```
const [GROUP_A, GROUP_B, GROUP_C] = SANDBOXED_GROUP_NAMES;
```

```
/**
```

```
 * Sample hardcoded users for JWT authentication.
```

```
 */
```

```
export const HARDCODED_USERS =...
```

### **enterprise/frontend/src/embedding-sdk/cli/constants/messages.ts**

```
import { blue, green, yellow } from "chalk";
```

```
import {
```

```
  CONTAINER_NAME,
```

```
  SAMPLE_CREDENTIALS_FILE_NAME,
```

```
  SDK_DOCS_LINK,
```

```
} from "../config";
```

## Analyst Base

```
export const SHOW_ON_STARTUP_MESSAGE = `
  This tool will...
```

### **enterprise/frontend/src/embedding-sdk/cli/constants/mock-server-package-json.ts**

```
export const MOCK_SERVER_PACKAGE_JSON = {
  name: "mock-server",
  version: "1.0.0",
  main: "server.js",
  scripts: {
    start: "node server.js",
  },
  license: "MIT",
  dependencies: {
    cors:...
```

### **enterprise/frontend/src/embedding-sdk/cli/run.ts**

```
import {
  addDatabaseConnectionStep,
  askForTenancyColumns,
  askIfHasDatabase,
  checkIfDockerContainerExists,
  checkIfReactProject,
  checkIsDockerRunning,
  checkSdkAvailable,
  createApiKey,
  ...
```

### **enterprise/frontend/src/embedding-sdk/cli/snippets/analytics-css-snippet.ts**

```
export const ANALYTICS_CSS_SNIPPET = `
body {
  margin: 0;
```

```
.theme-switcher {
  width: 28px;
  height: 28px;
  cursor: pointer;
}
```

```
.analytics-root {
  font-family:...
```

## Analyst Base

### **enterprise/frontend/src/embedding-sdk/cli/snippets/analytics-dashboard-snippet.ts**

```
import { SDK_PACKAGE_NAME } from "../constants/config";
import type { DashboardInfo } from "../types/dashboard";
```

```
interface Options {
  dashboards: DashboardInfo[];
  userSwitcherEnabled:...
```

### **enterprise/frontend/src/embedding-sdk/cli/snippets/analytics-page-snippet.ts**

\*

### **enterprise/frontend/src/embedding-sdk/cli/snippets/analytics-provider-snippet.ts**

```
export const ANALYTICS_PROVIDER_SNIPPET_MINIMAL = `
import { createContext, useState } from 'react'
```

```
/**
 * @typedef {Object} AnalyticsContextType
 * @property {'light'|'dark'} themeKey - The current...
```

### **enterprise/frontend/src/embedding-sdk/cli/snippets/embedding-provider-snippet.ts**

```
import { SDK_PACKAGE_NAME } from "../constants/config";
```

```
interface Options {
  instanceUrl: string;
  apiKey: string;
  userSwitcherEnabled: boolean;
}
```

```
export const getEmbeddingProviderSnippet =...
```

### **enterprise/frontend/src/embedding-sdk/cli/snippets/express-server-snippet.ts**

```
import { sample } from "underscore";
```

```
import { HARDCODED_JWT_SHARED_SECRET } from "../constants/config";
import { HARDCODED_USERS } from "../constants/hardcoded-users";
import type { CliState } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/snippets/get-component-snippets.ts**

```
import type { DashboardInfo } from "../types/dashboard";
import { withNextJsUseClientDirective } from "../utils/nextjs-helpers";
```

## Analyst Base

```
import { getAnalyticsDashboardSnippet } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/snippets/index.ts**

```
export * from "./analytics-provider-snippet";
export * from "./embedding-provider-snippet";
export * from "./analytics-dashboard-snippet";
export * from "./analytics-page-snippet";
export * from...
```

### **enterprise/frontend/src/embedding-sdk/cli/snippets/nextjs-snippets.ts**

```
*
```

### **enterprise/frontend/src/embedding-sdk/cli/snippets/theme-switcher-snippet.ts**

```
eslint-disable-next-line no-unconditional-metabase-links-render -- cli snippets
```

### **enterprise/frontend/src/embedding-sdk/cli/snippets/user-switcher-snippet.ts**

```
import { HARDCODED_USERS } from "../constants/hardcoded-users";
```

```
export const getUserSwitcherSnippet = () => {
  const users = HARDCODED_USERS.map((user) => ({
    email: user.email,
    firstName:...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/add-database-connection.ts**

```
import { search } from "@inquirer/prompts";
import ora from "ora";
```

```
import type { CliStepMethod } from "embedding-sdk/cli/types/cli";
import type { Settings } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/ask-if-has-database.ts**

```
import toggle from "inquirer-toggle";
```

```
import type { CliStepMethod } from "embedding-sdk/cli/types/cli";
import { printHelperText } from "embedding-sdk/cli/utils/print";
```

```
/**
```

```
 * Asks the user first...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/ask-tenancy-columns.ts**

```
import { search } from "@inquirer/prompts";
import toggle from "inquirer-toggle";
```

```
import type { CliStepMethod } from "../types/cli";
```

## Analyst Base

```
import { printHelperText } from "../utils/print";
```

```
export const...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/check-docker-container-exist.ts**

```
import { exec } from "child_process";
```

```
import toggle from "inquirer-toggle";
```

```
import { promisify } from "util";
```

```
import { CONTAINER_NAME } from "../constants/config";
```

```
import {...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/check-docker-running.ts**

```
import { exec as execCallback } from "child_process";
```

```
import ora from "ora";
```

```
import { promisify } from "util";
```

```
import type { CliError, CliStepMethod } from "embedding-sdk/cli/types/cli";
```

```
const...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/check-if-react-project.ts**

```
import ora from "ora";
```

```
import semver from "semver";
```

```
import {  
  MISSING_REACT_DEPENDENCY,  
  PACKAGE_JSON_NOT_FOUND_MESSAGE,  
  UNSUPPORTED_REACT_VERSION,  
} from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/check-sdk-available.ts**

```
import ora from "ora";
```

```
import { SDK_PACKAGE_NAME } from "../constants/config";
```

```
import { installSdk } from "../steps/install-sdk";
```

```
import type { CliStepMethod } from "../types/cli";
```

```
import {...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/create-api-key.ts**

```
import ora from "ora";
```

## Analyst Base

```
import type { CliStepMethod } from "embedding-sdk/cli/types/cli";
```

```
export const createApiKey: CliStepMethod = async (state) => {  
  if (!state.instanceUrl || !state.cookie) {  
    ...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/create-models-and-xrays.ts**

```
import ora from "ora";
```

```
import { createCollection } from "embedding-sdk/cli/utils/create-collection";
```

```
import type { CliStepMethod } from "../types/cli";  
import type { DashboardInfo } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/generate-component-files.ts**

```
import fs from "fs/promises";
```

```
import { input } from "@inquirer/prompts";
```

```
import { getGeneratedComponentFilesMessage } from "../constants/messages";  
import { ANALYTICS_CSS_SNIPPET } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/generate-credentials-file.ts**

```
import fs from "fs/promises";
```

```
import { SAMPLE_CREDENTIALS_FILE_NAME } from "../constants/config";  
import type { CliStepMethod } from "../types/cli";  
import { addFileToGitIgnore } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/generate-credentials.ts**

```
import { input } from "@inquirer/prompts";
```

```
import { isEmail } from "metabase/lib/email";
```

```
import type { CliStepMethod } from "../types/cli";  
import { generateRandomDemoPassword } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/generate-express-server-file.ts**

```
import fs from "fs/promises";
```

```
import { input } from "@inquirer/prompts";
```

```
import { installMockServerDeps } from "embedding-sdk/cli/utils/install-mock-server-deps";
```

## Analyst Base

```
import { MOCK_SERVER_PACKAGE_JSON...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/index.ts**

```
export * from "./check-docker-running";  
export * from "./check-docker-container-exist";  
export * from "./create-api-key";  
export * from "./generate-credentials";  
export * from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/install-sdk.ts**

```
import { exec as execCallback } from "child_process";  
  
import { detect } from "detect-package-manager";  
import toggle from "inquirer-toggle";  
import ora from "ora";  
import { promisify } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/pick-database-tables.ts**

```
import { checkbox } from "@inquirer/prompts";  
import ora from "ora";  
  
import type { CliStepMethod } from "embedding-sdk/cli/types/cli";  
import type { Table, TableId } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/poll-metabase-instance.ts**

```
import ora from "ora";  
  
import type { CliStepMethod } from "embedding-sdk/cli/types/cli";  
  
const delay = (duration: number) =>  
  new Promise((resolve) => setTimeout(resolve, duration));
```

```
const...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/setup-embedding-settings.ts**

```
import ora from "ora";  
  
import { HARDCODED_JWT_SHARED_SECRET } from "../constants/config";  
import { getEmbeddingFailedMessage } from "../constants/messages";  
import type { CliStepMethod } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/setup-license.ts**

```
import { input, select } from "@inquirer/prompts";  
import chalk from "chalk";
```

## Analyst Base

```
import toggle from "inquirer-toggle";
import open from "open";
import ora from "ora";
```

```
import type { CliStepMethod } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/setup-metabase-instance.ts**

```
import ora from "ora";
```

```
import { SITE_NAME } from "../constants/config";
import { EMBEDDING_DEMO_SETUP_TOKEN } from "../constants/env";
import { INSTANCE_CONFIGURED_MESSAGE } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/setup-permission.ts**

```
import { SANDBOXED_GROUP_NAMES } from "../constants/config";
import { getNoTenantMessage } from "../constants/messages";
import type { CliStepMethod } from "../types/cli";
import { createCollection }...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/show-metabase-cli-title.ts**

```
import { SDK_DOCS_LINK } from "embedding-sdk/cli/constants/config";
```

```
import { SHOW_ON_STARTUP_MESSAGE } from "../constants/messages";
import type { CliStepMethod } from "../types/cli";
import {...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/show-post-setup-steps.ts**

```
import { select } from "@inquirer/prompts";
import { green } from "chalk";
```

```
import {
  SDK_LEARN_MORE_MESSAGE,
  getMetabaseInstanceSetupCompleteMessage,
} from "../constants/messages";
import type {...
```

### **enterprise/frontend/src/embedding-sdk/cli/steps/start-local-metabase-container.ts**

```
import { exec as execCallback } from "child_process";
```

```
import chalk from "chalk";
import ora from "ora";
import { promisify } from "util";
```



## Analyst Base

```
import {  
  CONTAINER_NAME,  
  DEFAULT_PORT,  
  IMAGE_NAME,  
}...
```

### **enterprise/frontend/src/embedding-sdk/cli/types/cli.ts**

```
import type { CLI_STEPS } from "embedding-sdk/cli/run";  
import type { Settings, Table } from "metabase-types/api";
```

```
import type { DashboardInfo } from "../types/dashboard";
```

```
export type CliState =...
```

### **enterprise/frontend/src/embedding-sdk/cli/types/dashboard.ts**

```
import type { DashboardId } from "metabase-types/api";
```

```
export type DashboardInfo = { id: DashboardId; name: string };
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/add-database-connection.ts**

```
import { propagateErrorResponse } from "../propagate-error-response";
```

```
interface Options {  
  name: string;  
  engine: string;  
  connection: Record<string, string | boolean | number>;
```

```
  cookie:...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/add-file-to-git-ignore.ts**

```
import fs from "fs/promises";
```

```
const GITIGNORE_PATH = ".gitignore";
```

```
/**
```

```
 * Adds the credential file to .gitignore if exists.
```

```
 */
```

```
export async function addFileToGitIgnore(fileName: string) {  
  try {  
    ...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/ask-for-db-connection-info.ts**

```
import fs from "fs/promises";
```

## Analyst Base

```
import { input, number, password, select } from "@inquirer/prompts";
import { EventEmitter } from "events";
import fileSelector from "inquirer-file-selector";
import...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/check-typescript-project.ts**

```
import fs from "fs";
import path from "path";

import { getProjectDependenciesFromPackageJson } from "../utils/get-package-version";
```

```
/**
 * Checks if the current project is a TypeScript project.
 *
 * ...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/create-collection.ts**

```
import { propagateErrorResponse } from "embedding-sdk/cli/utils/propagate-error-response";
```

```
interface Options {
  name: string;

  instanceUrl: string;
  cookie: string;
}
```

```
export async function...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/create-model-from-table.ts**

```
import type { Table } from "metabase-types/api";

import { propagateErrorResponse } from "../propagate-error-response";
```

```
interface Options {
  table: Table;
  databaseId: number;
  collectionId:...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/fetch-instance-settings.ts**

```
import type { Settings } from "metabase-types/api";

interface Options {
  instanceUrl: string;
}
```

## Analyst Base

```
export async function fetchInstanceSettings(
  options: Options,
): Promise<Settings | null> {
  ...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/generate-password.ts**

```
export function generateRandomDemoPassword(): string {
  const chars = "abcdefghijklmnopqrstuvwxyz";
  const upperCaseChars = chars.toUpperCase();
  const numbers = "0123456789";

  const allChars =...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/get-current-docker-port.ts**

```
const PORT_REGEX = /(?:0\.\.0\.\.0:|:::)(\d+)->3000\tcp/g;

export function getCurrentDockerPort(ports: string): number | null {
  if (!ports || ports.length === 0) {
    return null;
  }

  let...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/get-example-component-import-path.ts**

```
import path from "path";

import { GENERATED_COMPONENTS_DEFAULT_PATH } from "../constants/config";

export const getExampleComponentImportPath = (
  reactComponentDir =...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/get-example-component-import-path.unit.spec.ts**

```
import { getExampleComponentImportPath } from "../get-example-component-import-path";

describe("CLI > getExampleImportPath", () => {
  it("should return a valid example import path", () => {
    ...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/get-local-metabase-container.ts**

```
import { exec as execCallback } from "child_process";
```

## Analyst Base

```
import { promisify } from "util";
```

```
import { safeJsonParse } from "metabase/lib/json-parse";
```

```
import { CONTAINER_NAME } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/get-nextjs-setup-message.ts**

```
import { green } from "chalk";
```

```
import { NEXTJS_DEMO_ROUTE_NAME } from "../constants/config";
```

```
import {
```

```
  LINK_TO_NEXT_JS_GUIDE,
```

```
  LINK_TO_NEXT_JS_SAMPLE,
```

```
} from "../constants/messages";
```

```
import {...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/get-package-version.ts**

```
import fs from "fs/promises";
```

```
import path from "path";
```

```
export const hasPackageJson = async () => {
```

```
  try {
```

```
    await fs.access("package.json");
```

```
    return true;
```

```
  } catch (error) {
```

```
    return...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/get-permission-groups.ts**

```
import {
```

```
  DataPermission,
```

```
  DataPermissionValue,
```

```
} from "metabase/admin/permissions/types";
```

```
import type {
```

```
  DatabasePermissions,
```

```
  GroupsPermissions,
```

```
  Table,
```

```
} from "metabase-types/api";
```

```
type...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/get-sandboxed-collection-permissions.ts**

```
interface Options {
```

```
  groupIds: number[];
```

## Analyst Base

```
collectionIds: number[];
}
```

```
const ALL_USERS_GROUP_ID = 1;
```

```
export function getSandboxedCollectionPermissions(options: Options) {
  const { groupIds,...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/get-tenancy-isolation-sandboxes.ts**

```
import type {
  FieldReference,
  GroupTableAccessPolicy,
  Table,
} from "metabase-types/api";
```

```
type Options = {
  groupIds: number[];
  chosenTables: Table[];
  tenancyColumnNames: Record<string,...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/install-mock-server-deps.ts**

```
import { exec as execCallback } from "child_process";
import path from "path";
```

```
import { detect } from "detect-package-manager";
import ora from "ora";
import { match } from "ts-pattern";
import {...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/is-port-taken.ts**

```
import { type Server, createServer } from "net";
```

```
/**
 * Check if a port is taken.
 * Creates a TCP server on the given port and waits for it to be closed.
 */
```

```
export const checkIsPortTaken = (port:...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/nextjs-helpers.ts**

```
import fs from "fs";
import path from "path";

import { glob } from "glob";
```

## Analyst Base

```
import { NEXTJS_DEMO_ROUTE_NAME } from "../constants/config";
import {
  getNextJsAnalyticsPageSnippet,
  ...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/permissions-graph.unit.spec.ts**

```
import { createMockField, createMockTable } from "metabase-types/api/mocks";

import { getPermissionsForGroups } from "../get-permission-groups";
import { getSandboxedCollectionPermissions } from...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/print.ts**

```
import chalk from "chalk";
```

```
const MAX_WIDTH = 80;
```

```
export const OUTPUT_STYLES = {
  title: chalk.bold.bgHex("#509EE3").white,
  version: chalk.hex("#509EE3"),
  link: chalk.underline.blueBright,
  ...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/propagate-error-response.ts**

```
import type { CliError } from "../types/cli";
```

```
// Propagate the error from the API to the CLI.
```

```
export const propagateErrorResponse = async (res: Response) => {
  if (res.ok) {
    return;
  }
```

```
let...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/retry.ts**

```
const waitFor = (delayMs: number) =>
  new Promise((resolve) => setTimeout(resolve, delayMs));
```

```
export async function retry<T>(
  task: () => Promise<T>,
  options?: {
    retries?: number;
    ...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/sample-tenancy-column-values.**

## Analyst Base

**ts**

```
import type { Dataset, FieldReference, Table } from "metabase-types/api";
```

```
import { propagateErrorResponse } from "../propagate-error-response";
```

```
interface Options {  
  table: Table;  
  columnName:...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/show-warning-prompt.ts**

```
import { select } from "@inquirer/prompts";
```

```
import { CONTINUE_SETUP_ON_WARNING_MESSAGE } from "../constants/messages";
```

```
import { printWarning } from "../print";
```

```
/**  
 * @returns {boolean} whether the...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/snippets-helpers.ts**

```
import { GENERATED_COMPONENTS_DEFAULT_PATH } from "../constants/config";
```

```
/**  
 * Where should we save the generated components by default?  
 */
```

```
export const getGeneratedComponentsDefaultPath = ({  
  ...
```

### **enterprise/frontend/src/embedding-sdk/cli/utils/xray-models.ts**

```
import type {  
  MetabaseDashboard,  
  SdkDashboardId,  
} from "embedding-sdk/types/dashboard";  
import { uuid } from "metabase/lib/uuid";
```

```
import { propagateErrorResponse } from...
```

### **enterprise/frontend/src/embedding-sdk/components/private/PublicComponent Wrapper/index.ts**

```
export * from "../PublicComponentWrapper";  
export * from "../withPublicComponentWrapper";  
export * from "../SdkError";  
export * from "../SdkLoader";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkAdHocQuestion/index.ts**

```
export { SdkAdHocQuestion } from "./SdkAdHocQuestion";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/BackButton/index.ts**

```
export * from "./BackButton";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Breakout/BreakoutDropdown/index.ts**

```
export * from "./BreakoutDropdown";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Breakout/BreakoutPicker/index.ts**

```
export * from "./BreakoutPicker";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Breakout/index.ts**

```
export * from "./Breakout";
```

```
export * from "./BreakoutDropdown";
```

```
export * from "./BreakoutPicker";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Breakout/use-breakout-data.ts**

```
import { isNotNull } from "metabase/lib/types";
```

```
import {
```

```
  type ListItem as BreakoutListItem,
```

```
  getBreakoutListItem,
```

```
} from...
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/ChartTypeSelectorList/index.ts**

```
export * from "./ChartTypeDropdown";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/DownloadWidget/index.ts**

```
export * from "./DownloadWidget";
```

```
export * from "./DownloadWidgetDropdown";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/EditorButton/index.ts**

```
export * from "./EditorButton";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/comp**



### **onents/Filter/FilterDropdown/index.ts**

```
export * from "../FilterDropdown";
```

### **enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Filter/FilterPicker/index.ts**

```
export * from "../FilterPicker";
```

### **enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Filter/hooks/use-filter-data.ts**

```
import { useCallback, useMemo } from "react";
```

```
import { useSdkQuestionContext } from "embedding-sdk/components/private/SdkQuestion/context";
```

```
import type { FilterItem } from...
```

### **enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Filter/hooks/use-filter-handlers.ts**

```
import type { UpdateQueryHookProps } from "metabase/query_builder/hooks";
```

```
import * as Lib from "metabase-lib";
```

```
export const useFilterHandlers = ({
```

```
  query,
```

```
  stageIndex = -1,
```

```
  onQueryChange,
```

```
}):...
```

### **enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Filter/index.ts**

```
export * from "../FilterDropdown";
```

```
export * from "../Filter";
```

```
export * from "../FilterPicker";
```

### **enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/QuestionSettings/QuestionSettingsDropdown/index.ts**

```
export * from "../QuestionSettingsDropdown";
```

### **enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/QuestionSettings/index.ts**

```
export * from "../QuestionSettings";
```

```
export * from "../QuestionSettingsDropdown";
```

### **enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/ResetButton/index.ts**

```
export * from "../ResetButton";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Summarize/SummarizeDropdown/index.ts**

```
export * from "./SummarizeDropdown";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Summarize/SummarizePicker/index.ts**

```
export * from "./SummarizePicker";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Summarize/index.ts**

```
export * from "./Summarize";
```

```
export * from "./SummarizeDropdown";
```

```
export * from "./SummarizePicker";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/Summarize/use-summarize-data.ts**

```
import { useCallback, useMemo } from "react";
```

```
import {  
  type AggregationItem,  
  getAggregationItems,  
} from "metabase/query_builder/utils/get-aggregation-items";  
import * as Lib from...
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/VisualizationButton/index.ts**

```
export * from "./VisualizationButton";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/index.ts**

```
export * from "./BackButton";
```

```
export * from "./Breakout";
```

```
export * from "./ChartTypeSelector";
```

```
export * from "./ChartTypeSelectorList";
```

```
export * from "./DownloadWidget";
```

```
export * from...
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/util/BadgeList/AddBadgeListItem/index.ts**

```
export * from "./AddBadgeListItem";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/util/BadgeList/BadgeListItem/index.ts**

## Analyst Base

```
export * from "./BadgeListItem";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/util/BadgeList/index.ts**

```
export * from "./BadgeList";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/components/util/ToolbarButton/index.ts**

```
export * from "./ToolbarButton";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/context/index.ts**

```
export * from "./SdkQuestionProvider";
```

```
export * from "./types";
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/context/types.ts**

```
import type { PropsWithChildren } from "react";
```

```
import type { LoadQuestionHookResult } from "embedding-sdk/hooks/private/use-load-question";
```

```
import type { SdkCollectionId } from...
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/hooks/use-question-visualization.ts**

```
import { useQuestionVisualizationState } from "metabase/query_builder/components/chart-type-selector";
```

```
import { useSdkQuestionContext } from "../context";
```

```
export const useQuestionVisualization = ()...
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/hooks/use-run-visualization.ts**

```
import { useMemo } from "react";
```

```
import { transformSdkQuestion } from "embedding-sdk/lib/transform-question";
```

```
import { isQuestionDirty } from "metabase/query_builder/utils/question";
```

```
import {...
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestion/hooks/use-sensible-visualizations.ts**

```
import { useMemo } from "react";
```

```
import { getSensibleVisualizations } from "metabase/query_builder/components/chart-type-selector";
```

## Analyst Base

```
import { useSdkQuestionContext } from "../context";
```

```
export const...
```

**enterprise/frontend/src/embedding-sdk/components/private/SdkQuestionDefaultView/index.ts**

```
export * from "../SdkQuestionDefaultView";
```

**enterprise/frontend/src/embedding-sdk/components/private/util/MultiStepPopover/index.ts**

```
export * from "../MultiStepPopover";
```

**enterprise/frontend/src/embedding-sdk/components/public/CollectionBrowser/index.ts**

```
export * from "../CollectionBrowser";
```

**enterprise/frontend/src/embedding-sdk/components/public/CreateDashboardModal/index.ts**

```
export { CreateDashboardModal } from "../CreateDashboardModal";
```

```
export type { CreateDashboardModalProps } from "../CreateDashboardModal";
```

**enterprise/frontend/src/embedding-sdk/components/public/CreateQuestion/index.ts**

```
export * from "../CreateQuestion";
```

**enterprise/frontend/src/embedding-sdk/components/public/InteractiveQuestion/index.ts**

```
export * from "../InteractiveQuestion";
```

**enterprise/frontend/src/embedding-sdk/components/public/MetabotQuestion/index.ts**

```
export { MetabotQuestion } from "../MetabotQuestion";
```

**enterprise/frontend/src/embedding-sdk/components/public/SdkQuestion/index.ts**

```
export {
```

```
  type BackButtonProps as InteractiveQuestionBackButtonProps,
```

```
  type InteractiveQuestionBreakoutDropdownProps,
```

```
  type ChartTypeDropdownProps as InteractiveQuestionChartTypeDropdownProps,
```

```
  ...
```

**enterprise/frontend/src/embedding-sdk/components/public/SdkQuestion/types.ts**

```
import type { SdkQuestionId } from "embedding-sdk/types/question";
```

## Analyst Base

```
export type SdkQuestionIdProps = {  
  /**  
   * The ID of the question.  
   * <br/>  
   * This is either:  
   * <br/>  
   * - The...
```

### **enterprise/frontend/src/embedding-sdk/components/public/StaticQuestion/index.ts**

```
export * from "./StaticQuestion";
```

### **enterprise/frontend/src/embedding-sdk/components/public/StaticQuestion/module.ts**

```
import type { QueryClickActionsMode } from "metabase/visualizations/types";
```

```
export const StaticQuestionSdkMode: QueryClickActionsMode = {  
  name: "static-question-sdk",  
  hasDrills: false,  
  ...
```

### **enterprise/frontend/src/embedding-sdk/components/public/dashboard/EditableDashboard/index.ts**

```
export {  
  EditableDashboard,  
  type EditableDashboardProps,  
} from "./EditableDashboard";
```

### **enterprise/frontend/src/embedding-sdk/components/public/dashboard/InteractiveDashboard/index.ts**

```
export {  
  InteractiveDashboard,  
  type InteractiveDashboardProps,  
} from "./InteractiveDashboard";
```

### **enterprise/frontend/src/embedding-sdk/components/public/dashboard/StaticDashboard/index.ts**

```
export { StaticDashboard, type StaticDashboardProps } from "./StaticDashboard";
```

### **enterprise/frontend/src/embedding-sdk/components/public/dashboard/index.ts**

```
export * from "./EditableDashboard";  
export * from "./InteractiveDashboard";  
export * from "./StaticDashboard";
```

## Analyst Base

### **enterprise/frontend/src/embedding-sdk/components/public/index.ts**

```
export { CollectionBrowser } from "./CollectionBrowser";
export { CreateDashboardModal } from "./CreateDashboardModal";
export { CreateQuestion } from "./CreateQuestion";
export {
  StaticDashboard,
  ...
```

### **enterprise/frontend/src/embedding-sdk/config.ts**

```
export const DEFAULT_FONT = "Lato";

export const getEmbeddingSdkVersion = (): string | "unknown" =>
  (process.env.EMBEDDING_SDK_VERSION as string) ?? "unknown";
```

### **enterprise/frontend/src/embedding-sdk/conventional-changelog-config.js**

```
eslint-disable-next-line import/no-commonjs
```

### **enterprise/frontend/src/embedding-sdk/dev.md**

Docs for development on the sdk

### **enterprise/frontend/src/embedding-sdk/errors/base.ts**

```
export class MetabaseError<C extends string, P> extends Error {
  public readonly code: C;
  public readonly params?: P;

  constructor(code: C, message: string, params?: P) {
    super(message);
    ...
```

### **enterprise/frontend/src/embedding-sdk/errors/generic.ts**

```
import { MetabaseError } from "../base";

export function USER_FETCH_FAILED() {
  return new MetabaseError(
    "USER_FETCH_FAILED",
    "Failed to fetch the user, the session might be invalid.",
    ...
```

### **enterprise/frontend/src/embedding-sdk/errors/index.ts**

```
export * from "../base";
export * from "../jwt";
export * from "../version";
export * from "../saml";
export * from "../generic";
```

## Analyst Base

### **enterprise/frontend/src/embedding-sdk/errors/jwt.ts**

```
import { MetabaseError } from "./base";

export function INVALID_SESSION_OBJECT(params: {
  expected?: string;
  actual?: string;
}) {
  return new MetabaseError(
    "INVALID_SESSION_OBJECT",
    ...
```

### **enterprise/frontend/src/embedding-sdk/errors/saml.ts**

```
import { MetabaseError } from "./base";

export function SAML_POPUP_BLOCKED() {
  return new MetabaseError(
    "SAML_POPUP_BLOCKED",
    "Popup blocked. Please allow popups for this site.",
    ...
```

### **enterprise/frontend/src/embedding-sdk/errors/version.ts**

```
import { MetabaseError } from "./base";

export function SDK_VERSION_INCOMPATIBLE(params: {
  expected?: string;
  actual?: string;
}) {
  return new MetabaseError(
    "SDK_VERSION_INCOMPATIBLE",
    ...
```

### **enterprise/frontend/src/embedding-sdk/hooks/index.ts**

```
export * from "./private";
```

### **enterprise/frontend/src/embedding-sdk/hooks/private/index.ts**

```
export * from "./use-init-data";
```

### **enterprise/frontend/src/embedding-sdk/hooks/private/use-init-data/index.ts**

```
export * from "./use-init-data";
```

### **enterprise/frontend/src/embedding-sdk/hooks/private/use-init-data/use-init-data.ts**

```
import { useEffect, useRef } from "react";
import { useMount } from "react-use";
```

## Analyst Base

```
import _ from "underscore";
```

```
import { getEmbeddingSdkVersion } from "embedding-sdk/config";
```

```
import { useLazySelector }...
```

### **enterprise/frontend/src/embedding-sdk/hooks/private/use-load-question.ts**

```
import { useReducer, useRef, useState } from "react";
```

```
import { useAsyncFn, useUnmount } from "react-use";
```

```
import {
```

```
  loadQuestionSdk,
```

```
  runQuestionOnNavigateSdk,
```

```
  runQuestionQuerySdk,
```

```
  ...
```

### **enterprise/frontend/src/embedding-sdk/hooks/private/use-sdk-dashboard-params.ts**

```
import { pick } from "underscore";
```

```
import type { SdkDashboardId } from "embedding-sdk/types/dashboard";
```

```
import type { CommonStylingProps } from "embedding-sdk/types/props";
```

```
import {...
```

### **enterprise/frontend/src/embedding-sdk/hooks/private/use-sdk-element-size.ts**

```
import { useElementSize } from "@mantine/hooks";
```

```
import { getDefaultVizHeight } from "embedding-sdk/lib/default-height";
```

```
import type { VisualizationDisplay } from "metabase-types/api";
```

```
export const...
```

### **enterprise/frontend/src/embedding-sdk/hooks/private/use-sdk-usage-problem.ts**

```
import { useEffect, useMemo, useRef } from "react";
```

```
import { printUsageProblemToConsole } from "embedding-sdk/lib/print-usage-problem";
```

```
import { getSdkUsageProblem } from...
```

### **enterprise/frontend/src/embedding-sdk/hooks/private/use-translated-collection-id.ts**

```
import { isValidId } from "embedding-sdk/lib/is-valid-collection-id";
```

```
import { getCollectionIdSlugFromReference } from "embedding-sdk/store/collections";
```

```
import type { SdkCollectionId } from...
```



## Analyst Base

### enterprise/frontend/src/embedding-sdk/index.ts

```
import { EMBEDDING_SDK_CONFIG } from "metabase/embedding-sdk/config";
import { defineGlobalDependencies } from "metabase/embedding-sdk/lib/define-global-dependencies";
```

```
// Enable SDK mode as we are...
```

### enterprise/frontend/src/embedding-sdk/jest/console-restrictions.js

```
const RESTRICTED_CONSOLE_PATTERNS = [
  /UNSAFE_component.*Visualization/,
  /UNSAFE_component.*DashboardGrid/,
  /Warning: React does not recognize the `.*?` prop on a DOM element/,
  /Warning:.../
```

### enterprise/frontend/src/embedding-sdk/jest/setup-after-env.js

```
import { ensureMetabaseProviderPropsStore } from
"embedding-sdk/sdk-shared/lib/ensure-metabase-provider-props-store";
```

```
afterEach(() => {
  ensureMetabaseProviderPropsStore().cleanup();
});
```

### enterprise/frontend/src/embedding-sdk/jest/setup-env.js

```
import { EMBEDDING_SDK_CONFIG } from "metabase/embedding-sdk/config";
```

```
// eslint-disable-next-line no-undef
process.env.IS_EMBEDDING_SDK = "true";
```

```
EMBEDDING_SDK_CONFIG.isEmbeddingSdk = true;
```

### enterprise/frontend/src/embedding-sdk/lib/default-height.ts

```
import { getDefaultSize } from "metabase/visualizations/shared/utils/sizes";
import type { VisualizationDisplay } from "metabase-types/api";
```

```
/**
 * How many pixels are in each cell?
 *
 * The card...
```

### enterprise/frontend/src/embedding-sdk/lib/is-localhost.ts

```
export const getIsLocalhost = () => {
  const { hostname } = window.location;

  return hostname === "localhost" || hostname === "127.0.0.1";
};
```

## Analyst Base

### **enterprise/frontend/src/embedding-sdk/lib/is-valid-collection-id.ts**

```
import type { CollectionId } from "metabase-types/api";
```

```
export const isValidId = (
  collectionId: unknown,
): collectionId is CollectionId => {
  return (
    !!collectionId &&
    (typeof...
```

### **enterprise/frontend/src/embedding-sdk/lib/load-static-question.ts**

```
import type { Deferred } from "metabase/lib/promise";
import { CardApi } from "metabase/services";
import type { Card, Dataset, ParameterQueryObject } from "metabase-types/api";
```

```
interface Options {
  ...
```

### **enterprise/frontend/src/embedding-sdk/lib/log-utils.ts**

```
eslint-disable no-color-literals
```

Note: these functions need to return an array of two elements, when styling console.logs, the style needs to be passed as second argument  
To use them, do...

### **enterprise/frontend/src/embedding-sdk/lib/plugins/dashboard.ts**

```
import { merge } from "icepick";
```

```
import type {
  MetabaseDashboardPluginsConfig,
  MetabasePluginsConfig,
} from "embedding-sdk/types/plugins";
```

```
const DEFAULT_DASHCARD_MENU_ITEMS:...
```

### **enterprise/frontend/src/embedding-sdk/lib/polyfill/use-sync-external-store.ts**

```
import React from "react";
import { useSyncExternalStore } from "use-sync-external-store/shim";
```

```
// Monkey-patches useSyncExternalStore if we are in React 17,
// where useSyncExternalStore is not...
```

### **enterprise/frontend/src/embedding-sdk/lib/print-usage-problem.ts**

```
eslint-disable no-color-literals
```

## Analyst Base

### **enterprise/frontend/src/embedding-sdk/lib/sdk-question/index.ts**

```
export { updateQuestionSdk } from "./update-question";
export { loadQuestionSdk } from "./load-question";
export { runQuestionOnNavigateSdk } from "./run-question-on-navigate";
export {...
```

### **enterprise/frontend/src/embedding-sdk/lib/sdk-question/load-question.ts**

```
import _ from "underscore";

import type { LoadSdkQuestionParams } from "embedding-sdk/types/question";
import { resolveCards } from "metabase/query_builder/actions";
import {...
```

### **enterprise/frontend/src/embedding-sdk/lib/sdk-question/run-question-on-navigate.ts**

```
import { runQuestionQuerySdk } from "embedding-sdk/lib/sdk-question/run-question-query";
import type {
  NavigateToNewCardParams,
  SdkQuestionState,
} from "embedding-sdk/types/question";
import {...
```

### **enterprise/frontend/src/embedding-sdk/lib/sdk-question/run-question-query.ts**

```
import type { SdkQuestionState } from "embedding-sdk/types/question";
import type { Deferred } from "metabase/lib/promise";
import { runQuestionQuery } from "metabase/services";
import {...
```

### **enterprise/frontend/src/embedding-sdk/lib/sdk-question/update-question.ts**

```
import _ from "underscore";

import type { SdkQuestionState } from "embedding-sdk/types/question";
import type { Deferred } from "metabase/lib/promise";
import { computeQuestionPivotTable } from...
```

### **enterprise/frontend/src/embedding-sdk/lib/sdk-specific-imports.ts**

\*

### **enterprise/frontend/src/embedding-sdk/lib/theme/color-tuple.ts**

```
type ColorTuple = [
  string,
  string,
  string,
  string,
```

## Analyst Base

```
string,  
string,  
string,  
string,  
string,  
string,
```

```
];
```

```
export const colorTuple = (value: string): ColorTuple =>
```

```
...
```

### **enterprise/frontend/src/embedding-sdk/lib/theme/embedding-color-palette.unit.spec.ts**

```
import { getEmbeddingColorPalette } from "metabase/embedding-sdk/theme/embedding-color-palette";
```

```
describe("Embedding Color Palette", () => {  
  it("transforms chart color overrides into accent...
```

### **enterprise/frontend/src/embedding-sdk/lib/theme/get-embedding-theme.ts**

```
import { merge } from "icepick";
```

```
import _ from "underscore";
```

```
import { DEFAULT_FONT } from "embedding-sdk/config";
```

```
import type {
```

```
  MetabaseColor,
```

```
  MetabaseComponentTheme,
```

```
  MetabaseTheme,
```

```
} from...
```

### **enterprise/frontend/src/embedding-sdk/lib/theme/get-embedding-theme.unit.spec.ts**

```
import {
```

```
  DEFAULT_EMBEDDED_COMPONENT_THEME,
```

```
  getEmbeddingComponentOverrides,
```

```
} from "metabase/embedding-sdk/theme";
```

```
import { getEmbeddingThemeOverride } from...
```

### **enterprise/frontend/src/embedding-sdk/lib/theme/index.ts**

```
export { getEmbeddingThemeOverride } from "./get-embedding-theme";
```

### **enterprise/frontend/src/embedding-sdk/lib/transform-question.ts**

```
import type { MetabaseQuestion } from "metabase/embedding-sdk/types/question";
```

```
import type Question from "metabase-lib/v1/Question";
```

## Analyst Base

```
export function transformSdkQuestion(question: Question):...
```

### **enterprise/frontend/src/embedding-sdk/lib/transform-question.unit.spec.ts**

```
import { createMockMetadata } from "___support___/metadata";
import { transformSdkQuestion } from "embedding-sdk/lib/transform-question";
import Question from "metabase-lib/v1/Question";
import {...
```

### **enterprise/frontend/src/embedding-sdk/lib/usage-problem.ts**

```
import { match } from "ts-pattern";

import type { MetabaseAuthConfig } from "embedding-sdk/types";
import type {
  SdkUsageProblem,
  SdkUsageProblemKey,
} from...
```

### **enterprise/frontend/src/embedding-sdk/lib/version-utils.ts**

```
import { versionToNumericComponents } from "metabase/lib/utils";
```

```
// They are mostly used for local development
```

```
export const isInvalidMetabaseVersion = (mbVersion: string) => {
  return (
    ...
  )
}
```

### **enterprise/frontend/src/embedding-sdk/lib/version-utils.unit.spec.ts**

```
import {
  isInvalidMetabaseVersion,
  isSdkVersionCompatibleWithMetabaseVersion,
} from "../version-utils";
```

```
const expectCompatibility = ({
  mbVersion,
  sdkVersion,
  expected,
}): {
  mbVersion:...
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/components/public/CollectionBrowser/index.ts**

```
export * from "../CollectionBrowser";
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/components/public/CreateDashboardModal/index.ts**

## Analyst Base

```
export * from "./CreateDashboardModal";
```

**enterprise/frontend/src/embedding-sdk/sdk-package/components/public/CreateQuestion/index.ts**

```
export * from "./CreateQuestion";
```

**enterprise/frontend/src/embedding-sdk/sdk-package/components/public/InteractiveQuestion/index.ts**

```
export * from "./InteractiveQuestion";
```

**enterprise/frontend/src/embedding-sdk/sdk-package/components/public/MetabaseProvider/index.ts**

```
eslint-disable-next-line no-literal-metabase-strings -- Export
```

**enterprise/frontend/src/embedding-sdk/sdk-package/components/public/MetabotQuestion/index.ts**

```
export * from "./MetabotQuestion";
```

**enterprise/frontend/src/embedding-sdk/sdk-package/components/public/StaticQuestion/index.ts**

```
export * from "./StaticQuestion";
```

**enterprise/frontend/src/embedding-sdk/sdk-package/components/public/dashboard/EditableDashboard/index.ts**

```
export * from "./EditableDashboard";
```

**enterprise/frontend/src/embedding-sdk/sdk-package/components/public/dashboard/InteractiveDashboard/index.ts**

```
export * from "./InteractiveDashboard";
```

**enterprise/frontend/src/embedding-sdk/sdk-package/components/public/dashboard/StaticDashboard/index.ts**

```
export * from "./StaticDashboard";
```

**enterprise/frontend/src/embedding-sdk/sdk-package/components/public/debug/SdkDebugInfo/index.ts**

```
export * from "./SdkDebugInfo";
```

**enterprise/frontend/src/embedding-sdk/sdk-package/config.ts**

```
export const SDK_BUNDLE_SCRIPT_DATA_ATTRIBUTE_DASH_CASED =  
  "embedding-sdk-bundle";
```

```
export const SDK_BUNDLE_SCRIPT_DATA_ATTRIBUTE_PASCAL_CASED =  
  "embeddingSdkBundle";
```

## Analyst Base

export const...

### **enterprise/frontend/src/embedding-sdk/sdk-package/hooks/private/use-initialize-metabase-provider-props-store.ts**

```
import { useEffect, useMemo } from "react";
```

```
import { useMetabaseProviderPropsStore } from "embedding-sdk/sdk-shared/hooks/use-metabase-provider-props-store";
import { type...
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/hooks/private/use-lazy-selector.ts**

```
import { useCallback, useSyncExternalStore } from "react";
```

```
import { useMetabaseProviderPropsStore } from "embedding-sdk/sdk-shared/hooks/use-metabase-provider-props-store";
import type {...
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/hooks/private/use-load-sdk-bundle.ts**

```
import { useEffect } from "react";
```

```
// eslint-disable-next-line no-external-references-for-sdk-package-code
import { SDK_BUNDLE_FULL_PATH } from...
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/hooks/public/use-application-name.ts**

```
import { useLazySelector } from "embedding-sdk/sdk-package/hooks/private/use-lazy-selector";
import { getWindow } from "embedding-sdk/sdk-shared/lib/get-window";
```

```
/**
 * Returns application name.
 * ...
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/hooks/public/use-available-fonts.ts**

```
import { useMemo } from "react";
```

```
import { useLazySelector } from "embedding-sdk/sdk-package/hooks/private/use-lazy-selector";
import { getWindow } from...
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/hooks/public/use-create-dashboard-api/index.ts**

## Analyst Base

```
export { useCreateDashboardApi } from "./use-create-dashboard-api";
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/hooks/public/use-create-dash-board-api/use-create-dashboard-api.ts**

```
import { useCallback, useMemo } from "react";
```

```
import { useLazySelector } from "embedding-sdk/sdk-package/hooks/private/use-lazy-selector";
```

```
import { useMetabaseProviderPropsStore } from...
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/hooks/public/use-current-user.ts**

```
import { useLazySelector } from "embedding-sdk/sdk-package/hooks/private/use-lazy-selector";
```

```
import { getWindow } from "embedding-sdk/sdk-shared/lib/get-window";
```

```
import type { MetabaseUser } from...
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/hooks/public/use-metabase-auth-status.ts**

```
import { useLazySelector } from "embedding-sdk/sdk-package/hooks/private/use-lazy-selector";
```

```
import { getWindow } from "embedding-sdk/sdk-shared/lib/get-window";
```

```
/**
```

```
 * Returns the authentication...
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/lib/private/get-sdk-bundle-script-element.ts**

```
import { SDK_BUNDLE_SCRIPT_DATA_ATTRIBUTE_DASH_CASED } from "embedding-sdk/sdk-package/config";
```

```
export function getSdkBundleScriptElement(): HTMLScriptElement | null {  
  return...
```

### **enterprise/frontend/src/embedding-sdk/sdk-package/lib/public/define-metabase-auth-config.ts**

```
import type { MetabaseAuthConfig } from "embedding-sdk/types";
```

```
/**
```

```
 * Defines a Metabase auth config.
```

```
 *
```

```
 * @function
```

```
 * @category MetabaseProvider
```

```
 */
```

```
export const defineMetabaseAuthConfig = (  
  ...
```



## Analyst Base

### **enterprise/frontend/src/embedding-sdk/sdk-package/lib/public/define-metabase-theme.ts**

eslint-disable-next-line no-external-references-for-sdk-package-code

### **enterprise/frontend/src/embedding-sdk/sdk-shared/hooks/use-metabase-provider-props-store.ts**

```
import { useEffect, useRef, useSyncExternalStore } from "react";
```

```
import { ensureMetabaseProviderPropsStore } from "../lib/ensure-metabase-provider-props-store";
```

```
export function...
```

### **enterprise/frontend/src/embedding-sdk/sdk-shared/hooks/use-sdk-loading-state.ts**

```
import { useMetabaseProviderPropsStore } from "embedding-sdk/sdk-shared/hooks/use-metabase-provider-props-store";
```

```
import { SdkLoadingError, SdkLoadingState, } from...
```

### **enterprise/frontend/src/embedding-sdk/sdk-shared/hooks/use-single-instance-ids-data.ts**

```
import { useCallback } from "react";
```

```
import { useMetabaseProviderPropsStore } from "embedding-sdk/sdk-shared/hooks/use-metabase-provider-props-store";
```

```
export const useSingleInstanceIdsData = () =>...
```

### **enterprise/frontend/src/embedding-sdk/sdk-shared/lib/ensure-metabase-provider-props-store.ts**

```
import type { InternalMetabaseProviderProps } from "embedding-sdk/components/public/MetabaseProvider";
```

```
import { type SdkLoadingError, SdkLoadingState, } from...
```

### **enterprise/frontend/src/embedding-sdk/sdk-shared/lib/get-window.ts**

```
export function getWindow() {
  return typeof window !== "undefined" ? window : null;
}
```

### **enterprise/frontend/src/embedding-sdk/sdk-shared/types/sdk-loading.ts**

## Analyst Base

```
export enum SdkLoadingError {  
  NotStartedLoading = "NotStartedLoading",  
  Error = "Error",  
}
```

```
export enum SdkLoadingState {  
  Initial = 0,  
  Loading = 1,  
  Loaded = 2,  
  Initialized = 3,  
}
```

### **enterprise/frontend/src/embedding-sdk/store/auth/auth.ts**

```
import {  
  connectToInstanceAuthSso,  
  jwtDefaultRefreshTokenFunction,  
  openSamlLoginPopup,  
  samlTokenStorage,  
  validateSessionToken,  
} from "embedding/auth-common";  
import {...
```

### **enterprise/frontend/src/embedding-sdk/store/auth/index.ts**

```
export * from "./auth";  
export * from "../../embedding/auth-common/saml-token-storage";
```

### **enterprise/frontend/src/embedding-sdk/store/collections.ts**

```
import { createSelector } from "@reduxjs/toolkit";  
import { P, match } from "ts-pattern";
```

```
import { getUserPersonalCollectionId } from "metabase/selectors/user";  
import type { CollectionId,...
```

### **enterprise/frontend/src/embedding-sdk/store/index.ts**

```
eslint-disable no-restricted-imports
```

### **enterprise/frontend/src/embedding-sdk/store/reducer.ts**

```
import { createAction, createReducer } from "@reduxjs/toolkit";
```

```
import { samlTokenStorage } from "embedding/auth-common";  
import type { SdkState, SdkStoreState } from...
```

### **enterprise/frontend/src/embedding-sdk/store/selectors.ts**

```
import type { SdkStoreState } from "embedding-sdk/store/types";  
import { getSetting } from "metabase/selectors/settings";
```

## Analyst Base

```
import type { State } from "metabase-types/store";
```

```
export const...
```

### **enterprise/frontend/src/embedding-sdk/store/types.ts**

```
import type {  
  Action,  
  AnyAction,  
  SerializedError,  
  Store,  
  ThunkDispatch,  
} from "@reduxjs/toolkit";  
import type { JSX } from "react";
```

```
import type { MetabaseAuthConfig } from...
```

### **enterprise/frontend/src/embedding-sdk/store/use-sdk-selector.ts**

```
import { useContext } from "react";  
import type { TypedUseSelectorHook } from "react-redux";  
import { createSelectorHook } from "react-redux";
```

```
import type { SdkStoreState } from...
```

### **enterprise/frontend/src/embedding-sdk/store/use-sdk-selector.unit.spec.ts**

```
import { renderHook } from "__support__/ui";
```

```
import {  
  USE_OUTSIDE_OF_CONTEXT_MESSAGE,  
  useSdkSelector,  
} from "../use-sdk-selector";
```

```
describe("useSdkSelector", () => {  
  it("should throw an error...
```

### **enterprise/frontend/src/embedding-sdk/test/.eslintrc**

```
{  
  "rules": {  
    "no-color-literals": "off"  
  }  
}
```

### **enterprise/frontend/src/embedding-sdk/test/environment.unit.spec.ts**

```
import { EMBEDDING_SDK_CONFIG } from "metabase/embedding-sdk/config";
```

## Analyst Base

```
describe("SDK environment config", () => {  
  beforeEach(() => {  
    EMBEDDING_SDK_CONFIG.isEmbeddingSdk = false;  
  });  
});
```

...

### **enterprise/frontend/src/embedding-sdk/test/mocks/config.ts**

```
import type {  
  MetabaseAuthConfig,  
  MetabaseAuthConfigWithApiKey,  
} from "embedding-sdk/types";  
  
import { MOCK_INSTANCE_URL } from "../sso";
```

```
export const createMockSdkConfig = (  
  opts:...
```

### **enterprise/frontend/src/embedding-sdk/test/mocks/sso.ts**

```
import fetchMock from "fetch-mock";  
  
// ===== MOCK CONSTANTS =====
```

```
export const MOCK_INSTANCE_URL = "http://localhost";  
export const MOCK_JWT_PROVIDER_URI = "http://test_uri/sso/metabase";  
export...
```

### **enterprise/frontend/src/embedding-sdk/test/mocks/state.ts**

```
import type {  
  EmbeddingSessionTokenState,  
  SdkState,  
} from "embedding-sdk/store/types";  
import type { LoginStatus } from "embedding-sdk/types/user";
```

```
export const createMockTokenState = ({  
  ...
```

### **enterprise/frontend/src/embedding-sdk/test/server-mocks/sdk-init.ts**

```
import {  
  setupCurrentUserEndpoint,  
  setupPropertiesEndpoints,  
  setupSettingsEndpoints,  
} from "__support__/server-mocks";  
import { mockSettings } from "__support__/settings";
```

## Analyst Base

```
import type {...
```

### **enterprise/frontend/src/embedding-sdk/test/storybook-id-args.ts**

```
const QUESTION_ENTITY_ID = "_GiVL6zYmsnBb1oqLCp4u";
const DASHBOARD_ENTITY_ID = "xBLdW9FsgRuB2HGhWiBa_";
const COLLECTION_ENTITY_ID = "HyB3nRtqb7pBPhFG26evl";
```

```
export const questionIds =...
```

### **enterprise/frontend/src/embedding-sdk/test/storybook-themes.ts**

```
import {
  type MetabaseTheme,
  defineMetabaseTheme,
} from "metabase/embedding-sdk/theme";
```

```
export const darkColors = {
  primary: "#DF75E9",
  filter: "#7ABBF9",
  lighterGrey: "#E3E7E4",
  ...
```

### **enterprise/frontend/src/embedding-sdk/types/auth-config.ts**

```
import type { MetabaseFetchRequestTokenFn } from "embedding-sdk/types/refresh-token";
```

```
/**
 * @inline
 */
type BaseMetabaseAuthConfig = {
  metabaseInstanceUrl: string;
};
```

```
/**
 * @category...
```

### **enterprise/frontend/src/embedding-sdk/types/collection.ts**

```
import type { SdkUserId } from "embedding-sdk/types/user";
```

```
import type { SdkEntityId } from "../entity-id";
```

```
// "CollectionId" from core app also includes "root" | "users" and "trash", we don't want...
```

### **enterprise/frontend/src/embedding-sdk/types/dashboard.ts**

```
import type {
  MetabaseCollection,
```

## Analyst Base

```
    SdkCollectionId,  
  } from "embedding-sdk/types/collection";  
import type { CreateDashboardProperties } from...
```

### **enterprise/frontend/src/embedding-sdk/types/entity-id.ts**

eslint-disable-next-line @typescript-eslint/ban-types -- allows for other string literal types in other ID types

### **enterprise/frontend/src/embedding-sdk/types/events.ts**

```
import type { MetabaseDashboard } from "embedding-sdk/types/dashboard";
```

```
export type SdkDashboardLoadEvent = (  
  dashboard: MetabaseDashboard | null,  
) => void;
```

```
export type SdkEventHandlersConfig =...
```

### **enterprise/frontend/src/embedding-sdk/types/globalTypes.d.ts**

eslint-disable @typescript-eslint/consistent-type-imports

### **enterprise/frontend/src/embedding-sdk/types/index.ts**

```
export type * from "./auth-config";  
export type * from "./collection";  
export type * from "./dashboard";  
export type * from "./entity-id";  
export type * from "./events";  
export type * from...
```

### **enterprise/frontend/src/embedding-sdk/types/metabase-provider.ts**

```
import type { JSX, ReactNode } from "react";
```

```
import type { MetabaseTheme } from "metabase/embedding-sdk/theme";
```

```
import type { MetabaseAuthConfig } from "./auth-config";
```

```
import type {...
```

### **enterprise/frontend/src/embedding-sdk/types/plugins.ts**

```
import           type           {           DashboardCardMenu           }           from  
"metabase/dashboard/components/DashCard/DashCardMenu/dashcard-menu";
```

```
export type MetabaseClickAction = {  
  name: string;  
} & Record<string, any>;
```

```
export type...
```

## **enterprise/frontend/src/embedding-sdk/types/props.ts**

```
import type { CSSProperties } from "react";

/**
 * @inline
 */
export type CommonStylingProps = {
  /**
   * A custom class name to be added to the root element.
   */
  className?: string;

  /**
   ...
```

## **enterprise/frontend/src/embedding-sdk/types/question.ts**

```
import type { ReactNode } from "react";

import type { Deferred } from "metabase/lib/promise";
import type { QueryParams } from "metabase/query_builder/actions";
import type { ObjectId } from...
```

## **enterprise/frontend/src/embedding-sdk/types/refresh-token.ts**

```
export type MetabaseEmbeddingSessionToken = {
  id: string;
  exp: number;
};

/**
 * @inline
 */
export type UserBackendJwtResponse = {
  jwt: string;
};
```

```
export type MetabaseFetchRequestTokenFn =...
```

## **enterprise/frontend/src/embedding-sdk/types/ui.ts**

```
import type { JSX, ReactNode } from "react";

export type { ButtonProps } from "metabase/ui";
export type {
  ChartColor,
```

## Analyst Base

```
MetabaseTheme,  
MetabaseColors,  
MetabaseComponentTheme,  
} from...
```

### **enterprise/frontend/src/embedding-sdk/types/usage-problem.ts**

```
import type { USAGE_PROBLEM_MESSAGES } from "embedding-sdk/lib/usage-problem";  
  
export interface SdkUsageProblem {  
  type: SdkUsageProblemKey;  
  severity: "warning" | "error";  
  title: string;  
  ...  
}
```

### **enterprise/frontend/src/embedding-sdk/types/user.ts**

```
export type SdkUserId = number;  
  
/**  
 * The User entity  
 */  
export type MetabaseUser = {  
  id: SdkUserId;  
  first_name: string | null;  
  last_name: string | null;  
  common_name: string;  
  email:...
```

### **enterprise/frontend/src/metabase-enterprise/advanced\_permissions/component s/ImpersonationModal/index.ts**

```
export * from "./ImpersonationModal";
```

### **enterprise/frontend/src/metabase-enterprise/advanced\_permissions/component s/ImpersonationWarning/index.ts**

```
export * from "./ImpersonationWarning";
```

### **enterprise/frontend/src/metabase-enterprise/advanced\_permissions/graph.ts**

```
import _ from "underscore";  
  
import type { EntityId } from "metabase/admin/permissions/types";  
import {  
  DataPermission,  
  DataPermissionValue,  
} from "metabase/admin/permissions/types";  
import {
```



...

## **enterprise/frontend/src/metabase-enterprise/advanced\_permissions/graph.unit.spec.ts**

```
import {
  DataPermission,
  DataPermissionValue,
} from "metabase/admin/permissions/types";
import Database from "metabase-lib/v1/metadata/Database";
import Schema from...
```

## **enterprise/frontend/src/metabase-enterprise/advanced\_permissions/index.js**

```
import { push } from "react-router-redux";
import { t } from "ttag";

import { DataPermissionValue } from "metabase/admin/permissions/types";
import {
  getDatabaseFocusPermissionsUrl,
  ...
```

## **enterprise/frontend/src/metabase-enterprise/advanced\_permissions/reducer.ts**

```
import type { PayloadAction } from "@reduxjs/toolkit";
import { createAction, createSlice } from "@reduxjs/toolkit";
import { push } from "react-router-redux";

import {
  LOAD_DATA_PERMISSIONS,
  ...
```

## **enterprise/frontend/src/metabase-enterprise/advanced\_permissions/selectors.ts**

```
import type { DatabaseId, GroupId } from "metabase-types/api";

import type { AdvancedPermissionsStoreState } from "./types";

export const getImpersonation =
  (databaseId: DatabaseId, groupId:...
```

## **enterprise/frontend/src/metabase-enterprise/advanced\_permissions/services.js**

```
import { GET } from "metabase/lib/api";

export const ImpersonationApi = {
  get: GET("/api/ee/advanced-permissions/impersonation"),
};
```

## Analyst Base

### **enterprise/frontend/src/metabase-enterprise/advanced\_permissions/types.ts**

```
import type { PartialBy } from "metabase/common/types";
import type { EnterpriseSharedState } from "metabase-enterprise/shared/reducer";
import type { EnterpriseState } from...
```

### **enterprise/frontend/src/metabase-enterprise/advanced\_permissions/utils.ts**

```
import type { EntityId } from "metabase/admin/permissions/types";
import {
  getDatabaseFocusPermissionsUrl,
  getGroupFocusPermissionsUrl,
} from "metabase/admin/permissions/urls";
import type...
```

### **enterprise/frontend/src/metabase-enterprise/ai-entity-analysis/actions.ts**

```
import { setSidebar } from "metabase/dashboard/actions";
import { SIDEBAR_NAME } from "metabase/dashboard/constants";
import type { DashCardId } from "metabase-types/api";
import type { Dispatch }...
```

### **enterprise/frontend/src/metabase-enterprise/ai-entity-analysis/components/AIQuestionAnalysisSidebar/index.ts**

```
export { AIQuestionAnalysisSidebar } from "./AIQuestionAnalysisSidebar";
```

### **enterprise/frontend/src/metabase-enterprise/ai-entity-analysis/components/AIQuestionAnalysisSidebar/utils.ts**

```
import { isNotNull } from "metabase/lib/types";
import type { CollectionId, Timeline, TimelineEvent } from "metabase-types/api";
```

```
export const getTimelineEventsForAnalysis = (
  ...
```

### **enterprise/frontend/src/metabase-enterprise/ai-entity-analysis/hooks/useDashCardAnalysis.ts**

```
import { useEffect, useRef } from "react";
import { usePrevious } from "react-use";

import {
  getChartImagePngDataUri,
  getChartSelector,
} from "metabase/visualizations/lib/image-exports";
import...
```

### **enterprise/frontend/src/metabase-enterprise/ai-entity-analysis/index.ts**

```
import { t } from "ttag";
```

## Analyst Base

```
import {
  PLUGIN_AI_ENTITY_ANALYSIS,
  PLUGIN_DASHBOARD_MENU,
} from "metabase/plugins";
import { hasPremiumFeature } from "metabase-enterprise/settings";
import type...
```

### **enterprise/frontend/src/metabase-enterprise/ai-entity-analysis/utils.ts**

```
import type { VisualizationDisplay } from "metabase-types/api";
```

```
const CHART_ANALYSIS_ENABLED = {
  // enabled
  area: true,
  bar: true,
  combo: true,
  funnel: true,
  gauge: true,
  line: true,
  ...
```

### **enterprise/frontend/src/metabase-enterprise/ai-sql-fixer/components/FixSqlQueryButton/index.ts**

```
export * from "./FixSqlQueryButton";
```

### **enterprise/frontend/src/metabase-enterprise/ai-sql-fixer/index.ts**

```
import { PLUGIN_AI_SQL_FIXER } from "metabase/plugins";
import { hasPremiumFeature } from "metabase-enterprise/settings";

import { FixSqlQueryButton } from "./components/FixSqlQueryButton";

if...
```

### **enterprise/frontend/src/metabase-enterprise/ai-sql-generation/components/GenerateSqlQueryButton/index.ts**

```
export * from "./GenerateSqlQueryButton";
```

### **enterprise/frontend/src/metabase-enterprise/ai-sql-generation/index.ts**

```
import { PLUGIN_AI_SQL_GENERATION } from "metabase/plugins";
import { hasPremiumFeature } from "metabase-enterprise/settings";

import { GenerateSqlQueryButton } from...
```

### **enterprise/frontend/src/metabase-enterprise/ai-sql-generation/utils.ts**

## Analyst Base

```
import { t } from "ttag";
```

```
export function getPlaceholderText() {  
  return t`Write and select text to generate SQL with Metabot, or type SQL directly`;  
}
```

### **enterprise/frontend/src/metabase-enterprise/api/ai-entity-analysis.ts**

```
import type {  
  AIEntityAnalysisResponse,  
  AIQuestionAnalysisParams,  
} from "metabase-types/api";
```

```
import { EnterpriseApi } from "../api";
```

```
const DEFAULT_TIMEOUT = 30000;
```

```
export const...
```

### **enterprise/frontend/src/metabase-enterprise/api/ai-sql-generation.ts**

```
import type {  
  GenerateSqlQueryRequest,  
  GenerateSqlQueryResponse,  
} from "metabase-types/api";
```

```
import { EnterpriseApi } from "../api";
```

```
export const aiSqlGenerationApi =...
```

### **enterprise/frontend/src/metabase-enterprise/api/ai-streaming/index.ts**

```
export { aiStreamingQuery, getInflightRequestsForUrl } from "../requests";  
export type { JSONValue } from "../types";
```

### **enterprise/frontend/src/metabase-enterprise/api/ai-streaming/process-stream.ts**

```
import { match } from "ts-pattern";  
import _ from "underscore";
```

```
import type { MetabotHistory } from "metabase-types/api";
```

```
import {  
  type KnownDataPart,  
  dataPartSchema,  
  finishPartSchema,  
  ...  
}
```

### **enterprise/frontend/src/metabase-enterprise/api/ai-streaming/process-stream.u**

### **nit.spec.ts**

```
import { processChatResponse } from "../process-stream";
import { createMockReadableStream } from "../test-utils";
```

```
const getMockedCallbacks = () => ({
  onTextPart: jest.fn(),
  onDataPart:...
```

### **enterprise/frontend/src/metabase-enterprise/api/ai-streaming/requests.ts**

```
import { nanoid } from "@reduxjs/toolkit";
```

```
import api from "metabase/lib/api";
```

```
import { type AIStreamingConfig, processChatResponse } from "../process-stream";
import type { JSONValue } from...
```

### **enterprise/frontend/src/metabase-enterprise/api/ai-streaming/requests.unit.spec.ts**

```
import fetchMock from "fetch-mock";
```

```
import { aiStreamingQuery, getInflightRequestsForUrl } from "../requests";
import { mockStreamedEndpoint } from "../test-utils";
```

```
const ENDPOINT =...
```

### **enterprise/frontend/src/metabase-enterprise/api/ai-streaming/schemas.ts**

```
import * as Yup from "yup";
```

```
export const dataPartSchema = Yup.object({
  type: Yup.string().required(),
  version: Yup.number().required(),
  value: Yup.mixed(),
});
```

```
export const knownDataPartTypes...
```

### **enterprise/frontend/src/metabase-enterprise/api/ai-streaming/test-utils.ts**

```
import { defer } from "metabase/lib/promise";
```

```
async function delay(timeout: number) {
  return new Promise((resolve) => {
    setTimeout(() => resolve(undefined), timeout);
  });
}
```

## Analyst Base

```
}
```

```
export function...
```

### **enterprise/frontend/src/metabase-enterprise/api/ai-streaming/types.ts**

```
*
```

### **enterprise/frontend/src/metabase-enterprise/api/api.ts**

```
import { Api } from "metabase/api";
```

```
import { ENTERPRISE_TAG_TYPES } from "../tags";
```

```
export const EnterpriseApi = Api.enhanceEndpoints({  
  addTagTypes: ENTERPRISE_TAG_TYPES,  
});
```

### **enterprise/frontend/src/metabase-enterprise/api/audit-info.ts**

```
import type { AuditInfo } from "metabase-enterprise/audit_app/types/state";
```

```
import { EnterpriseApi } from "../api";
```

```
export const auditInfoApi = EnterpriseApi.injectEndpoints({  
  endpoints: (builder)...
```

### **enterprise/frontend/src/metabase-enterprise/api/billing-info.ts**

```
import type { BillingInfo } from "metabase-types/api";
```

```
import { EnterpriseApi } from "../api";
```

```
export const billingInfoApi = EnterpriseApi.injectEndpoints({  
  endpoints: (builder) => ({  
    ...
```

### **enterprise/frontend/src/metabase-enterprise/api/collection.ts**

```
import { provideCollectionItemListTags } from "metabase/api/tags";
```

```
import type {  
  ListStaleCollectionItemsRequest,  
  ListStaleCollectionItemsResponse,  
} from...
```

### **enterprise/frontend/src/metabase-enterprise/api/content-translation.ts**

```
import { invalidateTags, listTag } from "metabase/api/tags";  
import { contentTranslationEndpoints } from "metabase-enterprise/content_translation/constants";  
import type { DictionaryResponse } from...
```

## Analyst Base

### **enterprise/frontend/src/metabase-enterprise/api/database-replication.ts**

```
import { invalidateTags, tag } from "metabase/api/tags";
import type { Databaseld } from "metabase-types/api";
```

```
import { EnterpriseApi } from "./api";
```

```
export const DatabaseReplicationApi =...
```

### **enterprise/frontend/src/metabase-enterprise/api/db-routing.ts**

```
import _ from "underscore";
```

```
import { idTag, invalidateTags, listTag } from "metabase/api/tags";
```

```
import type {
  CreateDestinationDatabaseRequest,
  Database,
  UpdateDatabaseRouterRequest,
} from...
```

### **enterprise/frontend/src/metabase-enterprise/api/gdrive.ts**

```
import type { Databaseld, GdrivePayload } from "metabase-types/api";
```

```
import { EnterpriseApi } from "./api";
```

```
export const gdriveApi = EnterpriseApi.injectEndpoints({
  endpoints: (builder) => ({
    ...
```

### **enterprise/frontend/src/metabase-enterprise/api/index.ts**

```
export * from "./ai-entity-analysis";
export * from "./ai-sql-generation";
export * from "./audit-info";
export * from "./api";
export * from "./billing-info";
export * from "./metabot";
export *...
```

### **enterprise/frontend/src/metabase-enterprise/api/metabot.ts**

```
import type {
  DeleteSuggestedMetabotPromptRequest,
  MetabotApiEntity,
  MetabotEntity,
  MetabotId,
  MetabotInfo,
```

## Analyst Base

```
PaginationRequest,  
PaginationResponse,  
SuggestedMetabotPromptsRequest,  
...
```

### **enterprise/frontend/src/metabase-enterprise/api/saml.ts**

```
import { invalidateTags, tag } from "metabase/api/tags";  
import type { EnterpriseSettings } from "metabase-types/api";  
  
import { EnterpriseApi } from "../api";  
  
type SAMLSettings = Pick<  
...
```

### **enterprise/frontend/src/metabase-enterprise/api/scim.ts**

```
import type {  
  MaskedScimApiKey,  
  UnmaskedScimApiKey,  
} from "metabase-enterprise/user_provisioning/types";  
  
import { EnterpriseApi } from "../api";  
  
export const scimApi =...
```

### **enterprise/frontend/src/metabase-enterprise/api/smtp-override.ts**

```
import { invalidateTags, tag } from "metabase/api/tags";  
import type { EmailSMTPOverrideSettings } from "metabase-types/api";  
  
import { EnterpriseApi } from "../api";  
  
export const smtpOverrideApi =...
```

### **enterprise/frontend/src/metabase-enterprise/api/tags.ts**

```
import type { TagDescription } from "@reduxjs/toolkit/query";  
  
export const ENTERPRISE_TAG_TYPES = [  
  "scim",  
  "metabot",  
  "metabot-entities-list",  
  "metabot-prompt-suggestions",  
  ...
```

### **enterprise/frontend/src/metabase-enterprise/api/upload-management.ts**

```
import { invalidateTags, provideTableListTags, tag } from "metabase/api/tags";
```



## Analyst Base

```
import type {
  DeleteUploadTableRequest,
  UploadManagementResponse,
} from "metabase-types/api";
```

```
import {...
```

### **enterprise/frontend/src/metabase-enterprise/application\_permissions/api.js**

```
import { GET, PUT } from "metabase/lib/api";
```

```
export const ApplicationPermissionsApi = {
  graph: GET("/api/ee/advanced-permissions/application/graph"),
  updateGraph:...
```

### **enterprise/frontend/src/metabase-enterprise/application\_permissions/constants.ts**

```
import { t } from "ttag";
```

```
export const APPLICATION_PERMISSIONS_OPTIONS = {
  yes: {
    // eslint-disable-next-line ttag/no-module-declaration -- see metabase#55045
    label: t`Yes`,
    value:...
```

### **enterprise/frontend/src/metabase-enterprise/application\_permissions/index.ts**

```
import { t } from "ttag";
```

```
import {
  PLUGIN_ADMIN_ALLOWED_PATH_GETTERS,
  PLUGIN_APPLICATION_PERMISSIONS,
  PLUGIN_REDUCERS,
} from "metabase/plugins";
import { hasPremiumFeature } from...
```

### **enterprise/frontend/src/metabase-enterprise/application\_permissions/pages/ApplicationPermissionsPage/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **enterprise/frontend/src/metabase-enterprise/application\_permissions/reducer.js**

```
import { assocIn } from "icepick";
import { t } from "ttag";
```

```
import { getErrorMessage } from "metabase/api/utils";
```

## Analyst Base

```
import {
  combineReducers,
  createAction,
  createThunkAction,
  handleActions,
}...
```

### **enterprise/frontend/src/metabase-enterprise/application\_permissions/selectors.ts**

```
import { createSelector } from "@reduxjs/toolkit";
import { getIn } from "icepick";
import { t } from "ttag";
import _ from "underscore";
```

```
import { UNABLE_TO_CHANGE_ADMIN_PERMISSIONS } from...
```

### **enterprise/frontend/src/metabase-enterprise/application\_permissions/types/permissions.ts**

```
import type { GroupId } from "metabase-types/api";
```

```
export type ApplicationPermissionKey =
  | "subscription"
  | "monitoring"
  | "setting";
export type ApplicationPermissionValue = "yes" |...
```

### **enterprise/frontend/src/metabase-enterprise/application\_permissions/types/state.ts**

```
import type { State } from "metabase-types/store";
```

```
import type { ApplicationPermissions } from "../permissions";
import type { UserWithApplicationPermissions } from "../user";
```

```
export interface...
```

### **enterprise/frontend/src/metabase-enterprise/application\_permissions/types/user.ts**

```
export type { UserWithApplicationPermissions } from "metabase/plugins";
```

### **enterprise/frontend/src/metabase-enterprise/application\_permissions/utils.ts**

```
import type { AdminPathKey } from "metabase-types/store";
```

```
import type { UserWithApplicationPermissions } from "../types/user";
```

## Analyst Base

```
const canAccessMonitoringItems = (user?:...
```

### **enterprise/frontend/src/metabase-enterprise/audit\_app/components/AuditTableVisualization/index.ts**

```
export * from "./AuditTableVisualization";
```

### **enterprise/frontend/src/metabase-enterprise/audit\_app/containers/UnsubscribeUserModal/index.ts**

```
export { UnsubscribeUserModal } from "./UnsubscribeUserModal";
```

### **enterprise/frontend/src/metabase-enterprise/audit\_app/index.js**

```
import { t } from "ttag";
```

```
import { ForwardRefLink } from "metabase/common/components/Link";
```

```
import {  
  PLUGIN_ADMIN_USER_MENU_ITEMS,  
  PLUGIN_ADMIN_USER_MENU_ROUTES,  
  PLUGIN_AUDIT,  
} from...
```

### **enterprise/frontend/src/metabase-enterprise/audit\_app/lib/cards/queries.js**

```
import { t } from "ttag";
```

```
export const viewsAndAvgExecutionTimeByDay = () => ({  
  card: {  
    name: t`Query views and speed per day`,  
    display: "line",  
    dataset_query: {  
      type:...
```

### **enterprise/frontend/src/metabase-enterprise/audit\_app/lib/mode.js**

```
import { push } from "react-router-redux";  
import _ from "underscore";
```

```
export const getColumnName = (column) => column.remapped_to || column.name;
```

```
export const getRowValuesByColumns = (row, cols)...
```

### **enterprise/frontend/src/metabase-enterprise/audit\_app/types/state.ts**

```
import type { CardId, CollectionId, DashboardId } from "metabase-types/api";
```

```
export interface AuditInfo {  
  dashboard_overview: DashboardId;  
  question_overview: CardId;
```

## Analyst Base

custom\_reports:...

### **enterprise/frontend/src/metabase-enterprise/audit\_app/utils.ts**

```
import type { Database } from "metabase-types/api";
```

```
export const isAuditDb = (db: Database) => !!db.is_audit;
```

### **enterprise/frontend/src/metabase-enterprise/audit\_app/utils.unit.spec.ts**

```
import { createMockDatabase } from "metabase-types/api/mocks";
```

```
import { isAuditDb } from "../utils";
```

```
describe("enterprise audit utils", () => {  
  describe("isAuditDb", () => {  
    it("should return...
```

### **enterprise/frontend/src/metabase-enterprise/auth/actions.ts**

```
import { createAsyncThunk } from "@reduxjs/toolkit";
```

```
import { redirect } from "metabase/lib/dom";
```

```
import { getSetting } from "metabase/selectors/settings";
```

```
import type { State } from...
```

### **enterprise/frontend/src/metabase-enterprise/auth/components/SessionTimeoutSetting/index.ts**

```
export * from "../SessionTimeoutSetting";
```

### **enterprise/frontend/src/metabase-enterprise/auth/components/SettingsJWTForm/index.ts**

```
export * from "../SettingsJWTForm";
```

### **enterprise/frontend/src/metabase-enterprise/auth/components/SettingsSAMLForm/index.ts**

```
export * from "../SettingsSAMLForm";
```

### **enterprise/frontend/src/metabase-enterprise/auth/components/SsoButton/index.ts**

```
export * from "../SsoButton";
```

### **enterprise/frontend/src/metabase-enterprise/auth/constants.ts**

```
import * as Yup from "yup";
```

```
export const JWT_SCHEMA = Yup.object({  
  "jwt-enabled": Yup.boolean().default(false),  
  "jwt-user-provisioning-enabled?": Yup.boolean().default(null),
```

...

## **enterprise/frontend/src/metabase-enterprise/auth/containers/JwtAuthCard/index.ts**

```
export * from "./JwtAuthCard";
```

## **enterprise/frontend/src/metabase-enterprise/auth/containers/SamlAuthCard/index.ts**

```
export * from "./SamlAuthCard";
```

## **enterprise/frontend/src/metabase-enterprise/auth/index.ts**

```
import { LOGIN, LOGIN_GOOGLE } from "metabase/auth/actions";
import MetabaseSettings from "metabase/lib/settings";
import {
  PLUGIN_AUTH_PROVIDERS,
  PLUGIN_IS_PASSWORD_USER,
  ...
```

## **enterprise/frontend/src/metabase-enterprise/auth/middleware/session-middleware.js**

```
import Cookies from "js-cookie";
import { replace } from "react-router-redux";
```

```
import { logout, refreshSession } from "metabase/auth/actions";
import { isSameOrSiteUrlOrigin } from...
```

## **enterprise/frontend/src/metabase-enterprise/auth/middleware/session-middleware.unit.spec.js**

```
import FakeTimers from "@sinonjs/fake-timers";
import Cookie from "js-cookie";
import { replace } from "react-router-redux";
```

```
import { logout, refreshSession } from "metabase/auth/actions";
```

```
import...
```

## **enterprise/frontend/src/metabase-enterprise/auth/utils.ts**

```
import { useGetSettingsQuery } from "metabase/api";
import { hasAnySsoFeature } from "metabase/common/utils/plan";
```

```
export const getSSOUrl = (siteUrl: string, redirectUrl?: string): string => {
  if...
```

## **enterprise/frontend/src/metabase-enterprise/caching/constants.ts**

## Analyst Base

```
import { t } from "ttag";
import * as Yup from "yup";
```

```
import {
  getPerformanceTabMetadata,
  getPositiveIntegerSchema,
} from "metabase/admin/performance/constants/complex";
import {
  ...
```

### **enterprise/frontend/src/metabase-enterprise/caching/utils.js**

```
export function hasQuestionCacheSection(question) {
  const type = question.type();

  return (
    type !== "model" &&
    (question.canWrite() || question.lastQueryStart() !== null)
  );
}
```

### **enterprise/frontend/src/metabase-enterprise/clean\_up/CleanupCollectionModal/hooks.ts**

```
import { useCallback, useState } from "react";

export const usePagination = ({
  pageSize: inputPageSize,
  initialPage,
}): {
  pageSize: number;
  initialPage: number;
} => {
  const pageSize =...
```

### **enterprise/frontend/src/metabase-enterprise/clean\_up/CleanupCollectionModal/utils.ts**

```
import dayjs from "dayjs";
import { c, t } from "ttag";
import _ from "underscore";
```

```
import type { StaleCollectionItem } from "../types";
```

```
// constant portion of this string is how mantine calculates...
```

### **enterprise/frontend/src/metabase-enterprise/clean\_up/analytics.ts**

## Analyst Base

```
import { trackSchemaEvent } from "metabase/lib/analytics";
import type { RegularCollectionId } from "metabase-types/api";
```

```
export const trackStaleItemsArchived = ({
  collection_id,
  ...
```

### **enterprise/frontend/src/metabase-enterprise/clean\_up/types.ts**

```
import type {
  CollectionId,
  CollectionItem,
  PaginationRequest,
  PaginationResponse,
} from "metabase-types/api";
import type { SortingOptions } from "metabase-types/api/sorting";
```

```
export type...
```

### **enterprise/frontend/src/metabase-enterprise/clean\_up/utils.ts**

```
import {
  isInstanceAnalyticsCustomCollection,
  isTrashedCollection,
} from "metabase/collections/utils";
import type { Collection } from "metabase-types/api";
```

```
export function...
```

### **enterprise/frontend/src/metabase-enterprise/clean\_up/utils.unit.spec.ts**

```
import { PLUGIN_COLLECTIONS } from "metabase/plugins";
import { createMockCollection } from "metabase-types/api/mocks";
```

```
import { canCleanUp } from "../utils";
```

```
describe("canCleanUp", () => {
  ...
```

### **enterprise/frontend/src/metabase-enterprise/collections/constants.ts**

```
import { t } from "ttag";
```

```
import type {
  BaseEntityId,
  CollectionAuthorityLevelConfig,
  CollectionInstanceAnalyticsConfig,
} from "metabase-types/api";
```

```
export const REGULAR_COLLECTION:...
```

### **enterprise/frontend/src/metabase-enterprise/collections/use-get-default-collection-id/index.ts**

```
export * from "../use-get-default-collection-id";
```

### **enterprise/frontend/src/metabase-enterprise/collections/use-get-default-collection-id/use-get-default-collection-id.ts**

```
import { skipToken, useGetCollectionQuery } from "metabase/api";
import { _useGetDefaultCollectionId as useOSSGetDefaultCollectionId } from "metabase/collections/hooks";
import { useGetAuditInfoQuery...
```

### **enterprise/frontend/src/metabase-enterprise/collections/utils.ts**

```
import type { IconData, ObjectWithModel } from "metabase/lib/icon";
import { getIconBase } from "metabase/lib/icon";
import type { ItemWithCollection } from "metabase/plugins";
import type {
  ...
```

### **enterprise/frontend/src/metabase-enterprise/content\_translation/components/index.ts**

```
export {
  ContentTranslationConfiguration
} from
"./ContentTranslationConfiguration/ContentTranslationConfiguration";
```

### **enterprise/frontend/src/metabase-enterprise/content\_translation/constants.ts**

```
export const contentTranslationEndpoints = {
  /** This endpoint, which includes a JSON Web Token, is set in static embedding */
  getDictionary: null as string | null,
  uploadDictionary:...
```

### **enterprise/frontend/src/metabase-enterprise/content\_translation/index.ts**

```
import { PLUGIN_CONTENT_TRANSLATION } from "metabase/plugins";
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import { ContentTranslationConfiguration } from...
```

### **enterprise/frontend/src/metabase-enterprise/content\_translation/tests/utils.unit.spec.ts**

```
import type { HoveredObject } from "metabase/visualizations/types";
import type { DatasetColumn, DictionaryArray } from "metabase-types/api";
import { createMockColumn, createMockSeries } from...
```

### **enterprise/frontend/src/metabase-enterprise/content\_translation/use-translate-content.ts**



## Analyst Base

```
import { useCallback } from "react";
```

```
import { skipToken } from "metabase/api";
```

```
import { useLocale } from "metabase/common/hooks";
```

```
import type { ContentTranslationFunction } from...
```

### **enterprise/frontend/src/metabase-enterprise/content\_translation/utils.ts**

```
import * as I from "icepick";
```

```
import { useCallback, useMemo } from "react";
```

```
import _ from "underscore";
```

```
import type { ContentTranslationFunction } from "metabase/i18n/types";
```

```
import type {...
```

### **enterprise/frontend/src/metabase-enterprise/content\_verification/VerifiedFilter/index.ts**

```
export * from "./VerifiedFilter";
```

### **enterprise/frontend/src/metabase-enterprise/content\_verification/index.ts**

```
import { PLUGIN_CONTENT_VERIFICATION } from "metabase/plugins";
```

```
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import { VerifiedFilter } from "./VerifiedFilter";
```

```
import {...
```

### **enterprise/frontend/src/metabase-enterprise/database\_routing/DestinationDatabaseConnectionModal/utils.ts**

```
import _ from "underscore";
```

```
import type { Database } from "metabase-types/api";
```

```
const FILTER_DB_KEYS = ["id", "name"];
```

```
const FILTER_DB_DETAIL_KEYS = [
```

```
  "password",
```

```
  "schema-filters-type",
```

```
  ...
```

### **enterprise/frontend/src/metabase-enterprise/database\_routing/hooks.ts**

```
import { useEffect } from "react";
```

```
import { replace } from "react-router-redux";
```

```
import { useDispatch } from "metabase/lib/redux";
```

```
import * as Urls from "metabase-enterprise/urls";
```

## Analyst Base

```
import type {...
```

### **enterprise/frontend/src/metabase-enterprise/database\_routing/utils.ts**

```
import { skipToken } from "@reduxjs/toolkit/query";
```

```
export function paramIdToGetQuery(strId: string | undefined) {  
  const id = strId ? parseInt(strId, 10) : undefined;  
  return id ? { id } : ...
```

### **enterprise/frontend/src/metabase-enterprise/embedding/components/EmbeddingAppOriginDescription/index.ts**

```
export { EmbeddingAppOriginDescription } from "./EmbeddingAppOriginDescription";
```

### **enterprise/frontend/src/metabase-enterprise/embedding/components/EmbeddingAppSameSiteCookieDescription/index.ts**

```
export { SameSiteSelectWidget } from "./SameSiteSelectWidget";
```

### **enterprise/frontend/src/metabase-enterprise/embedding/index.ts**

```
import { DataSourceSelector } from "embedding/data-picker/DataSelector";  
import { SimpleDataPicker } from "embedding/data-picker/SimpleDataPicker";  
import { PLUGIN_ADMIN_SETTINGS, PLUGIN_EMBEDDING } ...
```

### **enterprise/frontend/src/metabase-enterprise/embedding/selectors.ts**

```
import { getPlan } from "metabase/common/utils/plan";  
import { getSetting } from "metabase/selectors/settings";  
import type { EnterpriseState } from "metabase-enterprise/settings/types";
```

```
export...
```

### **enterprise/frontend/src/metabase-enterprise/embedding-sdk/index.ts**

```
import { PLUGIN_EMBEDDING_SDK } from "metabase/plugins";  
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
if (hasPremiumFeature("embedding_sdk")) {  
  PLUGIN_EMBEDDING_SDK.isEnabled...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/constants.ts**

```
import type {  
  DashboardEmbedOptions,  
  ExplorationEmbedOptions,  
  QuestionEmbedOptions,  
  SdkIframeEmbedBaseSettings,  
  SdkIframeEmbedSettingKey,
```

## Analyst Base

```
} from "../types/embed";
```

```
/**
```

```
 * The timeout to wait...
```

**enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/embed.ts**

```
import {
  connectToInstanceAuthSso,
  jwtDefaultRefreshTokenFunction,
  openSamlLoginPopup,
  validateSessionToken,
} from "embedding/auth-common";
import { INVALID_AUTH_METHOD, MetabaseError } from...
```

**enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/embed.unit.spec.ts**

```
import { act } from "react-dom/test-utils";

import type { MetabaseEmbedElement } from "../embed";
```

```
const defineMetabaseConfig = (config: unknown) => {
  (window as any).metabaseConfig =...
```

**enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/hooks/use-param-rerender-key.ts**

```
import { useMemo } from "react";
import { P, match } from "ts-pattern";

import { sortObject } from "metabase-lib/v1/utls";
```

```
import type { SdkIframeEmbedSettings } from...
```

**enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/hooks/use-sdk-iframe-embed-event-bus.ts**

```
import { useEffect, useState } from "react";
import { match } from "ts-pattern";

import { isWithinIframe } from "metabase/lib/dom";
```

```
import type {
  SdkIframeEmbedMessage,
  ...
```

**enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/index.ts**

## Analyst Base

```
import { PLUGIN_EMBEDDING_IFRAME_SDK } from "metabase/plugins";
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import { SdkIframeEmbedRoute } from...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/types/embed.ts**

```
import type {
  EntityTypeFilterKeys,
  MetabaseTheme,
  SqlParameterValues,
} from "embedding-sdk";
import type { MetabaseError } from "embedding-sdk/errors";
import type { MetabaseAuthMethod } from...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/types/store.ts**

```
import type { Action, Store } from "@reduxjs/toolkit";

import type { SdkStoreState } from "embedding-sdk/store/types";
```

```
export type StoreWithSdkState = Store<SdkStoreState, Action>;
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/utils/index.ts**

```
export { requestSessionTokenFromEmbedJs } from "./request-session-token";
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/utils/request-session-token.ts**

```
import { samlTokenStorage } from "embedding/auth-common";
import { AUTH_TIMEOUT } from "embedding-sdk/errors";
import type { MetabaseEmbeddingSessionToken } from...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/webcomponents.ts**

```
import json5 from "json5";
```

```
// Convert kebab-case attribute names to their camelCase setting keys.
```

```
export const attributeToSettingKey = (attr: string): string =>
  attr.replace(/-([a-z])/g, (_, c) =>...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk/webcomponents.unit.spec.ts**

```
import { parseAttributeValue } from "./webcomponents";
```

## Analyst Base

// Note, these tests use ` when to wrap strings passed to parseAttributeValue so we can only focus on ' vs " inside the...

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/analytics.ts**

```
import { trackSimpleEvent } from "metabase/lib/analytics";
import type {
  SdkIframeEmbedSettingKey,
  SdkIframeEmbedSettings,
} from "metabase-enterprise/embedding_iframe_sdk/types/embed";
```

import...

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/components/ParameterSettings/hooks/use-hide-parameter.ts**

```
import { useSdkIframeEmbedSetupContext } from "../../context";

export function useHideParameter() {
  const { settings, updateSettings } = useSdkIframeEmbedSetupContext();
```

const...

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/components/ParameterSettings/utils/parameter-placeholder.ts**

```
import { match } from "ts-pattern";

import type { Parameter, ParameterType } from "metabase-types/api";

export const getParameterPlaceholder = (param: Parameter): string => {
  // If the parameter...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/components/ParameterSettings/utils/parameter-placeholder.unit.spec.ts**

```
import type { Parameter } from "metabase-types/api";

import { getParameterPlaceholder } from "../parameter-placeholder";

const createParameter = (overrides: Partial<Parameter> = {}): Parameter =>...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/constants.ts**

```
import { GetCodeStep } from "../components/GetCodeStep";
```

## Analyst Base

```
import { SelectEmbedExperienceStep } from "../components/SelectEmbedExperienceStep";
import { SelectEmbedOptionsStep } from...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/context.ts**

```
import { createContext, useContext } from "react";
```

```
import type { Parameter } from "metabase-types/api";
```

```
import type {
  SdkIframeEmbedSetupExperience,
  SdkIframeEmbedSetupRecentItem,
  ...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/hooks/index.ts**

```
export { useSdkIframeEmbedNavigation } from "../use-sdk-iframe-embed-navigation";
export { useRecentItems } from "../use-recent-items";
export { useParameterList } from "../use-parameter-list";
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/hooks/use-parameter-list.ts**

```
import { useMemo } from "react";
import { useLatest } from "react-use";
```

```
import { skipToken, useGetCardQuery, useGetDashboardQuery } from "metabase/api";
import { useSelector } from...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/hooks/use-recent-items.ts**

```
import { useCallback, useMemo, useState } from "react";
```

```
import { useListRecentsQuery } from "metabase/api";
import type { RecentItem } from "metabase-types/api";
```

```
import {...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/hooks/use-sdk-iframe-embed-navigation.ts**

```
import { useMemo } from "react";
```

```
import { EMBED_STEPS } from "../constants";
import { useSdkIframeEmbedSetupContext } from "../context";
```

## Analyst Base

```
export function useSdkIframeEmbedNavigation() {  
  const {...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/hooks/use-sdk-iframe-embed-snippet.ts**

```
import { useMemo } from "react";
```

```
import { useSetting } from "metabase/common/hooks";
```

```
import { useSdkIframeEmbedSetupContext } from "../context";
```

```
import { getEmbedSnippet } from...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/index.ts**

```
import { PLUGIN_EMBEDDING_IFRAME_SDK_SETUP } from "metabase/plugins";
```

```
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import { SdkIframeEmbedSetup } from...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/types.ts**

```
import type {
```

```
  SdkIframeEmbedBaseSettings,
```

```
  SdkIframeEmbedTemplateSettings,
```

```
} from "metabase-enterprise/embedding_iframe_sdk/types/embed";
```

```
import type { BaseRecentItem } from...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/utils/default-embed-setting.ts**

```
import { match } from "ts-pattern";
```

```
import type {
```

```
  DashboardEmbedOptions,
```

```
  ExplorationEmbedOptions,
```

```
  QuestionEmbedOptions,
```

```
} from "metabase-enterprise/embedding_iframe_sdk/types/embed";
```

```
import...
```

### **enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/utils/embed-snippet.ts**

```
import { match } from "ts-pattern";
```

## Analyst Base

```
import _ from "underscore";
```

```
import {  
  ALLOWED_EMBED_SETTING_KEYS_MAP,  
  type AllowedEmbedSettingKey,  
} from...
```

**enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/utils/filter-empty-settings.ts**

```
*
```

**enterprise/frontend/src/metabase-enterprise/embedding\_iframe\_sdk\_setup/utils/theme-colors.ts**

```
import { t } from "ttag";
```

```
import type { MetabaseColors } from "embedding-sdk";  
import type { ColorName } from "metabase/lib/colors/types";
```

```
export const getConfigurableThemeColors = () =>  
[  
  {  
    ...
```

**enterprise/frontend/src/metabase-enterprise/feature\_level\_permissions/index.ts**

```
import {  
  PLUGIN_ADMIN_ALLOWED_PATH_GETTERS,  
  PLUGIN_FEATURE_LEVEL_PERMISSIONS,  
} from "metabase/plugins";  
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import {...
```

**enterprise/frontend/src/metabase-enterprise/feature\_level\_permissions/permission-management/data-model-permission.ts**

```
import { push } from "react-router-redux";  
import { t } from "ttag";
```

```
import {  
  UNABLE_TO_CHANGE_ADMIN_PERMISSIONS  
} from "metabase/admin/permissions/constants/messages";  
import {  
  ...
```

**enterprise/frontend/src/metabase-enterprise/feature\_level\_permissions/permission-management/data-model-permission.unit.spec.ts**



## Analyst Base

```
import { DataPermissionValue } from "metabase/admin/permissions/types";
import type { Group, GroupsPermissions } from "metabase-types/api";
```

```
import {
  DATA_MODEL_PERMISSION_OPTIONS,
  ...
```

### **enterprise/frontend/src/metabase-enterprise/feature\_level\_permissions/permission-management/details-permission.ts**

```
import { getIn } from "icepick";
import { t } from "ttag";
```

```
import {
  {
    UNABLE_TO_CHANGE_ADMIN_PERMISSIONS
  }
  from
  "metabase/admin/permissions/constants/messages";
import {
  getPermissionWarning,
  ...
```

### **enterprise/frontend/src/metabase-enterprise/feature\_level\_permissions/permission-management/details-permission.unit.spec.ts**

```
import { DataPermissionValue } from "metabase/admin/permissions/types";
import type { Group, GroupsPermissions } from "metabase-types/api";
```

```
import {
  DETAILS_PERMISSION_OPTIONS,
  ...
```

### **enterprise/frontend/src/metabase-enterprise/feature\_level\_permissions/permission-management/download-permission.ts**

```
import { push } from "react-router-redux";
import { t } from "ttag";
```

```
import {
  {
    UNABLE_TO_CHANGE_ADMIN_PERMISSIONS
  }
  from
  "metabase/admin/permissions/constants/messages";
import {
  ...
```

### **enterprise/frontend/src/metabase-enterprise/feature\_level\_permissions/permission-management/download-permission.unit.spec.ts**

```
import { DataPermissionValue } from "metabase/admin/permissions/types";
import type { Group, GroupsPermissions } from "metabase-types/api";
```

```
import {
```

## Analyst Base

DOWNLOAD\_PERMISSION\_OPTIONS,

...

### **enterprise/frontend/src/metabase-enterprise/feature\_level\_permissions/permission-management/index.ts**

```
import type {
  DataPermissionValue,
  EntityId,
  PermissionSubject,
} from "metabase/admin/permissions/types";
import type { Group, GroupsPermissions } from "metabase-types/api";
```

import {...

### **enterprise/frontend/src/metabase-enterprise/feature\_level\_permissions/query-downloads.ts**

```
import { t } from "ttag";

import type { Dataset } from "metabase-types/api/dataset";

export const canDownloadResults = (result: Dataset) =>
  result.data?.download_perms !== "none";
```

export const...

### **enterprise/frontend/src/metabase-enterprise/feature\_level\_permissions/types/user.ts**

```
import type { User } from "metabase-types/api";

export interface UserWithFeaturePermissions extends User {
  permissions?: {
    can_access_data_model: boolean;
    can_access_db_details: boolean;
  };
  ...
}
```

### **enterprise/frontend/src/metabase-enterprise/feature\_level\_permissions/utils.ts**

```
import { t } from "ttag";

import type { PermissionSubject } from "metabase/admin/permissions/types";
import type { AdminPathKey } from "metabase-types/store";

import type {...
```

### **enterprise/frontend/src/metabase-enterprise/google\_drive/GdriveConnectionMo**

### **dal.strings.ts**

```
import { t } from "ttag";

export const getStrings = (objectType: "file" | "folder") => {
  if (objectType === "file") {
    return {
      clickInstruction: t`In Google Drive, right-click on the...
```

### **enterprise/frontend/src/metabase-enterprise/google\_drive/analytics.ts**

```
import { trackSimpleEvent } from "metabase/lib/analytics";
import type { GsheetsConnectionClickedEvent } from "metabase-types/analytics";

export function trackSheetImportClick() {
  ...
```

### **enterprise/frontend/src/metabase-enterprise/google\_drive/constants.ts**

```
export const SYNC_POLL_INTERVAL = 3 * 1000; // 3 seconds
```

### **enterprise/frontend/src/metabase-enterprise/google\_drive/index.ts**

```
export * from "./GdriveAddDataPanel";
export * from "./GdriveConnectionModal";
export * from "./GdriveSyncStatus";
export * from "./GdriveDbMenu";
export * from "./PausedModal";
```

### **enterprise/frontend/src/metabase-enterprise/google\_drive/utils.unit.spec.ts**

```
import { getStatus } from "./utils";

describe("google_drive > getStatus", () => {
  it("should return 'not-connected' if status is undefined", () => {
    const result = getStatus({ status:...
```

### **enterprise/frontend/src/metabase-enterprise/group\_managers/actions.js**

```
import { push } from "react-router-redux";

import { getAdminPaths } from "metabase/admin/app/selectors";
import { permissionApi } from "metabase/api";
import { createThunkAction } from...
```

### **enterprise/frontend/src/metabase-enterprise/group\_managers/components/Use rTypeCell/index.ts**

```
export * from "./UserTypeCell";
```

### **enterprise/frontend/src/metabase-enterprise/group\_managers/components/Use**

## **rTypeToggle/index.ts**

```
export * from "../UserTypeToggle";
```

## **enterprise/frontend/src/metabase-enterprise/group\_managers/index.ts**

```
import {
  PLUGIN_ADMIN_ALLOWED_PATH_GETTERS,
  PLUGIN_GROUP MANAGERS,
} from "metabase/plugins";
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import {
```

```
...
```

## **enterprise/frontend/src/metabase-enterprise/group\_managers/types/user.ts**

```
import type { User } from "metabase-types/api";
```

```
export interface UserWithGroupManagerPermission extends User {
  permissions?: {
    is_group_manager: boolean;
  };
}
```

## **enterprise/frontend/src/metabase-enterprise/group\_managers/utils.ts**

```
import { t } from "ttag";
```

```
import type { ConfirmationState } from "metabase/common/hooks/use-confirmation";
import type { Member, Membership } from "metabase-types/api";
import type { User } from...
```

## **enterprise/frontend/src/metabase-enterprise/license/components/BillingInfo/utils.ts**

```
import { formatDateTimeWithUnit, formatNumber } from "metabase/lib/formatting";
import type { BillingInfoLineItem } from "metabase-types/api";
import {
  supportedDisplayTypes,
  ...
```

## **enterprise/frontend/src/metabase-enterprise/license/components/LicenseAndBillingSettings/index.ts**

```
export * from "../LicenseAndBillingSettings";
```

## **enterprise/frontend/src/metabase-enterprise/license/components/StillNeedHelp/index.ts**

```
export * from "../StillNeedHelp";
```

## Analyst Base

### **enterprise/frontend/src/metabase-enterprise/license/index.ts**

```
import { PLUGIN_ADMIN_SETTINGS } from "metabase/plugins";

import { LicenseAndBillingSettings } from "../components/LicenseAndBillingSettings";
import { useUpsellFlow } from...
```

### **enterprise/frontend/src/metabase-enterprise/license/use-upsell-flow.ts**

```
import { useEffect, useRef } from "react";
import { t } from "ttag";

import { getCurrentUser } from "metabase/admin/datamodel/selectors";
import { useUpsellLink } from...
```

### **enterprise/frontend/src/metabase-enterprise/metabot/README.md**

Metabot v3  
# Sending requests to Metabot  
## Request format

### **enterprise/frontend/src/metabase-enterprise/metabot/components/MetabotAdmin/test-utils.ts**

```
import type { MetabotId, SuggestedMetabotPrompt } from "metabase-types/api";

export const mockSuggestedPrompts: Record<MetabotId, SuggestedMetabotPrompt[]> =
{
  "1": [
    {
      id: 1,
      ...
    }
  ]
}
```

### **enterprise/frontend/src/metabase-enterprise/metabot/components/MetabotAdmin/Utils.ts**

```
import { useSelector } from "metabase/lib/redux";
import { getLocation } from "metabase/selectors/routing";

export function useMetabotIdPath() {
  const location = useSelector(getLocation);
  const...
```

### **enterprise/frontend/src/metabase-enterprise/metabot/components/MetabotChat/hooks.ts**

```
import { type RefObject, useCallback, useEffect, useRef } from "react";

function calculateFillerHeight(
```

## Analyst Base

```
scrollContainerEl: HTMLElement,  
fillerEl: HTMLElement,  
): number {  
  const scrollContent =...
```

### **enterprise/frontend/src/metabase-enterprise/metabot/components/MetabotIcon/index.ts**

```
export * from "./MetabotIcon";
```

### **enterprise/frontend/src/metabase-enterprise/metabot/constants.ts**

```
import { t } from "ttag";
```

```
export const LONG_CONVO_MSG_LENGTH_THRESHOLD = 120000;
```

```
// NOTE: this is not ideal, but will get fixed w/ BOT-189 allowing us to use fixed entity_ids  
export const...
```

### **enterprise/frontend/src/metabase-enterprise/metabot/hooks.ts**

```
import { isFulfilled } from "@reduxjs/toolkit";  
import { useCallback } from "react";
```

```
import { useDispatch, useSelector } from "metabase/lib/redux";  
import { useMetabotContext } from...
```

### **enterprise/frontend/src/metabase-enterprise/metabot/reactions/errors.ts**

```
import { t } from "ttag";
```

```
import { addAgentErrorMessage, stopProcessing } from "../state";
```

```
import type { ReactionHandler } from "../types";
```

```
type UnknownReaction = { type: string } & Record<string,...
```

### **enterprise/frontend/src/metabase-enterprise/metabot/reactions/index.ts**

```
import type { MetabotReaction } from "metabase-types/api";
```

```
import { showMessage } from "../messages";  
import { redirect } from "../metabot";  
import type { ReactionHandler } from "../types";
```

```
export *...
```

### **enterprise/frontend/src/metabase-enterprise/metabot/reactions/messages.ts**

```
import type { MetabotMessageReaction } from "metabase-types/api";
```

## Analyst Base

```
import { addAgentMessage } from "../state";
```

```
import type { ReactionHandler } from "../types";
```

```
export const showMessage:...
```

**enterprise/frontend/src/metabase-enterprise/metabot/reactions/metabot.ts**

```
import { push } from "react-router-redux";
```

```
import type { MetabotRedirectReaction } from "metabase-types/api";
```

```
import type { ReactionHandler } from "../types";
```

```
export const redirect:...
```

**enterprise/frontend/src/metabase-enterprise/metabot/reactions/types.ts**

```
import type { Dispatch, GetState } from "metabase-types/store";
```

```
export type ReactionHandler<Reaction> = (
```

```
  reaction: Reaction,
```

```
) => (reduxApis: {
```

```
  dispatch: Dispatch;
```

```
  getState: GetState;
```

```
}) =>...
```

**enterprise/frontend/src/metabase-enterprise/metabot/state/actions.ts**

```
import { type UnknownAction, isRejected } from "@reduxjs/toolkit";
```

```
import { push } from "react-router-redux";
```

```
import { P, match } from "ts-pattern";
```

```
import { createAsyncThunk } from...
```

**enterprise/frontend/src/metabase-enterprise/metabot/state/index.ts**

```
export * from "./actions";
```

```
export * from "./reducer";
```

```
export * from "./selectors";
```

```
export * from "../types";
```

**enterprise/frontend/src/metabase-enterprise/metabot/state/reducer.ts**

```
import { type PayloadAction, createSlice } from "@reduxjs/toolkit";
```

```
import _ from "underscore";
```

```
import { logout } from "metabase/auth/actions";
```

## Analyst Base

```
import { uuid } from "metabase/lib/uuid";
import type...
```

### **enterprise/frontend/src/metabase-enterprise/metabot/state/selectors.ts**

```
import { createSelector } from "@reduxjs/toolkit";
import _ from "underscore";
```

```
import { getIsEmbedding } from "metabase/selectors/embed";
```

```
import {
  FIXED_METABOT_IDS,
  ...
```

### **enterprise/frontend/src/metabase-enterprise/metabot/state/selectors.unit.spec.ts**

```
import { setupEnterprisePlugins } from "___support___/enterprise";
import { createMockState } from "metabase-types/store/mocks";
```

```
import {
  type MetabotState,
  getLastAgentMessagesByType,
  ...
```

### **enterprise/frontend/src/metabase-enterprise/metabot/state/types.ts**

```
import type { EnterpriseSharedState } from "metabase-enterprise/shared/reducer";
import type { EnterpriseState } from "metabase-enterprise/shared/types";
```

```
import type { MetabotState } from...
```

### **enterprise/frontend/src/metabase-enterprise/metabot/state/utils.ts**

```
import { nanoid } from "@reduxjs/toolkit";
```

```
export const createMessageId = () => {
  return `msg_${nanoid()}`;
};
```

### **enterprise/frontend/src/metabase-enterprise/model\_persistence/components/ModelCacheControl/index.ts**

```
export * from "./ModelCacheControl";
```

### **enterprise/frontend/src/metabase-enterprise/model\_persistence/index.ts**

```
import { PLUGIN_MODEL_PERSISTENCE } from "metabase/plugins";
import { hasPremiumFeature } from "metabase-enterprise/settings";
```



## Analyst Base

```
import { ModelCacheToggle } from "../components/ModelCacheControl";
```

```
if...
```

**enterprise/frontend/src/metabase-enterprise/moderation/components/EntityModerationIcon/index.ts**

```
export * from "../EntityModerationIcon";
```

**enterprise/frontend/src/metabase-enterprise/moderation/components/ModerationReviewBanner/index.ts**

```
export * from "../ModerationReviewBanner";
```

**enterprise/frontend/src/metabase-enterprise/moderation/components/ModerationReviewIcon/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**enterprise/frontend/src/metabase-enterprise/moderation/components/ModerationStatusIcon/index.ts**

```
export * from "../ModerationStatusIcon";
```

**enterprise/frontend/src/metabase-enterprise/moderation/components/QuestionModerationSection/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**enterprise/frontend/src/metabase-enterprise/moderation/constants.ts**

```
import type { ColorName } from "metabase/lib/colors/types";
```

```
import type { IconName } from "metabase/ui";
```

```
export const MODERATION_STATUS = {  
  verified: "verified",  
};
```

```
export const...
```

**enterprise/frontend/src/metabase-enterprise/moderation/containers/ModerationReviewIcon/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**enterprise/frontend/src/metabase-enterprise/moderation/index.ts**

```
import { PLUGIN_MODERATION } from "metabase/plugins";
```

```
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import { EntityModerationIcon } from...
```

**enterprise/frontend/src/metabase-enterprise/moderation/selectors.js**

## Analyst Base

```
import { getUserIsAdmin } from "metabase/selectors/user";
```

```
export const getIsModerator = (state, props) => {  
  return getUserIsAdmin(state, props);  
};
```

### **enterprise/frontend/src/metabase-enterprise/moderation/service.ts**

```
import { c, t } from "ttag";  
import _ from "underscore";
```

```
import type { ColorName } from "metabase/lib/colors/types";  
import { ModerationReviewApi } from "metabase/services";  
import type { IconName }...
```

### **enterprise/frontend/src/metabase-enterprise/moderation/service.unit.spec.ts**

```
import { ModerationReviewApi } from "metabase/services";  
import type Question from "metabase-lib/v1/Question";  
import type { ModerationReview } from "metabase-types/api";  
import {  
  ...
```

### **enterprise/frontend/src/metabase-enterprise/moderation/utils.ts**

```
import { t } from "ttag";
```

```
import type Question from "metabase-lib/v1/Question";
```

```
export const getVerifyQuestionTitle = (question: Question): string => {  
  const type = question.type();
```

```
  if (type...
```

### **enterprise/frontend/src/metabase-enterprise/overrides.js**

```
import "../whitelabel/overrides";
```

### **enterprise/frontend/src/metabase-enterprise/plugins.js**

```
import { PLUGIN_IS_EE_BUILD } from "metabase/plugins";
```

```
// SETTINGS OVERRIDES:
```

```
PLUGIN_IS_EE_BUILD.isEEBuild = () => true;
```

```
import "../shared";
```

```
// PLUGINS:
```

## Analyst Base

```
import "../tools";
import...
```

### **enterprise/frontend/src/metabase-enterprise/redux.ts**

```
import type { TypedUseSelectorHook } from "react-redux";
import { createSelectorHook } from "react-redux";
```

```
import { MetabaseReduxContext } from "metabase/lib/redux";
```

```
// TODO: use the real type...
```

### **enterprise/frontend/src/metabase-enterprise/resource\_downloads/index.ts**

```
import "../resource_downloads_plugin";
```

### **enterprise/frontend/src/metabase-enterprise/resource\_downloads/resource\_downloads\_plugin.ts**

```
import { P, match } from "ts-pattern";
```

```
import { PLUGIN_RESOURCE_DOWNLOADS } from "metabase/plugins";
import type { EmbedResourceDownloadOptions } from "metabase/public/lib/types";
import {...
```

### **enterprise/frontend/src/metabase-enterprise/resource\_downloads/tests/common.unit.spec.ts**

```
import { PLUGIN_RESOURCE_DOWNLOADS } from "metabase/plugins";
```

```
import { downloadsEnabledTestData, setup } from "../setup";
```

```
describe("[OSS] resource downloads plugin", () => {
  beforeAll(() => {
    ...
```

### **enterprise/frontend/src/metabase-enterprise/resource\_downloads/tests/enterprise.unit.spec.ts**

```
import { PLUGIN_RESOURCE_DOWNLOADS } from "metabase/plugins";
```

```
import { downloadsEnabledTestData, setup } from "../setup";
```

```
describe("[EE - no features] resource downloads plugin", () => {
  ...
```

### **enterprise/frontend/src/metabase-enterprise/resource\_downloads/tests/premium.unit.spec.ts**

```
import { PLUGIN_RESOURCE_DOWNLOADS } from "metabase/plugins";
```

## Analyst Base

```
import { createMockTokenFeatures } from "metabase-types/api/mocks";
```

```
import { downloadsEnabledTestData, setup } from...
```

### **enterprise/frontend/src/metabase-enterprise/resource\_downloads/tests/setup.ts**

```
import { setupEnterprisePlugins } from "__support__/enterprise";
```

```
import { mockSettings } from "__support__/settings";
```

```
import type { EmbedResourceDownloadOptions } from...
```

### **enterprise/frontend/src/metabase-enterprise/sandboxes/actions.js**

```
import _ from "underscore";
```

```
import {
```

```
  LOAD_DATA_PERMISSIONS,
```

```
  SAVE_DATA_PERMISSIONS,
```

```
  UPDATE_DATA_PERMISSION,
```

```
  updateDataPermission,
```

```
} from "metabase/admin/permissions/permissions";
```

```
import {
```

```
  ...
```

### **enterprise/frontend/src/metabase-enterprise/sandboxes/components/AttributeMappingEditor/index.ts**

```
export * from "./DataAttributeMappingEditor";
```

```
export * from "./AttributeOptionsEmptyState";
```

### **enterprise/frontend/src/metabase-enterprise/sandboxes/components/EditSandboxingModal/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **enterprise/frontend/src/metabase-enterprise/sandboxes/components/LoginAttributesWidget/index.ts**

```
export * from "./LoginAttributesWidget";
```

### **enterprise/frontend/src/metabase-enterprise/sandboxes/confirmations.ts**

```
import { t } from "ttag";
```

```
import {
```

```
  DataPermissionValue,
```

```
  type EntityId,
```

```
} from "metabase/admin/permissions/types";
```

```
import { hasPermissionValueInGraph } from...
```

### **enterprise/frontend/src/metabase-enterprise/sandboxes/index.js**

## Analyst Base

```
import { push } from "react-router-redux";
import { t } from "ttag";
import _ from "underscore";
```

```
import { DataPermissionValue } from "metabase/admin/permissions/types";
import {
  ...
```

### **enterprise/frontend/src/metabase-enterprise/sandboxes/selectors.ts**

```
import { createSelector } from "@reduxjs/toolkit";
```

```
import type { GroupTableAccessPolicyParams, SandboxesState } from "../types";
import { getPolicyKeyFromParams } from "../utils";
```

```
export const...
```

### **enterprise/frontend/src/metabase-enterprise/sandboxes/types.ts**

```
import type { EnterpriseSharedState } from "metabase-enterprise/shared/reducer";
import type { EnterpriseState } from "metabase-enterprise/shared/types";
import type {
  DimensionRef,
  ...
```

### **enterprise/frontend/src/metabase-enterprise/sdk-iframe-embedding-plugins.js**

```
*
```

### **enterprise/frontend/src/metabase-enterprise/sdk-plugins.js**

```
import { PLUGIN_IS_EE_BUILD } from "metabase/plugins";
```

```
// SETTINGS OVERRIDES:
```

```
PLUGIN_IS_EE_BUILD.isEEBuild = () => true;
```

```
import "../shared";
```

```
// SDK PLUGINS:
```

```
import "../embedding";
import...
```

### **enterprise/frontend/src/metabase-enterprise/services.ts**

```
import { DELETE, GET } from "metabase/lib/api";
```

```
export const AuditApi = {
  unsubscribe_user: DELETE("/api/ee/audit-app/user/:id/subscriptions"),
```

## Analyst Base

```
};
```

```
export const ImpersonationApi = {  
  get:...
```

### **enterprise/frontend/src/metabase-enterprise/settings/hooks/use-license.ts**

```
import { useCallback, useEffect, useState } from "react";  
import { t } from "ttag";
```

```
import { reload } from "metabase/lib/dom";  
import { SettingsApi, StoreApi } from "metabase/services";  
import type...
```

### **enterprise/frontend/src/metabase-enterprise/settings/index.ts**

```
import { isTest } from "metabase/env";  
import MetabaseSettings from "metabase/lib/settings";  
import type { TokenFeature } from "metabase-types/api";
```

```
export function hasPremiumFeature(feature:...
```

### **enterprise/frontend/src/metabase-enterprise/settings/selectors.ts**

```
import noResultsSource from "assets/img/no_results.svg";  
import type { IllustrationValue } from "metabase/plugins";  
import { getSetting, getSettings } from "metabase/selectors/settings";  
import type...
```

### **enterprise/frontend/src/metabase-enterprise/settings/selectors.unit.spec.ts**

```
import {  
  createMockSettingsState,  
  createMockState,  
} from "metabase-types/store/mocks";
```

```
import { getIsWhiteLabeling, getLoadingMessage, getLogoUrl } from "../selectors";
```

```
describe("getLogoUrl",...
```

### **enterprise/frontend/src/metabase-enterprise/settings/types.ts**

```
import type { EnterpriseSettings } from "metabase-types/api";  
import type { SettingsState, State } from "metabase-types/store";
```

```
export interface EnterpriseState extends State {  
  settings:...
```

### **enterprise/frontend/src/metabase-enterprise/shared/index.ts**

```
import { PLUGIN_REDUCERS } from "metabase/plugins";
```

## Analyst Base

```
import { shared } from "../reducer";
```

```
PLUGIN_REDUCERS.shared = shared.reducer;
```

### **enterprise/frontend/src/metabase-enterprise/shared/reducer.ts**

```
import { createSlice } from "@reduxjs/toolkit";
```

```
import { createAsyncThunk } from "metabase/lib/redux";
```

```
import { GTAPApi } from "metabase/services";
```

```
import type { UserAttributeKey } from...
```

### **enterprise/frontend/src/metabase-enterprise/shared/selectors.ts**

```
import type { EnterpriseState } from "../types";
```

```
export const getUserAttributes = (state: EnterpriseState) => {
```

```
  return state.plugins.shared.attributes;
```

```
};
```

### **enterprise/frontend/src/metabase-enterprise/shared/types.ts**

```
import type { State } from "metabase-types/store";
```

```
import type { EnterpriseSharedState } from "../reducer";
```

```
export interface EnterpriseState extends State {
```

```
  plugins: {
```

```
    shared:...
```

### **enterprise/frontend/src/metabase-enterprise/sharing/components/utils.ts**

```
import { hasInlineParameters } from "metabase/dashboard/utils";
```

```
import type { UiParameter } from "metabase-lib/v1/parameters/types";
```

```
import type { Dashboard, DashboardCard } from...
```

### **enterprise/frontend/src/metabase-enterprise/sharing/index.ts**

```
import { PLUGIN_DASHBOARD_SUBSCRIPTION_PARAMETERS_SECTION_OVERRIDE } from "metabase/plugins";
```

```
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import { MutableParametersSection }...
```

### **enterprise/frontend/src/metabase-enterprise/smtplib/override/components/index.ts**

```
export { CloudSMTPConnectionCard } from "../CloudSMTPConnectionCard";
```

```
export { SMTPOverrideConnectionForm } from "../SMTPOverrideConnectionForm";
```

### **enterprise/frontend/src/metabase-enterprise/smtplib/override/index.ts**

## Analyst Base

```
import { PLUGIN_SMTP_OVERRIDE } from "metabase/plugins";
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import {
  CloudSMTPConnectionCard,
  SMTPOverrideConnectionForm,
} from...
```

### **enterprise/frontend/src/metabase-enterprise/snippets/index.js**

```
import { t } from "ttag";
```

```
import                                     CollectionPermissionsModal                                from
"metabase/admin/permissions/components/CollectionPermissionsModal/CollectionPermissionsModal";
import { canonicalCollectionId } from...
```

### **enterprise/frontend/src/metabase-enterprise/static-viz-overrides.js**

```
import { applyWhitelabelOverride } from "../whitelabel/static-viz-overrides";
```

```
export default function apply() {
  applyWhitelabelOverride();
}
```

### **enterprise/frontend/src/metabase-enterprise/tools/index.ts**

```
import { PLUGIN_ADMIN_TOOLS } from "metabase/plugins";
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import ErrorOverview from "../ErrorOverview";
```

```
if...
```

### **enterprise/frontend/src/metabase-enterprise/tools/mode.js**

```
import { push } from "react-router-redux";
import { t } from "ttag";
```

```
const CARD_ID_ROW_IDX = 0;
```

```
const ErrorDrill = ({ clicked }) => {
  if (!clicked) {
    return [];
  }
}
```

```
const cardId =...
```

### **enterprise/frontend/src/metabase-enterprise/upload\_management/index.ts**



## Analyst Base

```
import { PLUGIN_UPLOAD_MANAGEMENT } from "metabase/plugins";
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import {
  FileUploadErrorModal,
  GdriveAddDataPanel,
  ...
```

### **enterprise/frontend/src/metabase-enterprise/upload\_management/utils.ts**

```
import dayjs from "dayjs";
import relativeTime from "dayjs/plugin/relativeTime";
import updateLocale from "dayjs/plugin/updateLocale";
dayjs.extend(updateLocale);
dayjs.extend(relativeTime);
```

```
export...
```

### **enterprise/frontend/src/metabase-enterprise/urls.ts**

```
import type { Databaseld } from "metabase-types/api";
```

```
export * as Urls from "metabase/lib/urls";
```

```
export function viewDestinationDatabases(databaseld: Databaseld) {
  return...
```

### **enterprise/frontend/src/metabase-enterprise/user\_provisioning/index.ts**

```
import { PLUGIN_AUTH_PROVIDERS } from "metabase/plugins";
import { hasPremiumFeature } from "metabase-enterprise/settings";
```

```
import { UserProvisioning } from "../components/UserProvisioning";
```

```
if...
```

### **enterprise/frontend/src/metabase-enterprise/user\_provisioning/types.ts**

```
export interface MaskedScimApiKey {
  id: number;
  scope: "scim";
  key: string;
  key_prefix: string;
  masked_key: string;
  name: string;
  user_id: null;
  created_at: string;
  creator_id:...
```

## Analyst Base

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/BrandColorSettings/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/BrandColorSettings/types.ts**

```
export interface ColorOption {  
  name: string;  
  description: string;  
}
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/BrandColorSettings/utls.ts**

```
import { t } from "ttag";
```

```
import type { ColorOption } from "./types";
```

```
export const getBrandColorOptions = (): ColorOption[] => [  
  {  
    name: "brand",  
    description: t`The main color used...`  
  }  
]
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/ChartColorPreview/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/ChartColorPreview/utls.ts**

```
import _ from "underscore";
```

```
import { color } from "metabase/lib/colors";
```

```
export const getAccentColorGroups = (palette: Record<string, string>) => {  
  const groups = [  
    _.times(8, (i) =>...  
  ]  
}
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/ChartColorSample/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/ChartColorSettings/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **enterprise/frontend/src/metabase-enterprise/whitelabel/components/ChartColor Settings/utils.ts**

```
import Color from "color";
import _ from "underscore";

export const getChartColorGroups = (): string[][] => {
  return _.times(8, (i) => [
    `accent${i}`,
    `accent${i}-light`,
    ...
```

## **enterprise/frontend/src/metabase-enterprise/whitelabel/components/ChartColor Settings/utils.unit.spec.ts**

```
import Color from "color";

import { getAutoChartColors, getDefaultChartColors } from "../utils";

describe("getDefaultChartColors", () => {
  const groups = [["accent1"], ["accent2"], ["accent3"]];

  ...
```

## **enterprise/frontend/src/metabase-enterprise/whitelabel/components/ColorSettings/index.ts**

```
export { ColorSettings } from "../ColorSettings";
```

## **enterprise/frontend/src/metabase-enterprise/whitelabel/components/ColorSettingsWidget/index.ts**

```
export * from "../ColorSettingsWidget";
```

## **enterprise/frontend/src/metabase-enterprise/whitelabel/components/FontWidget/index.ts**

```
export * from "../FontWidget";
```

## **enterprise/frontend/src/metabase-enterprise/whitelabel/components/FontWidget/types.ts**

```
import type { FontFile, Settings } from "metabase-types/api";

export interface FontSetting {
  value: string | null;
  default: string;
}
```

## Analyst Base

```
export type FontSettingKeys = "application-font" |...
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/FontWidget/Utils.ts**

```
import { useMemo } from "react";
import { t } from "ttag";
import _ from "underscore";
```

```
import { useAdminSetting } from "metabase/api/utils";
import type { FontFile, FontFormat } from...
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/HelpLinkSettings/index.ts**

```
export { HelpLinkSettings } from "../HelpLinkSettings";
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/IllustrationWidget/index.ts**

```
export { IllustrationWidget } from "../IllustrationWidget";
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/ImageToggle/index.ts**

```
export { ImageToggle } from "../ImageToggle";
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/LandingPageWidget/Utils.ts**

```
interface GetRelativeLandingPageUrlResult {
  isSameOrigin: boolean;
  relativeUrl: string;
}
```

```
export const getRelativeLandingPageUrl = (
  value: string,
): GetRelativeLandingPageUrlResult => {
  ...
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/LandingPageWidget/Utils.unit.spec.ts**

```
import { getRelativeLandingPageUrl } from "../utils";
```

```
describe("utils", () => {
  describe("getRelativeLandingPageUrl", () => {
    it("should return relative url for valid inputs", () => {
      [
        ...
```

## Analyst Base

### **enterprise/frontend/src/metabase-enterprise/whitelabel/components/MetabotToggleWidget/index.ts**

```
export { MetabotToggleWidget } from "./MetabotToggleWidget";
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/index.ts**

```
import MetabaseSettings from "metabase/lib/settings";
```

```
import {  
  PLUGIN_APP_INIT_FUNCTIONS,  
  PLUGIN_LANDING_PAGE,  
  PLUGIN_LOGO_ICON_COMPONENTS,  
  PLUGIN_SELECTORS,  
  PLUGIN_WHITELABEL,  
} from...
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/lib/loading-message.ts**

```
import { t } from "ttag";
```

```
export const LOADING_MESSAGE_BY_SETTING = {  
  "doing-science": {  
    // eslint-disable-next-line ttag/no-module-declaration -- see metabase#55045  
    name: t`Doing...
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/lib/whitelabel.js**

```
import { colors } from "metabase/lib/colors/colors";  
import MetabaseSettings from "metabase/lib/settings";
```

```
export function updateColors() {  
  const scheme =...
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/overrides.js**

```
import { hasPremiumFeature } from "metabase-enterprise/settings";  
import { updateColors } from "metabase-enterprise/whitelabel/lib/whitelabel";
```

```
if (hasPremiumFeature("whitelabel")) {  
  ...
```

### **enterprise/frontend/src/metabase-enterprise/whitelabel/static-viz-overrides.js**

```
import { hasPremiumFeature } from "metabase-enterprise/settings";  
import { updateColors } from "metabase-enterprise/whitelabel/lib/whitelabel";
```

```
export function applyWhitelabelOverride() {  
  if...
```

### **frontend/CLAUDE.md**

## Analyst Base

Working in /frontend

# Structure

**frontend/lint/eslint-rules/no-color-literals.js**

\*

**frontend/lint/eslint-rules/no-direct-helper-import.js**

\*

**frontend/lint/eslint-rules/no-external-references-for-sdk-package-code.js**

eslint-disable import/no-commonjs

eslint-env node

**frontend/lint/eslint-rules/no-literal-metabase-strings.js**

\*

**frontend/lint/eslint-rules/no-locale-with-intl-functions.js**

-----  
Rule Definition

-----  
eslint-disable-next-line...

**frontend/lint/eslint-rules/no-unconditional-metabase-links-render.js**

-----  
Rule Definition

-----  
The following cases are...

**frontend/lint/eslint-rules/no-unordered-test-helpers.js**

\*

**frontend/lint/eslint-rules/no-unsafe-element-filtering.js**

eslint-disable-next-line import/no-commonjs

**frontend/lint/eslint-rules/no-unscoped-text-selectors.js**

-----  
Rule Definition

-----  
**frontend/lint/tests/no-literal-metabase-strings.unit.spec.js**

import { RuleTester } from "eslint";

import noLiteralMetabaseString from "../eslint-rules/no-literal-metabase-strings";

const ruleTester = new RuleTester({

## Analyst Base

```
parserOptions: {  
  ecmaVersion:...
```

### **frontend/lint/tests/no-locale-with-intl-functions.unit.spec.js**

```
import { RuleTester } from "eslint";
```

```
import noLocaleWithIntlFunctions from "../eslint-rules/no-locale-with-intl-functions";
```

```
const ruleTester = new RuleTester({  
  parserOptions: {  
    ecmaVersion:...
```

### **frontend/lint/tests/no-unconditional-metabase-links-render.unit.spec.js**

```
import { RuleTester } from "eslint";
```

```
import noUnconditionalMetabaseLinksRender from "../eslint-rules/no-unconditional-metabase-links-render";
```

```
const ruleTester = new RuleTester({  
  parserOptions: {  
    ...
```

### **frontend/lint/tests/no-unordered-test-helpers.unit.spec.js**

```
import { RuleTester } from "eslint";
```

```
import rule from "../eslint-rules/no-unordered-test-helpers";
```

```
const ruleTester = new RuleTester({ parserOptions: { ecmaVersion: 2015 } });
```

```
const orderError =...
```

### **frontend/lint/tests/no-unsafe-element-filtering.unit.spec.js**

```
import { RuleTester } from "eslint";
```

```
import rule from "../eslint-rules/no-unsafe-element-filtering";
```

```
const ruleTester = new RuleTester({ parserOptions: { ecmaVersion: 2015 } });
```

```
const unsafeError...
```

### **frontend/lint/tests/no-unscoped-text-selectors.unit.spec.js**

```
import { RuleTester } from "eslint";
```

```
import rule from "../eslint-rules/no-unscoped-text-selectors";
```

## Analyst Base

```
const ruleTester = new RuleTester({ parserOptions: { ecmaVersion: 2015 } });
```

```
const scopeError =...
```

### **frontend/parse-deps.js**

```
#!/usr/bin/env node
```

```
eslint-disable import/no-commonjs, no-undef, no-console
```

### **frontend/src/metabase/.eslintrc**

```
{
  "rules": {
    // Note: adding this rule to a eslint config file in a subfolder will remove
    // *not* carry over the restricted imports from parent folders, you will
    // need to copy them...
```

### **frontend/src/metabase/account/app/components/AccountHeader/index.ts**

```
export { AccountHeader } from "./AccountHeader";
```

### **frontend/src/metabase/account/app/components/AccountLayout/AccountLayout.unit.spec.js**

```
import { render, screen } from "__support__/ui";
```

```
import AccountLayout from "./AccountLayout";
```

```
const getUser = () => ({
  id: 1,
  first_name: "John",
  last_name: "Doe",
  email:...
```

### **frontend/src/metabase/account/app/components/AccountLayout/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/account/app/components/AccountApp/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/account/login-history/components/LoginHistory/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/account/login-history/components/LoginHistoryApp/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/account/notifications/actions.js**

```
import { push } from "react-router-redux";
```



## Analyst Base

```
const PREFIX = `/account/notifications`;
```

```
const TYPE_MAP = {  
  "question-notification": "alert",  
  pulse: "pulse",  
};
```

```
export const navigateToUnsubscribe =...
```

### **frontend/src/metabase/account/notifications/components/ArchiveModal/ArchiveModal.unit.spec.js**

```
import { render, screen, waitFor } from "__support__/ui";
```

```
import ArchiveModal from "../ArchiveModal";
```

```
const getAlert = ({ creator = getUser(), channels = [getChannel()] } = {}) => ({  
  creator,  
  ...
```

### **frontend/src/metabase/account/notifications/components/ArchiveModal/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/account/notifications/components/HelpModal/HelpModal.unit.spec.js**

```
import { mockSettings } from "__support__/settings";  
import { renderWithProviders, screen } from "__support__/ui";
```

```
import HelpModal from "../HelpModal";
```

```
function setup({ adminEmail, onClose } = {})...
```

### **frontend/src/metabase/account/notifications/components/HelpModal/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/account/notifications/components/NotificationCard/index.ts**

```
export { DashboardNotificationCard } from "../DashboardNotificationCard";  
export { NotificationCard } from "../NotificationCard";
```

### **frontend/src/metabase/account/notifications/components/NotificationCard/Utils.ts**

```
import { t } from "ttag";
```

## Analyst Base

```
import type { NotificationListItem } from "metabase/account/notifications/types";
import { formatDateTimeWithUnit } from "metabase/lib/formatting/date";
import Settings...
```

### **frontend/src/metabase/account/notifications/components/NotificationList/index.ts**

```
export { NotificationList } from "./NotificationList";
```

### **frontend/src/metabase/account/notifications/components/UnsubscribeModal/UnsubscribeModal.unit.spec.js**

```
import { render, screen, waitFor } from "__support__/ui";
```

```
import UnsubscribeModal from "./UnsubscribeModal";
```

```
const getAlert = ({ creator = getUser({ id: 1 }) } = {}) => ({
  name: "Alert",
  ...
```

### **frontend/src/metabase/account/notifications/components/UnsubscribeModal/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/account/notifications/containers/ArchiveAlertModal/index.ts**

```
export { DeleteAlertModal } from "./DeleteAlertModal";
```

### **frontend/src/metabase/account/notifications/containers/ArchivePulseModal/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/account/notifications/containers/NotificationsApp/index.ts**

```
export { NotificationsApp } from "./NotificationsApp";
```

### **frontend/src/metabase/account/notifications/containers/UnsubscribeAlertModal/index.ts**

```
export { UnsubscribeAlertModal } from "./UnsubscribeAlertModal";
```

### **frontend/src/metabase/account/notifications/containers/UnsubscribePulseModal/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/account/notifications/selectors.ts**

## Analyst Base

```
export const getAlertId = (rawAlertId: unknown): number | null => {
  if (rawAlertId && typeof rawAlertId === "string") {
    return parseInt(rawAlertId, 10);
  }
```

```
  return null;
};
```

export const...

### **frontend/src/metabase/account/notifications/types.ts**

```
import type { Alert, Notification } from "metabase-types/api";
```

```
export type DashboardAlertListItem = {
  item: Alert;
  type: "pulse";
};
```

```
export type QuestionNotificationListItem = {
  item:...
```

### **frontend/src/metabase/account/password/actions.ts**

```
import { getIn } from "icepick";
```

```
import MetabaseSettings from "metabase/lib/settings";
import { UtilApi } from "metabase/services";
```

```
export const validatePassword = async (password: string) => {
  ...
```

### **frontend/src/metabase/account/password/components/UserPasswordForm/index.ts**

```
export * from "../UserPasswordForm";
```

### **frontend/src/metabase/account/password/containers/UserPasswordApp/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/account/password/types.ts**

```
export interface UserPasswordData {
  old_password: string;
  password: string;
  password_confirm: string;
}
```

## Analyst Base

### **frontend/src/metabase/account/profile/actions.ts**

```
import { userApi } from "metabase/api";
import { createThunkAction } from "metabase/lib/redux";
import type { User } from "metabase-types/api";
```

```
import type { UserProfileData } from...
```

### **frontend/src/metabase/account/profile/components/UserProfileForm/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/account/profile/containers/UserProfileApp/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/account/profile/selectors.ts**

```
import { createSelector } from "@reduxjs/toolkit";

import { PLUGIN_IS_PASSWORD_USER } from "metabase/plugins";
import { getSettings } from "metabase/selectors/settings";
import { getUser } from...
```

### **frontend/src/metabase/account/profile/types.ts**

```
export interface UserProfileData {
  first_name?: string | null;
  last_name?: string | null;
  email?: string;
  locale: string | null;
}
```

### **frontend/src/metabase/actions/actions.ts**

```
import { addUndo } from "metabase/redux/undo";
import { ActionsApi } from "metabase/services";
import type {
  ActionFormSubmitResult,
  ParametersForActionExecution,
  WritebackAction,
} from...
```

### **frontend/src/metabase/actions/components/ActionForm/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/actions/components/ActionFormFieldWidget/index.ts**

```
export { ActionFormFieldWidget } from "./ActionFormFieldWidget";
```

### **frontend/src/metabase/actions/components/ActionViz/ExplainerText/index.ts**

```
export { ExplainerText } from "./ExplainerText";
```

## **frontend/src/metabase/actions/components/ActionViz/ExplainerText/tests/common.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import { setup } from "../setup";
```

```
describe("ExplainerText (OSS)", () => {  
  it("should render help link when `show-metabase-links: true`", () => {  
    ...
```

## **frontend/src/metabase/actions/components/ActionViz/ExplainerText/tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import type { SetupOpts } from "../setup";  
import { setup as baseSetup } from "../setup";
```

```
function setup(opts: SetupOpts) {  
  baseSetup({...
```

## **frontend/src/metabase/actions/components/ActionViz/ExplainerText/tests/premium.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import type { SetupOpts } from "../setup";  
import { setup as baseSetup } from "../setup";
```

```
function setup(opts: SetupOpts) {  
  baseSetup({  
    ...
```

## **frontend/src/metabase/actions/components/ActionViz/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

## **frontend/src/metabase/actions/components/ActionViz/utils.ts**

```
import _ from "underscore";
```

```
import { isImplicitDeleteAction } from "metabase/actions/utils";  
import { isNotNull } from "metabase/lib/types";  
import { isEmpty } from "metabase/lib/validate";  
import...
```

## **frontend/src/metabase/actions/components/ImplicitActionIcon/index.ts**

## Analyst Base

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/actions/constants.ts**

```
import { t } from "ttag";
```

```
import type { FieldType, InputSettingType } from "metabase-types/api";
```

```
interface FieldOptionType {  
  value: FieldType;  
  name: string;  
}
```

```
interface InputOptionType {  
  ...
```

### **frontend/src/metabase/actions/containers/ActionCreator/ActionContext/ActionContext.ts**

```
import { createContext, useContext } from "react";  
import _ from "underscore";
```

```
import type { ActionFormSettings, WritebackAction } from "metabase-types/api";
```

```
import { getDefaultFormSettings } from...
```

### **frontend/src/metabase/actions/containers/ActionCreator/ActionContext/ImplicitActionContextProvider/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/actions/containers/ActionCreator/ActionContext/QueryActionContextProvider/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/actions/containers/ActionCreator/ActionContext/QueryActionContextProvider/utils.ts**

```
import type Question from "metabase-lib/v1/Question";  
import type NativeQuery from "metabase-lib/v1/queries/NativeQuery";  
import type {  
  ActionFormSettings,  
  FieldType,  
  InputSettingType,  
  ...
```

### **frontend/src/metabase/actions/containers/ActionCreator/ActionContext/QueryActionContextProvider/utils.unit.spec.ts**

## Analyst Base

```
import { createMockMetadata } from "__support__/metadata";
import {
  getDefaultFieldSettings,
  getDefaultFormSettings,
} from "metabase/actions/utils";
import Question from...
```

### **frontend/src/metabase/actions/containers/ActionCreator/ActionContext/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/actions/containers/ActionCreator/ActionContext/types.ts**

```
import type { ReactNode } from "react";

import type { WritebackAction } from "metabase-types/api";
```

```
export type EditableActionParams = Pick<
  Partial<WritebackAction>,
  "name" |...
```

### **frontend/src/metabase/actions/containers/ActionCreator/ActionContext/utils.ts**

```
import type { WritebackQueryAction } from "metabase-types/api";

import { getDefaultFormSettings } from "../../utils";
```

```
export function createEmptyWritebackAction():...
```

### **frontend/src/metabase/actions/containers/ActionCreator/CreateActionForm/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/Description/index.ts**

```
export { Description } from "../Description";
```

### **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/Description/test/common.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import { setup } from "../setup";
```

```
describe("FormCerator > Description (OSS)", () => {
  it("should show a help link when `show-metabase-links: true`, () =>...
```

### **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/Description/test/enterprise.unit.spec.ts**

## Analyst Base

```
import { screen } from "__support__/ui";

import type { SetupOpts } from "./setup";
import { setup as baseSetup } from "./setup";

function setup(opts: SetupOpts) {
  return baseSetup({
    ...
```

### **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/Description/test/premium.unit.spec.ts**

```
import { screen } from "__support__/ui";

import type { SetupOpts } from "./setup";
import { setup as baseSetup } from "./setup";

function setup(opts: SetupOpts) {
  return baseSetup({
    ...
```

### **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/EmptyFormPlaceholder/index.ts**

```
export { EmptyFormPlaceholder } from "./EmptyFormPlaceholder";
```

### **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/EmptyFormPlaceholder/tests/common.unit.spec.ts**

```
import { screen } from "__support__/ui";

import { setup } from "./setup";

describe("EmptyFormPlaceholder (OSS)", () => {
  it("should render help link when `show-metabase-links: true`", () => {
    ...
```

### **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/EmptyFormPlaceholder/tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";

import type { SetupOpts } from "./setup";
import { setup as baseSetup } from "./setup";

function setup(opts: SetupOpts) {
  baseSetup({
```



...

## **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/EmptyFormPlaceholder/tests/premium.unit.spec.ts**

```
import { screen } from "__support__/ui";

import type { SetupOpts } from "./setup";
import { setup as baseSetup } from "./setup";
```

```
function setup(opts: SetupOpts) {
  baseSetup({
    ...
```

...

## **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/FormFieldEditor/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/index.ts**

```
export { FormCreator } from "./FormCreator";
export * from "./utils";
```

## **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/utils.ts**

```
import _ from "underscore";

import { moveElement } from "metabase/lib/arrays";
import type { FieldSettingsMap, InputSettingType } from "metabase-types/api";
```

```
const inputTypeMap:...
```

## **frontend/src/metabase/actions/containers/ActionCreator/FormCreator/utils.unit.spec.ts**

```
import { getDefaultFieldSettings } from "metabase/actions/utils";
import type { FieldSettingsMap } from "metabase-types/api";
```

```
import { reorderFields } from "./utils";
```

```
describe("actions > containers...
```

## **frontend/src/metabase/actions/containers/ActionCreator/tests/ActionCreator-ImplicitActions.unit.spec.ts**

```
import { queryIcon, screen } from "__support__/ui";
import {
  createMockActionParameter,
  createMockImplicitQueryAction,
```

## Analyst Base

```
} from "metabase-types/api/mocks";
```

```
import type { SetupOpts } from...
```

### **frontend/src/metabase/actions/containers/ActionCreator/tests/ActionCreator-QueryActions.unit.spec.ts**

```
import userEvent from "@testing-library/user-event";
```

```
import { getIcon, queryIcon, screen, waitFor, within } from "__support__/ui";
```

```
import {
```

```
  createMockActionParameter,
```

```
  createMockCard,
```

```
  ...
```

### **frontend/src/metabase/actions/containers/ActionCreator/types.ts**

```
export type SideView = "dataReference" | "actionForm" | "actionSettings";
```

```
export interface ActionCreatorUIProps {
```

```
  canRename: boolean;
```

```
  canChangeFieldSettings: boolean;
```

```
}
```

### **frontend/src/metabase/actions/containers/ActionCreator/utils.ts**

```
import _ from "underscore";
```

```
import { getDefaultFieldSettings } from "metabase/actions/utils";
```

```
import type { ActionFormSettings, Parameter } from "metabase-types/api";
```

```
export const...
```

### **frontend/src/metabase/actions/containers/ActionCreator/utils.unit.spec.ts**

```
import {
```

```
  createMockActionFormSettings,
```

```
  createMockFieldSettings,
```

```
  createMockParameter,
```

```
} from "metabase-types/api/mocks";
```

```
import { syncFieldsWithParameters } from...
```

### **frontend/src/metabase/actions/containers/ActionCreatorModal/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/actions/containers/ActionExecuteModal/index.ts**

```
export { ActionExecuteModal } from "./ActionExecuteModal";
```

## Analyst Base

### **frontend/src/metabase/actions/containers/ActionParametersInputForm/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/actions/containers/ActionPicker/index.ts**

export { ActionPicker, ConnectedActionPicker } from "./ActionPicker";

### **frontend/src/metabase/actions/containers/ActionPicker/types.ts**

import type { CardId, WritebackAction } from "metabase-types/api";

type ModelId = CardId;

export type ModelActionMap = Record<ModelId, WritebackAction[]>;

### **frontend/src/metabase/actions/containers/ActionPicker/utils.ts**

import \_ from "underscore";

import type { WritebackAction } from "metabase-types/api";

import type { ModelActionMap } from "./types";

export const sortAndGroupActions = (  
 actions?:...

### **frontend/src/metabase/actions/containers/ActionPicker/utils.unit.spec.ts**

import { createMockQueryAction } from "metabase-types/api/mocks";

import { sortAndGroupActions } from "./utils";

const testActions = [  
 createMockQueryAction({ id: 2, name: "Bear Action",...

### **frontend/src/metabase/actions/hooks/use-action-form/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/actions/hooks/use-action-form/use-action-form.ts**

import { useCallback, useMemo } from "react";

import \_ from "underscore";

import { getForm, getFormValidationSchema } from "metabase/actions/utils";

import type {  
 ActionFormInitialValues,

...

### **frontend/src/metabase/actions/hooks/use-action-form/use-action-form.unit.spec.ts**

## Analyst Base

```
import { renderHook } from "@testing-library/react";
```

```
import {  
  createMockActionParameter,  
  createMockFieldSettings,  
  createMockQueryAction,  
} from "metabase-types/api/mocks";
```

```
import...
```

### **frontend/src/metabase/actions/hooks/use-action-form/utils.ts**

```
import dayjs from "dayjs";
```

```
import type { FieldSettings as LocalFieldSettings } from "metabase/actions/types";  
import { getDefaultFieldSettings } from "metabase/actions/utils";  
import { isEmpty } from "lodash";
```

### **frontend/src/metabase/actions/hooks/use-action-form/utils.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";  
import { checkNotNull } from "metabase/lib/types";  
import type { FieldSettingsMap } from "metabase-types/api";  
import {  
  createMockField,  
  ...  
}
```

### **frontend/src/metabase/actions/hooks/use-action-initial-values.ts**

```
import { useEffect, useMemo } from "react";  
import { useAsyncFn } from "react-use";
```

```
import type { ParametersForActionExecution } from "metabase-types/api";
```

```
const NO_VALUES:...
```

### **frontend/src/metabase/actions/types.ts**

```
import type Field from "metabase-lib/v1/metadata/Field";  
import type {  
  ActionFormOption,  
  FieldSettings as BaseFieldSettings,  
  InputComponentType,  
} from "metabase-types/api";
```

```
export type...
```

### **frontend/src/metabase/actions/utils.ts**

## Analyst Base

```
import { t } from "ttag";
import * as Yup from "yup";
```

```
import * as Errors from "metabase/lib/errors";
import type Field from "metabase-lib/v1/metadata/Field";
import { TYPE } from...
```

### **frontend/src/metabase/actions/utils.unit.spec.ts**

```
import {
  getDefaultFieldSettings,
  getDefaultFormSettings,
  sortActionParams,
} from "./utils";
```

```
const createParameter = (options?: any) => {
  return {
    id: "test_parameter",
    name: "Test..."
```

### **frontend/src/metabase/admin/admin.js**

Reducers needed for admin section (only used in "main" app)

### **frontend/src/metabase/admin/app/actions.ts**

```
import { updateSetting } from "metabase/admin/settings/settings";
import { createAsyncThunk } from "metabase/lib/redux";
```

```
import { getCurrentVersion } from "../selectors";
```

```
export const disableNotice...
```

### **frontend/src/metabase/admin/app/components/AdminApp/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/admin/app/components/DeprecationNotice/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/admin/app/components/DeprecationNotice/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/admin/app/reducers.ts**

```
import { createReducer } from "@reduxjs/toolkit";
import { t } from "ttag";
```

```
import { combineReducers } from "metabase/lib/redux";
import { isNotNull } from "metabase/lib/types";
```

## Analyst Base

```
import {  
  ...
```

### **frontend/src/metabase/admin/app/selectors.ts**

```
import { getEngines } from "metabase/databases/selectors";  
import { isDeprecatedEngine } from "metabase/lib/engine";  
import { getSetting } from "metabase/selectors/settings";  
import type Database...
```

### **frontend/src/metabase/admin/components/AdminNav/index.ts**

```
export * from "./AdminNav";
```

### **frontend/src/metabase/admin/components/SettingsSection/index.ts**

```
export * from "./SettingsSection";
```

### **frontend/src/metabase/admin/databases/components/ContentRemovalConfirmation/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/admin/databases/components/DatabaseList/index.ts**

```
export * from "./DatabaseList";
```

### **frontend/src/metabase/admin/databases/components/DatabaseModelFeaturesSection/ModelActionsSection/index.ts**

```
export * from "./ModelActionsSection";
```

### **frontend/src/metabase/admin/databases/components/DatabaseModelFeaturesSection/ModelCachingControl/index.ts**

```
export * from "./ModelCachingControl";
```

### **frontend/src/metabase/admin/databases/database.ts**

```
import { createAction, createReducer } from "@reduxjs/toolkit";  
import { push } from "react-router-redux";
```

```
import Databases from "metabase/entities/databases";  
import { combineReducers } from...
```

### **frontend/src/metabase/admin/databases/editParamsForUserControlledScheduling.ts**

```
import _ from "underscore";
```

```
import type { DatabaseData } from "metabase-types/api";
```

```
export const editParamsForUserControlledScheduling = _.compose(  
  editScheduleParamsForUserControlledScheduling,
```

## Analyst Base

...

### **frontend/src/metabase/admin/databases/editParamsForUserControlledScheduling.unit.spec.ts**

```
import { createMockDatabase } from "metabase-types/api/mocks";
```

```
import { editParamsForUserControlledScheduling } from "../editParamsForUserControlledScheduling";
```

```
it("adds full_sync param if user will...
```

### **frontend/src/metabase/admin/databases/selectors.ts**

```
import type { State } from "metabase-types/store";
```

```
export const getDeletes = (state: State) => state.admin.databases.deletes;
```

```
export const getDeletionError = (state: State) =>
```

...

### **frontend/src/metabase/admin/databases/utils.ts**

```
import type { Database, DatabaseFeature, Databaseld } from "metabase-types/api";
```

```
export const isDbModifiable = (  
  database: { id?: Databaseld; is_attached_dwh?: boolean } | undefined,  
) => {
```

...

### **frontend/src/metabase/admin/datamodel/components/FormInput/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/admin/datamodel/components/FormLabel/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/admin/datamodel/components/FormTextArea/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/admin/datamodel/components/QueryDefinition/index.ts**

```
export * from "../QueryDefinition";
```

### **frontend/src/metabase/admin/datamodel/components/SegmentForm/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/admin/datamodel/datamodel.js**

```
import {  
  combineReducers,  
  createAction,
```

## Analyst Base

```
createThunkAction,  
handleActions,  
} from "metabase/lib/redux";  
import { MetabaseApi, RevisionsApi } from "metabase/services";
```

```
export const...
```

### **frontend/src/metabase/admin/datamodel/selectors.js**

```
export const getPreviewSummary = (state) =>  
  state.admin.datamodel.previewSummary;  
export const getRevisions = (state) => state.admin.datamodel.revisions;  
export const getCurrentUser = (state) =>...
```

### **frontend/src/metabase/admin/datamodel/utils/segments.ts**

```
import * as Lib from "metabase-lib";  
import type Metadata from "metabase-lib/v1/metadata/Metadata";  
import type { StructuredQuery, TableId } from "metabase-types/api";
```

```
export function...
```

### **frontend/src/metabase/admin/people/colors.ts**

```
import { color } from "metabase/lib/colors";  
import type { GroupId, User } from "metabase-types/api";
```

```
export const userToColor = (user: User) => {  
  return user.is_superuser ? color("accent2") : ...
```

### **frontend/src/metabase/admin/people/components/GroupMembersTable/index.ts**

```
export * from "./GroupMembersTable";
```

### **frontend/src/metabase/admin/people/components/MembershipSelect/index.ts**

```
export * from "./MembershipSelect";
```

### **frontend/src/metabase/admin/people/constants.ts**

```
export type UserStatus = "active" | "deactivated";
```

```
export const USER_STATUS: Record<string, UserStatus> = {  
  active: "active",  
  deactivated: "deactivated",  
};
```

```
export const...
```

### **frontend/src/metabase/admin/people/containers/EditUserModal/index.ts**

```
export * from "./EditUserModal";
```



## Analyst Base

### **frontend/src/metabase/admin/people/events.js**

```
export const CLEAR_MEMBERSHIPS = "metabase/admin/people/CLEAR_MEMBERSHIPS";
export const STORE_TEMPORARY_PASSWORD =
  "metabase/admin/people/STORE_TEMPORARY_PASSWORD";
export const...
```

### **frontend/src/metabase/admin/people/hooks/use-people-query.ts**

```
import { useEffect, useMemo, useState } from "react";

import { usePagination } from "metabase/common/hooks/use-pagination";
import { SEARCH_DEBOUNCE_DURATION } from "metabase/lib/constants";

import...
```

### **frontend/src/metabase/admin/people/people.js**

```
import {
  combineReducers,
  createAction,
  handleActions,
} from "metabase/lib/redux";

import { CLEAR_TEMPORARY_PASSWORD, STORE_TEMPORARY_PASSWORD } from "../events";

// ACTION CREATORS

export...
```

### **frontend/src/metabase/admin/people/selectors.js**

```
import { createSelector } from "@reduxjs/toolkit";

import { ACTIVE_USERS_NUDGE_THRESHOLD } from "metabase/admin/people/constants";
import { hasAnySsoFeature } from...
```

### **frontend/src/metabase/admin/performance/constants/complex.ts**

```
import { t } from "ttag";
import * as Yup from "yup";

import type { CacheableModel } from "metabase-types/api";
import type { AdminPath } from "metabase-types/store";

import { PerformanceTabId,...
```

### **frontend/src/metabase/admin/performance/constants/simple.ts**

```
export const rootId = 0;
```

## Analyst Base

```
export const defaultMinDurationMs = 1000;
export const defaultCronSchedule = "0 0 * * * ? *";
```

### **frontend/src/metabase/admin/performance/types.ts**

```
import type { AnySchema } from "yup";

import type { CacheableModel } from "metabase-types/api";

export type UpdateTargetId = (
  newTargetId: number | null,
  isFormDirty: boolean,
) =>...
```

### **frontend/src/metabase/admin/performance/utils.unit.spec.ts**

```
import type {
  AdaptiveStrategy,
  InheritStrategy,
  ScheduleSettings,
} from "metabase-types/api";

import {
  cronToScheduleSettings,
  getShortStrategyLabel,
  hourTo24HourFormat,
  ...
```

### **frontend/src/metabase/admin/permissions/components/ApplicationPermissionsHelp/index.ts**

```
export * from "../ApplicationPermissionsHelp";
```

### **frontend/src/metabase/admin/permissions/components/CollectionPermissionsHelp/index.ts**

```
export * from "../CollectionPermissionsHelp";
```

### **frontend/src/metabase/admin/permissions/components/DataPermissionsHelp/index.ts**

```
export * from "../DataPermissionsHelp";
```

### **frontend/src/metabase/admin/permissions/components/EntityViewSwitch/index.ts**

```
export * from "../EntityViewSwitch";
```

### **frontend/src/metabase/admin/permissions/components/FilterableTree/index.ts**

```
export * from "../FilterableTree";
```

## **frontend/src/metabase/admin/permissions/components/FilterableTree/utils.ts**

```
export const searchItems = (items: any[], filter: string) => {
  const matchingItems = items.filter((item) =>
    item.name.toLowerCase().includes(filter),
  );

  const children = items
    .map((c)...
```

## **frontend/src/metabase/admin/permissions/components/FilterableTree/utils.unit.spec.ts**

```
import { searchItems } from "./utils";

const items = [
  {
    name: "Parent node",
    children: [
      { name: "child", children: [{ name: "Lowest node" }] },
      { name: "aLsO cHiLd" },
    ],
  },
  ...
```

## **frontend/src/metabase/admin/permissions/components/PermissionHelpDescription/index.ts**

```
export * from "./PermissionHelpDescription";
```

## **frontend/src/metabase/admin/permissions/components/PermissionsEditor/index.ts**

```
export * from "./PermissionsEditor";
export * from "./PermissionsEditorEmptyState";
```

## **frontend/src/metabase/admin/permissions/components/PermissionsPageLayout/index.ts**

```
export * from "./PermissionsPageLayout";
```

## **frontend/src/metabase/admin/permissions/components/PermissionsSelect/PermissionsSelect.unit.spec.js**

```
import userEvent from "@testing-library/user-event";

import { fireEvent, getIcon, render, screen } from "__support__/ui";

import { DataPermissionValue } from "../../types";

import {...
```

## **frontend/src/metabase/admin/permissions/components/PermissionsSelect/index.ts**

```
export * from "./PermissionsSelect";
```

## **frontend/src/metabase/admin/permissions/components/PermissionsSidebar/index.ts**

```
export * from "./PermissionsSidebar";
```

## **frontend/src/metabase/admin/permissions/components/PermissionsTable/index.ts**

```
export * from "./PermissionsTable";
```

## **frontend/src/metabase/admin/permissions/components/ToolbarButton/index.ts**

```
export * from "./ToolbarButton";
```

## **frontend/src/metabase/admin/permissions/constants/collections-permissions.js**

```
eslint-disable ttag/no-module-declaration -- see metabase#55045
```

## **frontend/src/metabase/admin/permissions/constants/messages.ts**

```
import { t } from "ttag";
```

```
// eslint-disable-next-line ttag/no-module-declaration -- see metabase#55045
```

```
export const UNABLE_TO_CHANGE_ADMIN_PERMISSIONS = t`Administrators always have the highest...`
```

## **frontend/src/metabase/admin/permissions/pages/DataPermissionsPage/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

## **frontend/src/metabase/admin/permissions/permissions.js**

```
import { assocIn, merge } from "icepick";
```

```
import { push } from "react-router-redux";
```

```
import { t } from "ttag";
```

```
import {  
  inferAndUpdateEntityPermissions,  
  ...
```

## **frontend/src/metabase/admin/permissions/selectors/collection-permissions.ts**

```
import { createSelector } from "@reduxjs/toolkit";
```

```
import { getIn } from "icepick";
```

```
import { t } from "ttag";
```

```
import _ from "underscore";
```

```
import {
```

## Analyst Base

```
isInstanceAnalyticsCollection,
```

```
...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/breadcrumbs.ts**

```
import { t } from "ttag";
```

```
import { isNotFalsy } from "metabase/lib/types";
```

```
import type Metadata from "metabase-lib/v1/metadata/Metadata";
```

```
import type Schema from...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/breadcrumbs.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
```

```
import {
```

```
  createMockDatabase,
```

```
  createMockSchema,
```

```
  createMockTable,
```

```
} from "metabase-types/api/mocks";
```

```
import {...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/data-permissions.unit.spec.fixtures.ts**

```
import { createMockSettingsState } from "metabase-types/store/mocks";
```

```
import { DataPermission, DataPermissionValue } from "../../types";
```

```
// Database 2 contains an imaginary multi-schema database...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/data-sidebar.ts**

```
import type { Selector } from "@reduxjs/toolkit";
```

```
import { createSelector } from "@reduxjs/toolkit";
```

```
import { t } from "ttag";
```

```
import type { ITreeNodeItem } from...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/data-sidebar.unit.spec.ts**

```
import type { State } from "metabase-types/store";
```

```
import type { RawDataRouteParams } from "../../types";
```

## Analyst Base

```
import { state as mockState } from "../data-permissions.unit.spec.fixtures";
```

```
import {...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/diff.ts**

```
import { createSelector } from "@reduxjs/toolkit";
```

```
import _ from "underscore";
```

```
import { diffDataPermissions } from "metabase/admin/permissions/utils/graph";
```

```
import { PLUGIN_DATA_PERMISSIONS } from...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/fields.ts**

```
import _ from "underscore";
```

```
import { getNativePermissionDisabledTooltip } from "metabase/admin/permissions/selectors/data-permissions/shared";
```

```
import { getFieldsPermission,
```

```
...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/group-si debar.ts**

```
import { createSelector } from "@reduxjs/toolkit";
```

```
import { t } from "ttag";
```

```
import { getGroupNameLocalized } from "metabase/lib/groups";
```

```
import type { Group } from "metabase-types/api";
```

```
import type...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/group-si debar.unit.spec.ts**

```
import { assocIn } from "icepick";
```

```
import { PLUGIN_ADVANCED_PERMISSIONS } from "metabase/plugins";
```

```
import type { State } from "metabase-types/store";
```

```
import { DataPermission, DataPermissionValue }...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/groups.t s**

```
import { createSelector } from "@reduxjs/toolkit";
```

```
import _ from "underscore";
```

## Analyst Base

```
import Groups from "metabase/entities/groups";
import { isAdminGroup, isDefaultGroup } from...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/index.ts**

```
export * from "./breadcrumbs";
export * from "./data-sidebar";
export * from "./diff";
export * from "./fields";
export * from "./group-sidebar";
export * from "./permission-editor";
export * from...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/permission-editor.ts**

```
import type { Selector } from "@reduxjs/toolkit";
import { createSelector } from "@reduxjs/toolkit";
import { msgid, ngettext, t } from "ttag";
import _ from "underscore";
```

```
import Groups from...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/revision.ts**

```
import { createSelector } from "@reduxjs/toolkit";

import type { State } from "metabase-types/store";

import { getIsDirty } from "./diff";
```

```
export const showRevisionChangedModal = createSelector(
  ...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/schemas.ts**

```
import _ from "underscore";
```

```
import {
  getNativePermissionDisabledTooltip
} from
"metabase/admin/permissions/selectors/data-permissions/shared";
import { getSchemasPermission } from...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/shared.ts**

```
import {
  NATIVE_PERMISSION_REQUIRES_DATA_ACCESS,
```

## Analyst Base

```
UNABLE_TO_CHANGE_ADMIN_PERMISSIONS,  
UNABLE_TO_CHANGE_LEGACY_PERMISSIONS,  
} from "metabase/admin/permissions/constants/messages";  
import {...
```

### **frontend/src/metabase/admin/permissions/selectors/data-permissions/tables.ts**

```
import _ from "underscore";
```

```
import { getNativePermissionDisabledTooltip } from  
"metabase/admin/permissions/selectors/data-permissions/shared";  
import {  
  getSchemasPermission,  
  ...
```

### **frontend/src/metabase/admin/permissions/selectors/help-reference.ts**

```
import type { State } from "metabase-types/store";
```

```
export const getIsHelpReferenceOpen = (state: State) => {  
  return state.admin.permissions.isHelpReferenceOpen;  
};
```

### **frontend/src/metabase/admin/permissions/types.ts**

```
import type { ReactNode } from "react";
```

```
export type GroupRouteParams = {  
  groupId?: number;  
  databaseId?: number;  
  schemaName?: string;  
};
```

```
export type RawGroupRouteParams = {  
  groupId?:...
```

### **frontend/src/metabase/admin/permissions/utils/data-entity-id.ts**

```
import { checkNotNull } from "metabase/lib/types";  
import type Database from "metabase-lib/v1/metadata/Database";  
import type Schema from "metabase-lib/v1/metadata/Schema";  
import type Table from...
```

### **frontend/src/metabase/admin/permissions/utils/graph/data-permissions/get.ts**

```
import { getIn } from "icepick";
```

```
import type {  
  DatabaseEntityId,
```



## Analyst Base

```
EntityId,  
SchemaEntityId,  
TableEntityId,  
} from "metabase/admin/permissions/types";  
import {  
  DataPermission,  
  ...  
}
```

### **frontend/src/metabase/admin/permissions/utils/graph/data-permissions/has.ts**

```
import _ from "underscore";  
  
import type {  
  DataPermission,  
  DataPermissionValue,  
  DatabaseEntityId,  
  EntityWithGroupId,  
  SchemaEntityId,  
} from "metabase/admin/permissions/types";  
import {...
```

### **frontend/src/metabase/admin/permissions/utils/graph/data-permissions/has.unit.spec.ts**

```
import _ from "underscore";  
  
import {  
  DataPermission,  
  DataPermissionValue,  
} from "metabase/admin/permissions/types";  
import { PLUGIN_ADVANCED_PERMISSIONS } from "metabase/plugins";  
import type {...
```

### **frontend/src/metabase/admin/permissions/utils/graph/data-permissions/index.ts**

```
export * from "./get";  
export * from "./has";  
export * from "./update";  
export * from "./utils";
```

### **frontend/src/metabase/admin/permissions/utils/graph/data-permissions/update.ts**

```
import { getIn, setIn } from "icepick";  
import _ from "underscore";
```

## Analyst Base

```
import type {  
  DatabaseEntityId,  
  EntityId,  
  SchemaEntityId,  
  TableEntityId,  
} from "metabase/admin/permissions/types";  
import...
```

### **frontend/src/metabase/admin/permissions/utils/graph/data-permissions/utils.ts**

```
import { DataPermissionValue } from "metabase/admin/permissions/types";  
import { PLUGIN_ADVANCED_PERMISSIONS } from "metabase/plugins";
```

```
export const isRestrictivePermission = (value:...
```

### **frontend/src/metabase/admin/permissions/utils/graph/index.ts**

```
export * from "./data-permissions";  
export * from "./permissions-diff";
```

### **frontend/src/metabase/admin/permissions/utils/graph/partial-updates.ts**

```
import _ from "underscore";
```

```
import type {  
  CollectionPermissions,  
  GroupsPermissions,  
} from "metabase-types/api";
```

```
// utils for dealing with partial graph updates
```

```
// select only the parts of the...
```

### **frontend/src/metabase/admin/permissions/utils/graph/partial-updates.unit.spec.ts**

```
import type { GroupsPermissions } from "metabase-types/api";
```

```
import { DataPermission, DataPermissionValue } from "../types";
```

```
import {  
  getModifiedCollectionPermissionsGraphParts,  
  ...
```

### **frontend/src/metabase/admin/permissions/utils/graph/permissions-diff.ts**

```
import type Database from "metabase-lib/v1/metadata/Database";  
import type {  
  ConcreteTableId,
```

## Analyst Base

```
Group,  
GroupsPermissions,  
} from "metabase-types/api";
```

```
import { DataPermission } from...
```

### **frontend/src/metabase/admin/permissions/utils/metadata.ts**

```
import type Metadata from "metabase-lib/v1/metadata/Metadata";  
import type Table from "metabase-lib/v1/metadata/Table";  
import type { ConcreteTableId } from "metabase-types/api";
```

```
import type {...
```

### **frontend/src/metabase/admin/permissions/urls.js**

```
import {  
  isDatabaseEntityId,  
  isSchemaEntityId,  
  isTableEntityId,  
} from "../data-entity-id";
```

```
export const DATABASES_BASE_PATH = `/admin/permissions/data/database`;  
export const GROUPS_BASE_PATH...
```

### **frontend/src/metabase/admin/permissions/urls.unit.spec.js**

```
import {  
  getDatabaseFocusPermissionsUrl,  
  getGroupFocusPermissionsUrl,  
} from "../urls";
```

```
describe("getDatabaseFocusPermissionsUrl", () => {  
  it("when entityId is not specified it returns base...
```

### **frontend/src/metabase/admin/settings/analytics.ts**

```
import { trackSchemaEvent } from "metabase/lib/analytics";
```

```
export const trackTrackingPermissionChanged = (isEnabled: boolean) => {  
  trackSchemaEvent("settings", {  
    event: isEnabled  
    ?...
```

### **frontend/src/metabase/admin/settings/auth/components/AuthCard/index.ts**

```
export * from "../AuthCard";
```

### **frontend/src/metabase/admin/settings/auth/components/GoogleAuthForm/index.ts**

## Analyst Base

```
export * from "./GoogleAuthForm";
```

### **frontend/src/metabase/admin/settings/auth/constants.ts**

```
import * as Yup from "yup";
```

```
import * as Errors from "metabase/lib/errors";
```

```
import type { SettingDefinition } from "metabase-types/api";
```

```
const REQUIRED_SCHEMA = {  
  is: {isEnabled: boolean,...
```

### **frontend/src/metabase/admin/settings/auth/containers/GoogleAuthCard/index.ts**

```
export * from "./GoogleAuthCard";
```

### **frontend/src/metabase/admin/settings/auth/containers/LdapAuthCard/index.ts**

```
export * from "./LdapAuthCard";
```

### **frontend/src/metabase/admin/settings/components/ApiKeys/utils.ts**

```
import { t } from "ttag";
```

```
import * as Yup from "yup";
```

```
import type { ApiKey } from "metabase-types/api";
```

```
export function formatMaskedKey(maskedKey: string) {  
  return maskedKey.substring(0, 7) +...
```

### **frontend/src/metabase/admin/settings/components/CloudPanel/index.ts**

```
export * from "./CloudPanel";
```

### **frontend/src/metabase/admin/settings/components/CloudPanel/utils.ts**

```
import dayjs from "dayjs";
```

```
import type {  
  CloudMigration,  
  CloudMigrationState,  
} from "metabase-types/api/cloud-migration";
```

```
export type InternalCloudMigrationState = CloudMigrationState |...
```

### **frontend/src/metabase/admin/settings/components/Email/SMTPConnectionCard/index.ts**

```
export * from "./SMTPConnectionCard";
```

### **frontend/src/metabase/admin/settings/components/Email/analytics.ts**

## Analyst Base

```
import { trackSimpleEvent } from "metabase/lib/analytics";
import type {
  CustomSMTPSetupClickedEvent,
  CustomSMTPSetupSuccessEvent,
} from "metabase-types/analytics";
```

```
export function...
```

**frontend/src/metabase/admin/settings/components/EmbeddingSettings/EmbeddingLegaleseModal/index.ts**

```
export * from "./EmbeddingLegaleseModal";
```

**frontend/src/metabase/admin/settings/components/EmbeddingSettings/EmbeddingOption/EmbeddingSdkOptionCard/index.ts**

```
export * from "./EmbeddingSdkOptionCard";
```

**frontend/src/metabase/admin/settings/components/EmbeddingSettings/EmbeddingOption/InteractiveEmbeddingOptionCard/index.ts**

```
export * from "./InteractiveEmbeddingOptionCard";
```

**frontend/src/metabase/admin/settings/components/EmbeddingSettings/EmbeddingOption/LinkButton/index.ts**

```
export * from "./LinkButton";
```

**frontend/src/metabase/admin/settings/components/EmbeddingSettings/EmbeddingOption/StaticEmbeddingOptionCard/index.ts**

```
export * from "./StaticEmbeddingOptionCard";
```

**frontend/src/metabase/admin/settings/components/EmbeddingSettings/EmbeddingOption/index.ts**

```
export { StaticEmbeddingOptionCard } from "./StaticEmbeddingOptionCard";
```

```
export { EmbeddingSdkOptionCard } from "./EmbeddingSdkOptionCard";
```

```
export { InteractiveEmbeddingOptionCard } from...
```

**frontend/src/metabase/admin/settings/components/EmbeddingSettings/EmbeddingOption/types.ts**

```
export type EmbeddingOptionIconProps = {
  disabled?: boolean;
};
```

**frontend/src/metabase/admin/settings/components/EmbeddingSettings/EmbeddingSdkSettings/index.ts**

```
export * from "./EmbeddingSdkSettings";
```

**frontend/src/metabase/admin/settings/components/EmbeddingSettings/Embed**

### **dingToggle/index.ts**

```
export * from "../EmbeddingToggle";
```

### **frontend/src/metabase/admin/settings/components/EmbeddingSettings/index.ts**

```
export { StaticEmbeddingSettings } from "../StaticEmbeddingSettings";  
export { EmbeddingSdkSettings } from "../EmbeddingSdkSettings/EmbeddingSdkSettings";
```

### **frontend/src/metabase/admin/settings/components/LicenseInput/index.ts**

```
export * from "../LicenseInput";
```

### **frontend/src/metabase/admin/settings/components/SettingHeader/index.ts**

```
export * from "../SettingHeader";
```

### **frontend/src/metabase/admin/settings/components/SettingsLicense/index.ts**

```
export * from "../SettingsLicense";
```

### **frontend/src/metabase/admin/settings/components/SettingsNav/index.ts**

```
export * from "../SettingsNav";
```

### **frontend/src/metabase/admin/settings/components/UploadSettings/utils.ts**

```
import type { Database, SchemaName } from "metabase-types/api";
```

```
export const getDatabaseOptions = (databases: Database[]) =>  
  databases.map((db) => ({  
    label: db.router_user_attribute  
    ? ...
```

### **frontend/src/metabase/admin/settings/components/UploadSettings/utils.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";  
import { checkNotNull } from "metabase/lib/types";  
import type { Database, Schema } from "metabase-types/api";  
import { createMockDatabase, ...
```

### **frontend/src/metabase/admin/settings/components/widgets/GroupMappingsWidget/index.ts**

```
export * from "../GroupMappingsWidget";
```

### **frontend/src/metabase/admin/settings/components/widgets/Notifications/utils.ts**

```
import type { NotificationChannel } from "metabase-types/api";
```

```
import type { WebhookFormProps } from "../WebhookForm";
```

```
export const buildAuthInfo = (
```

## Analyst Base

```
form: WebhookFormProps,  
): Record<string,...
```

### **frontend/src/metabase/admin/settings/components/widgets/Notifications/Utils.unit.spec.ts**

```
import {  
  createMockChannel,  
  createMockChannelDetails,  
} from "metabase-types/api/mocks/channel";  
  
import type { WebhookFormProps } from "../WebhookForm";  
import { buildAuthInfo, channelToForm }...
```

### **frontend/src/metabase/admin/settings/components/widgets/PublicLinksListing/index.ts**

```
export * from "../PublicLinksListing";  
export * from "../EmbeddedResources";  
export * from "../PublicResources";
```

### **frontend/src/metabase/admin/settings/components/widgets/VersionUpdateNotice/index.ts**

```
export { VersionUpdateNotice } from "../VersionUpdateNotice";
```

### **frontend/src/metabase/admin/settings/settings.js**

```
import {  
  combineReducers,  
  createAction,  
  createThunkAction,  
  handleActions,  
} from "metabase/lib/redux";  
import { refreshSiteSettings } from "metabase/redux/settings";  
import { SettingsApi }...
```

### **frontend/src/metabase/admin/settings/types.ts**

```
import type { ComponentType } from "react";  
  
import type {  
  SettingDefinition,  
  SettingKey,  
  SettingValue,  
  Settings,  
} from "metabase-types/api";  
  
export type SettingElement<Key extends...
```

## **frontend/src/metabase/admin/settings/utils.js**

```
import { t } from "ttag";

import { useDocsUrl } from "metabase/common/hooks";
```

```
export const settingToFormField = (setting) => ({
  name: setting.key,
  label: setting.display_name,
  description:...
```

## **frontend/src/metabase/admin/tools/components/Help/index.ts**

```
export { Help } from "./Help";
```

## **frontend/src/metabase/admin/tools/components/LogLevelsModal/index.ts**

```
export * from "./LogLevelsModal";
```

## **frontend/src/metabase/admin/tools/components/LogLevelsModal/types.ts**

```
import type { LoggerDurationUnit } from "metabase-types/api";
```

```
// Some options are not practically useful
```

```
export type AllowedTimeUnit = Exclude<
  LoggerDurationUnit,
  "nanoseconds" | "microseconds"...
```

## **frontend/src/metabase/admin/tools/components/LogLevelsModal/utils.ts**

```
import type { LoggerPreset } from "metabase-types/api";
```

```
export function getPresetJson(preset: LoggerPreset) {
  const logLevels = Object.fromEntries(
    preset.loggers.map(({ level, name }) =>...
```

## **frontend/src/metabase/admin/tools/components/Logs/hooks.ts**

```
import { useInterval } from "@mantine/hooks";
import { useCallback, useEffect, useRef, useState } from "react";
import { useMount, useUnmount } from "react-use";
import { t } from "ttag";
```

```
import {...
```

## **frontend/src/metabase/admin/tools/components/Logs/index.ts**

```
export * from "./Logs";
```

## **frontend/src/metabase/admin/tools/components/Logs/utils.ts**

```
import dayjs from "dayjs";
```



## Analyst Base

```
import orderBy from "lodash.orderby";
import _ from "underscore";
```

```
import {
  type QueryParam,
  type UrlStateConfig,
  getFirstParamValue,
} from...
```

### **frontend/src/metabase/admin/tools/components/ModelCacheRefreshJobs/index.ts**

```
export * from "./ModelCacheRefreshJobs";
export * from "./ModelCacheRefreshJobModal";
```

### **frontend/src/metabase/admin/tools/components/TaskModal/index.ts**

```
export * from "./TaskModal";
```

### **frontend/src/metabase/admin/tools/components/TaskPicker/index.ts**

```
export * from "./TaskPicker";
```

### **frontend/src/metabase/admin/tools/components/TaskStatusPicker/index.ts**

```
export * from "./TaskStatusPicker";
```

### **frontend/src/metabase/admin/tools/components/TasksApp/index.ts**

```
export * from "./TasksApp";
```

### **frontend/src/metabase/admin/tools/components/TasksApp/utils.ts**

```
import {
  type QueryParam,
  type UrlStateConfig,
  getFirstParamValue,
} from "metabase/common/hooks/use-url-state";
import type { ListTasksSortColumn, TaskStatus } from...
```

### **frontend/src/metabase/admin/tools/components/ToolsUpsell/index.ts**

```
export * from "./ToolsUpsell";
```

### **frontend/src/metabase/admin/types.ts**

```
export type MappingsType = Record<string, number[]>;
export type GroupIds = number[];
export type DeleteMappingModalValueType = "nothing" | "clear" | "delete";
export type UserGroupType = { id:...
```

### **frontend/src/metabase/admin/upsells/components/analytics.ts**

```
import { trackSchemaEvent } from "metabase/lib/analytics";
```

## Analyst Base

```
type UpsellEventProps = {  
  location: string;  
  campaign: string;  
};
```

```
export const trackUpsellViewed = ({ location, campaign }:...
```

### **frontend/src/metabase/admin/upsells/components/index.ts**

```
export * from "./UpsellBanner";  
export * from "./UpsellBigCard";  
export * from "./UpsellCard";  
export * from "./UpsellPill";  
export { UpsellGem } from "./UpsellGem";
```

### **frontend/src/metabase/admin/upsells/components/use-upsell-link.ts**

```
import { useUrlWithUtm } from "metabase/common/hooks";
```

```
interface UpsellLinkProps {  
  /* The URL we're sending them to */  
  url: string | undefined;  
  /* The name of the feature we're trying to sell...  
  feature: string;  
}
```

### **frontend/src/metabase/admin/upsells/constants.ts**

```
export const UPGRADE_URL = "https://www.metabase.com/upgrade";
```

### **frontend/src/metabase/admin/upsells/index.ts**

```
export * from "./UpsellBetterSupport";  
export * from "./UpsellCacheConfig";  
export * from "./UpsellCloud";  
export * from "./UpsellEmailWhitelabel";  
export * from "./UpsellHosting";  
export * from...
```

### **frontend/src/metabase/admin/upsells/readme.md**

Upsells Components

### **frontend/src/metabase/admin/utils.js**

```
import { push, replace, routerActions } from "react-router-redux";  
import { connectedReduxRedirect } from "redux-auth-wrapper/history3/redirect";
```

```
import { getAdminPaths } from...
```

### **frontend/src/metabase/api/action.ts**

```
import _ from "underscore";
```

```
import type {  
  CreateActionRequest,  
  GetActionRequest,  
  ListActionsRequest,  
  UpdateActionRequest,  
  WritebackAction,  
  WritebackActionId,  
} from...
```

### **frontend/src/metabase/api/activity.ts**

```
import type {  
  CreateRecentRequest,  
  Field,  
  PopularItem,  
  PopularItemsResponse,  
  RecentItem,  
  RecentsRequest,  
  RecentsResponse,  
  VisualizationDisplay,  
} from "metabase-types/api";
```

```
import {...
```

### **frontend/src/metabase/api/api-key.ts**

```
import type {  
  ApiKey,  
  ApiKeyId,  
  CreateApiKeyRequest,  
  CreateApiKeyResponse,  
  RegenerateApiKeyResponse,  
  UpdateApiKeyRequest,  
  UpdateApiKeyResponse,  
} from...
```

### **frontend/src/metabase/api/api.ts**

```
import {  
  buildCreateApi,  
  coreModule,  
  reactHooksModule,  
  skipToken,  
} from "@reduxjs/toolkit/query/react";  
import {  
  createDispatchHook,
```

## Analyst Base

```
createSelectorHook,  
createStoreHook,  
} from...
```

### **frontend/src/metabase/api/automagic-dashboards.ts**

```
import type {  
  Dashboard,  
  DashboardQueryMetadata,  
  DatabaseId,  
  DatabaseXray,  
  GetXrayDashboardQueryMetadataRequest,  
} from "metabase-types/api";
```

```
import { Api } from "../api";  
import {  
  ...
```

### **frontend/src/metabase/api/bookmark.ts**

```
import type {  
  Bookmark,  
  CreateBookmarkRequest,  
  DeleteBookmarkRequest,  
  ReorderBookmarksRequest,  
} from "metabase-types/api";
```

```
import { Api } from "../api";  
import {  
  idTag,  
  invalidateTags,  
  ...
```

### **frontend/src/metabase/api/bug-report.ts**

```
import type { ErrorPayload } from "metabase-types/api";
```

```
import { Api } from "../api";
```

```
interface BugReportResponse {  
  success: boolean;  
}
```

```
export const bugReportApi = Api.injectEndpoints({  
  ...
```

### **frontend/src/metabase/api/card.ts**

## Analyst Base

```
import { PLUGIN_API } from "metabase/plugins";
import type {
  Card,
  CardId,
  CardQueryMetadata,
  CardQueryRequest,
  CollectionItem,
  CreateCardFromCsvRequest,
  CreateCardRequest,
  ...
```

### frontend/src/metabase/api/channel.ts

```
import type { ChannelDetails, NotificationChannel } from "metabase-types/api";

import { Api } from "../api";
import { idTag, invalidateTags, listTag, provideChannelListTags } from "../tags";

const...
```

### frontend/src/metabase/api/cloud-migration.ts

```
import type { CloudMigration } from "metabase-types/api/cloud-migration";

import { Api } from "../api";
import { listTag } from "../tags";

export const cloudMigrationApi = Api.injectEndpoints({
  ...
```

### frontend/src/metabase/api/collection.ts

```
import type {
  Collection,
  CreateCollectionRequest,
  DeleteCollectionRequest,
  GetCollectionDashboardQuestionCandidatesRequest,
  GetCollectionDashboardQuestionCandidatesResult,
  ...
```

### frontend/src/metabase/api/dashboard.ts

```
import { PLUGIN_API } from "metabase/plugins";
import type {
  CopyDashboardRequest,
  CreateDashboardRequest,
  Dashboard,
  DashboardId,
```

## Analyst Base

```
DashboardQueryMetadata,  
FieldId,  
FieldValue,  
...
```

### **frontend/src/metabase/api/database.ts**

```
import type {  
  AutocompleteRequest,  
  AutocompleteSuggestion,  
  CardAutocompleteRequest,  
  CardAutocompleteSuggestion,  
  CreateDatabaseRequest,  
  Database,  
  Databaseld,  
  Field,  
  ...
```

### **frontend/src/metabase/api/dataset.ts**

```
import type {  
  CardQueryMetadata,  
  Dataset,  
  DatasetQuery,  
  FieldValue,  
  GetRemappedParameterValueRequest,  
  NativeDatasetResponse,  
} from "metabase-types/api";  
  
import { Api } from...
```

### **frontend/src/metabase/api/email.ts**

```
import type { EmailSMTPSettings } from "metabase-types/api";  
  
import { Api } from "../api";  
import { invalidateTags, tag } from "../tags";  
  
export const settingsApi = Api.injectEndpoints({  
  endpoints:...
```

### **frontend/src/metabase/api/entity-id.ts**

```
import {  
  type BaseEntityId,  
  isBaseEntityID,  
} from "metabase-types/api/entity-id";
```

## Analyst Base

```
import { Api } from "../api";
```

```
const validEntityTypes = [  
  "action",  
  "card",  
  "collection",  
  "dashboard",  
  ...  
]
```

### frontend/src/metabase/api/field.ts

```
import type {  
  CreateFieldDimensionRequest,  
  Field,  
  FieldDimension,  
  FieldId,  
  FieldValue,  
  GetFieldRequest,  
  GetFieldValuesResponse,  
  GetRemappedFieldValueRequest,  
  ...  
}
```

### frontend/src/metabase/api/geojson.ts

```
import type { Feature, FeatureCollection } from "geojson";  
import { t } from "ttag";
```

```
import { computeMinimalBounds } from "metabase/visualizations/lib/mapping";  
import type { GeoJSONData } from...
```

### frontend/src/metabase/api/google.ts

```
import type { EnterpriseSettings } from "metabase-types/api";
```

```
import { Api } from "../api";  
import { invalidateTags, tag } from "../tags";
```

```
type GoogleAuthSettings = Pick<  
  EnterpriseSettings,  
  |...  
>
```

### frontend/src/metabase/api/index.ts

```
export * from "../action";  
export * from "../activity";  
export * from "../api";  
export * from "../api-key";  
export * from "../automagic-dashboards";
```

## Analyst Base

```
export * from "./bookmark";  
export * from...
```

### frontend/src/metabase/api/ldap.ts

```
import type { EnterpriseSettings } from "metabase-types/api";  
  
import { Api } from "./api";  
import { invalidateTags, tag } from "./tags";  
  
type LdapSettings = Pick<  
  EnterpriseSettings,  
  |...
```

### frontend/src/metabase/api/logger.ts

```
import type { AdjustLogLevelsRequest, LoggerPreset } from "metabase-types/api";  
  
import { Api } from "./api";  
import { provideLoggerPresetListTags } from "./tags";  
  
export const loggerApi =...
```

### frontend/src/metabase/api/login-history.ts

```
import type { UserLoginHistory } from "metabase-types/api";  
  
import { Api } from "./api";  
  
export const loginHistoryApi = Api.injectEndpoints({  
  endpoints: (builder) => ({  
    getLoginHistory:...
```

### frontend/src/metabase/api/model-index.ts

```
import type {  
  ModelIndex,  
  ModelIndexCreateQuery,  
  ModelIndexDeleteQuery,  
  ModelIndexesListQuery,  
} from "metabase-types/api";  
  
import { Api } from "./api";  
import { invalidateTags, listTag,...
```

### frontend/src/metabase/api/moderation.ts

```
import type { VerifyItemRequest } from "metabase-types/api";
```



## Analyst Base

```
import { Api } from "./api";  
import { invalidateTags, provideModeratedItemTags } from "./tags";
```

```
export const contentVerificationApi =...
```

### **frontend/src/metabase/api/notification.ts**

```
import type {  
  CreateNotificationRequest,  
  ListNotificationsRequest,  
  Notification,  
  NotificationId,  
  UpdateNotificationRequest,  
} from "metabase-types/api/notification";
```

```
import { Api } from...
```

### **frontend/src/metabase/api/parameters.ts**

```
import type {  
  GetParameterValuesRequest,  
  ParameterValues,  
  SearchParameterValuesRequest,  
} from "metabase-types/api";
```

```
import { Api } from "./api";  
import { idTag } from "./tags";
```

```
export const...
```

### **frontend/src/metabase/api/permission.ts**

```
import type {  
  BaseGroupInfo,  
  CreateMembershipRequest,  
  Group,  
  GroupId,  
  GroupListQuery,  
  ListUserMembershipsResponse,  
  Membership,  
} from "metabase-types/api";
```

```
import { Api } from...
```

### **frontend/src/metabase/api/persist.ts**

```
import type {  
  CardId,
```

## Analyst Base

```
ListPersistedInfoRequest,  
ListPersistedInfoResponse,  
ModelCacheRefreshStatus,  
PersistedInfoFold,  
PersistedInfoRefreshSchedule,  
} from "metabase-types/api";
```

```
import {...
```

### **frontend/src/metabase/api/product-feedback.ts**

```
import { Api } from "../api";
```

```
const productFeedbackApi = Api.injectEndpoints({  
  endpoints: (builder) => ({  
    sendProductFeedback: builder.mutation<  
      void,  
      { comment?: string; email?:...
```

### **frontend/src/metabase/api/query.ts**

```
import type { BaseQueryFn } from "@reduxjs/toolkit/query/react";
```

```
import api from "metabase/lib/api";
```

```
type AllowedHTTPMethods = "GET" | "POST" | "PUT" | "DELETE";  
const allowedHTTPMethods = new...
```

### **frontend/src/metabase/api/revision.ts**

```
import type {  
  ListRevisionRequest,  
  RevertRevisionRequest,  
  Revision,  
} from "metabase-types/api";
```

```
import { Api } from "../api";  
import { invalidateTags, listTag, provideRevisionListTags } from...
```

### **frontend/src/metabase/api/search.ts**

```
import { trackSearchRequest } from "metabase/search/analytics";  
import type { SearchRequest, SearchResponse } from "metabase-types/api";
```

```
import { Api } from "../api";  
import {...
```

### **frontend/src/metabase/api/segment.ts**

## Analyst Base

```
import type {  
  CreateSegmentRequest,  
  DeleteSegmentRequest,  
  Segment,  
  SegmentId,  
  UpdateSegmentRequest,  
} from "metabase-types/api";
```

```
import { Api } from "../api";  
import {  
  idTag,  
  ...
```

### frontend/src/metabase/api/session.ts

```
import MetabaseSettings from "metabase/lib/settings";  
import { loadSettings } from "metabase/redux/settings";  
import type {  
  EnterpriseSettings,  
  PasswordResetTokenStatus,  
} from...
```

### frontend/src/metabase/api/settings.ts

```
import _ from "underscore";  
  
import type {  
  EnterpriseSettingKey,  
  EnterpriseSettingValue,  
  EnterpriseSettings,  
  SettingDefinition,  
  SettingDefinitionMap,  
} from "metabase-types/api";
```

```
import {...
```

### frontend/src/metabase/api/slack.ts

```
import type { EnterpriseSettings } from "metabase-types/api";  
  
import { Api } from "../api";
```

```
type SlackSettings = Pick<  
  EnterpriseSettings,  
  "slack-app-token" | "slack-bug-report-channel" |...
```

### frontend/src/metabase/api/snippet.ts

## Analyst Base

```
import type {  
  CreateSnippetRequest,  
  ListSnippetsParams,  
  NativeQuerySnippet,  
  NativeQuerySnippetId,  
  UpdateSnippetRequest,  
} from "metabase-types/api";
```

```
import { Api } from "../api";  
import {  
  ...
```

### **frontend/src/metabase/api/subscription.ts**

```
import type {  
  ChannelApiResponse,  
  CreateSubscriptionRequest,  
  DashboardSubscription,  
  ListSubscriptionsRequest,  
  UpdateSubscriptionRequest,  
} from "metabase-types/api";
```

```
import { Api } from...
```

### **frontend/src/metabase/api/table.ts**

```
import type {  
  Field,  
  GetTableQueryMetadataRequest,  
  GetTableRequest,  
  Table,  
  TableId,  
  TableListQuery,  
  UpdateTableFieldsOrderRequest,  
  UpdateTableListRequest,  
  UpdateTableRequest,  
} from...
```

### **frontend/src/metabase/api/tags/constants.ts**

```
export type TagType = (typeof TAG_TYPES)[number];
```

```
export const TAG_TYPES = [  
  "action",  
  "alert",  
  "api-key",  
  "bookmark",
```

```
"card",  
"channel",  
"cloud-migration",  
"collection",  
...
```

### **frontend/src/metabase/api/tags/index.ts**

```
export * from "../constants";  
export * from "../utils";
```

### **frontend/src/metabase/api/tags/utils.ts**

```
import type { TagDescription } from "@reduxjs/toolkit/query";  
  
import { isVirtualDashCard } from "metabase/dashboard/utils";  
import type {  
  Alert,  
  ApiKey,  
  Bookmark,  
  Card,  
  CardId,  
  ...  

```

### **frontend/src/metabase/api/task.ts**

```
import type {  
  ListTasksRequest,  
  ListTasksResponse,  
  Task,  
  TaskInfo,  
} from "metabase-types/api";  
  
import { Api } from "../api";  
import {  
  provideTaskListTags,  
  provideTaskTags,  
  ...  

```

### **frontend/src/metabase/api/timeline-event.ts**

```
import type {  
  CreateTimelineEventRequest,  
  TimelineEvent,  
  TimelineEventId,  
  UpdateTimelineEventRequest,  
} from "metabase-types/api";
```

## Analyst Base

```
import { Api } from "../api";
import {
  idTag,
  ...
```

### frontend/src/metabase/api/timeline.ts

```
import type {
  CreateTimelineRequest,
  GetTimelineRequest,
  ListCollectionTimelinesRequest,
  ListTimelinesRequest,
  Timeline,
  TimelineId,
  UpdateTimelineRequest,
} from...
```

### frontend/src/metabase/api/user-key-value.ts

```
import type {
  DeleteUserKeyValueRequest,
  GetUserKeyValueRequest,
  UpdateUserKeyValueRequest,
} from "metabase-types/api";
```

```
import { Api } from "../api";
```

```
export const userKeyValueApi =...
```

### frontend/src/metabase/api/user.ts

```
import { STORE_TEMPORARY_PASSWORD } from "metabase/admin/people/events";
import { userUpdated } from "metabase/redux/user";
import type {
  CreateUserRequest,
  ListUsersRequest,
  ...
```

### frontend/src/metabase/api/utils/errors.ts

```
import { t } from "ttag";
```

```
type ErrorPayload =
  | { message: string }
  | { error: string }
  | { error_message: string }
  | string;
```

## Analyst Base

```
export const getErrorMessage = (  
  payload:  
    | unknown  
    | ...
```

### **frontend/src/metabase/api/utils/errors.unit.spec.ts**

```
import { getErrorMessage } from "../errors";  
  
describe("getErrorMessage", () => {  
  it("should return a message from a string payload", () => {  
    const result = getErrorMessage("Some error...
```

### **frontend/src/metabase/api/utils/index.ts**

```
export * from "../errors";  
export * from "../settings";  
export * from "../use-token-refresh";
```

### **frontend/src/metabase/api/utils/readme.md**

```
# API Utils
```

### **frontend/src/metabase/api/utils/settings.ts**

```
import { useCallback } from "react";  
import { t } from "ttag";  
  
import { useToast } from "metabase/common/hooks";  
import type {  
  EnterpriseSettingKey,  
  EnterpriseSettingValue,  
  ...
```

### **frontend/src/metabase/api/utils/use-token-refresh.ts**

```
import { useEffect } from "react";  
  
import { Api, useGetSettingsQuery } from "metabase/api";  
import { useDispatch } from "metabase/lib/redux";
```

```
const REFRESH_INTERVAL = 10 * 1000; // 10 seconds
```

```
/**
```

```
...
```

### **frontend/src/metabase/app-embed.js**

```
/*  
 * This file is subject to the terms and conditions defined in  
 * file 'LICENSE-EMBEDDING.txt', which is part of this source code package.
```

\*/

```
import { isWithinIframe } from...
```

### **frontend/src/metabase/app-main.js**

Enables hot reload in development and noop in production

MUST be imported BEFORE `react` and `react-dom`

### **frontend/src/metabase/app-public.js**

```
import { init } from "./app";
import { publicReducers } from "./reducers-public";
import { getRoutes } from "./routes-public";
```

```
init(publicReducers, getRoutes, () => {});
```

### **frontend/src/metabase/app.js**

```
import "@mantine/core/styles.css";
import "@mantine/dates/styles.css";
```

```
import "regenerator-runtime/runtime";
```

```
// This is conditionally aliased in the webpack config.
```

```
// If EE isn't enabled, it loads...
```

### **frontend/src/metabase/archive/actions.ts**

```
import { push } from "react-router-redux";
import { t } from "ttag";
```

```
import { createThunkAction } from "metabase/lib/redux";
import { addUndo } from "metabase/redux/undo";
```

```
export const...
```

### **frontend/src/metabase/archive/analytics.ts**

```
import { trackSchemaEvent } from "metabase/lib/analytics";
import type { MoveToTrashEvent } from "metabase-types/analytics";
```

```
export const archiveAndTrack = async ({
  archive,
  model,
  modelId,
  ...

```

### **frontend/src/metabase/archive/utils.ts**

```
import { c } from "ttag";
```



## Analyst Base

```
import _ from "underscore";
```

```
import type { Collection, CollectionItem } from "metabase-types/api";
```

```
/**
```

```
 * @param updateActionResult - result value of await...
```

### **frontend/src/metabase/auth/actions.ts**

```
import { type UnknownAction, createAction } from "@reduxjs/toolkit";
```

```
import { getIn } from "icepick";
```

```
import { push } from "react-router-redux";
```

```
import { deleteSession, initiateSLO } from...
```

### **frontend/src/metabase/auth/components/AuthButton/index.ts**

```
export * from "./AuthButton";
```

### **frontend/src/metabase/auth/components/AuthLayout/index.ts**

```
export * from "./AuthLayout";
```

### **frontend/src/metabase/auth/components/ForgotPassword/index.ts**

```
export * from "./ForgotPassword";
```

### **frontend/src/metabase/auth/components/ForgotPasswordForm/index.ts**

```
export * from "./ForgotPasswordForm";
```

### **frontend/src/metabase/auth/components/GoogleButton/index.ts**

```
export * from "./GoogleButton";
```

### **frontend/src/metabase/auth/components/Login/index.ts**

```
export * from "./Login";
```

### **frontend/src/metabase/auth/components/LoginForm/index.ts**

```
export * from "./LoginForm";
```

### **frontend/src/metabase/auth/components/Logout/index.ts**

```
export * from "./Logout";
```

### **frontend/src/metabase/auth/components/PasswordButton/index.ts**

```
export * from "./PasswordButton";
```

### **frontend/src/metabase/auth/components/PasswordPanel/index.ts**

```
export * from "./PasswordPanel";
```

### **frontend/src/metabase/auth/components/ResetPassword/index.ts**

```
export * from "./ResetPassword";
```

## Analyst Base

### **frontend/src/metabase/auth/components/ResetPasswordForm/index.ts**

```
export * from "./ResetPasswordForm";
```

### **frontend/src/metabase/auth/selectors.ts**

```
import { createSelector } from "@reduxjs/toolkit";
```

```
import { PLUGIN_AUTH_PROVIDERS } from "metabase/plugins";
```

```
import type { AuthProvider } from "metabase/plugins/types";
```

```
import { getSetting,...
```

### **frontend/src/metabase/auth/types.ts**

```
export interface LoginData {
```

```
  username: string;
```

```
  password: string;
```

```
  remember?: boolean;
```

```
}
```

```
export interface ForgotPasswordData {
```

```
  email: string;
```

```
}
```

```
export interface ResetPasswordData {
```

```
  ...
```

### **frontend/src/metabase/browse/constants.ts**

```
export const RELOAD_INTERVAL = 2000;
```

### **frontend/src/metabase/browse/containers/TableBrowser/TableBrowser.unit.spec.js**

```
import fetchMock from "fetch-mock";
```

```
import {
```

```
  renderWithProviders,
```

```
  screen,
```

```
  waitFor,
```

```
  waitForLoaderToBeRemoved,
```

```
} from "__support__/ui";
```

```
// import { RELOAD_INTERVAL } from...
```

### **frontend/src/metabase/browse/containers/TableBrowser/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/browse/databases/analytics.ts**

## Analyst Base

```
import { trackSimpleEvent } from "metabase/lib/analytics";
```

```
export const trackAddDatabaseDBList = () => {  
  trackSimpleEvent({  
    event: "database_add_clicked",  
    triggered_from: "db-list",  
    ...  
  })  
}
```

### frontend/src/metabase/browse/metrics/analytics.ts

```
import { trackSimpleEvent } from "metabase/lib/analytics";
```

```
export const trackNewMetricInitiated = () =>  
  trackSimpleEvent({  
    event: "plus_button_clicked",  
    triggered_from: "metric",  
  });
```

### frontend/src/metabase/browse/metrics/test-utils.ts

```
import {  
  createMockRecentCollectionItem,  
  createMockSearchResult,  
} from "metabase-types/api/mocks";
```

```
import type { MetricResult, RecentMetric } from "../types";
```

```
export const...
```

### frontend/src/metabase/browse/metrics/utils.ts

```
import { t } from "ttag";
```

```
import { getCollectionPathAsString } from "metabase/collections/utils";  
import { formatValue } from "metabase/lib/formatting";  
import { isDate } from...
```

### frontend/src/metabase/browse/metrics/utils.unit.spec.ts

```
import {  
  createMockCollection,  
  createMockColumn,  
  createMockDataset,  
  createMockDatasetData,  
} from "metabase-types/api/mocks";  
import { SortDirection } from...
```

### frontend/src/metabase/browse/models/EmptyStates/index.ts

## Analyst Base

```
export { ModelsVideo, ModelsVideoThumbnail } from "../ModelsVideo";
```

### **frontend/src/metabase/browse/models/analytics.ts**

```
import { trackSchemaEvent, trackSimpleEvent } from "metabase/lib/analytics";
import type { CardId } from "metabase-types/api";
```

```
export const trackModelClick = (modelId: CardId) =>
```

```
...
```

### **frontend/src/metabase/browse/models/test-utils.ts**

```
import type { RecentCollectionItem } from "metabase-types/api";
```

```
import {
  createMockRecentCollectionItem,
  createMockSearchResult,
} from "metabase-types/api/mocks";
```

```
import type { ModelResult, ...
```

### **frontend/src/metabase/browse/models/utils.ts**

```
import { t } from "ttag";
```

```
import { getCollectionPathAsString } from "metabase/collections/utils";
import { entityForObject } from "metabase/lib/schema";
import type { IconName } from ...
```

### **frontend/src/metabase/browse/tables/TableBrowser/TableBrowser.unit.spec.js**

```
import { setupDatabaseEndpoints } from "__support__/server-mocks";
import { renderWithProviders, screen } from "__support__/ui";
import { createMockDatabase } from "metabase-types/api/mocks";
```

```
import ...
```

### **frontend/src/metabase/browse/tables/TableBrowser/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/browse/tables/TableBrowser/useDatabaseCrumb.ts**

```
import { t } from "ttag";
```

```
import { skipToken, useGetDatabaseQuery } from "metabase/api";
import Databases from "metabase/entities/databases";
import { useSelector } from "metabase/lib/redux";
import ...
```

### **frontend/src/metabase/browse/tables/analytics.ts**

```
import { trackSchemaEvent } from "metabase/lib/analytics";
```

## Analyst Base

```
import type { ConcreteTableId } from "metabase-types/api";
```

```
export const trackTableClick = (tableId: ConcreteTableId) =>
```

```
...
```

### **frontend/src/metabase/collections/components/ActionMenu/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/collections/components/CollectionContent/constants.ts**

```
import type { CollectionItemModel } from "metabase-types/api";
```

```
export const COLLECTION_PAGE_SIZE = 25;
```

```
export const COLLECTION_CONTENT_COLUMNS = [
```

```
  "type",
```

```
  "name",
```

```
  "lastEditedBy",
```

```
...
```

### **frontend/src/metabase/collections/components/CollectionContent/index.ts**

```
export * from "./CollectionContent";
```

```
export * from "./constants";
```

### **frontend/src/metabase/collections/components/CollectionContent/utils.ts**

```
import type { DragEvent, DragEventHandler } from "react";
```

```
import type { DropzoneRootProps } from "react-dropzone";
```

```
export const composeFileEventHandler =
```

```
(fn: DragEventHandler<HTMLInputElement> |...
```

### **frontend/src/metabase/collections/components/CollectionContent/utils.unit.spec.ts**

```
import type { DragEvent } from "react";
```

```
import type { DropzoneRootProps } from "react-dropzone";
```

```
import { composeFileEventHandler, getComposedDragProps } from "./utils";
```

```
describe("Collections >...
```

### **frontend/src/metabase/collections/components/CollectionEmptyState/analytics.ts**

```
import { trackSimpleEvent } from "metabase/lib/analytics";
```

```
export const trackCollectionNewButtonClicked = () =>
```

## Analyst Base

```
trackSimpleEvent({  
  event: "new_button_clicked",  
  triggered_from:...
```

### **frontend/src/metabase/collections/components/CollectionEmptyState/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/collections/components/CollectionHeader/analytics.ts**

```
import { trackSimpleEvent } from "metabase/lib/analytics";
```

```
export const trackNewCollectionFromHeaderInitiated = () =>  
  trackSimpleEvent({  
    event: "plus_button_clicked",  
    triggered_from:...
```

### **frontend/src/metabase/collections/components/CollectionHeader/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/collections/components/CollectionInfoSidebar/index.ts**

```
export { CollectionInfoSidebar } from "./CollectionInfoSidebar";
```

### **frontend/src/metabase/collections/components/CollectionLanding/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/collections/components/CollectionMenu/index.ts**

```
export { CollectionMenu } from "./CollectionMenu";
```

### **frontend/src/metabase/collections/components/CreateCollectionForm/index.ts**

```
export { default as CreateCollectionForm } from "./CreateCollectionForm";  
export type { CreateCollectionFormOwnProps } from "./CreateCollectionForm";
```

### **frontend/src/metabase/collections/components/MoveCollectionModal/index.ts**

```
export * from "./MoveCollectionModal";
```

### **frontend/src/metabase/collections/components/PinDropZone/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/collections/components/PinnedItemCard/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/collections/components/PinnedItemOverview/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/collections/components/PinnedItemSortDropTarget/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## Analyst Base

### **frontend/src/metabase/collections/components/PinnedQuestionCard/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/collections/components/TrashCollectionLanding/index.ts**

```
export * from "./TrashCollectionLanding";
```

### **frontend/src/metabase/collections/components/UploadOverlay/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/collections/components/utils.ts**

```
import type { CollectionItem } from "metabase-types/api";
```

```
export const findLastEditedCollectionItem = (
  collectionItems: CollectionItem[],
) => {
  return collectionItems.reduce((latest, item) =>...
```

### **frontend/src/metabase/collections/containers/CollectionHeader/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/collections/containers/FormCollectionAndDashboardPicker/index.ts**

```
export * from "./FormCollectionAndDashboardPicker";
```

### **frontend/src/metabase/collections/containers/FormCollectionPicker/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/collections/hooks.ts**

```
import getInitialCollectionId from "metabase/entities/collections/getInitialCollectionId";
import { useSelector } from "metabase/lib/redux";
import { PLUGIN_COLLECTIONS } from...
```

### **frontend/src/metabase/collections/types.ts**

```
import type {
  BookmarkId,
  BookmarkType,
  CardId,
  Collection,
  CollectionId,
  CollectionItem,
  Dashboard,
  TableId,
} from "metabase-types/api";
```

## Analyst Base

```
import type { UploadMode } from...
```

### **frontend/src/metabase/collections/utils.ts**

```
import { t } from "ttag";
```

```
import { PLUGIN_COLLECTIONS } from "metabase/plugins";
```

```
import type {  
  Collection,  
  CollectionEssentials,  
  CollectionId,  
  CollectionItem,  
} from...
```

### **frontend/src/metabase/collections/utils.unit.spec.ts**

```
import {  
  canonicalCollectionId,  
  getCollectionPathAsString,  
  isExamplesCollection,  
  isItemCollection,  
  isReadOnlyCollection,  
  isRootCollection,  
  isRootPersonalCollection,  
  ...
```

### **frontend/src/metabase/common/components/AccordionList/index.ts**

```
export { AccordionList } from "./AccordionList";  
export type { Section, SearchProp, SearchProps } from "./types";
```

### **frontend/src/metabase/common/components/AccordionList/types.ts**

```
import type { ReactNode } from "react";
```

```
import type { IconName } from "metabase/ui";
```

```
export type Item = object | string;
```

```
export type Section<TItem extends Item = Item> = {  
  key?: string;  
  name?:...
```

### **frontend/src/metabase/common/components/AccordionList/utils.ts**

```
import { shallowEqual } from "@mantine/hooks";  
import { getIn } from "icepick";  
import { type ReactNode, isValidElement } from "react";  
import { isFragment } from "react-is";
```



## Analyst Base

```
import type { Item,...
```

### **frontend/src/metabase/common/components/AccordionList/utils.unit.spec.ts**

```
import { getNextCursor, getPrevCursor } from "../utils";
```

```
const canSelectSection = () => true;
const cannotSelectSection = () => false;
```

```
const sections = [
  {
    id: 1,
    items: [
      {
        ...

```

### **frontend/src/metabase/common/components/AdHocQuestionLoader.unit.spec.js**

```
import { render } from "__support__/ui";
import { delay } from "__support__/utils";
import Question from "metabase-lib/v1/Question";
import * as ML_Urls from "metabase-lib/v1/urls";
```

```
import {...
```

### **frontend/src/metabase/common/components/AdminPaneLayout/types.ts**

```
import type { ReactNode } from "react";
```

```
export type AdminPaneProps = {
  title?: React.ReactNode;
  description?: string;
  buttonText?: string;
  buttonAction?: () => void;
  buttonDisabled?:...
```

### **frontend/src/metabase/common/components/AggregationPicker/index.ts**

```
export * from "../AggregationPicker";
```

### **frontend/src/metabase/common/components/Alert/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/AppBanner/index.ts**

```
export { AppBanner } from "../AppBanner";
```

### **frontend/src/metabase/common/components/AppBanner/utils.ts**

## Analyst Base

```
import dayjs from "dayjs";
import utc from "dayjs/plugin/utc";
```

```
dayjs.extend(utc);
```

```
type Props = {
  daysRemaining: number;
  lastDismissed?: string | null;
  tokenExpiryTimestamp: string;
  ...
}
```

### **frontend/src/metabase/common/components/AppBanner/utils.unit.spec.ts**

```
import dayjs from "dayjs";
```

```
import { getCurrentUTCTimestamp, shouldShowTrialBanner } from "../utils";
```

```
describe("app banner utils", () => {
  describe("shouldShowBanner", () => {
    it("should return...
```

### **frontend/src/metabase/common/components/AutocompleteInput/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/Banner/index.ts**

```
export { Banner } from "../Banner";
```

### **frontend/src/metabase/common/components/BookmarkToggle/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/Button/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/ButtonGroup/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/CheckBox/constants.ts**

```
export const DEFAULT_SIZE = 16;
export const DEFAULT_ICON_PADDING = 4;
export const DEFAULT_CHECKED_COLOR = "brand";
export const DEFAULT_UNCHECKED_COLOR = "text-light";
```

### **frontend/src/metabase/common/components/CheckBox/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/CheckBox/types.ts**

```
export interface CheckBoxInputProps {
```

## Analyst Base

```
size: number;  
}
```

```
export interface CheckBoxContainerProps {  
  disabled: boolean | undefined;  
}
```

```
export interface CheckBoxIconProps {  
  checked: boolean;  
  ...
```

### **frontend/src/metabase/common/components/CheckBox/utils.ts**

```
export function isEllipsisActive(node: HTMLElement): boolean {  
  return node.offsetWidth < node.scrollWidth;  
}
```

### **frontend/src/metabase/common/components/CodeEditor/index.ts**

```
export * from "./CodeEditor";  
export type { CodeLanguage } from "./types";
```

### **frontend/src/metabase/common/components/CodeEditor/types.ts**

```
export type CodeLanguage =  
  | "clojure"  
  | "html"  
  | "json"  
  | "mustache"  
  | "pug"  
  | "python"  
  | "ruby"  
  | "typescript";
```

### **frontend/src/metabase/common/components/CodeEditor/utils.ts**

```
import { html } from "@codemirror/lang-html";  
import { javascript } from "@codemirror/lang-javascript";  
import { json } from "@codemirror/lang-json";  
import { python } from...
```

### **frontend/src/metabase/common/components/CodeMirror/highlights.ts**

```
import { type Range, StateEffect, StateField } from "@codemirror/state";  
import { Decoration, EditorView } from "@codemirror/view";  
import { type RefObject, useEffect } from "react";
```

```
import S from...
```

### **frontend/src/metabase/common/components/CodeMirror/index.ts**

## Analyst Base

```
export { CodeMirror, type CodeMirrorProps } from "./CodeMirror";
export { type CodeMirrorRef } from "./types";
```

### **frontend/src/metabase/common/components/CodeMirror/types.ts**

```
export { type ReactCodeMirrorRef as CodeMirrorRef } from "@uiw/react-codemirror";
```

### **frontend/src/metabase/common/components/CodeMirror/utils.ts**

```
import {
  acceptCompletion,
  moveCompletionSelection,
  nextSnippetField,
  prevSnippetField,
} from "@codemirror/autocomplete";
import { indentMore } from "@codemirror/commands";
import {
  ...
```

### **frontend/src/metabase/common/components/CollapseSection/index.ts**

```
eslint-disable-next-line import/no-default-export -- legacy usage
```

### **frontend/src/metabase/common/components/ColorInput/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/ColorPicker/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/ColorPill/index.ts**

```
export { ColorPill } from "./ColorPill";
export type { PillSize } from "./types";
```

### **frontend/src/metabase/common/components/ColorPill/types.ts**

```
export type PillSize = "small" | "medium";
```

### **frontend/src/metabase/common/components/ColorRange/index.ts**

```
export { ColorRange } from "./ColorRange";
```

### **frontend/src/metabase/common/components/ColorRangeSelector/index.ts**

```
export { ColorRangeSelector } from "./ColorRangeSelector";
```

### **frontend/src/metabase/common/components/ColorSelector/index.ts**

```
export { ColorSelector } from "./ColorSelector";
```

### **frontend/src/metabase/common/components/CommunityLocalizationNotice/index.ts**

```
export {
  getLocalizationNoticeText,
```

## Analyst Base

```
CommunityLocalizationNotice,  
} from "../CommunityLocalizationNotice";
```

### **frontend/src/metabase/common/components/CopyButton/index.ts**

```
export * from "../CopyButton";
```

### **frontend/src/metabase/common/components/CopyTextInput/index.ts**

```
export * from "../CopyTextInput";
```

### **frontend/src/metabase/common/components/DashboardSelector/index.ts**

```
export * from "../DashboardSelector";
```

### **frontend/src/metabase/common/components/DragDropContext/index.ts**

```
export { DragDropContext } from "../DragDropContext";
```

### **frontend/src/metabase/common/components/EditBar/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/EditableDescription/index.ts**

```
export { EditableDescription } from "../EditableDescription";
```

### **frontend/src/metabase/common/components/EditableText/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/Ellipsified/index.ts**

```
export { Ellipsified } from "../Ellipsified";
```

### **frontend/src/metabase/common/components/EllipsifiedPath/index.ts**

```
export { EllipsifiedPath } from "../EllipsifiedPath";
```

### **frontend/src/metabase/common/components/EntityIdCard/index.ts**

```
export * from "../EntityIdCard";
```

### **frontend/src/metabase/common/components/EntityMenuItem/EntityMenuItem.unit.spec.js**

```
import userEvent from "@testing-library/user-event";
```

```
import { fireEvent, getIcon, render, screen } from "__support__/ui";
```

```
import { delay } from "__support__/utils";
```

```
import EntityMenuItem from...
```

### **frontend/src/metabase/common/components/EntityMenuTrigger/EntityMenuTrigger.unit.spec.js**

```
import { fireEvent, render, screen } from "__support__/ui";
```

```
import EntityMenuTrigger from "metabase/common/components/EntityMenuTrigger";
```

## Analyst Base

```
describe("EntityMenuTrigger", () => {  
  it("should render...
```

### **frontend/src/metabase/common/components/EntityPicker/components/AutoScrollBox/index.ts**

```
export * from "./AutoScrollBox";
```

### **frontend/src/metabase/common/components/EntityPicker/components/EntityPickerModal/index.ts**

```
export * from "./EntityPickerModal";
```

### **frontend/src/metabase/common/components/EntityPicker/components/ItemList/index.ts**

```
export * from "./ItemList";
```

### **frontend/src/metabase/common/components/EntityPicker/components/LoadingSpinner/index.ts**

```
export * from "./LoadingSpinner";
```

### **frontend/src/metabase/common/components/EntityPicker/components/NestedItemPicker/index.ts**

```
export * from "./NestedItemPicker";  
export * from "./NestedItemPicker.styled";
```

### **frontend/src/metabase/common/components/EntityPicker/components/NestedItemPicker/Utils.ts**

```
import type { PickerState } from "../../types";
```

```
// reverse-traverse the statePath to find the last selected item  
export const findLastSelectedItem = <Item, Query>(  
  statePath: PickerState<Item,...
```

### **frontend/src/metabase/common/components/EntityPicker/components/RecentsTab/index.ts**

```
export * from "./RecentsTab";
```

### **frontend/src/metabase/common/components/EntityPicker/components/RecentsTab/Utils.ts**

```
import dayjs from "dayjs";  
import relativeTime from "dayjs/plugin/relativeTime";  
import { t } from "ttag";  
dayjs.extend(relativeTime);
```

```
import type { RecentItem } from "metabase-types/api";
```

import...

## **frontend/src/metabase/common/components/EntityPicker/components/RecentsTab/utils.unit.spec.ts**

import dayjs from "dayjs";

import { createMockRecentCollectionItem } from "metabase-types/api/mocks";

import { getRecentGroups } from "../utils";

```
const items = [  
  createMockRecentCollectionItem({  
    ...
```

## **frontend/src/metabase/common/components/EntityPicker/components/ResultItem/index.ts**

export \* from "../ResultItem";

export \* from "../ResultItem.styled";

## **frontend/src/metabase/common/components/EntityPicker/components/SearchTab/index.ts**

export \* from "../SearchTab";

## **frontend/src/metabase/common/components/EntityPicker/components/index.ts**

export \* from "../AutoScrollBox";

export \* from "../EntityPickerModal";

export \* from "../ItemList";

export \* from "../LoadingSpinner";

export \* from "../NestedItemPicker";

## **frontend/src/metabase/common/components/EntityPicker/constants.ts**

export const RECENTS\_TAB\_ID = "recents-tab";

export const SEARCH\_TAB\_ID = "search-tab";

## **frontend/src/metabase/common/components/EntityPicker/hooks/index.ts**

export \* from "../use-log-recent-item";

export \* from "../use-scoped-search-results";

## **frontend/src/metabase/common/components/EntityPicker/hooks/use-scoped-search-results.ts**

import { useMemo } from "react";

## Analyst Base

```
import {
  skipToken,
  useGetDatabaseQuery,
  useListCollectionItemsQuery,
  useListDashboardItemsQuery,
  useListDatabaseSchemaTablesQuery,
} from...
```

### **frontend/src/metabase/common/components/EntityPicker/index.ts**

```
export * from "./components";
export * from "./types";
```

### **frontend/src/metabase/common/components/EntityPicker/types.ts**

```
import type { ReactNode } from "react";

import type { IconName } from "metabase/ui";
import type { SearchResult, SearchResultId } from "metabase-types/api";

import type { EntityPickerModalOptions }...
```

### **frontend/src/metabase/common/components/EntityPicker/utils.ts**

```
import { c, msgid, t } from "ttag";

import { color } from "metabase/lib/colors";
import type { ObjectWithModel } from "metabase/lib/icon";
import { getIcon } from "metabase/lib/icon";
import {
  ...
```

### **frontend/src/metabase/common/components/ErrorDetails/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/ErrorDetails/types.ts**

```
export type ErrorDetails = string | Record<string, any>;

export interface ErrorDetailsProps {
  details?: ErrorDetails;
  centered?: boolean;
  className?: string;
}
```

### **frontend/src/metabase/common/components/ErrorPages/analytics.ts**

```
import { trackSimpleEvent } from "metabase/lib/analytics";
import type {
  ErrorDiagnosticModalOpenedEvent,
```



## Analyst Base

```
ErrorDiagnosticModalSubmittedEvent,  
} from "metabase-types/analytics/event";
```

```
export...
```

### **frontend/src/metabase/common/components/ErrorPages/index.ts**

```
export * from "./ErrorPages";  
export * from "./ErrorDiagnosticModal";
```

### **frontend/src/metabase/common/components/ErrorPages/types.ts**

```
import type {  
  Card,  
  Collection,  
  Dashboard,  
  DatasetData,  
  Log,  
  MetabaseInfo,  
} from "metabase-types/api";
```

```
export type ReportableEntityName =  
  | "question"  
  | "model"  
  | "metric"  
  | ...
```

### **frontend/src/metabase/common/components/ErrorPages/use-error-info.ts**

```
import { useAsync } from "react-use";  
import { t } from "ttag";
```

```
import { getCurrentUser } from "metabase/admin/datamodel/selectors";  
import { useSelector } from "metabase/lib/redux";  
import { ...
```

### **frontend/src/metabase/common/components/ErrorPages/utils.ts**

```
import Browser from "browser";
```

```
import { b64url_to_utf8 } from "metabase/lib/encoding";  
import { CardApi, CollectionsApi, DashboardApi } from "metabase/services";
```

```
import type { ReportableEntityName }...
```

### **frontend/src/metabase/common/components/ExplicitSize/\_\_mocks\_\_/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/ExplicitSize/index.ts**

## Analyst Base

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/ExportSettingsWidget/index.ts**

export \* from "./ExportSettingsWidget";

**frontend/src/metabase/common/components/ExternalLink/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/FileInput/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/FormErrorMessage/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/FormField/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/FormField/types.ts**

export type FieldAlignment = "start" | "end";

export type FieldOrientation = "horizontal" | "vertical";

**frontend/src/metabase/common/components/FormFileInput/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/FormFooter/index.ts**

export \* from "./FormFooter";

**frontend/src/metabase/common/components/FormInput/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/FormNumericInput/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/FormRadio/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/FormSelect/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/FormSubmitButton/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/FormTextArea/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/common/components/FormToggle/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## Analyst Base

### **frontend/src/metabase/common/components/HelpCard/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/common/components/Input/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/common/components/InputWithSelectPrefix/index.ts**

export \* from "../InputWithSelectPrefix";

### **frontend/src/metabase/common/components/ItemsTable/BaselItemsTable/index.ts**

export \* from "../BaselItemsTable";

### **frontend/src/metabase/common/components/ItemsTable/utils.ts**

```
import type {
  CollectionContentTableColumn,
  CollectionContentTableColumnsMap,
} from "metabase/collections/components/CollectionContent";
import { type BreakpointName, breakpoints } from...
```

### **frontend/src/metabase/common/components/LastEditInfoLabel/LastEditInfoLabel.unit.spec.js**

```
import dayjs from "dayjs";
import mockDate from "mockdate";
```

```
import { renderWithProviders, screen } from "__support__/ui";
import { createMockUser } from "metabase-types/api/mocks";
```

import...

### **frontend/src/metabase/common/components/LeaveConfirmModal/index.ts**

```
export { LeaveRouteConfirmModal } from "../LeaveRouteConfirmModal";
export { LeaveConfirmModal } from "../LeaveConfirmModal";
```

### **frontend/src/metabase/common/components/Link/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/common/components/Link/types.ts**

```
import type { LinkProps as RouterLinkProps } from "react-router";
```

```
import type { TooltipProps } from "metabase/ui";
```

```
export interface LinkProps extends RouterLinkProps {
  variant?: "default" |...
```

## Analyst Base

### **frontend/src/metabase/common/components/ListItem/ListItem.unit.spec.js**

```
import { Route } from "react-router";

import { getIcon, renderWithProviders, screen } from "__support__/ui";

import ListItem from "./ListItem";

const ITEM_NAME = "Table Foo";
const...
```

### **frontend/src/metabase/common/components/ListSearchField/ListSearchField.unit.spec.js**

```
import { render, screen } from "__support__/ui";

import ListSearchField from "./ListSearchField";

describe("ListSearchField", () => {
  it("should render", async () => {
    render(<ListSearchField...
```

### **frontend/src/metabase/common/components/LoadingSpinner/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/common/components/Markdown/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/common/components/MarkdownPreview/index.ts**

```
export { MarkdownPreview } from "./MarkdownPreview";
```

### **frontend/src/metabase/common/components/MetabotLogo/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/common/components/MetadataInfo/ColumnFingerprintInfo/ColumnFingerprintInfo.unit.spec.js**

```
import { createMockEntitiesState } from "__support__/store";
import { renderWithProviders, screen } from "__support__/ui";
import { getMetadata } from "metabase/selectors/metadata";
import {
  ...
```

### **frontend/src/metabase/common/components/MetadataInfo/ColumnFingerprintInfo/GlobalFingerprint.unit.spec.js**

```
import { setupFieldValuesEndpoint } from "__support__/server-mocks";
import { createMockEntitiesState } from "__support__/store";
```

## Analyst Base

```
import { renderWithProviders, screen } from "__support__/ui";
import...
```

### **frontend/src/metabase/common/components/MetadataInfo/ColumnFingerprintInfo/index.ts**

```
export * from "./ColumnFingerprintInfo";
```

### **frontend/src/metabase/common/components/MetadataInfo/ColumnInfo/ColumnInfo.unit.spec.js**

```
import { setupFieldsValuesEndpoints } from "__support__/server-mocks";
import { createMockEntitiesState } from "__support__/store";
import { renderWithProviders, screen } from...
```

### **frontend/src/metabase/common/components/MetadataInfo/ColumnInfo/index.ts**

```
export { QueryColumnInfo, TableColumnInfo } from "./ColumnInfo";
export type { QueryColumnInfoProps, TableColumnInfoProps } from "./ColumnInfo";
```

### **frontend/src/metabase/common/components/MetadataInfo/ColumnInfoPopover/index.ts**

```
export * from "./ColumnInfoPopover";
```

### **frontend/src/metabase/common/components/MetadataInfo/SemanticTypeLabel/SemanticTypeLabel.unit.spec.js**

```
import { createMockEntitiesState } from "__support__/store";
import { getIcon, renderWithProviders, screen } from "__support__/ui";
import { getMetadata } from "metabase/selectors/metadata";
import...
```

### **frontend/src/metabase/common/components/MetadataInfo/SemanticTypeLabel/index.ts**

```
export { SemanticTypeLabel } from "./SemanticTypeLabel";
```

### **frontend/src/metabase/common/components/MetadataInfo/TableInfo/index.ts**

```
export type { TableInfoProps } from "./TableInfo";
// eslint-disable-next-line import/no-default-export -- deprecated usage
export { default } from "./TableInfo";
```

### **frontend/src/metabase/common/components/MetadataInfo/TableInfoPopover/index.ts**

```
export * from "./TableInfoPopover";
```

### **frontend/src/metabase/common/components/ModalContent/ModalContent.style.d.ts**

```
eslint-disable-next-line no-restricted-imports
```

## Analyst Base

### **frontend/src/metabase/common/components/ModalContent/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/common/components/ModalContent/types.ts**

```
import type { ReactNode } from "react";
```

```
export interface CommonModalProps {  
  // takes over the entire screen  
  fullPageModal?: boolean;  
  // standard modal  
  formModal?: boolean;  
  centeredTitle?:...
```

### **frontend/src/metabase/common/components/MoveQuestionsIntoDashboardsModal/hooks.ts**

```
import {  
  skipToken,  
  useListCollectionDashboardQuestionCandidatesQuery,  
} from "metabase/api";  
import { useSelector } from "metabase/lib/redux";  
import { getUserIsAdmin } from...
```

### **frontend/src/metabase/common/components/MultiContainerDraggableContext/index.ts**

```
export * from "../MultiContainerDraggableContext";
```

### **frontend/src/metabase/common/components/NewItemMenu/analytics.ts**

```
import { trackSimpleEvent } from "metabase/lib/analytics";
```

```
export const trackNewItemClicked = (  
  item: "question" | "native-query" | "dashboard",  
) =>  
  trackSimpleEvent({  
    event:...
```

### **frontend/src/metabase/common/components/NewItemMenu/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/common/components/NumericInput/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/common/components>PasswordReveal/PasswordReveal.unit.spec.js**

```
import { fireEvent, render, screen } from "__support__/ui";
```

## Analyst Base

```
import PasswordReveal from "metabase/common/components/PasswordReveal";
```

```
describe("password reveal", () => {  
  it("should toggle the...
```

### **frontend/src/metabase/common/components/Pickers/CollectionPicker/components/index.ts**

```
export * from "./CollectionPicker";  
export * from "./CollectionPickerModal";
```

### **frontend/src/metabase/common/components/Pickers/CollectionPicker/hooks.ts**

```
import { useEffect, useMemo, useState } from "react";  
import { t } from "ttag";
```

```
import {  
  skipToken,  
  useGetCollectionQuery,  
  useListCollectionItemsQuery,  
} from "metabase/api";  
import {...
```

### **frontend/src/metabase/common/components/Pickers/CollectionPicker/index.ts**

```
export * from "./components";  
export * from "./hooks";  
export * from "./types";
```

### **frontend/src/metabase/common/components/Pickers/CollectionPicker/types.ts**

```
import type {  
  CardId,  
  CollectionId,  
  CollectionItemModel,  
  DashboardId,  
  ListCollectionItemsRequest,  
  SearchResult,  
} from "metabase-types/api";
```

```
import type {  
  EntityPickerModalOptions,  
  ...
```

### **frontend/src/metabase/common/components/Pickers/CollectionPicker/utils.ts**

```
import type {  
  CollectionId,  
  CollectionItemModel,
```

## Analyst Base

```
ListCollectionItemsRequest,  
} from "metabase-types/api";
```

```
import type { PickerState } from "../../EntityPicker";  
import type {...
```

### **frontend/src/metabase/common/components/Pickers/CollectionPicker/utils.unit.spec.ts**

```
import { getParentCollectionId } from "./utils";
```

```
describe("CollectionPicker > utils", () => {  
  describe("getParentCollectionId", () => {  
    it("should get the root collection for null values", ()...
```

### **frontend/src/metabase/common/components/Pickers/DashboardPicker/components/index.ts**

```
export * from "./DashboardPicker";  
export * from "./DashboardPickerModal";
```

### **frontend/src/metabase/common/components/Pickers/DashboardPicker/index.ts**

```
export * from "./components";  
export * from "./types";
```

### **frontend/src/metabase/common/components/Pickers/DashboardPicker/types.ts**

```
import type {  
  CollectionId,  
  DashboardId,  
  ListCollectionItemsRequest,  
  SearchModel,  
} from "metabase-types/api";
```

```
import type {  
  EntityPickerModalOptions,  
  ListProps,  
  PickerState,  
} from...
```

### **frontend/src/metabase/common/components/Pickers/DashboardPicker/utils.ts**

```
import _ from "underscore";
```

```
import type { CollectionId, Dashboard } from "metabase-types/api";
```

```
import {  
  getParentCollectionId,
```



## Analyst Base

```
    getPathLevelForItem,  
  } from "../CollectionPicker/utils";
```

```
import...
```

**frontend/src/metabase/common/components/Pickers/DataPicker/components/index.ts**

```
export * from "./DataPickerModal";
```

**frontend/src/metabase/common/components/Pickers/DataPicker/hooks/index.ts**

```
export * from "./useAvailableData";
```

**frontend/src/metabase/common/components/Pickers/DataPicker/hooks/useAvailableData.ts**

```
import { useSearchQuery } from "metabase/api";  
import type { DatabaseId } from "metabase-types/api";
```

```
interface Props {  
  databaseId?: DatabaseId;  
}
```

```
export const useAvailableData = ({ databaseId } :...
```

**frontend/src/metabase/common/components/Pickers/DataPicker/index.ts**

```
export * from "./components";  
export * from "./types";  
export * from "./utils";
```

**frontend/src/metabase/common/components/Pickers/DataPicker/types.ts**

```
import type {  
  CardId,  
  Collection,  
  DashboardId,  
  DatabaseId,  
  SchemaName,  
  TableId,  
} from "metabase-types/api";
```

```
import type { EntityPickerModalOptions } from "../../EntityPicker";  
import...
```

**frontend/src/metabase/common/components/Pickers/DataPicker/utils.ts**

```
import { humanize, titleize } from "metabase/lib/formatting";  
import { isNullOrUndefined } from "metabase/lib/types";
```

## Analyst Base

```
import * as Lib from "metabase-lib";
import { getSchemaName } from...
```

### **frontend/src/metabase/common/components/Pickers/QuestionPicker/components/index.ts**

```
export * from "./QuestionPicker";
export * from "./QuestionPickerModal";
```

### **frontend/src/metabase/common/components/Pickers/QuestionPicker/index.ts**

```
export * from "./components";
export * from "./types";
export { getQuestionPickerValue } from "./utils";
```

### **frontend/src/metabase/common/components/Pickers/QuestionPicker/types.ts**

```
import type {
  CardId,
  ListCollectionItemsRequest,
  SearchModel,
} from "metabase-types/api";
```

```
import type {
  EntityPickerModalOptions,
  ListProps,
  PickerState,
} from...
```

### **frontend/src/metabase/common/components/Pickers/QuestionPicker/utils.ts**

```
import _ from "underscore";
```

```
import type { Card, CardType, CollectionItemModel } from "metabase-types/api";
```

```
import type {
  QuestionPickerItem,
  QuestionPickerValue,
  QuestionPickerValueModel,
}...
```

### **frontend/src/metabase/common/components/Pickers/hooks.ts**

```
import {
  skipToken,
  useGetCardQuery,
  useGetCollectionQuery,
  useGetDashboardQuery,
} from "metabase/api";
```

## Analyst Base

```
import { isValidCollectionId } from "metabase/collections/utils";
```

```
import type {...
```

### **frontend/src/metabase/common/components/Pickers/utils.ts**

```
import { PERSONAL_COLLECTIONS } from "metabase/entities/collections/constants";
```

```
import { isNullOrUndefined } from "metabase/lib/types";
```

```
import type {
```

```
  CollectionId,
```

```
  CollectionItemModel,
```

```
...
```

### **frontend/src/metabase/common/components/Pickers/utils.unit.spec.ts**

```
import { getCollectionIdPath } from "./utils";
```

```
describe("getCollectionIdPath", () => {
```

```
  it("should handle the current user's personal collection", () => {
```

```
    const path = getCollectionIdPath(
```

```
    ...
```

### **frontend/src/metabase/common/components/Popover/SizeToFitModifier.ts**

```
import * as popper from "@popperjs/core";
```

```
const PAGE_PADDING = 10;
```

```
const SIZE_TO_FIT_MIN_HEIGHT = 200;
```

```
export type SizeToFitOptions = {
```

```
  minHeight: number;
```

```
};
```

```
export function...
```

### **frontend/src/metabase/common/components/Popover/TippyPopover.unit.spec.js**

```
eslint-disable react/prop-types
```

### **frontend/src/metabase/common/components/Popover/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/PopoverWithTrigger/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/QueryColumnPicker/BucketPicker Popover/index.ts**

```
export * from "./BucketPickerPopover";
```

## Analyst Base

### **frontend/src/metabase/common/components/QueryColumnPicker/BucketPicker Popover/types.ts**

```
import type * as Lib from "metabase-lib";
```

```
import type { BaseBucketPickerPopoverProps } from "../BaseBucketPickerPopover";
```

```
type CommonProps = Pick<  
  BaseBucketPickerPopoverProps,  
  | "query"  
  | ...
```

### **frontend/src/metabase/common/components/QueryColumnPicker/index.ts**

```
export * from "../QueryColumnPicker";
```

### **frontend/src/metabase/common/components/QuestionResultLoader.unit.spec.js**

```
import { setupCardQueryEndpoints } from "__support__/server-mocks";  
import { render } from "__support__/ui";  
import Question from "metabase-lib/v1/Question";  
import { createMockDataset } from...
```

### **frontend/src/metabase/common/components/QuestionSavedModal/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/Radio/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/Radio/types.ts**

```
export type RadioVariant = "normal" | "underlined" | "bubble";  
export type RadioColorScheme = "default" | "accent7";
```

### **frontend/src/metabase/common/components/SaveQuestionForm/index.ts**

```
export * from "../SaveQuestionTitle";  
export * from "../SaveQuestionForm";
```

### **frontend/src/metabase/common/components/SaveQuestionForm/schema.ts**

```
import * as Yup from "yup";
```

```
import * as Errors from "metabase/lib/errors";  
import { QUESTION_NAME_MAX_LENGTH } from "metabase/questions/constants";
```

```
export const SAVE_QUESTION_SCHEMA = Yup.object({  
  ...
```

### **frontend/src/metabase/common/components/SaveQuestionForm/types.ts**

## Analyst Base

```
import type Question from "metabase-lib/v1/Question";
import type {
  CollectionId,
  DashboardId,
  DashboardTabId,
} from "metabase-types/api";
```

```
export type SaveQuestionProps<C = CollectionId> = {
  ...
```

### **frontend/src/metabase/common/components/SaveQuestionForm/util.ts**

```
import { P, match } from "ts-pattern";
import { t } from "ttag";
```

```
import { canonicalCollectionId } from "metabase/collections/utils";
import { isNullOrUndefined } from "metabase/lib/types";
import...
```

### **frontend/src/metabase/common/components/SaveQuestionForm/util.unit.spec.ts**

```
import Question from "metabase-lib/v1/Question";
import { createMockCard } from "metabase-types/api/mocks";
```

```
import type { CreateQuestionOptions } from "../types";
import {
  createQuestion,
  ...
```

### **frontend/src/metabase/common/components/SaveQuestionModal/index.ts**

```
export * from "./SaveQuestionModal";
```

### **frontend/src/metabase/common/components/SavedQuestionLoader.unit.spec.js**

```
import {
  setupCardEndpoints,
  setupCardQueryMetadataEndpoint,
  setupDatabaseEndpoints,
  setupUnauthorizedCardEndpoints,
  setupUnauthorizedSchemaEndpoints,
} from...
```

### **frontend/src/metabase/common/components/Schedule/constants.ts**

```
export const defaultDay = "mon";
export const defaultHour = 8;
```

### **frontend/src/metabase/common/components/Schedule/strings.ts**

## Analyst Base

```
import { c, msgid, ngettext, t } from "ttag";
import _ from "underscore";
```

```
import { has24HourModeSetting } from "metabase/lib/time";
import type { ScheduleDayType, ScheduleFrameType } from...
```

### **frontend/src/metabase/common/components/Schedule/strings.unit.spec.ts**

```
import { has24HourModeSetting } from "metabase/lib/time";
```

```
import { getHours } from "../strings";
```

```
jest.mock("metabase/lib/time", () => ({
  has24HourModeSetting: jest.fn(),
}));
```

```
describe("getHours",...
```

### **frontend/src/metabase/common/components/Schedule/types.ts**

```
import type { ScheduleSettings, ScheduleType } from "metabase-types/api";
```

```
type ScheduleProperty = keyof ScheduleSettings;
export type ScheduleChangeProp = { name: ScheduleProperty; value: unknown...
```

### **frontend/src/metabase/common/components/SchedulePicker/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/SegmentedControl/index.ts**

```
export * from "../SegmentedControl";
```

### **frontend/src/metabase/common/components/Select/Select.unit.spec.js**

```
import { render, screen } from "__support__/ui";
import Select, { Option } from "metabase/common/components/Select";
```

```
describe("Select", () => {
  it("should render selected option", () => {
    ...
```

### **frontend/src/metabase/common/components/Select/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/SelectButton/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/SelectList/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

## Analyst Base

### **frontend/src/metabase/common/components/Sidesheet/components/InsightsTab/index.ts**

```
export { InsightsTab } from "./InsightsTab";
```

### **frontend/src/metabase/common/components/Sidesheet/components/InsightsTabOrLink/index.ts**

```
export { InsightsTabOrLink } from "./InsightsTabOrLink";
```

### **frontend/src/metabase/common/components/Sidesheet/index.ts**

```
export * from "./Sidesheet";
export * from "./SidesheetButton";
export * from "./SidesheetCard";
export * from "./SidesheetCardSection";
export * from "./SidesheetSubPage";
export * from...
```

### **frontend/src/metabase/common/components/Sortable/index.ts**

```
export * from "./Sortable";
export * from "./SortableList";
```

### **frontend/src/metabase/common/components/Swapper/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/Tab/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/Tab/utils.ts**

```
export function getTabId<T>(idPrefix: string, value: T): string {
  return `${idPrefix}-T-${value}`;
}
```

```
export function getTabPanelId<T>(idPrefix: string, value: T): string {
  return...
```

### **frontend/src/metabase/common/components/TabButton/index.ts**

```
export * from "./TabButton";
```

### **frontend/src/metabase/common/components/TabContent/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/TabList/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/TabPanel/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

## Analyst Base

### **frontend/src/metabase/common/components/TabRow/collision-detection.ts**

```
import type { CollisionDetection } from "@dnd-kit/core";

export const tabsCollisionDetection: CollisionDetection = ({
  active,
  collisionRect,
  droppableRects,
  droppableContainers,
}) => {
  //...
```

### **frontend/src/metabase/common/components/TabRow/index.ts**

```
export * from "./TabRow";
```

### **frontend/src/metabase/common/components/Table/index.ts**

```
export * from "./ClientSortableTable";
export * from "./Table";
```

### **frontend/src/metabase/common/components/Table/types.ts**

```
export type BaseRow = Record<string, any> & { id: number | string };

export type ColumnItem = {
  name: string;
  key: string;
  sortable?: boolean;
};
```

### **frontend/src/metabase/common/components/TextArea/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/Timeline/index.ts**

```
export * from "./Timeline";
```

### **frontend/src/metabase/common/components/Timeline/utils.ts**

```
import { t } from "ttag";

import type { Revision, User } from "metabase-types/api";

export function getTimelineEvents({
  revisions = [],
  currentUser,
}): {
  revisions: Revision[] | undefined;
  ...
```



## Analyst Base

### **frontend/src/metabase/common/components/TitleAndDescription/index.ts**

```
export { TitleAndDescription } from "./TitleAndDescription";
```

### **frontend/src/metabase/common/components/Toaster/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/Toggle/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/TokenField/TokenField.unit.spec.js**

```
eslint-disable react/prop-types
```

### **frontend/src/metabase/common/components/TokenField/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/TokenField/utils.ts**

```
export function parseStringValue(value: any): string | null {
  const trimmedValue = trim(value);
  if (trimmedValue === "") {
    return null;
  }

  return trimmedValue;
}
```

```
function trim(value:...
```

### **frontend/src/metabase/common/components/TokenField/utils.unit.spec.ts**

```
import { parseStringValue } from "./utils";
```

```
describe("metabase/common/components/TokenField/utils", () => {
  describe("parseStringValue", () => {
    it("should return null for falsy and whitespace...
```

### **frontend/src/metabase/common/components/TokenFieldItem/TokenFieldItem.styled.ts**

```
eslint-disable-next-line no-restricted-imports
```

### **frontend/src/metabase/common/components/TokenFieldItem/index.ts**

```
export * from "./TokenFieldItem.styled";
```

### **frontend/src/metabase/common/components/ToolbarButton/index.ts**

```
export * from "./ToolbarButton";
```

## Analyst Base

### **frontend/src/metabase/common/components/Tooltip/Tooltip.unit.spec.js**

```
import userEvent from "@testing-library/user-event";
import { forwardRef, useState } from "react";
```

```
import { render, screen } from "__support__/ui";
```

```
import Tooltip from "../Tooltip";
```

```
const...
```

### **frontend/src/metabase/common/components/Tooltip/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/UserAvatar/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/components/ViewFooterButton/index.ts**

```
export * from "../ViewFooterButton";
```

### **frontend/src/metabase/common/components/VirtualizedList/index.ts**

```
export * from "../VariableHeightVirtualizedList";
```

### **frontend/src/metabase/common/components/dnd/index.ts**

NOTE: we currently use object's `model` property for the drag type

### **frontend/src/metabase/common/components/tree/index.ts**

```
export * from "../Tree";
```

### **frontend/src/metabase/common/components/tree/types.ts**

```
import type * as React from "react";
```

```
import type { IconName, IconProps } from "metabase/ui";
```

```
export interface ITreeNodeItem {
```

```
  id: string | number;
```

```
  name: string;
```

```
  icon: IconName | IconProps;
```

```
  ...
```

### **frontend/src/metabase/common/components/upload/constants.ts**

```
export const DEFAULT_UPLOAD_INPUT_ID = "upload-input";
```

### **frontend/src/metabase/common/components/upload/index.ts**

```
export * from "../UploadTooltip";
```

```
export * from "../UploadInput";
```

## Analyst Base

```
export * from "./UploadLabel";
```

### **frontend/src/metabase/common/hooks/constants.ts**

```
export const LOAD_COMPLETE_FAVICON = "app/assets/img/blue_check.png";
```

### **frontend/src/metabase/common/hooks/entity-framework/index.ts**

```
/*
```

```
 * This directory is deprecated.
```

```
 * Use "metabase/api" instead.
```

```
*/
```

```
export * from "./use-bookmark-list-query";
```

```
export * from "./use-collection-query";
```

```
export * from...
```

### **frontend/src/metabase/common/hooks/entity-framework/use-bookmark-list-query/index.ts**

```
export * from "./use-bookmark-list-query";
```

### **frontend/src/metabase/common/hooks/entity-framework/use-bookmark-list-query/use-bookmark-list-query.ts**

```
import Bookmarks from "metabase/entities/bookmarks";
```

```
import type { Bookmark } from "metabase-types/api";
```

```
import type {
```

```
  UseEntityListQueryProps,
```

```
  UseEntityListQueryResult,
```

```
} from...
```

### **frontend/src/metabase/common/hooks/entity-framework/use-collection-query/index.ts**

```
export * from "./use-collection-query";
```

### **frontend/src/metabase/common/hooks/entity-framework/use-collection-query/use-collection-query.ts**

```
import Collections from "metabase/entities/collections";
```

```
import type { Collection, CollectionId } from "metabase-types/api";
```

```
import type {
```

```
  UseEntityQueryProps,
```

```
  UseEntityQueryResult,
```

```
} from...
```

### **frontend/src/metabase/common/hooks/entity-framework/use-dashboard-query/i**

## **index.ts**

```
export * from "../use-dashboard-query";
```

## **frontend/src/metabase/common/hooks/entity-framework/use-dashboard-query/use-dashboard-query.ts**

```
import type {  
  UseEntityQueryProps,  
  UseEntityQueryResult,  
} from "metabase/common/hooks/entity-framework/use-entity-query";  
import { useEntityQuery } from...
```

## **frontend/src/metabase/common/hooks/entity-framework/use-database-list-query/index.ts**

```
export * from "../use-database-list-query";
```

## **frontend/src/metabase/common/hooks/entity-framework/use-database-list-query/use-database-list-query.ts**

```
import type {  
  UseEntityListQueryProps,  
  UseEntityListQueryResult,  
} from "metabase/common/hooks/entity-framework/use-entity-list-query";  
import { useEntityListQuery } from...
```

## **frontend/src/metabase/common/hooks/entity-framework/use-database-query/index.ts**

```
export * from "../use-database-query";
```

## **frontend/src/metabase/common/hooks/entity-framework/use-database-query/use-database-query.ts**

```
import type {  
  UseEntityQueryProps,  
  UseEntityQueryResult,  
} from "metabase/common/hooks/entity-framework/use-entity-query";  
import { useEntityQuery } from...
```

## **frontend/src/metabase/common/hooks/entity-framework/use-entity-list-query/index.ts**

```
export * from "../use-entity-list-query";
```

## **frontend/src/metabase/common/hooks/entity-framework/use-entity-list-query/use-entity-list-query.ts**

```
import type { Action } from "@reduxjs/toolkit";  
import { useDeepCompareEffect, usePrevious } from "react-use";
```

## Analyst Base

```
import { useDispatch, useSelector } from "metabase/lib/redux";
import type { State }...
```

### **frontend/src/metabase/common/hooks/entity-framework/use-entity-query/index.ts**

```
export * from "./use-entity-query";
```

### **frontend/src/metabase/common/hooks/entity-framework/use-entity-query/use-entity-query.ts**

```
import type { Action } from "@reduxjs/toolkit";
import { useDeepCompareEffect } from "react-use";
```

```
import { useDispatch, useSelector } from "metabase/lib/redux";
import type { State } from...
```

### **frontend/src/metabase/common/hooks/entity-framework/use-group-list-query/index.ts**

```
export * from "./use-group-list-query";
```

### **frontend/src/metabase/common/hooks/entity-framework/use-group-list-query/use-group-list-query.ts**

```
import type {
  UseEntityListQueryProps,
  UseEntityListQueryResult,
} from "metabase/common/hooks/entity-framework/use-entity-list-query";
import { useEntityListQuery } from...
```

### **frontend/src/metabase/common/hooks/entity-framework/use-question-list-query/index.ts**

```
export * from "./use-question-list-query";
```

### **frontend/src/metabase/common/hooks/entity-framework/use-question-list-query/use-question-list-query.ts**

```
import type {
  UseEntityListQueryProps,
  UseEntityListQueryResult,
} from "metabase/common/hooks/entity-framework/use-entity-list-query";
import { useEntityListQuery } from...
```

### **frontend/src/metabase/common/hooks/entity-framework/use-question-query/index.ts**

```
export * from "./use-question-query";
```

## **frontend/src/metabase/common/hooks/entity-framework/use-question-query/use-question-query.ts**

```
import type {
  UseEntityQueryProps,
  UseEntityQueryResult,
} from "metabase/common/hooks/entity-framework/use-entity-query";
import { useEntityQuery } from...
```

## **frontend/src/metabase/common/hooks/entity-framework/use-revision-list-query/index.ts**

```
export * from "./use-revision-list-query";
```

## **frontend/src/metabase/common/hooks/entity-framework/use-revision-list-query/use-revision-list-query.ts**

```
import type {
  UseEntityListQueryProps,
  UseEntityListQueryResult,
} from "metabase/common/hooks/entity-framework/use-entity-list-query";
import { useEntityListQuery } from...
```

## **frontend/src/metabase/common/hooks/entity-framework/use-search-list-query/index.ts**

```
export * from "./use-search-list-query";
```

## **frontend/src/metabase/common/hooks/entity-framework/use-search-list-query/use-search-list-query.ts**

```
import Search from "metabase/entities/search";
import type {
  CollectionItem,
  SearchRequest,
  SearchResponse,
} from "metabase-types/api";
```

```
import type {
  UseEntityListQueryProps,
  ...
```

## **frontend/src/metabase/common/hooks/entity-framework/use-table-list-query/index.ts**

```
export * from "./use-table-list-query";
```

## **frontend/src/metabase/common/hooks/entity-framework/use-table-list-query/use-table-list-query.ts**

## Analyst Base

```
import type {
  UseEntityListQueryProps,
  UseEntityListQueryResult,
} from "metabase/common/hooks/entity-framework/use-entity-list-query";
import { useEntityListQuery } from...
```

### **frontend/src/metabase/common/hooks/entity-framework/use-table-query/index.ts**

```
export * from "./use-table-query";
```

### **frontend/src/metabase/common/hooks/entity-framework/use-table-query/use-table-query.ts**

```
import type {
  UseEntityQueryProps,
  UseEntityQueryResult,
} from "metabase/common/hooks/entity-framework/use-entity-query";
import { useEntityQuery } from...
```

### **frontend/src/metabase/common/hooks/index.ts**

```
export * from "./entity-framework";
export * from "./use-confirmation";
export * from "./use-docs-url";
export * from "./use-has-token-feature";
export * from "./use-locale";
export * from...
```

### **frontend/src/metabase/common/hooks/use-action-button-label/index.ts**

```
export * from "./use-action-button-label";
```

### **frontend/src/metabase/common/hooks/use-action-button-label/use-action-button-label.ts**

```
import { type ReactNode, useRef, useState } from "react";
```

```
interface UseActionButtonLabelProps {
  defaultLabel: string | ReactNode;
  timeout?: number;
}
```

```
/**
```

```
 * Small hook to temporarily update a...
```

### **frontend/src/metabase/common/hooks/use-before-unload/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/common/hooks/use-before-unload/use-before-unload.ts**

## Analyst Base

```
import { useBeforeUnload as useBeforeUnloadHook } from "react-use";
import { t } from "ttag";
```

// most browsers don't use a custom message with beforeunload anymore, just putting here to retain...

### **frontend/src/metabase/common/hooks/use-callback-effect/index.ts**

```
export {
  useCallbackEffect,
  type ScheduleCallback,
} from "../use-callback-effect";
```

### **frontend/src/metabase/common/hooks/use-callback-effect/use-callback-effect.ts**

```
import { useCallback, useEffect, useState } from "react";

type Callback = () => void | Promise<void>;

type IsScheduled = boolean;

export type ScheduleCallback = (callback: Callback) => void;
```

/\*\*

...

### **frontend/src/metabase/common/hooks/use-confirm-route-leave-modal.ts**

```
import type { Location } from "history";
import { useCallback, useEffect, useState } from "react";
import type { InjectedRouter, Route } from "react-router";
import { goBack, push, replace } from...
```

### **frontend/src/metabase/common/hooks/use-current-ref.ts**

```
import { useRef } from "react";

export const useCurrentRef = function <T>(value: T) {
  const ref = useRef(value);
  ref.current = value;

  return ref;
};
```

### **frontend/src/metabase/common/hooks/use-debounced-value.ts**

```
import { useEffect, useState } from "react";

export function useDebouncedValue<TVALUE>()
```



## Analyst Base

```
value: TVALUE,  
delay: number,  
onlyFor: (lastValue: TVALUE, newValue: TVALUE) => boolean = () => true,  
):...
```

### **frontend/src/metabase/common/hooks/use-docs-url/index.ts**

```
export * from "./use-docs-url";
```

### **frontend/src/metabase/common/hooks/use-docs-url/use-docs-url.ts**

```
import { useSelector } from "metabase/lib/redux";  
import {  
  type UtmProps,  
  getDocsUrl,  
  getLearnUrl,  
} from "metabase/selectors/settings";  
import { getShowMetabaseLinks } from...
```

### **frontend/src/metabase/common/hooks/use-escape-to-close-modal/index.ts**

```
export * from "./use-escape-to-close-modal";
```

### **frontend/src/metabase/common/hooks/use-favicon.ts**

```
import { useEffect } from "react";  
  
import { useSetting } from "metabase/common/hooks";  
  
export const useFavicon = ({ favicon }: { favicon: string | null }) => {  
  const defaultFavicon =...
```

### **frontend/src/metabase/common/hooks/use-force-update.ts**

```
import { useReducer } from "react";  
  
// https://reactjs.org/docs/hooks-faq.html#is-there-something-like-forceupdate  
export function useForceUpdate() {  
  const [, forceUpdate] = useReducer((x) => x +...
```

### **frontend/src/metabase/common/hooks/use-has-token-feature/index.ts**

```
export * from "./use-has-token-feature";
```

### **frontend/src/metabase/common/hooks/use-has-token-feature/use-has-token-feature.ts**

```
import { useSelector } from "metabase/lib/redux";  
import { getTokenFeature } from "metabase/setup/selectors";  
import type { TokenFeature } from "metabase-types/api";  
  
export const useHasTokenFeature...
```

## Analyst Base

### **frontend/src/metabase/common/hooks/use-instance-locale/index.ts**

```
export * from "../use-instance-locale";
```

### **frontend/src/metabase/common/hooks/use-instance-locale/use-instance-locale.ts**

```
import { useSelector } from "metabase/lib/redux";
import { getSetting } from "metabase/selectors/settings";
```

```
export const useInstanceLocale = () => {
  return useSelector((state) => getSetting(state, ...
```

### **frontend/src/metabase/common/hooks/use-is-at-homepage-dashboard.ts**

```
import { useLocation } from "react-use";
```

```
import { useSelector } from "metabase/lib/redux";
import { getCustomHomePageDashboardId } from "metabase/selectors/app";
```

```
import { useSetting } from...
```

### **frontend/src/metabase/common/hooks/use-is-small-screen.ts**

```
import { useMedia } from "react-use";
```

```
const useIsSmallScreen = () => {
  return useMedia("(max-width: 40em)");
};
```

```
// eslint-disable-next-line import/no-default-export -- deprecated usage
export...
```

### **frontend/src/metabase/common/hooks/use-is-truncated.ts**

```
import { useEffect, useRef, useState } from "react";
import _ from "underscore";
```

```
import ResizeObserver from "metabase/lib/resize-observer";
```

```
type UseIsTruncatedProps = {
  disabled?:...
```

### **frontend/src/metabase/common/hooks/use-keyboard-shortcut.ts**

```
import { useEffect } from "react";
```

```
export function useKeyboardShortcut(
  key: string,
  callback: (e: KeyboardEvent) => void,
```

## Analyst Base

```
) {  
  useEffect(() => {  
    function keyboardListener(e: KeyboardEvent)...
```

### **frontend/src/metabase/common/hooks/use-list-keyboard-navigation/index.ts**

```
export * from "../use-list-keyboard-navigation";
```

### **frontend/src/metabase/common/hooks/use-list-keyboard-navigation/use-list-keyboard-navigation.ts**

```
import type { MutableRefObject } from "react";  
import { useCallback, useEffect, useRef, useState } from "react";
```

```
interface ListKeyboardNavigationInput<T, R> {  
  ref?: MutableRefObject<R | null>;  
  ...
```

### **frontend/src/metabase/common/hooks/use-list-keyboard-navigation/use-list-keyboard-navigation.unit.spec.ts**

```
import { renderHook } from "@testing-library/react";
```

```
import { fireEvent } from "__support__/ui";
```

```
import { useListKeyboardNavigation } from "../use-list-keyboard-navigation";
```

```
const list = [  
  { id:...
```

### **frontend/src/metabase/common/hooks/use-list-select.ts**

```
import { useCallback, useState } from "react";
```

```
export type UseListSelectReturnValue<T> = {  
  clear: () => void;  
  getIsSelected: (item: T) => boolean;  
  selected: T[];  
  selectOnlyTheseItems:...
```

### **frontend/src/metabase/common/hooks/use-list-select.unit.spec.ts**

```
import { act, renderHook } from "@testing-library/react";
```

```
import { useListSelect } from "../use-list-select";
```

```
interface objectType {  
  id: number;  
  name: string;
```

## Analyst Base

```
}
```

```
const OBJECT_LIST = [  
  { id: 1,...
```

### **frontend/src/metabase/common/hooks/use-loading-timer.ts**

```
import { useEffect } from "react";
```

```
interface LoadingTimerProps {  
  timer: number;  
  onTimeout: () => void;  
}
```

```
export function useLoadingTimer(isLoading: boolean, props: LoadingTimerProps) {  
  const...
```

### **frontend/src/metabase/common/hooks/use-locale/index.ts**

```
export * from "./use-locale";
```

### **frontend/src/metabase/common/hooks/use-locale/use-locale.ts**

```
import { useContext } from "react";
```

```
import { getCurrentUser } from "metabase/admin/datamodel/selectors";  
import { useInstanceLocale } from "metabase/common/hooks/use-instance-locale";  
import {...
```

### **frontend/src/metabase/common/hooks/use-memoized-callback.ts**

```
import { useMemo } from "react";
```

```
type MapLike<K, V> = Map<K, V> | WeakMap<object & K, V>;
```

```
function getWithFallback<K, V>(  
  map: MapLike<K, V>,  
  key: K,  
  fallback: () => V,  
)> V {  
  if ("has" in...
```

### **frontend/src/metabase/common/hooks/use-memoized-callback.unit.spec.ts**

```
import { memoize } from "./use-memoized-callback";
```

```
describe("memoize", () => {  
  it("should return same result for same arguments", () => {  
    const add = jest.fn((a: number, b: number) => a + b);
```

...

### **frontend/src/metabase/common/hooks/use-modal-open.ts**

```
import { useEffect, useState } from "react";

// this is a custom hook that is used to open a modal after a delay
// so that the modal can be animated in
export function useModalOpen() {
  const...
```

### **frontend/src/metabase/common/hooks/use-most-recent-call/use-most-recent-call.ts**

```
import { useCallback, useRef } from "react";

type AsyncFn = (...args: any[]) => Promise<any>;

type AsyncReturnType<T extends (...args: any) => Promise<any>> = T extends (
  ...args: any
) =>...
```

### **frontend/src/metabase/common/hooks/use-notification-channels/index.ts**

```
export * from "./use-notification-channels";
```

### **frontend/src/metabase/common/hooks/use-notification-channels/use-notification-channels.ts**

```
import { useGetChannelInfoQuery } from "metabase/api";

export const useHasAnyNotificationChannel = (): boolean => {
  const { data: channelInfo } = useGetChannelInfoQuery();

  return...
```

### **frontend/src/metabase/common/hooks/use-number-formatter.ts**

```
import { useCallback } from "react";

import { useSetting } from "metabase/common/hooks";
import {
  type FormatNumberOptions,
  formatNumber,
} from "metabase/lib/formatting/numbers";

export type...
```

### **frontend/src/metabase/common/hooks/use-number-formatter.unit.spec.ts**

## Analyst Base

```
import { mockSettings } from "__support__/settings";
import { renderHookWithProviders } from "__support__/ui";
import type { NumberFormattingSettings } from "metabase-types/api";
```

```
import {
  type...
```

### **frontend/src/metabase/common/hooks/use-on-click-outside.ts**

```
import type { RefObject } from "react";
import { useEffect } from "react";
```

```
interface ValidRefTarget {
  contains(target: EventTarget | null): boolean;
}
```

```
export function useOnClickOutside<T extends...
```

### **frontend/src/metabase/common/hooks/use-pagination.ts**

```
import { useCallback, useState } from "react";
```

```
export const usePagination = (initialPage = 0) => {
  const [page, setPage] = useState(initialPage);
```

```
  const handleNextPage = useCallback(
    () =>...
```

### **frontend/src/metabase/common/hooks/use-pagination.unit.spec.ts**

```
import { act, renderHook } from "@testing-library/react";
```

```
import { usePagination } from "../use-pagination";
```

```
describe("usePagination", () => {
  it("should set 'page' to 'initialPage' upon calling...
```

### **frontend/src/metabase/common/hooks/use-palette.ts**

```
import { useMemo } from "react";
```

```
import type { ColorPalette } from "metabase/lib/colors/types";
import { useMantineTheme } from "metabase/ui";
```

```
/**
```

```
 * Extracts a color palette from a subset of...
```

### **frontend/src/metabase/common/hooks/use-safe-async-function.ts**

## Analyst Base

```
import { useCallback } from "react";
import { useMountedState } from "react-use";

type AsyncFn = (...args: any[]) => Promise<any>;
```

```
/**
 * wraps the given async function in a promise that does not...
```

### **frontend/src/metabase/common/hooks/use-scroll-on-mount.js**

```
import { useEffect, useRef } from "react";

export const useScrollOnMount = () => {
  const ref = useRef(null);
```

```
  useEffect(() => {
    if (ref.current) {
      ref.current.scrollToView?({ block:...
```

### **frontend/src/metabase/common/hooks/use-sequenced-content-close-handler.ts**

```
import { useCallback, useRef } from "react";
import _ from "underscore";

import { isElement } from "metabase-types/guards";
```

```
type PopoverData = {
  contentEl: Element;
  backdropEl?: Element;
  ...
```

### **frontend/src/metabase/common/hooks/use-setting/index.ts**

```
export * from "./use-setting";
```

### **frontend/src/metabase/common/hooks/use-setting/use-setting.ts**

```
import { useCallback, useMemo } from "react";
import _ from "underscore";
```

```
import { useDispatch, useSelector } from "metabase/lib/redux";
import { updateUserSetting } from...
```

### **frontend/src/metabase/common/hooks/use-store-url/use-store-url.ts**

```
import { useSelector } from "metabase/lib/redux";
import { type StorePaths, getStoreUrl } from "metabase/selectors/settings";
```

```
export function useStoreUrl(storePath: StorePaths = "") {
```

## Analyst Base

return...

### **frontend/src/metabase/common/hooks/use-temp-storage/index.ts**

```
export * from "./use-temp-storage";
```

### **frontend/src/metabase/common/hooks/use-temp-storage/use-temp-storage.ts**

```
import { useCallback } from "react";
```

```
import { useDispatch, useSelector } from "metabase/lib/redux";
```

```
import { setTempSetting } from "metabase/redux/app";
```

```
import type {
```

```
  State,
```

```
  TempStorageKey,
```

```
  ...
```

### **frontend/src/metabase/common/hooks/use-temporary-state.ts**

```
import { useCallback, useRef, useState } from "react";
```

```
import { useUnmount } from "react-use";
```

```
/**
```

```
 * A hook that temporarily changes the value of a stateful value
```

```
 * and automatically resets it to...
```

### **frontend/src/metabase/common/hooks/use-toast/index.ts**

```
export * from "./use-toast";
```

### **frontend/src/metabase/common/hooks/use-toast/use-toast.ts**

```
import { useCallback } from "react";
```

```
import { useDispatch } from "metabase/lib/redux";
```

```
import { addUndo, dismissUndo } from "metabase/redux/undo";
```

```
import type { Undo } from...
```

### **frontend/src/metabase/common/hooks/use-toggle.ts**

```
import { useCallback, useState } from "react";
```

```
type ToggleHookResult = [
```

```
  boolean,
```

```
  {
```

```
    turnOn: () => void;
```

```
    turnOff: () => void;
```

```
    toggle: () => void;
```

```
  },
```

```
];
```



```
/**  
 * @deprecated use...
```

### **frontend/src/metabase/common/hooks/use-unique-id.ts**

```
import { useRef } from "react";  
import _ from "underscore";  
  
export const useUniqueid = (prefix?: string): string => {  
  const idRef = useRef("");  
  if (!idRef.current) {  
    idRef.current = ...
```

### **frontend/src/metabase/common/hooks/use-unmount-layout.ts**

```
import { useEffect, useRef } from "react";  
  
// identical to useUnmount from react-use but leverages useEffect  
// instead of useLayoutEffect in the case you need access to the DOM elements  
//...
```

### **frontend/src/metabase/common/hooks/use-url-state/index.ts**

```
export * from "./types";  
export * from "./use-url-state";  
export * from "./utils";
```

### **frontend/src/metabase/common/hooks/use-url-state/types.ts**

```
import type { Query } from "history";  
  
export type QueryParam = Query[keyof Query];
```

### **frontend/src/metabase/common/hooks/use-url-state/use-url-state.ts**

```
import type { Location, Query } from "history";  
import { useCallback, useEffect, useState } from "react";  
import { push, replace } from "react-router-redux";  
import { useEffectOnce, useLatest } from ...
```

### **frontend/src/metabase/common/hooks/use-url-state/use-url-state.unit.spec.ts**

```
import type { Location } from "history";  
  
import { act, renderHookWithProviders, waitFor } from "__support__/ui";  
import { createMockLocation } from "metabase-types/store/mocks";  
  
import type { ...
```

### **frontend/src/metabase/common/hooks/use-url-state/utils.ts**

```
import type { QueryParam } from "./types";
```

## Analyst Base

```
export function getFirstParamValue(param: QueryParam) {  
  return Array.isArray(param) ? param[0] : param;  
}
```

### **frontend/src/metabase/common/hooks/use-url-with-utm/index.ts**

```
export { useUrlWithUtm } from "./use-url-with-utm";
```

### **frontend/src/metabase/common/hooks/use-url-with-utm/use-url-with-utm.ts**

```
import { useSelector } from "metabase/lib/redux";  
import { type UtmProps, getUrlWithUtm } from "metabase/selectors/settings";
```

```
export function useUrlWithUtm(url: string, utm: UtmProps):...
```

### **frontend/src/metabase/common/hooks/use-user-acknowledgement.ts**

```
import { useCallback } from "react";
```

```
import { useUserKeyValue } from "./use-user-key-value";
```

```
type UseUserAcknowledgementResult = [  
  acknowledged: boolean,  
  { ack: () => void; unack: () => void;...  
]
```

### **frontend/src/metabase/common/hooks/use-user-key-value.ts**

```
import { useCallback } from "react";
```

```
import {  
  skipToken,  
  useDeleteUserKeyValueMutation,  
  useGetUserKeyValueQuery,  
  useUpdateKeyValueMutation,  
} from "metabase/api";  
import { useSelector } from...
```

### **frontend/src/metabase/common/hooks/use-web-notification.ts**

```
import { useCallback } from "react";
```

```
const hasNotificationAPI = "Notification" in window;
```

```
export function useWebNotification() {  
  const requestPermission = useCallback(async () => {  
    if...
```

### **frontend/src/metabase/common/style/input.ts**

```
eslint-disable-next-line no-restricted-imports
```

## Analyst Base

### **frontend/src/metabase/common/style/types.ts**

```
export type InputSize = "small" | "medium" | "large";
```

### **frontend/src/metabase/common/types/export.ts**

```
export type TableExportFormat = "csv" | "xlsx" | "json";  
export type ExportFormat = TableExportFormat | "png";
```

### **frontend/src/metabase/common/types.ts**

```
export type PartialBy<T, K extends keyof T> = Omit<T, K> & Partial<Pick<T, K>>;
```

### **frontend/src/metabase/common/utils/column-groups.ts**

```
import type { IconName } from "metabase/ui";  
import type { ColumnGroupDisplayInfo } from "metabase-lib";
```

```
export function getColumnGroupIcon(  
  groupInfo: ColumnGroupDisplayInfo,  
) : IconName {  
  if...
```

### **frontend/src/metabase/common/utils/column-groups.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";  
import { checkNotNull } from "metabase/lib/types";  
import * as Lib from "metabase-lib";  
import { SAMPLE_METADATA } from...
```

### **frontend/src/metabase/common/utils/columns.ts**

```
import type { IconName } from "metabase/ui";  
import * as Lib from "metabase-lib";
```

```
export function getColumnIcon(  
  column: Lib.ColumnMetadata | Lib.ColumnTypeInfo,  
) : IconName {  
  if...
```

### **frontend/src/metabase/common/utils/columns.unit.spec.ts**

```
import * as Lib from "metabase-lib";  
import { columnFinder, createQuery } from "metabase-lib/test-helpers";
```

```
import { getColumnIcon } from "../columns";
```

```
describe("common/utils/columns", () => {  
  ...
```

### **frontend/src/metabase/common/utils/doNotForwardProps.ts**

## Analyst Base

```
export const doNotForwardProps = (...propNamesToBlock: string[]) => ({
  shouldForwardProp: (propName: string) => !propNamesToBlock.includes(propName),
});
```

### frontend/src/metabase/common/utils/keyboard.ts

\* Returns true if e.key is the given key and no modifier keys (ctrl, meta, alt, shift) were pressed

### frontend/src/metabase/common/utils/model-names.ts

```
import { t } from "ttag";
```

```
const TRANSLATED_NAME_BY_MODEL_TYPE: Record<string, string> = {
  get action() {
    return t`Action`;
  },
  get card() {
    return t`Question`;
  },
  get collection()...
```

### frontend/src/metabase/common/utils/model-names.unit.spec.ts

```
import { getTranslatedEntityName } from "../model-names";
```

```
describe("common/utils/model-names", () => {
  it("returns null if the model doesn't exist", () => {
    ...
```

### frontend/src/metabase/common/utils/plan.ts

```
import type { TokenFeatures } from "metabase-types/api";
import { tokenFeatures } from "metabase-types/api";
```

```
export type Plan =
  | "oss"
  | "starter"
  | "starter-with-dwh"
  | "pro-cloud"
  | ...
```

### frontend/src/metabase/common/utils/plan.unit.spec.ts

```
import { getPlan } from "metabase/common/utils/plan";
import { createMockTokenFeatures } from "metabase-types/api/mocks";
```

```
describe("common/utils/columns", () => {
  describe("getPlan", () => {
    ...
```

## Analyst Base

### **frontend/src/metabase/css/core/base.styled.ts**

eslint-disable-next-line no-restricted-imports

### **frontend/src/metabase/css/core/fonts.styled.ts**

eslint-disable-next-line no-restricted-imports

### **frontend/src/metabase/css/core/overlays/constants.ts**

export const OVERLAY\_Z\_INDEX = 200;

/\*\* This constant is used to work around a bug in Mantine: when certain Mantine  
\* overlays appear above a Mantine Modal, only the Modal's portal has  
\* ...

### **frontend/src/metabase/dashboard/actions/actions.ts**

```
import {  
  getActionErrorMessage,  
  getActionExecutionMessage,  
} from "metabase/actions/utils";  
import { SIDEBAR_NAME } from "metabase/dashboard/constants";  
import { addUndo } from...
```

### **frontend/src/metabase/dashboard/actions/actions.unit.spec.js**

```
import { DashboardApi } from "metabase/services";  
import { createMockDashboard } from "metabase-types/api/mocks";  
import {  
  createMockDashboardState,  
  createMockRoutingState,  
  createMockState,  
}...
```

### **frontend/src/metabase/dashboard/actions/auto-wire-parameters/actions.ts**

```
import {  
  closeAutoWireParameterToast,  
  showAddedCardAutoWireParametersToast,  
  showAutoWireParametersToast,  
} from "metabase/dashboard/actions/auto-wire-parameters/toasts";  
import {  
  ...
```

### **frontend/src/metabase/dashboard/actions/auto-wire-parameters/constants.ts**

```
export const AUTO_WIRE_TOAST_TIMEOUT = 12000;  
export const AUTO_WIRE_UNDO_TOAST_TIMEOUT = 8000;
```

### **frontend/src/metabase/dashboard/actions/auto-wire-parameters/toasts.ts**

## Analyst Base

```
import { t } from "ttag";
import _ from "underscore";
```

```
import type { SetMultipleDashCardAttributesOpts } from "metabase/dashboard/actions";
import {
  setDashCardAttributes,
  ...
```

### **frontend/src/metabase/dashboard/actions/auto-wire-parameters/utils.ts**

```
import _ from "underscore";

import { isActionDashCard } from "metabase/actions/utils";
import { getExistingDashCards } from "metabase/dashboard/actions/utils";
import {
  ...
```

### **frontend/src/metabase/dashboard/actions/cards-typed.ts**

```
import { createAction } from "@reduxjs/toolkit";
import { t } from "ttag";
import _ from "underscore";

import Questions from "metabase/entities/questions";
import {
  DEFAULT_CARD_SIZE,
  ...
```

### **frontend/src/metabase/dashboard/actions/cards.js**

```
import { t } from "ttag";

import { createCard } from "metabase/lib/card";

import { trackCardCreated } from "../analytics";

import { addDashCardToDashboard } from "./cards-typed";

export const...
```

### **frontend/src/metabase/dashboard/actions/cards.unit.spec.ts**

```
import type { Store } from "@reduxjs/toolkit";
import _ from "underscore";

import { getStore } from "__support__/entities-store";
import {
  setupCardQueryEndpoints,
```

...

### **frontend/src/metabase/dashboard/actions/core.ts**

```
import { createAction } from "@reduxjs/toolkit";
import { push } from "react-router-redux";

import { getLocation } from "metabase/selectors/routing";
import type {
  DashCardId,
  ...
```

### **frontend/src/metabase/dashboard/actions/data-fetching.ts**

```
import { createAction } from "@reduxjs/toolkit";
import { getIn } from "icepick";
import { denormalize, normalize, schema } from "normalizr";
import { match } from "ts-pattern";
import { t } from...
```

### **frontend/src/metabase/dashboard/actions/data-fetching.unit.spec.js**

```
import { getStore } from "__support__/entities-store";
import {
  setupDashboardQueryMetadataEndpoint,
  setupDashboardsEndpoints,
  setupDatabaseEndpoints,
} from "__support__/server-mocks";
import...
```

### **frontend/src/metabase/dashboard/actions/getNewCardUrl.ts**

```
import _ from "underscore";

import { getCardAfterVisualizationClick } from "metabase/visualizations/lib/utils";
import * as Lib from "metabase-lib";
import Question from...
```

### **frontend/src/metabase/dashboard/actions/getNewCardUrl.unit.spec.ts**

```
import {
  createMockCard,
  createMockDashboard,
  createMockDashboardCard,
  createMockParameter,
} from "metabase-types/api/mocks";
```

```
import { getParametersMappedToCard } from...
```

### **frontend/src/metabase/dashboard/actions/index.ts**

## Analyst Base

```
export * from "./cards-typed";
export * from "./cards";
export * from "./core";
export * from "./data-fetching";
export * from "./navigation";
export * from "./parameters";
export * from...
```

### **frontend/src/metabase/dashboard/actions/navigation.js**

```
import { createThunkAction } from "metabase/lib/redux";
import * as Urls from "metabase/lib/urls";
import { openUrl } from "metabase/redux/app";
import { getMetadata } from...
```

### **frontend/src/metabase/dashboard/actions/parameters.unit.spec.ts**

```
import { getStore } from "__support__/entities-store";
import { getParameters } from "metabase/dashboard/selectors";
import { mainReducers } from "metabase/reducers-main";
import {
  createMockCard,
  ...
```

### **frontend/src/metabase/dashboard/actions/revisions.js**

```
import {
  fetchDashboard,
  fetchDashboardCardData,
} from "metabase/dashboard/actions";
import Revisions from "metabase/entities/revisions";
import { createThunkAction } from...
```

### **frontend/src/metabase/dashboard/actions/save.js**

```
import { assocIn, dissocIn, getIn } from "icepick";
import _ from "underscore";
```

```
import {
  fetchDashboard,
  fetchDashboardCardData,
} from "metabase/dashboard/actions";
import Dashboards from...
```

### **frontend/src/metabase/dashboard/actions/sharing.ts**

```
import { SIDEBAR_NAME } from "metabase/dashboard/constants";
import type { Dispatch } from "metabase-types/store";
```



## Analyst Base

```
import { closeSidebar, setSidebar } from "../ui";
```

```
export const setSharing =...
```

### **frontend/src/metabase/dashboard/actions/tabs.ts**

```
import { arrayMove } from "@dnd-kit/sortable";  
import type { Draft } from "@reduxjs/toolkit";  
import { createAction, createReducer } from "@reduxjs/toolkit";  
import { t } from "ttag";
```

```
import {
```

```
...
```

### **frontend/src/metabase/dashboard/actions/tabs.unit.spec.ts**

```
import { TEST_DASHBOARD_STATE } from "../../components/DashboardTabs/test-utils";
```

```
import { getIdFromSlug, moveTab, tabsReducer } from "../tabs";
```

```
/**
```

```
 * It's preferred to write tests in...
```

### **frontend/src/metabase/dashboard/actions/theme.ts**

```
import { createAction } from "@reduxjs/toolkit";
```

```
import type { DisplayTheme } from "metabase/public/lib/types";
```

```
export const SET_DISPLAY_THEME = "metabase/dashboard/SET_DISPLAY_THEME";
```

```
export const...
```

### **frontend/src/metabase/dashboard/actions/trash.ts**

```
import { t } from "ttag";
```

```
import _ from "underscore";
```

```
import { getTrashUndoMessage } from "metabase/archive/utils";
```

```
import { canonicalCollectionId } from "metabase/collections/utils";
```

```
import {...
```

### **frontend/src/metabase/dashboard/actions/ui.ts**

```
import { SIDEBAR_NAME } from "metabase/dashboard/constants";
```

```
import { createAction, createThunkAction } from "metabase/lib/redux";
```

```
import type { DashCardId } from "metabase-types/api";
```

```
import type {
```

```
...
```

### **frontend/src/metabase/dashboard/actions/utils.ts**

## Analyst Base

```
import { t } from "ttag";
import _ from "underscore";
```

```
import { getIframeDomainName } from "metabase/visualizations/visualizations/IFrameViz/utils";
import type {
  DashCardId,
  Dashboard,
  ...
```

### **frontend/src/metabase/dashboard/actions/utils.unit.spec.ts**

```
import {
  createMockActionDashboardCard,
  createMockDashboard,
  createMockDashboardCard,
  createMockHeadingDashboardCard,
  createMockLinkDashboardCard,
  createMockTextDashboardCard,
} from...
```

### **frontend/src/metabase/dashboard/analytics.ts**

```
import { trackSchemaEvent, trackSimpleEvent } from "metabase/lib/analytics";
import type {
  DashboardId,
  DashboardWidth,
  VisualizationDisplay,
} from "metabase-types/api";
```

```
import type {...
```

### **frontend/src/metabase/dashboard/components/ActionSidebar/index.ts**

```
export * from "./ActionSidebar";
```

### **frontend/src/metabase/dashboard/components/AddCardSidebar/index.ts**

```
export * from "./AddCardSidebar";
```

### **frontend/src/metabase/dashboard/components/AddFilterParameterMenu/index.ts**

```
export * from "./AddFilterParameterMenu";
```

### **frontend/src/metabase/dashboard/components/ClickBehaviorSidebar/SidebarItem/index.ts**

```
export * from "./SidebarItem";
```

### **frontend/src/metabase/dashboard/components/ClickBehaviorSidebar/hooks.ts**

```
import { t } from "ttag";
```

## Analyst Base

```
import { hasActionsMenu } from "metabase/lib/click-behavior";
import { useSelector } from "metabase/lib/redux";
import { getApplicationName } from...
```

### **frontend/src/metabase/dashboard/components/ClickBehaviorSidebar/utils.ts**

```
import type { IconName } from "metabase/ui";
import { getColumnSettings } from "metabase-lib/v1/queries/utils/column-key";
import type {
  ClickBehaviorType,
  DashboardCard,
  DatasetColumn,
} from...
```

### **frontend/src/metabase/dashboard/components/CollapsibleDashboardParameterList/index.ts**

```
export * from "../CollapsibleDashboardParameterList";
```

### **frontend/src/metabase/dashboard/components/DashCard/DashCardActionsPanel/DashCardActionButton/index.ts**

```
export * from "../DashCardActionButton";
```

### **frontend/src/metabase/dashboard/components/DashCard/DashCardMenu/dash-card-menu.ts**

```
import type { MantineColor } from "@mantine/core/lib/core";
import type { ReactNode } from "react";
```

```
import type { DashboardContextProps } from "metabase/dashboard/context";
import type { IconName } from...
```

### **frontend/src/metabase/dashboard/components/DashCard/DashCardMenu/utils.ts**

```
import { PLUGIN_FEATURE_LEVEL_PERMISSIONS } from "metabase/plugins";
import * as Lib from "metabase-lib";
import type Question from "metabase-lib/v1/Question";
import type { Dataset } from...
```

### **frontend/src/metabase/dashboard/components/DashCard/DashCardParameterMapper/DisabledNativeCardHelpText/index.ts**

```
export { DisabledNativeCardHelpText } from "../DisabledNativeCardHelpText";
```

### **frontend/src/metabase/dashboard/components/DashCard/DashCardParameterMapper/DisabledNativeCardHelpText/tests/common.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

## Analyst Base

```
import { createMockParameter } from "metabase-types/api/mocks";
```

```
import { setup } from "../setup";
```

```
describe("DashCardParameterMapper >...
```

**frontend/src/metabase/dashboard/components/DashCard/DashCardParameterMapper/DisabledNativeCardHelpText/tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import type { SetupOpts } from "../setup";
```

```
import { setup as baseSetup } from "../setup";
```

```
function setup(opts: SetupOpts) {  
  baseSetup({...
```

**frontend/src/metabase/dashboard/components/DashCard/DashCardParameterMapper/DisabledNativeCardHelpText/tests/premium.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import type { SetupOpts } from "../setup";
```

```
import { setup as baseSetup } from "../setup";
```

```
function setup(opts: SetupOpts) {  
  baseSetup({  
    ...
```

**frontend/src/metabase/dashboard/components/DashCard/dashcard-ids.ts**

```
import { isJWT } from "metabase/lib/utils";
```

```
import { isUuid } from "metabase/lib/uuid";
```

```
import type { DashboardCard } from "metabase-types/api";
```

```
export const getDashcardTokenId = (dashcard:...
```

**frontend/src/metabase/dashboard/components/DashCard/types.ts**

```
import type { Card, DashboardCard } from "metabase-types/api";
```

```
export type CardSlownessStatus = "usually-fast" | "usually-slow" | boolean;
```

```
export type NavigateToNewCardFromDashboardOpts = {  
  ...
```

**frontend/src/metabase/dashboard/components/DashCard/utils.ts**

```
import { getVirtualCardType } from "metabase/dashboard/utils";
```

## Analyst Base

```
import type {
  BaseDashboardCard,
  Series,
  VisualizerColumnValueSource,
  VisualizerDataSourceId,
  VisualizerVizDefinition,
} from...
```

### **frontend/src/metabase/dashboard/components/DashCard/utils.unit.spec.ts**

```
import type { VisualizerVizDefinition } from "metabase-types/api";
import {
  createMockCard,
  createMockColumn,
  createMockDataset,
  createMockDatasetData,
  createMockSeries,
  ...
```

### **frontend/src/metabase/dashboard/components/Dashboard/components/index.ts**

```
export { Grid } from "./Grid";
export { ParametersList } from "./ParametersList";
```

### **frontend/src/metabase/dashboard/components/Dashboard/use-set-dashboard-attributes.ts**

```
import { useCallback } from "react";

import { setDashboardAttributes } from "metabase/dashboard/actions";
import { getDashboardComplete } from "metabase/dashboard/selectors";
import { useDispatch, ...
```

### **frontend/src/metabase/dashboard/components/DashboardHeader/DashboardHeaderButtonRow/constants.ts**

```
import { DASHBOARD_ACTION } from "./dashboard-action-keys";
```

```
// Buttons visible in public dashboard
export const DASHBOARD_DISPLAY_ACTIONS = [
  DASHBOARD_ACTION.REFRESH_WIDGET,
  ...
```

### **frontend/src/metabase/dashboard/components/DashboardHeader/DashboardHeaderButtonRow/types.ts**

```
import type { ComponentType } from "react";
```

## Analyst Base

```
import type { DashboardContextReturned } from "metabase/dashboard/context";
import type { Collection, Dashboard } from "metabase-types/api";
```

```
import type...
```

**frontend/src/metabase/dashboard/components/DashboardHeader/buttons/AddFilterParameterButton/index.ts**

```
export * from "../AddFilterParameterButton";
```

**frontend/src/metabase/dashboard/components/DashboardHeader/buttons/AddSectionButton/index.ts**

```
export * from "../AddSectionButton";
```

**frontend/src/metabase/dashboard/components/DashboardHeader/index.ts**

```
export * from "../DashboardHeader";
```

**frontend/src/metabase/dashboard/components/DashboardHeader/tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";
import {
  createMockCollection,
  createMockDashboard,
} from "metabase-types/api/mocks";
```

```
import { setup } from "../setup";
```

```
const setupEnterprise = (opts:...
```

**frontend/src/metabase/dashboard/components/DashboardInfoSidebar/DashboardEntityIdCard/index.ts**

```
export { DashboardEntityIdCard } from "../DashboardEntityIdCard";
```

**frontend/src/metabase/dashboard/components/DashboardInfoSidebar/index.ts**

```
export * from "../DashboardInfoSidebar";
```

**frontend/src/metabase/dashboard/components/DashboardInfoSidebar/tests/common.unit.spec.ts**

```
import userEvent from "@testing-library/user-event";
```

```
import { screen } from "__support__/ui";
import type { Dashboard } from "metabase-types/api";
import {
  createMockCollection,
  ...
```

## **frontend/src/metabase/dashboard/components/DashboardInfoSidebar/tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";
import type { Dashboard } from "metabase-types/api";
import { createMockDashboard } from "metabase-types/api/mocks";
```

```
import type { SetupOpts } from...
```

## **frontend/src/metabase/dashboard/components/DashboardInfoSidebar/tests/premium.unit.spec.ts**

```
import { screen } from "__support__/ui";
import type { Dashboard } from "metabase-types/api";
import {
  createMockDashboard,
  createMockSettings,
  createMockTokenFeatures,
} from...
```

## **frontend/src/metabase/dashboard/components/DashboardLeaveConfirmationModal/utils.ts**

```
import type { Location } from "history";

import { deserializeCard, parseHash } from "metabase/query_builder/actions";

export const isNavigatingToCreateADashboardQuestion = (
  nextLocation?:...
)
```

## **frontend/src/metabase/dashboard/components/DashboardParameterList/index.ts**

```
export * from "./DashboardParameterList";
```

## **frontend/src/metabase/dashboard/components/DashboardParameterPanel/index.ts**

```
export * from "./DashboardParameterPanel";
```

## **frontend/src/metabase/dashboard/components/DashboardSettingsSidebar/index.ts**

```
export * from "./DashboardSettingsSidebar";
```

## **frontend/src/metabase/dashboard/components/DashboardSettingsSidebar/tests/common.unit.spec.ts**

```
import userEvent from "@testing-library/user-event";
```

## Analyst Base

```
import { screen } from "__support__/ui";
```

```
import { setup } from "../setup";
```

```
jest.mock("metabase/dashboard/constants", () => ({
```

```
...
```

### **frontend/src/metabase/dashboard/components/DashboardSettingsSidebar/tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import { setupEnterprise } from "../setup";
```

```
describe("DashboardSettingsSidebar > enterprise", () => {
```

```
  it("should render the component", async () => {
```

```
    ...
```

### **frontend/src/metabase/dashboard/components/DashboardSettingsSidebar/tests/premium.unit.spec.ts**

```
import userEvent from "@testing-library/user-event";
```

```
import { screen } from "__support__/ui";
```

```
import {  
  createMockCollection,  
  createMockDashboard,  
} from "metabase-types/api/mocks";
```

```
import {...
```

### **frontend/src/metabase/dashboard/components/DashboardTabs/index.ts**

```
export * from "../DashboardTabs";
```

```
export * from "../use-dashboard-tabs";
```

### **frontend/src/metabase/dashboard/components/DashboardTabs/test-utils.ts**

```
import { getDefaultTab } from "metabase/dashboard/actions";
```

```
import { INITIAL_DASHBOARD_STATE } from "metabase/dashboard/constants";
```

```
import {  
  createMockCard,  
  createMockDashboardCard,  
} from...
```

### **frontend/src/metabase/dashboard/components/DashboardTabs/use-dashboard-tabs.ts**

```
import type { UniqueIdentifier } from "@dnd-kit/core";
```



## Analyst Base

```
import { t } from "ttag";
```

```
import { trackTabDuplicated } from "metabase/dashboard/analytics";
```

```
import { useDashboardContext } from...
```

### **frontend/src/metabase/dashboard/components/DashboardTitle/index.ts**

```
export * from "./DashboardTitle";
```

### **frontend/src/metabase/dashboard/components/ExtraEditButtonsMenu/index.ts**

```
export * from "./ExtraEditButtonsMenu";
```

### **frontend/src/metabase/dashboard/components/QuestionPicker/actions.ts**

```
import { push } from "react-router-redux";
```

```
import { getDashboard } from "metabase/dashboard/selectors";
```

```
import { createThunkAction } from "metabase/lib/redux";
```

```
import * as Urls from...
```

### **frontend/src/metabase/dashboard/components/QuestionPicker/index.ts**

```
export * from "./QuestionPicker";
```

### **frontend/src/metabase/dashboard/components/RefreshWidget/RefreshOption/index.ts**

```
export * from "./RefreshOption";
```

### **frontend/src/metabase/dashboard/components/RefreshWidget/index.ts**

```
export * from "./RefreshWidget";
```

### **frontend/src/metabase/dashboard/components/grid/utils.ts**

```
import _ from "underscore";
```

```
import { getMobileHeight } from "metabase/visualizations/shared/utils/sizes";
```

```
import type { BaseDashboardCard } from "metabase-types/api";
```

```
function...
```

### **frontend/src/metabase/dashboard/components/grid/utils.unit.spec.ts**

```
import { MOBILE_DEFAULT_CARD_HEIGHT } from "metabase/visualizations/shared/utils/sizes";
```

```
import type { VisualizationDisplay } from "metabase-types/api";
```

```
import {
```

```
  createMockCard,
```

```
  ...
```

### **frontend/src/metabase/dashboard/constants.ts**

```
import type {
```

## Analyst Base

```
DashboardSidebarName,  
DashboardState,  
} from "metabase-types/store";
```

```
import type { EmbedDisplayParams } from "../types";
```

```
export const DASHBOARD_DESCRIPTION_MAX_LENGTH =...
```

### **frontend/src/metabase/dashboard/containers/AutomaticDashboardApp/index.ts**

```
export * from "../AutomaticDashboardApp";
```

### **frontend/src/metabase/dashboard/containers/DashboardApp/use-dashboard-location-sync.ts**

```
import type { WithRouterProps } from "react-router";
```

```
import { useDashboardContext } from "metabase/dashboard/context";
```

```
import { useLocationSync } from "metabase/dashboard/hooks";
```

```
import type {...
```

### **frontend/src/metabase/dashboard/containers/DashboardApp/use-slow-card-notification.ts**

```
import { useCallback, useEffect } from "react";
```

```
import { t } from "ttag";
```

```
import { useLoadingTimer } from "metabase/common/hooks/use-loading-timer";
```

```
import { useUniqueld } from...
```

### **frontend/src/metabase/dashboard/containers/DashboardSharingEmbeddingModal/index.ts**

```
export * from "../DashboardSharingEmbeddingModal";
```

### **frontend/src/metabase/dashboard/context/context.redux.ts**

```
import type { ConnectedProps } from "react-redux";
```

```
import { push } from "react-router-redux";
```

```
import { deletePermanently } from "metabase/archive/actions";
```

```
import {
```

```
  addCardToDashboard,
```

```
  ...
```

### **frontend/src/metabase/dashboard/context/index.ts**

```
export * from "../context";
```

### **frontend/src/metabase/dashboard/grid-utils.ts**

## Analyst Base

```
import _ from "underscore";

import { DEFAULT_CARD_SIZE } from "metabase/lib/dashboard_grid";
import { getVisualizationRaw } from "metabase/visualizations";
import type {
  BaseDashboardCard,
  ...
```

### **frontend/src/metabase/dashboard/hooks/index.ts**

```
export * from "./use-click-behavior-data";
export * from "./use-dashboard-fullscreen";
export * from "./use-dashboard-refresh-period";
export * from "./use-dashboard-theme";
export * from...
```

### **frontend/src/metabase/dashboard/hooks/use-auto-scroll-to-dashcard.ts**

```
import type { LocationDescriptorObject } from "history";
import { useCallback, useMemo } from "react";
import { replace } from "react-router-redux";
```

```
import { parseHashOptions, stringifyHashOptions }...
```

### **frontend/src/metabase/dashboard/hooks/use-click-behavior-data.js**

```
import { useMemo } from "react";
import _ from "underscore";
```

```
import {
  getDashCardById,
  getDashboardComplete,
  getParameterValuesBySlugMap,
  getParameters,
} from...
```

### **frontend/src/metabase/dashboard/hooks/use-dashboard-fullscreen.ts**

```
import { useFullscreen } from "@mantine/hooks";
import { useCallback, useEffect, useState } from "react";
```

```
import type { DashboardFullscreenControls } from "../types";
```

```
export const...
```

### **frontend/src/metabase/dashboard/hooks/use-dashboard-refresh-period.ts**

```
import { useCallback, useRef, useState } from "react";
import { useUnmount } from "react-use";
```

## Analyst Base

```
import type { DashboardRefreshPeriodControls } from "../types";
```

```
import { useInterval } from...
```

### **frontend/src/metabase/dashboard/hooks/use-dashboard-theme.ts**

```
import { useCallback, useEffect } from "react";
```

```
import { setDisplayTheme } from "metabase/dashboard/actions";
```

```
import { getDisplayTheme } from "metabase/dashboard/selectors";
```

```
import { useDispatch,...
```

### **frontend/src/metabase/dashboard/hooks/use-dashboard-title.ts**

```
import { useCallback } from "react";
```

```
import { useSetDashboardAttributeHandler } from "../components/Dashboard/use-set-dashboard-attribute";
```

```
import { useDashboardContext } from "../context";
```

```
export...
```

### **frontend/src/metabase/dashboard/hooks/use-dashboard-url-query.ts**

```
import type { Location } from "history";
```

```
import { useEffect, useMemo } from "react";
```

```
import type { InjectedRouter } from "react-router";
```

```
import { push, replace } from "react-router-redux";
```

```
import {...
```

### **frontend/src/metabase/dashboard/hooks/use-interval.ts**

```
import { useCallback, useEffect, useRef, useState } from "react";
```

```
/**
```

```
 * it's a copy of mantine's useInterval, but with memoization of the callbacks
```

```
 */
```

```
export function useInterval(fn: () => void,...
```

### **frontend/src/metabase/dashboard/hooks/use-is-parameter-panel-sticky.ts**

```
import { type RefObject, useEffect, useState } from "react";
```

```
export function useIsParameterPanelSticky({
```

```
  parameterPanelRef,
```

```
  disabled = false,
```

```
}): {
```

```
  parameterPanelRef: RefObject<HTMLElement>;
```

```
...
```

## Analyst Base

### **frontend/src/metabase/dashboard/hooks/use-location-sync.ts**

```
import type { Location } from "history";
import { useEffect, useMemo } from "react";
import { replace } from "react-router-redux";
import { usePrevious } from "react-use";
import { omit } from...
```

### **frontend/src/metabase/dashboard/hooks/use-refresh-dashboard.ts**

```
import type { Query } from "history";
import { useCallback } from "react";
```

```
import {
  fetchDashboard,
  fetchDashboardCardData,
} from "metabase/dashboard/actions";
import { useDispatch } from...
```

### **frontend/src/metabase/dashboard/hooks/use-register-dashboard-metabot-context.ts**

```
import { useRegisterMetabotContextProvider } from "metabase/metabot";

import { getDashboard } from "../selectors";
```

```
export const useRegisterDashboardMetabotContext = () => {
  ...
}
```

### **frontend/src/metabase/dashboard/hooks/use-responsive-parameter-list.ts**

```
import { useCallback, useEffect, useRef, useState } from "react";

import { useDashboardContext } from "metabase/dashboard/context";
import resizeObserver from "metabase/lib/resize-observer";
```

```
type...
```

### **frontend/src/metabase/dashboard/reducers-typed.ts**

```
import { createReducer } from "@reduxjs/toolkit";
import { assocIn, dissocIn } from "icepick";
import { omit } from "underscore";
```

```
import {
  createDashboardPublicLink,
  deleteDashboardPublicLink,
  ...
}
```

## Analyst Base

### **frontend/src/metabase/dashboard/reducers.js**

```
import { assoc, assocIn, chain, dissoc, merge, updateIn } from "icepick";
import reduceReducers from "reduce-reducers";
import _ from "underscore";
```

```
import Actions from...
```

### **frontend/src/metabase/dashboard/reducers.unit.spec.js**

```
import { createMockDashboard } from "metabase-types/api/mocks";
```

```
import {
  ADD_DASHBOARD_IDS_TO_LOADING_QUEUE,
  CLOSE_SIDEBAR,
  FETCH_DASHBOARD_CARD_DATA,
  INITIALIZE,
  REMOVE_PARAMETER,
  ...
```

### **frontend/src/metabase/dashboard/sections.ts**

```
import { t } from "ttag";
```

```
import { GRID_WIDTH } from "metabase/lib/dashboard_grid";
import type {
  DashboardCardLayoutAttrs,
  VirtualCard,
  VirtualDashboardCard,
} from...
```

### **frontend/src/metabase/dashboard/selectors.ts**

```
import { createSelector } from "@reduxjs/toolkit";
import { createCachedSelector } from "re-reselect";
import _ from "underscore";
```

```
import { LOAD_COMPLETE_FAVICON } from...
```

### **frontend/src/metabase/dashboard/selectors.unit.spec.js**

```
import { chain } from "icepick";
```

```
import { createMockEntitiesState } from "__support__/store";
import {
  getClickBehaviorSidebarDashcard,
  getDashboardComplete,
  getEditingParameterId,
```

...

### **frontend/src/metabase/dashboard/types/display-options.ts**

```
import type { EmbedResourceDownloadOptions } from "metabase/public/lib/types";
```

```
export type DashboardFullscreenControls = {  
  isFullscreen: boolean;  
  onFullscreenChange: (  
    newIsFullscreen:...
```

### **frontend/src/metabase/dashboard/types/embed-display-options.ts**

```
import type { DashboardNightModeControls } from "metabase/dashboard/types/display-options";  
import type {  
  DisplayTheme,  
  EmbedResourceDownloadOptions,  
} from "metabase/public/lib/types";  
import...
```

### **frontend/src/metabase/dashboard/types/fetch-dashboard-result.ts**

```
import type { Dashboard } from "metabase-types/api";
```

```
export type SuccessfulFetchDashboardResult = {  
  payload: { dashboard: Dashboard };  
};  
export type FailedFetchDashboardResult = { error: unknown;...
```

### **frontend/src/metabase/dashboard/types/hash-options.ts**

```
import type {  
  EmbeddingAdditionalHashOptions,  
  EmbeddingDisplayOptions,  
} from "metabase/public/lib/types";
```

```
/**  
 * This is a type that controls some dashboard states.  
 */  
export type...
```

### **frontend/src/metabase/dashboard/types/index.ts**

```
export * from "./display-options";  
export * from "./hash-options";  
export * from "./embed-display-options";  
export * from "./fetch-dashboard-result";
```

### **frontend/src/metabase/dashboard/utils.ts**

```
import type { Location } from "history";
```

## Analyst Base

```
import _ from "underscore";
```

```
import { SERVER_ERROR_TYPES } from "metabase/lib/errors";
```

```
import { isJWT } from "metabase/lib/utils";
```

```
import { isUuid } from...
```

### **frontend/src/metabase/dashboard/utils.unit.spec.ts**

```
import type { Location } from "history";
```

```
import {
```

```
  canResetFilter,
```

```
  createTabSlug,
```

```
  fetchDataOrError,
```

```
  findDashCardForInlineParameter,
```

```
  getCurrentTabDashboardCards,
```

```
  getDashcardResultsError,
```

```
  ...
```

### **frontend/src/metabase/data-grid/constants.ts**

```
export const ROW_HEIGHT = 36;
```

```
export const HEADER_HEIGHT = 36;
```

```
export const ADD_COLUMN_BUTTON_WIDTH = 36;
```

```
export const MIN_COLUMN_WIDTH = 60;
```

```
export const ROW_ID_COLUMN_ID = `0_INDEX`;
```

```
export const...
```

### **frontend/src/metabase/data-grid/guards.ts**

```
import type { MaybeVirtualRow, VirtualRow } from "./types";
```

```
export const isVirtualRow = <TData>(  
  maybeVirtualRow: MaybeVirtualRow<TData>,  
)  
): maybeVirtualRow is VirtualRow<TData> => {  
  return...
```

### **frontend/src/metabase/data-grid/hooks/index.ts**

```
export { useDataGridTheme, DataGridThemeProvider } from "./use-table-theme";
```

### **frontend/src/metabase/data-grid/index.ts**

```
export * from "./components/DataGrid/DataGrid";
```

```
export * from "./components/BaseCell/BaseCell";
```

```
export * from "./components/HeaderCell/HeaderCell";
```

```
export * from...
```

### **frontend/src/metabase/data-grid/types.ts**

```
import type {
```



## Analyst Base

Cell,  
CellContext,  
ColumnDefTemplate,  
ColumnPinningState,  
ColumnSizingState,  
HeaderContext,  
OnChangeFn,  
Row,  
RowData,  
RowSelectionOptions,  
RowSelectionState,  
...

### **frontend/src/metabase/data-grid/utils/column-sizing.ts**

```
import type { ColumnSizingState } from "@tanstack/react-table";
```

```
export const pickRowsToMeasure = <TData, TValue>(  
  data: TData[],  
  accessorFn: (row: TData) => TValue,  
  count = 10,  
) => {  
  const...
```

### **frontend/src/metabase/data-grid/utils/column-sizing.unit.spec.ts**

```
import { getTruncatedColumnSizing, pickRowsToMeasure } from "../column-sizing";
```

```
describe("pickRowsToMeasure", () => {  
  const createData = (values: (string | null | undefined)[]) =>  
    ...
```

### **frontend/src/metabase/data-grid/utils/formatting.ts**

```
import type { PlainCellFormatter } from "../types";
```

```
/**  
 * Formats cell value for copying, applying formatters when available.  
 *  
 * Null value handling:  
 * - Raw copy: null/undefined -> "null"  
 * -...
```

### **frontend/src/metabase/data-grid/utils/maybe-expand-column-widths.ts**

```
import type { ColumnSizingState } from "@tanstack/react-table";
```

```
/**
```

## Analyst Base

\* Proportionally expands column widths to fill available space while preserving fixed-width columns.

\* Only expands columns when...

### **frontend/src/metabase/data-grid/utils/maybe-expand-column-widths.unit.spec.ts**

```
import { maybeExpandColumnWidths } from "../maybe-expand-column-widths";
```

```
describe("maybeExpandColumnWidths", () => {  
  it("returns the original map when minGridWidth is undefined, 0, or negative", ()...
```

### **frontend/src/metabase/databases/components/DatabaseAuthCodeDescription/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/databases/components/DatabaseAuthProviderSectionField/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/databases/components/DatabaseCacheScheduleField/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/databases/components/DatabaseClientIdDescription/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/databases/components/DatabaseConnectionSectionField/index.ts**

```
export { DatabaseConnectionSectionField } from "../DatabaseConnectionSectionField";
```

### **frontend/src/metabase/databases/components/DatabaseConnectionUri/analytics.ts**

```
import { trackSimpleEvent } from "metabase/lib/analytics";
```

```
export function connectionStringParsedSuccess(  
  location: "admin" | "setup" | "embedding_setup",  
) {  
  trackSimpleEvent({  
    event:...
```

### **frontend/src/metabase/databases/components/DatabaseConnectionUri/database-field-mapper.ts**

```
import { P, match } from "ts-pattern";
```

## Analyst Base

```
import type { EngineKey } from "metabase-types/api/settings";
```

```
import type { RegexFields } from "../parse-connection-regex";
```

```
function...
```

### **frontend/src/metabase/databases/components/DatabaseConnectionUri/engines-config.ts**

```
import type { EngineKey } from "metabase-types/api/settings";
```

```
type Placeholder = string;
```

```
export const enginesConfig: Record<EngineKey, Placeholder> = {  
  athena:...
```

### **frontend/src/metabase/databases/components/DatabaseConnectionUri/index.ts**

```
export { DatabaseConnectionStringField } from "../DatabaseConnectionStringField";
```

### **frontend/src/metabase/databases/components/DatabaseConnectionUri/parse-connection-regex.ts**

```
import type { EngineKey } from "metabase-types/api";
```

```
export interface RegexFields {  
  host?: string;  
  port?: string;  
  database?: string;  
  catalog?: string;  
  schema?: string;  
  username?:...
```

### **frontend/src/metabase/databases/components/DatabaseConnectionUri/parse-connection-regex.unit.spec.ts**

```
import { enginesConfig } from "../engines-config";  
import { parseConnectionUriRegex } from "../parse-connection-regex";
```

```
describe("parseConnectionUriRegex - Amazon Athena", () => {  
  it("should parse a...
```

### **frontend/src/metabase/databases/components/DatabaseDetailField/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/databases/components/DatabaseEngineField/index.ts**

```
export { DatabaseEngineField } from "../DatabaseEngineField";
```

### **frontend/src/metabase/databases/components/DatabaseEngineList/index.ts**

## Analyst Base

```
export { DatabaseEngineList } from "./DatabaseEngineList";
```

**frontend/src/metabase/databases/components/DatabaseEngineWarning/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/databases/components/DatabaseForm/index.ts**

```
export * from "./DatabaseForm";
```

**frontend/src/metabase/databases/components/DatabaseHelpCard/index.ts**

```
export * from "./DatabaseHelpCard";
```

**frontend/src/metabase/databases/components/DatabaseHostnameSectionField/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/databases/components/DatabaseInfoField/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/databases/components/DatabaseNameField/index.ts**

```
export { DatabaseNameField } from "./DatabaseNameField";
```

**frontend/src/metabase/databases/components/DatabaseScheduleToggleField/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/databases/components/DatabaseSectionField/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/databases/components/DatabaseSshDescription/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/databases/components/DatabaseSslKeyDescription/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/databases/components/DatabaseSyncScheduleField/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/databases/selectors.ts**

```
import { getSetting } from "metabase/selectors/settings";
```

```
import type { State } from "metabase-types/store";
```

```
export const getEngines = (state: State) => {
```

## Analyst Base

```
return getSetting(state,...
```

### **frontend/src/metabase/databases/types.ts**

```
import type { ComponentType, ReactNode } from "react";
```

```
import type { EngineFieldOption, EngineFieldType } from "metabase-types/api";
```

```
export interface EngineOption {  
  name: string;  
  value:...
```

### **frontend/src/metabase/databases/utils/engine.ts**

```
import type { Engine } from "metabase-types/api";
```

```
import { ELEVATED_ENGINES, ENGINE_LOGO } from "../constants";  
import type { EngineOption } from "../types";
```

```
export const getEngineOptions = (  
  ...
```

### **frontend/src/metabase/databases/utils/schema.ts**

```
import type { TestContext } from "yup";  
import * as Yup from "yup";
```

```
import * as Errors from "metabase/lib/errors";  
import type { DatabaseData, Engine, EngineField } from...
```

### **frontend/src/metabase/dev-noop.js**

Make sure this gets recognized as an ES module

### **frontend/src/metabase/dev.js**

```
if (  
  // eslint-disable-next-line no-undef  
  process.env.ENABLE_CLJS_HOT_RELOAD === "true" ||  
  // eslint-disable-next-line no-undef  
  process.env.ENABLE_CLJS_DEV_TOOLS === "true"  
) {  
  ...
```

### **frontend/src/metabase/embedding/components/PublicLinkPopover/index.ts**

```
export { QuestionPublicLinkPopover } from "./QuestionPublicLinkPopover";  
export { DashboardPublicLinkPopover } from "./DashboardPublicLinkPopover";
```

### **frontend/src/metabase/embedding/components/PublicLinkPopover/types.ts**

```
import type { exportFormats } from "metabase/lib/urls";
```

## Analyst Base

```
export type ExportFormatType = (typeof exportFormats)[number] | null;
```

### **frontend/src/metabase/embedding/components/SharingMenu/index.ts**

```
export * from "./DashboardSharingMenu";
```

```
export * from "./QuestionSharingMenu";
```

### **frontend/src/metabase/embedding/components/SharingMenu/types.ts**

```
export type EmbedModalType = "question-embed" | "dashboard-embed";
```

```
export type PublicLinkModalType =
```

```
  | "question-public-link"
```

```
  | "dashboard-public-link";
```

```
export type QuestionSharingModalType =
```

```
...
```

### **frontend/src/metabase/embedding-sdk/config.ts**

```
import type { SettingKey, TokenFeature } from "metabase-types/api";
```

```
export const EMBEDDING_SDK_PORTAL_ROOT_ELEMENT_ID = "metabase-sdk-portal-root";
```

```
type InternalSdkConfig = {
```

```
  isEmbeddingSdk:...
```

### **frontend/src/metabase/embedding-sdk/lib/define-global-dependencies.ts**

```
import * as React from "react";
```

```
import * as ReactJSXRuntime from "react/jsx-runtime";
```

```
import * as ReactDOM from "react-dom";
```

```
import * as ReactDOMClient from "react-dom/client";
```

```
import * as...
```

### **frontend/src/metabase/embedding-sdk/mocks/config-mock.ts**

```
*
```

### **frontend/src/metabase/embedding-sdk/test/rename-conflicting-cljs-globals.ts**

```
const CONFLICTING_CLJS_GLOBALS = ["cljs", "malli", "metabase"];
```

```
// To properly work with CLJS runtime that comes with Embedding SDK bundle
```

```
export function renameConflictingCljsGlobals() {
```

```
  for...
```

### **frontend/src/metabase/embedding-sdk/theme/MetabaseTheme.ts**

```
import type { CSSProperties } from "react";
```

```
import type { ColorName } from "metabase/lib/colors/types";
```

## Analyst Base

```
import type { DeepPartial } from "../types/utils";
```

```
import type { MetabaseFontFamily } from...
```

### **frontend/src/metabase/embedding-sdk/theme/css-vars-to-sdk-theme.ts**

```
import type { FlattenObjectKeys } from "../types/utils";
```

```
import type { MetabaseComponentTheme } from "../MetabaseTheme";
```

```
type MetabaseComponentThemeKey =...
```

### **frontend/src/metabase/embedding-sdk/theme/default-component-theme.ts**

```
import { merge } from "icepick";
```

```
import { OVERLAY_Z_INDEX } from "metabase/css/core/overlays/constants";
```

```
import { EMBEDDING_SDK_PORTAL_ROOT_ELEMENT_ID } from "metabase/embedding-sdk/config";
```

```
import...
```

### **frontend/src/metabase/embedding-sdk/theme/define-metabase-theme.ts**

```
import type { MetabaseTheme } from "../MetabaseTheme";
```

```
export const defineMetabaseTheme = (theme: MetabaseTheme): MetabaseTheme =>
  theme;
```

### **frontend/src/metabase/embedding-sdk/theme/dynamic-css-vars-config.ts**

```
import type { DynamicCssVarConfig } from "../types/private/css-variables";
```

```
/**
```

```
 * These CSS variables are dynamically generated based on the theme.
```

```
 */
```

```
export const DYNAMIC_CSS_VARIABLES:...
```

### **frontend/src/metabase/embedding-sdk/theme/dynamic-css-vars.ts**

```
eslint-disable-next-line no-restricted-imports
```

### **frontend/src/metabase/embedding-sdk/theme/embedding-color-palette.ts**

```
import type {
```

```
  MetabaseColor,
```

```
  MetabaseColors,
```

```
} from "metabase/embedding-sdk/theme";
```

```
import { colors } from "metabase/lib/colors";
```

```
import type { ColorName, ColorPalette } from...
```

### **frontend/src/metabase/embedding-sdk/theme/fonts.ts**

## Analyst Base

```
export type MetabaseFontFamily =  
  | "Roboto"  
  | "Merriweather"  
  | "Open Sans"  
  | "Lato"  
  | "Noto Sans"  
  | "Roboto Slab"  
  | "Source Sans Pro"  
  | "Raleway"  
  | "Slabo 27px"  
  | "PT Sans"  
  | ...
```

### frontend/src/metabase/embedding-sdk/theme/get-embedding-chart-colors.ts

```
import type { ChartColor } from "metabase/embedding-sdk/theme";
```

```
/**  
 * Map the input chart colors from the theme settings to the  
 * color names we use in our charts.  
 *  
 * @param chartColors chart...
```

### frontend/src/metabase/embedding-sdk/theme/index.ts

```
export * from "./default-component-theme";  
// eslint-disable-next-line no-literal-metabase-strings -- file name  
export * from "./MetabaseTheme";
```

```
export * from "./define-metabase-theme";
```

### frontend/src/metabase/embedding-sdk/theme/private.ts

```
import type { MetabaseComponentTheme, MetabaseTheme } from ".";
```

```
/**  
 * Mantine theme options specific to React embedding.  
 */  
export type EmbeddingThemeOptions = MetabaseComponentTheme &  
  ...
```

### frontend/src/metabase/embedding-sdk/types/components/data-picker.ts

```
import type { CardType, TableId } from "metabase-types/api";
```

```
export interface DataSourceSelectorProps {  
  isInitiallyOpen: boolean;  
  /** Type of the the query's first stage */
```



## Analyst Base

querySourceType:

...

### frontend/src/metabase/embedding-sdk/types/dashboard.ts

```
export type ParameterValues = Record<
  string,
  string | string[] | undefined | null
>;
```

### frontend/src/metabase/embedding-sdk/types/icon.ts

```
import type { IconName as InternalIconName } from "metabase/ui";

/**
 * Inline wrapper to properly display the `IconName` type without referencing the `internal` type
 *
 * @hidden
 * @inline
 */
...
```

### frontend/src/metabase/embedding-sdk/types/plugins.ts

```
import type { ReactNode } from "react";

import type {
  DashCardMenuItem
} from
"metabase/dashboard/components/DashCard/DashCardMenu/dashcard-menu";
import type { ClickAction, ClickObject } from...
```

### frontend/src/metabase/embedding-sdk/types/private/css-variables.ts

```
import type { SemanticColorKey } from "metabase/embedding-sdk/theme/embedding-color-palette";
import type { ColorName } from "metabase/lib/colors/types";

export type SourceColorKey = ColorName |...
```

### frontend/src/metabase/embedding-sdk/types/question.ts

```
export interface MetabaseQuestion {
  id: number;
  name: string;
  description: string | null;
  entityId: string;

  isSavedQuestion: boolean;
}
```

### frontend/src/metabase/embedding-sdk/types/utils.ts

\*

## Analyst Base

### **frontend/src/metabase/entities/actions/actions.ts**

```
import { updateIn } from "icepick";
import { t } from "ttag";
```

```
import {
  actionApi,
  useGetActionQuery,
  useListActionsQuery,
} from "metabase/api";
import {
  createEntity,
  ...
```

### **frontend/src/metabase/entities/actions/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/entities/bookmarks.js**

```
import { createSelector } from "@reduxjs/toolkit";
import { assoc, dissoc, getIn, updateIn } from "icepick";
import { t } from "ttag";
import _ from "underscore";
```

```
import { bookmarkApi,...
```

### **frontend/src/metabase/entities/collections/collections.ts**

```
import { createSelector } from "@reduxjs/toolkit";
import { t } from "ttag";
import _ from "underscore";
```

```
import {
  collectionApi,
  skipToken,
  useGetCollectionQuery,
  useListCollectionsQuery,
  ...
```

### **frontend/src/metabase/entities/collections/constants.ts**

```
import { t } from "ttag";
```

```
export const DEFAULT_COLLECTION_COLOR_ALIAS = "brand";
```

```
export const ROOT_COLLECTION = {
  id: "root" as const,
```

## Analyst Base

```
get name() {  
  return t`Our analytics`;  
},  
location:...
```

### **frontend/src/metabase/entities/collections/getExpandedCollectionsById.js**

```
import { t } from "ttag";  
import _ from "underscore";  
  
import {  
  PERSONAL_COLLECTION,  
  PERSONAL_COLLECTIONS,  
  ROOT_COLLECTION,  
} from "../constants";
```

```
// given list of collections with { id, name,...
```

### **frontend/src/metabase/entities/collections/getInitialCollectionId.ts**

```
import { createSelector } from "@reduxjs/toolkit";  
import type { Location } from "history";  
  
import {  
  canonicalCollectionId,  
  isRootTrashCollection,  
} from "metabase/collections/utils";  
import *...
```

### **frontend/src/metabase/entities/collections/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/entities/collections/tests/selectors.unit.spec.js**

```
import { collections as Collections } from "metabase/entities";  
import { ROOT_COLLECTION } from "metabase/entities/collections/constants";  
  
describe("Collection selectors", () => {  
  const...
```

### **frontend/src/metabase/entities/collections/utils.ts**

```
import {  
  isRootCollection,  
  isRootPersonalCollection,  
  isRootTrashCollection,  
} from "metabase/collections/utils";  
import { color } from "metabase/lib/colors";
```

## Analyst Base

```
import { PLUGIN_COLLECTIONS } from...
```

### **frontend/src/metabase/entities/collections/utils.unit.spec.ts**

```
import { setupEnterpriseTest } from "__support__/enterprise";
import { createMockCollection } from "metabase-types/api/mocks";
```

```
import { PERSONAL_COLLECTIONS } from "../constants";
import {
  ...

```

### **frontend/src/metabase/entities/containers/EntityListLoader.unit.spec.js**

```
import "mutationobserver-shim";
```

```
import { renderWithProviders } from "__support__/ui";
import { EntityListLoader } from "metabase/entities/containers/rtk-query";
import { Api } from...
```

### **frontend/src/metabase/entities/containers/index.js**

```
eslint-disable react/prop-types
```

### **frontend/src/metabase/entities/containers/rtk-query/index.ts**

```
export * from "../EntityListLoader";
export * from "../EntityObjectLoader";
export * from "../types";
```

### **frontend/src/metabase/entities/containers/rtk-query/types/entities.ts**

```
import type { BaseQueryFn, QueryDefinition } from "@reduxjs/toolkit/query";
```

```
import type { TagType } from "metabase/api/tags";
import type { IconName } from "metabase/ui";
import type { Collection } from...
```

### **frontend/src/metabase/entities/containers/rtk-query/types/index.ts**

```
export * from "../entities";
```

### **frontend/src/metabase/entities/containers/rtk-query/types/rtk.ts**

```
import type {
  QueryActionCreatorResult,
  QueryArgFrom,
  QueryDefinition,
  QueryStatus,
  QuerySubState,
  ResultTypeFrom,
  SkipToken,
  SubscriptionOptions,

```

## Analyst Base

```
TSHelpersId,  
TSHelpersNoInfer,  
...
```

### **frontend/src/metabase/entities/containers/rtk-query/usePaginatedQuery.ts**

```
import { useCallback, useMemo, useState } from "react";  
  
import type { EntityQuery } from "../types";  
  
/**  
 * @deprecated exists for backwards compatibility  
 */  
export const usePaginatedQuery = (  
  ...  
  ...
```

### **frontend/src/metabase/entities/dashboards.js**

```
import { t } from "ttag";  
  
import { automagicDashboardsApi } from "metabase/api/automagic-dashboards";  
import {  
  dashboardApi,  
  useGetDashboardQuery,  
  useListDashboardsQuery,  
} from...
```

### **frontend/src/metabase/entities/databases.js**

```
import { createSelector } from "@reduxjs/toolkit";  
import { normalize } from "normalizr";  
import _ from "underscore";  
  
import {  
  databaseApi,  
  useGetDatabaseMetadataQuery,  
  useGetDatabaseQuery,  
  ...
```

### **frontend/src/metabase/entities/fields.js**

```
import { assocIn, updateIn } from "icepick";  
import { normalize } from "normalizr";  
import { useMemo } from "react";  
import { t } from "ttag";  
  
import {  
  fieldApi,
```

## Analyst Base

```
skipToken,  
useGetFieldQuery,  
...
```

### **frontend/src/metabase/entities/groups.js**

```
import { assocIn } from "icepick";  
  
import { CLEAR_MEMBERSHIPS } from "metabase/admin/people/events";  
import {  
  permissionApi,  
  useGetPermissionsGroupQuery,  
  useListPermissionsGroupsQuery,  
} from...
```

### **frontend/src/metabase/entities/index.ts**

```
export { default as actions } from "./actions";  
export { default as collections } from "./collections";  
export { default as snippetCollections } from "./snippet-collections";  
export { default as...
```

### **frontend/src/metabase/entities/indexed-entities/index.ts**

```
export * from "./indexed-entities";
```

### **frontend/src/metabase/entities/indexed-entities/indexed-entities.ts**

```
*
```

### **frontend/src/metabase/entities/model-indexes/actions.ts**

```
import { dissocIn } from "icepick";  
import _ from "underscore";
```

```
import {  
  createModelIndex,  
  deleteModelIndex,  
  listModelIndexes,  
} from "metabase/api";  
import type Question from...
```

### **frontend/src/metabase/entities/model-indexes/actions.unit.spec.ts**

```
import fetchMock from "fetch-mock";  
  
import { getStore } from "__support__/entities-store";  
import { setupModelIndexEndpoints } from "__support__/server-mocks";  
import { Api } from...
```

### **frontend/src/metabase/entities/model-indexes/utils.ts**

## Analyst Base

```
import _ from "underscore";

import type Question from "metabase-lib/v1/Question";
import type FieldEntity from "metabase-lib/v1/metadata/Field";
import {
  isBoolean,
  isInteger,
  isPK,
  ...
```

### **frontend/src/metabase/entities/model-indexes/utils.unit.spec.ts**

```
import Question from "metabase-lib/v1/Question";
import Field from "metabase-lib/v1/metadata/Field";
import type { Field as FieldAPI } from "metabase-types/api";
import { createMockCard, ...
```

### **frontend/src/metabase/entities/persisted-models.js**

```
import { createSelector } from "@reduxjs/toolkit";

import {
  cardApi,
  persistApi,
  skipToken,
  useGetPersistedInfoByCardQuery,
  useGetPersistedInfoQuery,
  useListPersistedInfoQuery,
} from ...
```

### **frontend/src/metabase/entities/pulses.js**

```
import { t } from "ttag";

import {
  subscriptionApi,
  useGetSubscriptionQuery,
  useListSubscriptionsQuery,
} from "metabase/api";
import { getCollectionType } from ...
```

### **frontend/src/metabase/entities/questions.js**

```
import { updateIn } from "icepick";
import { t } from "ttag";

import {
  cardApi,
```

## Analyst Base

```
datasetApi,  
useGetCardQuery,  
useListCardsQuery,  
} from "metabase/api";  
import {  
  canonicalCollectionId,  
  ...
```

### **frontend/src/metabase/entities/revisions.js**

```
import { revisionApi, useListRevisionsQuery } from "metabase/api";  
import { createEntity, entityCompatibleQuery } from "metabase/lib/entities";  
  
import Dashboards from "../dashboards";  
import...
```

### **frontend/src/metabase/entities/schemas.js**

```
import { assocIn, updateIn } from "icepick";  
import { useMemo } from "react";  
  
import {  
  databaseApi,  
  skipToken,  
  useListDatabaseSchemaTablesQuery,  
  useListDatabaseSchemasQuery,  
  ...
```

### **frontend/src/metabase/entities/schemas.unit.spec.js**

```
import fetchMock from "fetch-mock";  
  
import { getStore } from "___support___/entities-store";  
import { Api } from "metabase/api";  
import Questions from "metabase/entities/questions";  
import Schemas...
```

### **frontend/src/metabase/entities/search.js**

```
import { useMemo } from "react";  
  
import {  
  cardApi,  
  collectionApi,  
  searchApi,  
  skipToken,  
  useListCollectionItemsQuery,  
  useSearchQuery,
```



## Analyst Base

```
} from "metabase/api";  
import { canonicalCollectionId...
```

### **frontend/src/metabase/entities/segments.js**

```
import {  
  segmentApi,  
  useGetSegmentQuery,  
  useListSegmentsQuery,  
} from "metabase/api";  
import { color } from "metabase/lib/colors";  
import { createEntity, entityCompatibleQuery } from...
```

### **frontend/src/metabase/entities/snippet-collections.js**

```
import { createSelector } from "@reduxjs/toolkit";  
import { t } from "ttag";  
import _ from "underscore";  
  
import { skipToken, useGetCollectionQuery } from "metabase/api";  
import {...
```

### **frontend/src/metabase/entities/snippets.js**

```
import {  
  snippetApi,  
  useGetSnippetQuery,  
  useListSnippetsQuery,  
} from "metabase/api";  
import { createEntity, entityCompatibleQuery } from "metabase/lib/entities";
```

```
/**  
 * @deprecated use...
```

### **frontend/src/metabase/entities/tables.js**

```
import { createSelector } from "@reduxjs/toolkit";  
import { updateIn } from "icepick";  
import { useEffect, useMemo } from "react";  
import { match } from "ts-pattern";  
import { t } from "ttag";  
import...
```

### **frontend/src/metabase/entities/tables.unit.spec.js**

```
import Questions from "metabase/entities/questions";  
import Tables from "metabase/entities/tables";  
import { convertSavedQuestionToVirtualTable } from...
```

### **frontend/src/metabase/entities/timeline-events.js**

## Analyst Base

```
import { t } from "ttag";
```

```
import { timelineEventApi, useGetTimelineEventQuery } from "metabase/api";
```

```
import {  
  createEntity,  
  entityCompatibleQuery,  
  undo,  
} from "metabase/lib/entities";  
import...
```

### **frontend/src/metabase/entities/timelines.js**

```
import { updateIn } from "icepick";  
import { t } from "ttag";  
import _ from "underscore";
```

```
import {  
  skipToken,  
  timelineApi,  
  timelineEventApi,  
  useGetTimelineQuery,  
  ...
```

### **frontend/src/metabase/env.ts**

```
const tryOrDefault = <T>(fn: () => T, defaultValue: T): T => {  
  try {  
    return fn();  
  } catch (e) {  
    console.warn(  
      "Error while trying to get env",  
      e,  
      `returning default:...`  
    );  
  }  
}
```

### **frontend/src/metabase/forms/components/Form/index.ts**

```
export * from "./Form";
```

### **frontend/src/metabase/forms/components/FormCheckbox/index.ts**

```
export * from "./FormCheckbox";
```

### **frontend/src/metabase/forms/components/FormCheckboxGroup/index.ts**

```
export * from "./FormCheckboxGroup";
```

### **frontend/src/metabase/forms/components/FormChipGroup/index.ts**

```
export * from "./FormChipGroup";
```

### **frontend/src/metabase/forms/components/FormDateInput/index.ts**

## Analyst Base

export \* from "./FormDateInput";

**frontend/src/metabase/forms/components/FormErrorMessage/index.ts**

export \* from "./FormErrorMessage";

**frontend/src/metabase/forms/components/FormGroupWidget/index.ts**

export \* from "./FormGroupWidget";

**frontend/src/metabase/forms/components/FormGroupsWidget/index.ts**

export \* from "./FormGroupsWidget";

**frontend/src/metabase/forms/components/FormMessage/index.ts**

export \* from "./FormMessage";

**frontend/src/metabase/forms/components/FormNumberInput/index.ts**

export \* from "./FormNumberInput";

**frontend/src/metabase/forms/components/FormObserver/index.ts**

export \* from "./FormObserver";

**frontend/src/metabase/forms/components/FormProvider/index.ts**

export \* from "./FormProvider";

**frontend/src/metabase/forms/components/FormRadioGroup/index.ts**

export \* from "./FormRadioGroup";

**frontend/src/metabase/forms/components/FormSecretKey/index.ts**

export \* from "./FormSecretKey";

**frontend/src/metabase/forms/components/FormSection/index.ts**

export \* from "./FormSection";

**frontend/src/metabase/forms/components/FormSelect/index.ts**

export \* from "./FormSelect";

**frontend/src/metabase/forms/components/FormSubmitButton/index.ts**

export \* from "./FormSubmitButton";

**frontend/src/metabase/forms/components/FormSwitch/index.ts**

export \* from "./FormSwitch";

**frontend/src/metabase/forms/components/FormTextInput/index.ts**

export \* from "./FormTextInput";

**frontend/src/metabase/forms/components/FormTextarea/index.ts**

export \* from "./FormTextarea";

**frontend/src/metabase/forms/components/index.ts**

## Analyst Base

```
export * from "./Form";
export * from "./FormCheckbox";
export * from "./FormCheckboxGroup";
export * from "./FormChipGroup";
export * from "./FormDateInput";
export * from...
```

### **frontend/src/metabase/forms/contexts/FormContext/index.ts**

```
export * from "./FormContext";
```

### **frontend/src/metabase/forms/contexts/index.ts**

```
export * from "./FormContext";
```

### **frontend/src/metabase/forms/hooks/index.ts**

```
export * from "./use-form-context";
export * from "./use-form-error-message";
export * from "./use-form-submit";
export * from "./use-form-submit-button";
export * from "./use-form-validation";
```

### **frontend/src/metabase/forms/hooks/use-form-context/index.ts**

```
export * from "./use-form-context";
```

### **frontend/src/metabase/forms/hooks/use-form-context/use-form-context.ts**

```
import { useContext } from "react";
```

```
import type { IFormContext } from "../../contexts";
import { FormContext } from "../../contexts";
```

```
export const useFormContext = (): IFormContext => {
  return...
```

### **frontend/src/metabase/forms/hooks/use-form-error-message/index.ts**

```
export * from "./use-form-error-message";
```

### **frontend/src/metabase/forms/hooks/use-form-error-message/use-form-error-message.ts**

```
import { useFormikContext } from "formik";
import { useEffect, useState } from "react";
import { t } from "ttag";
```

```
import { useFormContext } from "../use-form-context";
```

```
export const...
```

## Analyst Base

### **frontend/src/metabase/forms/hooks/use-form-submit/index.ts**

```
export * from "../use-form-submit";
```

### **frontend/src/metabase/forms/hooks/use-form-submit/types.ts**

```
import type { FormikErrors } from "formik";

export interface FormError<T> extends FormErrorData<T> {
  data?: string | FormErrorData<T>;
}
```

```
export interface FormErrorData<T> {
  errors?:...
```

### **frontend/src/metabase/forms/hooks/use-form-submit/use-form-submit.ts**

```
import type { FormikHelpers } from "formik";
import type { Dispatch, SetStateAction } from "react";
import { useCallback, useState } from "react";
```

```
import { getResponseErrorMessage } from...
```

### **frontend/src/metabase/forms/hooks/use-form-submit-button/index.ts**

```
export * from "../use-form-submit-button";
```

### **frontend/src/metabase/forms/hooks/use-form-submit-button/use-form-submit-button.ts**

```
import { useFormikContext } from "formik";
import { useEffect, useLayoutEffect, useState } from "react";
```

```
import type { FormStatus } from "../../contexts";
import { useFormContext } from...
```

### **frontend/src/metabase/forms/hooks/use-form-validation/index.ts**

```
export * from "../use-form-validation";
```

### **frontend/src/metabase/forms/hooks/use-form-validation/use-form-validation.ts**

```
import type { FormikErrors, FormikValues } from "formik";
import { prepareDataForValidation, yupToFormErrors } from "formik";
import { useCallback, useMemo } from "react";
import type { AnySchema } from...
```

### **frontend/src/metabase/forms/index.ts**

```
export * from "../components";
export * from "../contexts";
export * from "../hooks";
```

## Analyst Base

```
export * from "./utils";
```

### **frontend/src/metabase/forms/utils/index.ts**

```
export * from "./messages";
```

### **frontend/src/metabase/forms/utils/messages.ts**

```
import { t } from "ttag";
```

```
export interface MaxLengthParams {  
  max: number;  
}
```

```
export const requiredErrorMessage = () => t`Required`;
```

```
export const emailErrorMessage = () => t`Must be a valid email...`;
```

### **frontend/src/metabase/hoc/ModalRoute.unit.spec.js**

```
import { mockSettings } from "__support__/settings";
```

```
import { getParentPath } from "../ModalRoute";
```

```
const setup = (routePath, locationPath, siteURL = undefined) => {  
  if (siteURL) {  
    ...  
  }  
}
```

### **frontend/src/metabase/hoc/ScrollToTop.js**

```
eslint-disable react/prop-types
```

### **frontend/src/metabase/hoc/utils.js**

```
export function getDisplayName(WrappedComponent) {  
  return WrappedComponent.displayName || WrappedComponent.name || "Component";  
}
```

### **frontend/src/metabase/home/components/CustomHomePageModal/index.ts**

```
export * from "../CustomHomePageModal";
```

### **frontend/src/metabase/home/components/EmbedHomepage/Badge.ts**

```
eslint-disable-next-line no-restricted-imports
```

### **frontend/src/metabase/home/components/EmbedHomepage/actions.ts**

```
import { updateSetting } from "metabase/admin/settings/settings";  
import { createAsyncThunk } from "metabase/lib/redux";  
import type { EmbeddingHomepageDismissReason } from...
```

### **frontend/src/metabase/home/components/EmbedHomepage/analytics.ts**

```
import { trackSchemaEvent } from "metabase/lib/analytics";
```

## Analyst Base

```
import type { EmbeddingHomepageDismissReason } from "metabase-types/api";
```

```
import type { EmbeddingHomepageInitialTab } from...
```

### **frontend/src/metabase/home/components/EmbedHomepage/index.ts**

```
export { EmbedHomepage } from "./EmbedHomepage";  
export { EmbedHomepageView } from "./EmbedHomepageView";  
export type { EmbedHomepageViewProps } from "./EmbedHomepageView";
```

### **frontend/src/metabase/home/components/EmbedHomepage/types.ts**

```
export type EmbeddingHomepageInitialTab = "static" | "interactive";
```

### **frontend/src/metabase/home/components/HomeCaption/index.ts**

```
export * from "./HomeCaption";
```

### **frontend/src/metabase/home/components/HomeCard/index.ts**

```
export * from "./HomeCard";
```

### **frontend/src/metabase/home/components/HomeContent/index.ts**

```
export * from "./HomeContent";
```

### **frontend/src/metabase/home/components/HomeGreeting/index.ts**

```
export * from "./HomeGreeting";
```

### **frontend/src/metabase/home/components/HomeHelpCard/index.ts**

```
export * from "./HomeHelpCard";
```

### **frontend/src/metabase/home/components/HomeLayout/index.ts**

```
export * from "./HomeLayout";
```

### **frontend/src/metabase/home/components/HomeModelCard/index.ts**

```
export * from "./HomeModelCard";
```

### **frontend/src/metabase/home/components/HomePage/index.ts**

```
export * from "./HomePage";
```

### **frontend/src/metabase/home/components/HomePopularSection/index.ts**

```
export * from "./HomePopularSection";
```

### **frontend/src/metabase/home/components/HomeRecentSection/index.ts**

```
export * from "./HomeRecentSection";
```

### **frontend/src/metabase/home/components/HomeXrayCard/index.ts**

```
export * from "./HomeXrayCard";
```

### **frontend/src/metabase/home/components/HomeXraySection/index.ts**

```
export * from "./HomeXraySection";
```

## Analyst Base

### **frontend/src/metabase/home/components/Onboarding/analytics.ts**

```
import { trackSimpleEvent } from "metabase/lib/analytics";

import type { ChecklistItemCTA, ChecklistItemValue } from "./types";

export const trackChecklistItemExpanded = (value: ChecklistItemValue)...
```

### **frontend/src/metabase/home/components/Onboarding/index.ts**

```
export * from "./Onboarding";
```

### **frontend/src/metabase/home/components/Onboarding/types.ts**

```
export type ChecklistItemValue =
  | "database"
  | "invite"
  | "x-ray"
  | "notebook"
  | "sql"
  | "dashboard"
  | "subscription"
  | "alert";

export type ChecklistItemCTA = "primary" |...
```

### **frontend/src/metabase/home/selectors.ts**

```
import { createSelector } from "@reduxjs/toolkit";
import dayjs from "dayjs";

import { getIsEmbeddingIframe } from "metabase/selectors/embed";
import { getSetting } from...
```

### **frontend/src/metabase/home/utils.ts**

```
import dayjs from "dayjs";

import { parseTimestamp } from "metabase/lib/time-dayjs";

export const isWithinWeeks = (
  timestamp: string,
  weekCount: number,
): boolean => {
  const date =...
```

### **frontend/src/metabase/i18n/hooks.ts**

```
import { PLUGIN_CONTENT_TRANSLATION } from "metabase/plugins";
```



## Analyst Base

```
import type { ContentTranslationFunction } from "./types";
```

```
/** To keep the components that require content translation tidier, they...
```

### **frontend/src/metabase/i18n/test-utils.ts**

```
import type { ReactElement } from "react";
```

```
import { setupEnterprisePlugins } from "__support__/enterprise";
```

```
import { setupContentTranslationEndpoints } from...
```

### **frontend/src/metabase/i18n/types.ts**

```
export type NonEmpty<ArrayType> = ArrayType extends (infer ItemType)[]  
  ? [ItemType, ...ItemType[]]  
  : never;
```

```
export type ContentTranslationFunction = <T = string | null | undefined>(  
  msgid:...
```

### **frontend/src/metabase/lib/analytics-untyped.js**

```
import * as Snowplow from "@snowplow/browser-tracker";
```

```
import Settings from "metabase/lib/settings";
```

```
import { getUserId } from "metabase/selectors/user";
```

```
export const trackPageView = (url) => {  
  ...
```

### **frontend/src/metabase/lib/analytics.ts**

```
import * as Snowplow from "@snowplow/browser-tracker";
```

```
import { shouldLogAnalytics } from "metabase/env";
```

```
import Settings from "metabase/lib/settings";
```

```
import type {  
  SchemaEvent,  
  SchemaType,  
  ...
```

### **frontend/src/metabase/lib/api.js**

```
import EventEmitter from "events";
```

```
import querystring from "querystring";
```

```
import { isTest } from "metabase/env";
```

```
import { isWithinIframe } from "metabase/lib/dom";
```

```
import { delay } from...
```

### **frontend/src/metabase/lib/arrays.ts**

```
export function moveElement<T>(array: T[], oldIndex: number, newIndex: number) {  
  const arrayCopy = [...array];  
  arrayCopy.splice(newIndex, 0, arrayCopy.splice(oldIndex, 1)[0]);  
  return...
```

### **frontend/src/metabase/lib/auth.js**

```
import { SessionApi } from "metabase/services";
```

```
export const deleteSession = async () => {  
  try {  
    await SessionApi.delete();  
  } catch (error) {  
    if (error.status !== 404) {  
      ...
```

### **frontend/src/metabase/lib/browser.ts**

```
import querystring from "querystring";
```

```
import { safeJsonParse } from "metabase/lib/json-parse";
```

```
function parseQueryStringOptions(s: string) {  
  const options: Record<string, string | string[] |...
```

### **frontend/src/metabase/lib/card.js**

```
import { b64hash_to_utf8, utf8_to_b64url } from "metabase/lib/encoding";  
import { equals } from "metabase/lib/utils";
```

```
export function createCard(name = null) {  
  return {  
    name: name,  
    ...
```

### **frontend/src/metabase/lib/click-behavior.js**

```
import { getIn } from "icepick";  
import { msgid, ngettext, t } from "ttag";
```

```
import Dashboards from "metabase/entities/dashboards";  
import Questions from "metabase/entities/questions";
```

```
export...
```

### **frontend/src/metabase/lib/collections.ts**

```
import { t } from "ttag";
```

## Analyst Base

```
import { isNotNull } from "metabase/lib/types";
import type { Collection, CollectionId } from "metabase-types/api";
```

```
export const getCrumbs = (
  collection: Collection,
  ...
```

### **frontend/src/metabase/lib/collections.unit.spec.ts**

```
import { createMockCollection } from "metabase-types/api/mocks";
```

```
import { getCrumbs } from "../collections";
```

```
const collectionsById = {
  root: createMockCollection({ id: "root", name: "Our..."
```

### **frontend/src/metabase/lib/colors/charts.ts**

```
import { getAccentColors, getPreferredColor } from "../groups";
import { ACCENT_COUNT } from "../palette";
import type { ColorPalette } from "../types";
```

```
export const getColorsForValues = (
  keys:...
```

### **frontend/src/metabase/lib/colors/charts.unit.spec.ts**

```
import _ from "underscore";
```

```
import { getColorsForValues } from "../charts";
import { color } from "../palette";
```

```
describe("charts", () => {
  it("should use accent colors for <= 8 series", () => {
    ...
```

### **frontend/src/metabase/lib/colors/colors.ts**

```
eslint-disable no-color-literals
```

NOTE: DO NOT ADD COLORS WITHOUT EXTREMELY GOOD REASON AND DESIGN REVIEW  
NOTE: KEEP SYNCHRONIZED...

### **frontend/src/metabase/lib/colors/groups.ts**

```
import _ from "underscore";
```

```
import { ACCENT_COUNT, color } from "../palette";
import type { AccentColorOptions, ColorName, ColorPalette } from "../types";
```

## Analyst Base

```
export const getAccentColors = (  
  {  
    ...
```

### **frontend/src/metabase/lib/colors/groups.unit.spec.ts**

```
import { getAccentColors } from "../groups";  
import { color } from "../palette";
```

```
describe("groups", () => {  
  describe("getAccentColors", () => {  
    it("should return main accent colors without gray...
```

### **frontend/src/metabase/lib/colors/index.ts**

```
export * from "../palette";  
export * from "../colors";
```

### **frontend/src/metabase/lib/colors/palette.ts**

```
import Color from "color";  
  
import type { ColorGetter } from "metabase/visualizations/types";  
  
import { colors } from "../colors";  
import type { ColorPalette } from "../types";  
  
export const...
```

### **frontend/src/metabase/lib/colors/palette.unit.spec.ts**

```
import { colors } from "../colors";  
import { color } from "../palette";  
  
describe("palette", () => {  
  it("should get a color from the palette", () => {  
    expect(color("brand")).toBeDefined();  
    ...
```

### **frontend/src/metabase/lib/colors/scales.ts**

```
import { scaleLinear, scaleQuantile } from "d3-scale";  
  
export const getColorScale = (  
  extent: [number, number],  
  colors: string[],  
  isQuantile: boolean = false,  
) => {
```

## Analyst Base

```
if (isQuantile) {
```

```
...
```

### **frontend/src/metabase/lib/colors/scales.unit.spec.ts**

```
import { color } from "../palette";
```

```
import { getColorScale } from "../scales";
```

```
describe("scales", () => {
```

```
  const colors = [color("bg-white"), color("bg-black")];
```

```
  it("should interpolate colors by...
```

### **frontend/src/metabase/lib/colors/types.ts**

```
import type { colors } from "../colors";
```

```
export type ColorPalette = Partial<Record<keyof typeof colors, string>>;
```

```
export type ColorName = keyof ColorPalette;
```

```
export interface AccentColorOptions {
```

```
...
```

### **frontend/src/metabase/lib/compat/check-version.ts**

```
import React from "react";
```

```
export const getMajorReactVersion = () => {
```

```
  const versionParts = React.version.split(".").map(Number);
```

```
  return versionParts[0];
```

```
};
```

### **frontend/src/metabase/lib/compat/check-version.unit.spec.ts**

```
import React from "react";
```

```
import { getMajorReactVersion } from "../check-version";
```

```
describe("getMajorReactVersion", () => {
```

```
  const versions = [
```

```
    { version: "0.14.0", expected: 0 },
```

```
    { ...
```

### **frontend/src/metabase/lib/compose-event-handlers.ts**

```
type HandlerType<E> = ((event: E) => void) | undefined;
```

## Analyst Base

```
export const composeEventHandlers = <E>(...handlers: HandlerType<E>[]) => {  
  return function handleEvent(event: E) {  
    ...  
  }  
}
```

### frontend/src/metabase/lib/constants.js

```
import { t } from "ttag";  
  
export const SEARCH_DEBOUNCE_DURATION = 300;  
  
export const DEFAULT_SEARCH_LIMIT = 50;  
  
// eslint-disable-next-line ttag/no-module-declaration -- see metabase#55045  
export...
```

### frontend/src/metabase/lib/cookies.js

METABASE\_SESSION\_COOKIE is only used for e2e tests. In normal usage cookie is set automatically by login endpoints

### frontend/src/metabase/lib/core.ts

```
import { t } from "ttag";  
  
import type { IconName } from "metabase/ui";  
import { TYPE } from "metabase-lib/v1/types/constants";  
import type { Field } from "metabase-types/api";  
  
interface...
```

### frontend/src/metabase/lib/cron.ts

```
import { isValidCronExpression } from "cron-expression-validator";  
import cronstrue from "cronstrue";  
import { t } from "ttag";  
import { memoize } from "underscore";  
  
import MetabaseSettings from...
```

### frontend/src/metabase/lib/csp.js

\*

### frontend/src/metabase/lib/dashboard\_grid.js

NOTE: If we make changes to the algorithm or default values below we should change [the backend version](<https://github.com/metabase/metabase/blob/master/src/metabase/dashboards/autoplace.clj>).  
If...

### frontend/src/metabase/lib/data\_grid.js

## Analyst Base

```
import _ from "underscore";
```

```
import * as Pivot from "cljs/metabase.pivot.js";  
import { formatValue } from "metabase/lib/formatting";  
import { makeCellBackgroundGetter } from...
```

### frontend/src/metabase/lib/date-time.ts

```
import dayjs from "dayjs";  
import { t } from "ttag";  
import _ from "underscore";
```

```
import type { DayOfWeekId } from "metabase-types/api";
```

```
// returns 0-6 where Sunday as 0 and Saturday as 6  
// Note:...
```

### frontend/src/metabase/lib/dayjs-parse-zone-plugin.js

This is the copy of <https://github.com/iamkun/dayjs/pull/2060> which implements parseZone which we rely on when using momentjs so we need this plugin to be able to migrate from momentjs

@authors:...

### frontend/src/metabase/lib/dayjs.ts

```
import dayjs from "dayjs";  
import advancedFormat from "dayjs/plugin/advancedFormat";  
import customParseFormat from "dayjs/plugin/customParseFormat";  
import dayOfYear from...
```

### frontend/src/metabase/lib/delay.ts

```
declare global {  
  interface Window {  
    // Set REMOVE_DELAYS to true in environments where we want to remove them.  
    // For example, in Storybook we want to remove delays to make Loki tests more  
    ...  
  }  
}
```

### frontend/src/metabase/lib/dom.js

```
import querystring from "querystring";  
import _ from "underscore";
```

```
import { isCypressActive, isStorybookActive } from "metabase/env";  
import MetabaseSettings from "metabase/lib/settings";
```

```
// IE...
```

### frontend/src/metabase/lib/email.ts

```
const EMAIL_REGEX =
```

## Analyst Base

```
/^((([<>()[]\.,;:\s@"]+(\.[<>()[]\.,;:\s@"]+)*)(\".+\"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\)|((([a-zA-Z\d-0-9]+\.)+[a-zA-Z]{2,})))$);
```

export function...

### frontend/src/metabase/lib/embed.js

```
import { push } from "react-router-redux";
import _ from "underscore";
```

```
import CS from "metabase/css/core/index.css";
import { IFRAMED_IN_SELF, isWithinIframe } from "metabase/lib/dom";
import {...
```

### frontend/src/metabase/lib/encoding.js

escaping before base64 encoding is necessary for non-ASCII characters  
<https://developer.mozilla.org/en-US/docs/Web/API/WindowBase64/btoa>

### frontend/src/metabase/lib/engine.ts

```
import type { Engine } from "metabase-types/api";
```

```
export function getEngineNativeType(engine?: string): "sql" | "json" {
  switch (engine) {
    case "mongo":
    case "druid":
      return "json";
  }
}
```

...

### frontend/src/metabase/lib/engine.unit.spec.ts

```
import {
  formatNativeQuery,
  getEngineNativeType,
  getNativeQueryLanguage,
  isDeprecatedEngine,
} from "metabase/lib/engine";
import type { Engine } from...
```

### frontend/src/metabase/lib/entities.js

```
/*
 * # Entities abstract the interface between the back-end and the front-end.
 *
 * ## Endpoint requirements for entities:
 *
 * When fetching a list, each item of the list must include an `id` ...
```



## Analyst Base

### **frontend/src/metabase/lib/entities.unit.spec.js**

```
import { combineReducers, configureStore } from "@reduxjs/toolkit";
import promise from "redux-promise";
```

```
import { combineEntities, createEntity } from "metabase/lib/entities";
import requestsReducer...
```

### **frontend/src/metabase/lib/entity-id/hooks/use-validated-entity-id.ts**

```
import { useMemo } from "react";
import _ from "underscore";
```

```
import { skipToken } from "metabase/api";
import { useTranslateEntityIdQuery } from "metabase/api/entity-id";
import { isBaseEntityID }...
```

### **frontend/src/metabase/lib/entity-id/types.ts**

```
import type {
  BaseEntityId,
  CardId,
  CollectionId,
  DashboardId,
} from "metabase-types/api";
```

```
export type SUPPORTED_ENTITIES = {
  dashboard: DashboardId;
  card: CardId;
  collection:...
```

### **frontend/src/metabase/lib/errors/console.js**

```
export const MAX_ERROR_LOGS = 20;
```

```
export function captureConsoleErrors() {
  console.errorBuffer = [];
```

```
  const originalError = console.error;
```

```
  console.error = function (...args) {
    if...
```

### **frontend/src/metabase/lib/errors/console.unit.spec.js**

```
import { MAX_ERROR_LOGS, captureConsoleErrors } from "./console";
```

```
describe("captureConsoleErrors", () => {
```

## Analyst Base

```
beforeEach(() => {  
  captureConsoleErrors();  
});
```

```
afterEach(() => {  
  ...
```

### **frontend/src/metabase/lib/errors/index.ts**

```
export * from "./messages";  
export * from "./validation";  
export * from "./server-error-types";  
export * from "./console";
```

```
export type { GenericErrorResponse } from "./types";
```

### **frontend/src/metabase/lib/errors/messages.ts**

```
import type { GenericErrorResponse } from "./types";
```

```
export function getResponseErrorMessage(error: unknown): string | undefined {  
  const response = error as GenericErrorResponse | undefined;
```

```
  if...
```

### **frontend/src/metabase/lib/errors/messages.unit.spec.ts**

```
import { merge } from "icepick";
```

```
import { getResponseErrorMessage } from "./messages";
```

```
const ERROR_DATA_MESSAGE = {  
  data: {  
    message: "Error from error.data.message",  
  },  
};
```

```
const...
```

### **frontend/src/metabase/lib/errors/server-error-types.ts**

```
export const SERVER_ERROR_TYPES = {  
  missingPermissions: "missing-required-permissions",  
};
```

### **frontend/src/metabase/lib/errors/types.ts**

```
export type GenericErrorResponse = {  
  data?:
```

## Analyst Base

```
| {  
  message?: string;  
  errors?: Record<string, string>;  
}  
| string;  
errors?: Record<string, string>;  
message?:...
```

### frontend/src/metabase/lib/errors/validation.ts

```
import { c, t } from "ttag";  
  
import type { LengthParams, MaxLengthParams } from "../types";  
  
export const required = () => t`required`;  
  
export const email = () => t`must be a valid email...
```

### frontend/src/metabase/lib/fetchWithTimeout.ts

```
const DEFAULT_TIMEOUT = 7000;  
  
/* wrapper around fetch that allows passing a timeout prop */  
export function fetchWithTimeout(  
  url: string,  
  options: RequestInit & { timeout?: number } = {},  
) {  
  ...
```

### frontend/src/metabase/lib/formatting/colors.ts

```
import { color } from "metabase/lib/colors";  
  
export function assignUserColors(  
  userIds: string[],  
  currentUserId: string,  
  colors = [  
    color("brand"),  
    color("accent2"),  
  ]  
  ...
```

### frontend/src/metabase/lib/formatting/colors.unit.spec.js

```
import { color } from "metabase/lib/colors";  
  
import { assignUserColors } from "../colors";  
  
describe("lib/formatting/colors", () => {
```

## Analyst Base

```
it("should assign colors to users when currentUserId is...
```

### **frontend/src/metabase/lib/formatting/column.ts**

```
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";
import type { DatasetColumn } from "metabase-types/api/dataset";

export function displayNameForColumn(column: DatasetColumn): string {
  ...
}
```

### **frontend/src/metabase/lib/formatting/constants.ts**

```
export const FK_SYMBOL = "";
```

### **frontend/src/metabase/lib/formatting/currency.ts**

```
import { t } from "tag";

import { currency } from "cljs/metabase.util.currency";

export interface CurrencyInfo {
  symbol: string;
  name: string;
  symbol_native: string;
  decimal_digits:...
```

### **frontend/src/metabase/lib/formatting/currency.unit.spec.ts**

```
import { getCurrencyStyleOptions } from "./currency";

describe("getCurrencyStyleOptions", () => {
  it("should get currency options - USD", () => {
    const options =...
```

### **frontend/src/metabase/lib/formatting/datetime-utils.ts**

```
import type { DatetimeUnit } from "metabase-types/api/query";

export const DEFAULT_TIME_STYLE = "h:mm A";
export const DEFAULT_DATE_STYLE = "MMMM D, YYYY";

const UNITS_WITH_HOUR = ["default",...
```

### **frontend/src/metabase/lib/formatting/field.ts**

```
import type { Field } from "metabase-types/api/field";

export function formatField(field: Field) {
  if (!field) {
    return "";
  }
}
```

## Analyst Base

```
return field.dimensions?.[0]?.name || field.display_name ||...
```

### **frontend/src/metabase/lib/formatting/geography.ts**

```
import * as d3 from "d3";
```

```
import { decimalCount } from "metabase/visualizations/lib/numeric";
```

```
import { isLatitude, isLongitude } from "metabase-lib/v1/types/utils/isa";
```

```
import type { OptionsType }...
```

### **frontend/src/metabase/lib/formatting/link.ts**

```
import { isSameOrSiteUrlOrigin } from "metabase/lib/dom";
```

```
import {
```

```
  formatValue,
```

```
  getUrlProtocol,
```

```
  isDefaultLinkProtocol,
```

```
} from "metabase/lib/formatting";
```

```
import { isDate } from...
```

### **frontend/src/metabase/lib/formatting/nullable.ts**

```
import { NULL_DISPLAY_VALUE } from "../constants";
```

```
export function formatNullable<T>(value: T | null | undefined) {
```

```
  return value ?? NULL_DISPLAY_VALUE;
```

```
}
```

### **frontend/src/metabase/lib/formatting/numbers.unit.spec.ts**

```
import { formatNumber, numberFormatterForOptions } from "./numbers";
```

```
describe("formatNumber", () => {
```

```
  it("should respect the decimals setting even when compact is true (metabase#54063)", () => {
```

```
    ...
```

### **frontend/src/metabase/lib/formatting/round-float.unit.spec.ts**

```
import { roundFloat } from "./numbers";
```

```
describe("roundFloat", () => {
```

```
  it.each([
```

```
    // Rounding defaults to 2 digits, defined in DEFAULT_NUMBER_OPTIONS
```

```
    [30, undefined, 30],
```

```
    [1.2345,...
```

### **frontend/src/metabase/lib/formatting/strings.ts**

```
import inflection from "inflection";
```

## Analyst Base

```
export function singularize(str: string, singular?: string) {  
  return inflection.singularize(str, singular);  
}
```

```
export function pluralize(str: string, plural?:...
```

### **frontend/src/metabase/lib/formatting/time.ts**

```
import type { Moment } from "moment-timezone"; // eslint-disable-line no-restricted-imports -- deprecated  
usage  
import { msgid, ngettext } from "ttag";
```

```
import type { TimeOnlyOptions } from...
```

### **frontend/src/metabase/lib/formatting/types.ts**

```
export interface TimeOnlyOptions {  
  local?: boolean;  
  time_enabled?: "minutes" | "milliseconds" | "seconds" | null;  
  time_format?: string;  
  time_style?: string;  
}
```

```
export interface OptionsType...
```

### **frontend/src/metabase/lib/formatting.js**

```
export * from "./formatting/colors";  
export * from "./formatting/column";  
export * from "./formatting/currency";  
export * from "./formatting/date";  
export * from "./formatting/email";  
export * from...
```

### **frontend/src/metabase/lib/groups.ts**

```
import { t } from "ttag";
```

```
import { color } from "metabase/lib/colors";  
import type { Group } from "metabase-types/api";
```

```
const SPECIAL_GROUP_NAMES = new Map([  
  // eslint-disable-next-line...
```

### **frontend/src/metabase/lib/i18n-debug.js**

```
import { HAS_LOCAL_STORAGE } from "metabase/lib/dom";
```

## Analyst Base

```
// If enabled this monkeypatches `t` and `jt` to return blacked out
// strings/elements to assist in finding untranslated strings.
//
//...
```

### frontend/src/metabase/lib/i18n.js

```
import dayjs from "dayjs";
import moment from "moment-timezone"; // eslint-disable-line no-restricted-imports -- deprecated usage
import { addLocale, useLocale } from "ttag";

import { isEmbeddingSdk...
```

### frontend/src/metabase/lib/i18n.unit.spec.ts

```
import moment from "moment-timezone"; // eslint-disable-line no-restricted-imports -- deprecated usage

import { setLocalization } from "./i18n";

function setup(language: string) {
  ...
}
```

### frontend/src/metabase/lib/icon.ts

```
import { PERSONAL_COLLECTIONS } from "metabase/entities/collections/constants";
import { PLUGIN_COLLECTIONS } from "metabase/plugins";
import type { IconName } from "metabase/ui";
import {...
```

### frontend/src/metabase/lib/icon.unit.spec.ts

```
import register from "metabase/visualizations/register";

import { getIcon } from "./icon";

describe("getIcon", () => {
  beforeAll(() => {
    // visualizations are weird and we have to register...
  })
})
```

### frontend/src/metabase/lib/json-parse.ts

```
export const safeJsonParse = (value: string | null | undefined) => {
  if (!value) {
    return null;
  }

  try {
    return JSON.parse(value);
  } catch (e) {
```

## Analyst Base

```
console.error("Unable to parse...
```

### **frontend/src/metabase/lib/keyboard.js**

```
export const KEYCODE_SPACE = 0;
export const KEYCODE_BACKSPACE = 8;
export const KEYCODE_TAB = 9;
export const KEYCODE_ENTER = 13;
export const KEYCODE_ESCAPE = 27;
```

```
export const KEYCODE_LEFT =...
```

### **frontend/src/metabase/lib/loki-utils.ts**

```
export const openImageBlobOnStorybook = ({
  canvas,
  blob,
}): {
  canvas: HTMLCanvasElement;
  blob: Blob;
}) => {
  const imgElement = document.createElement("img");
  imgElement.src =...
```

### **frontend/src/metabase/lib/measure-text.ts**

```
import _ from "underscore";

import type {
  FontStyle,
  TextMeasurer,
} from "metabase/visualizations/shared/types/measure-text";
```

```
let canvas: HTMLCanvasElement | null = null;
```

```
export const...
```

### **frontend/src/metabase/lib/name.ts**

```
type ItemWithName = {
  name: string;
  display_name?: string;
};
```

```
/** items may or may not have display names, but we should prefer displaying them when they do */
```

```
export const getName = (item:...
```

### **frontend/src/metabase/lib/noop.js**



## Analyst Base

there's nothing here!

This file is the alternative to importing EE plugins.

Either way we import something to ensure a consistent import order.

### **frontend/src/metabase/lib/notifications.ts**

```
import { c, msgid, ngettext, t } from "ttag";
import _ from "underscore";
```

```
import type { NotificationListItem } from "metabase/account/notifications/types";
import { cronToScheduleSettings } from...
```

### **frontend/src/metabase/lib/number.ts**

```
const INTEGER_REGEX = /^[+-]?\d+$/;
```

```
export type NumberValue = number | bigint;
```

```
export function parseNumber(value: string): NumberValue | null {
  const number = parseFloat(value);
  if...
```

### **frontend/src/metabase/lib/number.unit.spec.ts**

```
import { parseNumber, parseNumberValue } from "../number";
```

```
describe("metabase/lib/number", () => {
  describe("parseNumberValue", () => {
    it("should return null for non-number values", () => {
      ...
```

### **frontend/src/metabase/lib/objects.ts**

```
export const getObjectEntries = <K extends string, V>(
  obj: Record<K, V>,
): [K, V][] => {
  return Object.entries(obj) as [K, V][];
};
```

```
export const getObjectKeys = <K extends string>(
  obj:...
```

### **frontend/src/metabase/lib/promise.ts**

```
import { noop } from "underscore";
```

```
type CancellablePromise<T> = Promise<T> & {
  cancel: () => void;
};
```

## Analyst Base

```
// return a promise wrapping the provided one but with a "cancel" method  
export function...
```

### **frontend/src/metabase/lib/pulse.ts**

```
import { msgid, ngettext, t } from "ttag";  
import _ from "underscore";
```

```
import { getEmailDomain } from "metabase/lib/email";  
import { formatDateTimeWithUnit } from...
```

### **frontend/src/metabase/lib/react-compatible.ts**

Support React 17 backwards compatibility for the Embedding SDK

### **frontend/src/metabase/lib/redux/hooks.ts**

```
import type { AnyAction, Store, ThunkDispatch } from "@reduxjs/toolkit";  
import type { TypedUseSelectorHook } from "react-redux";  
import {  
  createDispatchHook,  
  createSelectorHook,  
  ...
```

### **frontend/src/metabase/lib/redux/index.ts**

```
export * from "./utils";  
export * from "./typed-utils";  
export * from "./hooks";  
export * from "./custom-context";
```

### **frontend/src/metabase/lib/redux/typed-utils.ts**

```
import type { ThunkDispatch } from "@reduxjs/toolkit";  
import { createAsyncThunk as createAsyncThunkOriginal } from "@reduxjs/toolkit";
```

```
import type { GetState, State } from...
```

### **frontend/src/metabase/lib/redux/utils.js**

```
import { compose } from "@reduxjs/toolkit";  
import { getIn } from "icepick";  
import { normalize } from "normalizr";  
import _ from "underscore";
```

```
import { delay } from "metabase/lib/promise";  
import...
```

### **frontend/src/metabase/lib/redux/utils.unit.spec.js**

```
import { delay } from "__support__/utils";
```

## Analyst Base

```
import { fetchData, mergeEntities, updateData } from "./utils";
```

```
describe("Metadata", () => {  
  const getDefaultArgs = ({  
    existingData = "data",  
    ...  

```

### **frontend/src/metabase/lib/resize-observer.ts**

```
import { ResizeObserver as JuggleResizeObserver } from "@juggle/resize-observer";
```

```
type ResizeObserverCallback = (  
  entry: ResizeObserverEntry,  
  observer: ResizeObserver,  
) => void;
```

```
// PR:...
```

### **frontend/src/metabase/lib/schema/index.ts**

```
export * from "./schema";
```

### **frontend/src/metabase/lib/schema/schema.js**

backend returns model = "card" instead of "question"

### **frontend/src/metabase/lib/schema/schema.unit.spec.js**

```
import { entityTypeForModel, entityTypeForObject } from "./schema";
```

```
describe("schemas", () => {  
  const MODEL_ENTITY_TYPE = [  
    { model: "card", entityType: "questions" },  
    { model: "dataset", ...  

```

### **frontend/src/metabase/lib/schema\_metadata.js**

```
import { FIELD_SEMANTIC_TYPES_MAP } from "metabase/lib/core";
```

```
export function foreignKeyCountsByOriginTable(fks) {  
  if (fks === null || !Array.isArray(fks)) {  
    return null;  
  }  

```

```
  return fks  
  ...  

```

### **frontend/src/metabase/lib/security.js**

```
import passwordGenerator from "password-generator";
```

## Analyst Base

```
import MetabaseSettings from "metabase/lib/settings";
```

```
// generate a password that satisfies `complexity` requirements, by default the ones that...
```

### **frontend/src/metabase/lib/security.unit.spec.js**

```
import MetabaseSettings from "metabase/lib/settings";
```

```
import { generatePassword } from "../security";
```

```
describe("generatePassword", () => {  
  it("defaults to at least 14 characters even if...
```

### **frontend/src/metabase/lib/settings.ts**

```
import { msgid, ngettext, t } from "ttag";
```

```
import _ from "underscore";
```

```
import { numberToWord } from "metabase/lib/utils";
```

```
import type {
```

```
  PasswordComplexity,
```

```
  SettingKey,
```

```
  Settings,
```

```
} from...
```

### **frontend/src/metabase/lib/string.js**

```
import _ from "underscore";
```

```
// Creates a regex that will find an order dependent, case insensitive substring. All whitespace will be  
rendered as "." in the regex, to create a fuzzy search.
```

```
export...
```

### **frontend/src/metabase/lib/syncing.js**

```
export const isSyncInProgress = (entity) => {  
  return entity.initial_sync_status === "incomplete";  
};
```

```
export const isSyncCompleted = (entity) => {  
  return entity.initial_sync_status ===...
```

### **frontend/src/metabase/lib/time-dayjs.ts**

```
import type { Dayjs } from "dayjs";
```

```
import dayjs from "dayjs";
```

```
import { t } from "ttag";
```

## Analyst Base

```
import type { DatetimeUnit } from "metabase-types/api/query";
```

```
const DAYLIGHT_SAVINGS_CHANGE_TOLERANCE:...
```

### **frontend/src/metabase/lib/time-dayjs.unit.spec.ts**

```
import dayjs from "dayjs";
```

```
import { parseTimestamp } from "metabase/lib/time-dayjs";
```

```
describe("parseTimestamp", () => {  
  afterEach(() => {  
    dayjs.updateLocale(dayjs.locale(), { weekStart: 0...
```

### **frontend/src/metabase/lib/time.ts**

```
import moment from "moment-timezone"; // eslint-disable-line no-restricted-imports -- deprecated usage  
import { t } from "ttag";
```

```
import MetabaseSettings from "metabase/lib/settings";  
import type {...
```

### **frontend/src/metabase/lib/time.unit.spec.ts**

```
import moment from "moment-timezone"; // eslint-disable-line no-restricted-imports -- deprecated usage
```

```
import { parseTime } from "../time";
```

```
describe("parseTime", () => {  
  it("should return a moment...
```

### **frontend/src/metabase/lib/timelines.ts**

```
import { t } from "ttag";  
import _ from "underscore";
```

```
import { canonicalCollectionId } from "metabase/collections/utils";  
import type { IconName } from "metabase/ui";  
import type { Collection,...
```

### **frontend/src/metabase/lib/types.ts**

```
import _ from "underscore";
```

```
export const isNotNull = <T>(value: T | null | undefined): value is T => {  
  return value != null;  
};
```

```
export const isNotFalsy = <T>(
```

## Analyst Base

value: T | null | undefined |...

### frontend/src/metabase/lib/types.unit.spec.ts

```
import { removeNullAndUndefinedValues } from "../types";
```

```
describe("removeNullAndUndefinedValues", () => {  
  it("removes nil values from an object", () => {  
    const obj = {  
      a: 1,  
      b:...
```

### frontend/src/metabase/lib/uploads.ts

```
import type { FileUpload } from "metabase-types/store/upload";
```

```
export const isUploadInProgress = (upload: FileUpload) =>  
  upload.status === "in-progress";
```

```
export const isUploadCompleted = (upload:...
```

### frontend/src/metabase/lib/urls/actions.ts

```
import type { CardId, WritebackActionId } from "metabase-types/api";
```

```
import { modelDetail } from "../models";
```

```
type ParentModelProps = {  
  id: CardId;  
  name?: string;  
};
```

```
export function...
```

### frontend/src/metabase/lib/urls/admin.ts

```
import type { DatabaseId, UserId } from "metabase-types/api";
```

```
export function newUser() {  
  return `/admin/people/new`;  
}
```

```
export function editUser(userId: UserId) {  
  return...
```

### frontend/src/metabase/lib/urls/auth.ts

```
export const login = (redirectUrl?: string) => {  
  return redirectUrl
```

## Analyst Base

```
? `/auth/login?redirect=${encodeURIComponent(redirectUrl)}`  
: "/auth/login";  
};
```

```
export const password = (redirectUrl?:...
```

### **frontend/src/metabase/lib/urls/bookmarks.ts**

```
import slugg from "slugg";
```

```
import type { Bookmark } from "metabase-types/api";
```

```
import { appendSlug } from "./utils";
```

```
function getBookmarkBasePath(bookmark: Pick<Bookmark, "type" | "card_type">) {  
  ...
```

### **frontend/src/metabase/lib/urls/browse.ts**

```
import slugg from "slugg";
```

```
import type DatabaseV1 from "metabase-lib/v1/metadata/Database";
```

```
import type Table from "metabase-lib/v1/metadata/Table";
```

```
import { SAVED_QUESTIONS_VIRTUAL_DB_ID } from...
```

### **frontend/src/metabase/lib/urls/collections.ts**

```
import slugg from "slugg";
```

```
import {  
  isRootPersonalCollection,  
  isRootTrashCollection,  
} from "metabase/collections/utils";
```

```
import type {  
  Collection as BaseCollection,  
  CollectionId,  
} from...
```

### **frontend/src/metabase/lib/urls/dashboards.ts**

```
import slugg from "slugg";
```

```
import { stringifyHashOptions } from "metabase/lib/browser";
```

```
import MetabaseSettings from "metabase/lib/settings";
```

```
import type {  
  DashCardId,  
  DashboardId,  
  ...
```

### **frontend/src/metabase/lib/urls/index.ts**

```
export * from "./actions";
export * from "./admin";
export * from "./auth";
export * from "./bookmarks";
export * from "./browse";
export * from "./collections";
export * from "./dashboards";
export...
```

### **frontend/src/metabase/lib/urls/indexed-entities.ts**

```
import slugg from "slugg";

import type { IndexedEntity } from "metabase-types/api";
```

```
export const indexedEntity = (entity: IndexedEntity) =>
  ...
```

### **frontend/src/metabase/lib/urls/misc.ts**

```
import type {
  ExportFormat,
  TableExportFormat,
} from "metabase/common/types/export";

export const exportFormats: TableExportFormat[] = ["csv", "xlsx", "json"];
export const exportFormatPng:...
```

### **frontend/src/metabase/lib/urls/modelToUrl.ts**

```
import { collection } from "./collections";
import { dashboard } from "./dashboards";
import { metric, model } from "./models";
import { question, tableRowsQuery } from "./questions";

type...
```

### **frontend/src/metabase/lib/urls/modelToUrl.unit.spec.ts**

```
import { modelToUrl } from "./modelToUrl";

describe("urls > modelToUrl", () => {
  it("should return null for unknown model", () => {
    expect(
      // @ts-expect-error - testing the error case
    ...
```



### **frontend/src/metabase/lib/urls/models.ts**

```
import slugg from "slugg";

import type { Card } from "metabase-types/api";

import type { QuestionUrlBuilderParams } from "../questions";
import { question } from "../questions";
import { appendSlug }...
```

### **frontend/src/metabase/lib/urls/pulses.ts**

```
export function pulse(pulseId: number) {
  return `/pulse/${pulseId}`;
}

export function pulseEdit(pulseId: number) {
  return `/pulse/${pulseId}`;
}
```

### **frontend/src/metabase/lib/urls/questions.ts**

```
import slugg from "slugg";

import { serializeCardForUrl } from "metabase/lib/card";
import MetabaseSettings from "metabase/lib/settings";
import type { QuestionCreatorOpts } from...
```

### **frontend/src/metabase/lib/urls/timelines.ts**

```
import type { Collection, Timeline, TimelineEvent } from "metabase-types/api";

import { collection as getCollectionUrl } from "../collections";

export function timelinesInCollection(collection?:...
```

### **frontend/src/metabase/lib/urls/urls.ts**

```
import api from "metabase/lib/api";

export function appendSlug(path: string | number, slug?: string) {
  return slug ? `${path}-${slug}` : String(path);
}

export function extractEntityId(slug = "")...
```

### **frontend/src/metabase/lib/urls/urls.unit.spec.ts**

```
import api from "metabase/lib/api";
```

## Analyst Base

```
import { getSubpathSafeUrl, openInNewTab } from "./utils";
```

```
const fakeBasename = "foobar";  
const originalBasename = api.basename;
```

```
const mockWindowOpen =...
```

### frontend/src/metabase/lib/user.ts

```
export function getFullName(user: NamedUser): string | null {  
  const firstName = user.first_name?.trim() || "";  
  const lastName = user.last_name?.trim() || "";  
  return [firstName, lastName].join("...");  
}
```

### frontend/src/metabase/lib/utils.ts

```
import { t } from "ttag";  
import _ from "underscore";
```

```
import { PLUGIN_IS_EE_BUILD } from "metabase/plugins";
```

```
export function isEmpty(str: string | null) {  
  if (str != null) {  
    str =...  
  }  
}
```

### frontend/src/metabase/lib/uuid.ts

```
export function uuid() {  
  return (  
    s4() +  
    s4() +  
    "-" +  
    s4() +  
    "-" +  
    s4() +  
    "-" +  
    s4() +  
    "-" +  
    s4() +  
    s4()  
  );  
}
```

```
function s4() {  
  return...  
}
```

### **frontend/src/metabase/lib/validate.js**

```
import { t } from "ttag";

import { isEmail } from "metabase/lib/email";
import Settings from "metabase/lib/settings";
```

```
// we need this to allow 0 as a valid form value
export const isEmpty = (value)...
```

### **frontend/src/metabase/metabot/context.ts**

```
import { type RefObject, useContext, useEffect, useMemo } from "react";

import { PLUGIN_METABOT } from "metabase/plugins";
import type { MetabotChatContext } from "metabase-types/api";
import type { ...
```

### **frontend/src/metabase/metabot/index.ts**

```
export * from "./context";
```

### **frontend/src/metabase/metabot-v3/selectors.ts**

```
import { createSelector } from "@reduxjs/toolkit";
import _ from "underscore";
```

```
import {
  getRawSeries,
  getTransformedSeries,
  getVisualizationSettings,
} from...
```

### **frontend/src/metabase/metadata/eslintrc**

```
{
  "rules": {
    // Note: adding this rule to a eslint config file in a subfolder will remove
    // *not* carry over the restricted imports from parent folders, you will
    // need to copy them...
```

### **frontend/src/metabase/metadata/README.md**

Metadata

### **frontend/src/metabase/metadata/components/CoercionStrategyPicker/index.ts**

```
export * from "./CoercionStrategyPicker";
```

### **frontend/src/metabase/metadata/components/CoercionStrategyPicker/utils.ts**

```
import { c, t } from "ttag";
```

## Analyst Base

```
const GET_LEFT_TERM_CONVERSIONS = (): Record<string, string> => ({
  ISO8601: t`ISO 8601`,
  UNIXSeconds: t`UNIX seconds`,
  UNIXMilliseconds: t`UNIX milliseconds`,
  ...
})
```

### **frontend/src/metabase/metadata/components/CoercionStrategyPicker/units.spec.ts**

```
import { humanizeCoercionStrategy } from "../utils";
```

```
describe("humanizeCoercionStrategy", () => {
  it("does not convert `Don't cast`", () => {
    const original = "Don't cast";
    const humanized...
```

### **frontend/src/metabase/metadata/components/CurrencyPicker/index.ts**

```
export * from "../CurrencyPicker";
```

### **frontend/src/metabase/metadata/components/DiscardFieldValuesButton/index.ts**

```
export * from "../DiscardFieldValuesButton";
```

### **frontend/src/metabase/metadata/components/DiscardTableFieldValuesButton/index.ts**

```
export * from "../DiscardTableFieldValuesButton";
```

### **frontend/src/metabase/metadata/components/FieldOrderPicker/index.ts**

```
export * from "../FieldOrderPicker";
```

### **frontend/src/metabase/metadata/components/FieldValuesTypePicker/index.ts**

```
export * from "../FieldValuesTypePicker";
```

### **frontend/src/metabase/metadata/components/FieldVisibilityPicker/index.ts**

```
export * from "../FieldVisibilityPicker";
```

### **frontend/src/metabase/metadata/components/FkTargetPicker/index.ts**

```
export * from "../FkTargetPicker";
```

### **frontend/src/metabase/metadata/components/NameDescriptionInput/index.ts**

```
export * from "../NameDescriptionInput";
```

### **frontend/src/metabase/metadata/components/RescanFieldButton/index.ts**

```
export * from "../RescanFieldButton";
```

### **frontend/src/metabase/metadata/components/RescanTableFieldsButton/index.ts**

**S**

```
export * from "./RescanTableFieldsButton";
```

**frontend/src/metabase/metadata/components/SemanticTypeAndTargetPicker/index.ts**

```
export { SemanticTypeAndTargetPicker } from "./SemanticTypeAndTargetPicker";
```

**frontend/src/metabase/metadata/components/SemanticTypePicker/index.ts**

```
export * from "./SemanticTypePicker";
```

**frontend/src/metabase/metadata/components/SemanticTypePicker/utils.ts**

```
import { FIELD_SEMANTIC_TYPES } from "metabase/lib/core";
```

```
import { LEVEL_ONE_TYPES, TYPE } from "metabase-lib/v1/types/constants";
```

```
import { isTypeFK, isTypePK, isa } from...
```

**frontend/src/metabase/metadata/components/SortableFieldItem/index.ts**

```
export * from "./SortableFieldItem";
```

**frontend/src/metabase/metadata/components/SortableFieldList/index.ts**

```
export * from "./SortableFieldList";
```

**frontend/src/metabase/metadata/components/SyncTableSchemaButton/index.ts**

```
export * from "./SyncTableSchemaButton";
```

**frontend/src/metabase/metadata/components/TableBreadcrumbs/index.ts**

```
export * from "./TableBreadcrumbs";
```

**frontend/src/metabase/metadata/components/UnfoldJsonPicker/index.ts**

```
export * from "./UnfoldJsonPicker";
```

**frontend/src/metabase/metadata/components/index.ts**

```
export * from "./CoercionStrategyPicker";
```

```
export * from "./CurrencyPicker";
```

```
export * from "./DiscardFieldValuesButton";
```

```
export * from "./DiscardTableFieldValuesButton";
```

```
export * from...
```

**frontend/src/metabase/metadata/hooks/index.ts**

```
export * from "./useMetadataToasts";
```

**frontend/src/metabase/metadata/hooks/useMetadataToasts.ts**

```
import { useCallback } from "react";
```

```
import { t } from "ttag";
```

```
import { useToast } from "metabase/common/hooks";
```

## Analyst Base

```
export const useMetadataToasts = () => {  
  const [sendToast] = useToast();
```

```
  const...
```

**frontend/src/metabase/metadata/pages/DataModel/components/FieldSection/BehaviorSection/RemappingPicker/CustomMappingModal/index.ts**

```
export * from "../CustomMappingModal";  
export * from "../types";
```

**frontend/src/metabase/metadata/pages/DataModel/components/FieldSection/BehaviorSection/RemappingPicker/CustomMappingModal/types.ts**

```
export type Mapping = Map<number, string>;
```

```
export interface ChangeOptions {  
  isAutomatic?: boolean;  
}
```

**frontend/src/metabase/metadata/pages/DataModel/components/FieldSection/BehaviorSection/RemappingPicker/CustomMappingModal/utils.ts**

```
import type { Mapping } from "../types";
```

```
export function areMappingsEqual(a: Mapping, b: Mapping): boolean {  
  return (  
    a.size === b.size && [...a].every(([key, value]) => b.get(key) === value)  
    ...  
  )  
}
```

**frontend/src/metabase/metadata/pages/DataModel/components/FieldSection/BehaviorSection/RemappingPicker/DisplayValuesPicker/index.ts**

```
export * from "../DisplayValuesPicker";  
export * from "../types";
```

**frontend/src/metabase/metadata/pages/DataModel/components/FieldSection/BehaviorSection/RemappingPicker/DisplayValuesPicker/types.ts**

```
export type RemappingValue = "original" | "foreign" | "custom";
```

**frontend/src/metabase/metadata/pages/DataModel/components/FieldSection/BehaviorSection/RemappingPicker/index.ts**

```
export * from "../RemappingPicker";
```

**frontend/src/metabase/metadata/pages/DataModel/components/FieldSection/BehaviorSection/RemappingPicker/utils.ts**

```
import { t } from "ttag";  
import _ from "underscore";
```

## Analyst Base

```
import { getColumnIcon } from "metabase/common/utils/columns";
import { getRawTableFieldId } from "metabase/metadata/utils/field";
import * as...
```

### **frontend/src/metabase/metadata/pages/DataModel/components/FieldSection/BehaviorSection/index.ts**

```
export * from "./BehaviorSection";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/FieldSection/DataSection/index.ts**

```
export * from "./DataSection";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/FieldSection/hooks.ts**

```
import { useElementSize } from "@mantine/hooks";
import { useState } from "react";
```

```
const BUTTONS_GAP = 16;
```

```
export const useResponsiveButtons = () => {
  const { ref: buttonsContainerRef, width:...
```

### **frontend/src/metabase/metadata/pages/DataModel/components/FieldSection/index.ts**

```
export * from "./FieldSection";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/PreviewSection/Error/index.ts**

```
export * from "./Error";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/PreviewSection/index.ts**

```
export * from "./PreviewSection";
export * from "./types";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/PreviewSection/types.ts**

```
export type PreviewType = "table" | "detail" | "filtering";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/PreviewSection/utils.ts**

```
import { t } from "ttag";
```

## Analyst Base

```
import type { Dataset } from "metabase-types/api";
import { isObject } from "metabase-types/guards";
```

```
export function getPreviewTypeData() {
  return [
    { label:...
```

### **frontend/src/metabase/metadata/pages/DataModel/components/SegmentsLink/index.ts**

```
export * from "./SegmentsLink";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/TablePicker/index.ts**

```
export * from "./TablePicker";
export * from "./wrappers";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/TablePicker/types.ts**

```
import type {
  Databaseld,
  SchemaName,
  Table,
  TableId,
} from "metabase-types/api";
```

```
export type NodeKey = string;
```

```
export type TreePath = {
  databaseld?: Databaseld;
  schemaName?:...
```

### **frontend/src/metabase/metadata/pages/DataModel/components/TablePicker/utils.ts**

```
import { useCallback, useEffect, useMemo, useState } from "react";
import { useDeepCompareEffect, useLatest } from "react-use";
import _ from "underscore";
```

```
import {
  skipToken,
  ...
```

### **frontend/src/metabase/metadata/pages/DataModel/components/TableSection/FieldList/FieldItem/index.ts**



## Analyst Base

```
export * from "./FieldItem";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/TableSection/FieldList/index.ts**

```
export * from "./FieldList";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/TableSection/hooks.ts**

```
import { useElementSize } from "@mantine/hooks";  
import { useState } from "react";
```

```
const BUTTONS_GAP = 16;  
const LOADER_WIDTH = 16;  
const FIELD_ORDER_PICKER_WIDTH = 136;
```

```
export const...
```

### **frontend/src/metabase/metadata/pages/DataModel/components/TableSection/index.ts**

```
export * from "./TableSection";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/TitledSection/index.ts**

```
export * from "./TitledSection";
```

### **frontend/src/metabase/metadata/pages/DataModel/components/index.ts**

```
export * from "./FieldSection";  
export * from "./FieldValuesModal";  
export * from "./NoDatabasesEmptyState";  
export * from "./PreviewSection";  
export * from "./ResponsiveButton";  
export * from...
```

### **frontend/src/metabase/metadata/pages/DataModel/constants.ts**

```
import type { Column, ColumnSizeConfig } from "./types";
```

```
const PREVIEW_COLUMN_PADDING = 2 * 32;
```

```
export const COLUMN_CONFIG: Record<Column, ColumnSizeConfig> = {  
  nav: {  
    flex: "6 1 0",  
    min:...
```

### **frontend/src/metabase/metadata/pages/DataModel/index.ts**

## Analyst Base

```
export * from "../DataModel";
```

### **frontend/src/metabase/metadata/pages/DataModel/types.ts**

```
import type {
  DatabaseId,
  FieldId,
  SchemaId,
  SchemaName,
  TableId,
} from "metabase-types/api";

export type RouteParams = {
  databaseId?: string;
  fieldId?: string;
  schemaId?: ...
```

### **frontend/src/metabase/metadata/pages/DataModel/utils.ts**

```
import { skipToken } from "metabase/api";
import * as Urls from "metabase/lib/urls";
import { PLUGIN_FEATURE_LEVEL_PERMISSIONS } from "metabase/plugins";
// eslint-disable-next-line...
```

### **frontend/src/metabase/metadata/pages/DataModel/utils.unit.spec.ts**

```
import type { ParsedRouteParams, RouteParams } from "../types";
import { getUrl, parseRouteParams } from "../utils";
```

```
describe("parseRouteParams", () => {
  it("should parse all route parameters...
```

### **frontend/src/metabase/metadata/utils/database.ts**

```
import type { Database, DatabaseFeature } from "metabase-types/api";
```

```
export function hasDatabaseFeature(
  database: Database,
  feature: DatabaseFeature | string | null | undefined,
): boolean {
  ...
```

### **frontend/src/metabase/metadata/utils/database.unit.spec.ts**

```
import { createMockDatabase } from "metabase-types/api/mocks";
```

```
import { hasDatabaseFeature } from "../database";
```

```
describe("hasDatabaseFeature", () => {
```

## Analyst Base

```
it("returns true when feature is null", ()...
```

### **frontend/src/metabase/metadata/utils/field.ts**

```
import { is_coerceable } from "cljs/metabase.types.core";
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";
import { getGlobalSettingsForColumn } from...
```

### **frontend/src/metabase/metadata/utils/field.unit.spec.ts**

```
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";
import MetabaseSettings from "metabase/lib/settings";
import {
  createMockDatabase,
  createMockField,
  createMockTable,
} from...
```

### **frontend/src/metabase/metadata/utils/schema.ts**

```
import { humanize, titleize } from "metabase/lib/formatting";
import type { SchemaName } from "metabase-types/api";

export function getSchemaDisplayName(schema: SchemaName): string {
  return...
```

### **frontend/src/metabase/metadata/utils/schema.unit.spec.ts**

```
import { getSchemaDisplayName } from "../schema";

describe("getSchemaDisplayName", () => {
  it("should handle empty string", () => {
    expect(getSchemaDisplayName("")).toBe("");
  });
});
```

...

### **frontend/src/metabase/models/components/ModelActions/ModelActionDetails/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/models/components/ModelActions/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/models/containers/FormModelPicker/index.ts**

```
export * from "../FormModelPicker";
```

### **frontend/src/metabase/models/containers/NewModelOptions/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/nav/components/AdminNavbar/index.ts**

```
export * from "./AdminNavbar";
```

## **frontend/src/metabase/nav/components/AppBar/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

## **frontend/src/metabase/nav/components/CollectionBreadcrumbs/utils.ts**

```
import { isRootCollection } from "metabase/collections/utils";  
import type { Collection, CollectionId } from "metabase-types/api";
```

```
type GetCollectionListProps = {  
  collection: Collection;  
  ...
```

## **frontend/src/metabase/nav/components/CollectionBreadcrumbs/utils.unit.spec.ts**

```
import type { CollectionEssentials, CollectionId } from "metabase-types/api";  
import {  
  createMockCollection,  
  createMockCollectionEssential,  
} from "metabase-types/api/mocks";
```

```
import {...
```

## **frontend/src/metabase/nav/components/DevModeBanner/index.ts**

```
export * from "./DevModeBanner";
```

## **frontend/src/metabase/nav/components/LicenseTokenMissingBanner/index.ts**

```
export { LicenseTokenMissingBanner } from "./LicenseTokenMissingBanner";  
export { useLicenseTokenMissingBanner } from "./useLicenseTokenMissingBanner";
```

## **frontend/src/metabase/nav/components/LicenseTokenMissingBanner/useLicenseTokenMissingBanner.ts**

```
import dayjs from "dayjs";  
import utc from "dayjs/plugin/utc";
```

```
import { useUpdateSettingMutation } from "metabase/api";  
import { useSetting } from "metabase/common/hooks";  
import { isEEBuild } from...
```

## **frontend/src/metabase/nav/components/LicenseTokenMissingBanner/useLicenseTokenMissingBanner.unit.spec.ts**

```
import { act } from "@testing-library/react";
```

## Analyst Base

```
import { setupEnterprisePlugins } from "__support__/enterprise";
import {
  findRequests,
  setupPropertiesEndpoints,
  setupSettingsEndpoints,
  ...

```

### **frontend/src/metabase/nav/components/NewItemButton/analytics.ts**

```
import { trackSimpleEvent } from "metabase/lib/analytics";

export const trackAppNewButtonClicked = () =>
  trackSimpleEvent({
    event: "new_button_clicked",
    triggered_from: "app-bar",
  });

```

### **frontend/src/metabase/nav/components/NewItemButton/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/nav/components/ProfileLink/index.ts**

```
export { default as ProfileLink } from "./ProfileLink";

```

### **frontend/src/metabase/nav/components/ProfileLink/useHelpLink.ts**

```
import { useEffect, useState } from "react";

import { useSetting } from "metabase/common/hooks";
import { useSelector } from "metabase/lib/redux";
import { getIsPaidPlan } from...
```

### **frontend/src/metabase/nav/components/QuestionLineage/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/nav/components/ReadOnlyBanner/index.ts**

```
export * from "./ReadOnlyBanner";

```

### **frontend/src/metabase/nav/components/StoreLink/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/nav/components/TrialBanner/index.ts**

```
export * from "./TrialBanner";

```

### **frontend/src/metabase/nav/components/WhatsNewNotification/index.ts**

```
export { WhatsNewNotification } from "./WhatsNewNotification";

```

### **frontend/src/metabase/nav/components/WhatsNewNotification/utils.ts**

```
import { isNotFalsy } from "metabase/lib/types";

```

## Analyst Base

```
import { compareVersions } from "metabase/lib/utils";
import type { VersionInfoRecord } from "metabase-types/api";
import type { VersionInfo } from...
```

### **frontend/src/metabase/nav/components/WhatsNewNotification/utils.unit.spec.ts**

```
import { createMockVersionInfoRecord as mockVersion } from "metabase-types/api/mocks";
import type {
  VersionInfo,
  VersionInfoRecord,
} from "metabase-types/api/settings";
```

```
import {...
```

### **frontend/src/metabase/nav/components/search/RecentsList/index.ts**

```
export { RecentsList } from "./RecentsList";
```

### **frontend/src/metabase/nav/components/search/RecentsList/util.ts**

```
import { isSyncCompleted } from "metabase/lib/syncing";
import * as Urls from "metabase/lib/urls";
import type { RecentItem } from "metabase-types/api";
```

```
export const isItemActive = (item:...
```

### **frontend/src/metabase/nav/components/search/SearchBar/index.ts**

```
export { default as SearchBar } from "./SearchBar";
```

### **frontend/src/metabase/nav/components/search/SearchButton/index.ts**

```
export * from "./SearchButton";
```

### **frontend/src/metabase/nav/components/search/SearchResults/index.ts**

```
export { SearchResults, SearchLoadingSpinner } from "./SearchResults";
export type { SearchResultsFooter } from "./SearchResults";
export * from "./SearchResults.styled";
```

### **frontend/src/metabase/nav/components/search/SearchResultsDropdown/constants.ts**

```
export const MIN_RESULTS_FOR_FOOTER_TEXT = 4;
```

### **frontend/src/metabase/nav/components/search/SearchResultsDropdown/index.ts**

```
export { SearchResultsDropdown } from "./SearchResultsDropdown";
```

### **frontend/src/metabase/nav/constants.ts**

```
export const APP_BAR_HEIGHT = "52px";
export const APP_SUBHEADER_HEIGHT = "48px";
```

## Analyst Base

```
export const APP_BAR_EXTENDED_HEIGHT = "98px";
export const ADMIN_NAVBAR_HEIGHT = "65px";
export const...
```

### **frontend/src/metabase/nav/containers/AppBar/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/nav/containers/CollectionBreadcrumbs/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/nav/containers/MainNavbar/MainNavbarContainer/AddDataModal/analytics.ts**

```
import { trackSimpleEvent } from "metabase/lib/analytics";
import type { Engine } from "metabase-types/api";
export const trackAddDataEvent = (
  event: "csv_tab_clicked" | "sheets_tab_clicked" |...
```

### **frontend/src/metabase/nav/containers/MainNavbar/MainNavbarContainer/AddDataModal/index.ts**

```
export { AddDataModal } from "../AddDataModal";
```

### **frontend/src/metabase/nav/containers/MainNavbar/MainNavbarContainer/AddDataModal/utils.ts**

```
const validTabArray = ["db", "csv", "gsheets"] as const;
const validTabs = new Set<string>(validTabArray);
```

```
type AddDataTab = (typeof validTabArray)[number];
```

```
export const isValidTab = (v: string |...
```

### **frontend/src/metabase/nav/containers/MainNavbar/MainNavbarContainer/BookmarkList/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/nav/containers/MainNavbar/MainNavbarContainer/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/nav/containers/MainNavbar/SidebarItems/index.ts**

```
export { DraggableSidebarLink } from "../DraggableSidebarLink";
export { default as SidebarCollectionLink } from "../SidebarCollectionLink";
export { default as SidebarLink } from "../SidebarLink";
```

### **frontend/src/metabase/nav/containers/MainNavbar/analytics.ts**

```
import { trackSimpleEvent } from "metabase/lib/analytics";
```

## Analyst Base

```
export const trackOnboardingChecklistOpened = () => {
  trackSimpleEvent({
    event: "onboarding_checklist_opened",
  });
};
```

export const...

### **frontend/src/metabase/nav/containers/MainNavbar/getSelectedItems.ts**

```
import { coerceCollectionId } from "metabase/collections/utils";
import * as Urls from "metabase/lib/urls";
import type Question from "metabase-lib/v1/Question";
import type { Collection, Dashboard }...
```

### **frontend/src/metabase/nav/containers/MainNavbar/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/nav/containers/MainNavbar/types.ts**

```
import type { Location } from "history";
```

```
export interface MainNavbarOwnProps {
  isOpen: boolean;
  location: Location;
  params: {
    slug?: string;
    pageId?: string;
  };
}
```

export interface...

### **frontend/src/metabase/nav/containers/QuestionLineage/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/notifications/AddEditSidebar/CaveatMessage/index.ts**

```
export { CaveatMessage } from "../CaveatMessage";
```

### **frontend/src/metabase/notifications/AddEditSidebar/CaveatMessage/tests/common.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import { setup } from "../setup";
```



## Analyst Base

```
describe("CaveatMessage (OSS)", () => {
  it("should show a help link when `show-metabase-links: true`", () => {
    ...
```

### **frontend/src/metabase/notifications/AddEditSidebar/CaveatMessage/tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";

import type { SetupOpts } from "../setup";
import { setup as baseSetup } from "../setup";
```

```
function setup(opts: SetupOpts) {
  baseSetup({...
```

### **frontend/src/metabase/notifications/AddEditSidebar/CaveatMessage/tests/premium.unit.spec.ts**

```
import { screen } from "__support__/ui";

import type { SetupOpts } from "../setup";
import { setup as baseSetup } from "../setup";
```

```
function setup(opts: SetupOpts) {
  baseSetup({
    ...
```

### **frontend/src/metabase/notifications/AddEditSidebar/constants.ts**

```
import { t } from "ttag";

import type { ChannelType } from "metabase-types/api";

export const CHANNEL_NOUN_PLURAL: Record<ChannelType, string> = {
  get email() {
    return t`Emails`;
  },
  get...
```

### **frontend/src/metabase/notifications/DashboardSubscriptionsSidebar/get-supported-cards-for-subscriptions.unit.spec.ts**

```
import {
  createMockCard,
  createMockDashboard,
  createMockDashboardCard,
} from "metabase-types/api/mocks";
```

## Analyst Base

```
import { getSupportedCardsForSubscriptions } from...
```

### **frontend/src/metabase/notifications/DashboardSubscriptionsSidebar/tests/com mon.unit.spec.ts**

```
import userEvent from "@testing-library/user-event";  
import fetchMock from "fetch-mock";
```

```
import { screen } from "__support__/ui";
```

```
import { dashcard, hasBasicFilterOptions, setup, user } from...
```

### **frontend/src/metabase/notifications/DashboardSubscriptionsSidebar/tests/ente rprise.unit.spec.ts**

```
import userEvent from "@testing-library/user-event";
```

```
import { screen } from "__support__/ui";
```

```
import { hasBasicFilterOptions, setup } from "./setup";
```

```
describe("DashboardSubscriptionsSidebar...
```

### **frontend/src/metabase/notifications/DashboardSubscriptionsSidebar/tests/pre mium.unit.spec.ts**

```
import userEvent from "@testing-library/user-event";
```

```
import { screen, within } from "__support__/ui";
```

```
import {  
  createMockCard,  
  createMockDashboardCard,  
  createMockParameter,  
} from...
```

### **frontend/src/metabase/notifications/EmailAttachmentPicker.unit.spec.js**

```
import { mockSettings } from "__support__/settings";  
import { fireEvent, renderWithProviders, screen } from "__support__/ui";  
import { createMockState } from "metabase-types/store/mocks";
```

```
import...
```

### **frontend/src/metabase/notifications/NotificationsActionsMenu/index.ts**

```
export * from "./DashboardSubscriptionMenuItem";  
export * from "./QuestionAlertsMenuItem";
```

## Analyst Base

### **frontend/src/metabase/notifications/modals/CreateOrEditQuestionAlertModal/index.ts**

```
export { CreateOrEditQuestionAlertModal } from "./CreateOrEditQuestionAlertModal";
```

### **frontend/src/metabase/notifications/modals/CreateOrEditQuestionAlertModal/types.ts**

```
import type { NotificationCardSendCondition } from "metabase-types/api";
```

```
// TODO: combine this with api types
```

```
export type NotificationTriggerOption = {  
  value: NotificationCardSendCondition;  
  ...
```

### **frontend/src/metabase/notifications/modals/QuestionAlertListModal/index.ts**

```
export { QuestionAlertListModal } from "./QuestionAlertListModal";
```

### **frontend/src/metabase/notifications/modals/index.ts**

```
export { CreateOrEditQuestionAlertModal } from "./CreateOrEditQuestionAlertModal";
```

```
export { QuestionAlertListModal } from "./QuestionAlertListModal";
```

### **frontend/src/metabase/notifications/pulse/actions.js**

```
import { createAction } from "redux-actions";
```

```
import { t } from "ttag";
```

```
import { getActionErrorMessage } from "metabase/actions/utils";
```

```
import Pulses from "metabase/entities/pulses";
```

```
import {  
  ...
```

### **frontend/src/metabase/notifications/pulse/reducers.js**

```
import { handleActions } from "redux-actions";
```

```
import {  
  CANCEL_EDITING_PULSE,  
  FETCH_PULSE_CARD_PREVIEW,  
  FETCH_PULSE_FORM_INPUT,  
  FETCH_PULSE_LIST_BY_DASHBOARD_ID,  
  SAVE_EDITING_PULSE,  
  ...
```

### **frontend/src/metabase/notifications/pulse/selectors.js**

```
export const getEditingPulse = (state) => state.pulse.editingPulse;
```

```
export const getPulseFormInput = (state) => state.pulse?.formInput;
```

## **frontend/src/metabase/notifications/utils.ts**

```
import type {  
  CardId,  
  ChannelApiResponse,  
  CreateAlertNotificationRequest,  
  NotificationChannel,  
  NotificationHandler,  
  ScheduleSettings,  
  UserId,  
} from "metabase-types/api";  
  
import type {...
```

## **frontend/src/metabase/palette/components/PaletteShortcutsModal/index.ts**

```
export * from "./PaletteShortcutsModal";
```

## **frontend/src/metabase/palette/constants.ts**

```
import { t } from "ttag";  
  
import type { Settings } from "metabase-types/api";  
  
export const GROUP_LABELS = {  
  get global() {  
    return t`General`;  
  },  
  get dashboard() {  
    return...
```

## **frontend/src/metabase/palette/shortcuts/admin.ts**

```
import { t } from "ttag";  
  
import { ELLIPSIS } from "../constants";  
  
export const adminShortcuts = {  
  "admin-change-tab": {  
    get name() {  
      return t`Change admin tab`;  
    },  
    shortcut:...
```

## **frontend/src/metabase/palette/shortcuts/collection.ts**

```
import { t } from "ttag";
```

## Analyst Base

```
export const collectionShortcuts = {  
  "collection-send-items-to-trash": {  
    get name() {  
      return t`Move collection items to trash`;  
    },  
    shortcut:...
```

### **frontend/src/metabase/palette/shortcuts/dashboard.ts**

```
import { t } from "ttag";  
  
import { ELLIPSIS } from "../constants";
```

```
export const dashboardShortcuts = {  
  "dashboard-bookmark": {  
    get name() {  
      return t`Bookmark dashboard`;  
    },  
    ...
```

### **frontend/src/metabase/palette/shortcuts/global.ts**

```
import { t } from "ttag";  
  
export const globalShortcuts = {  
  "create-new-question": {  
    get name() {  
      return t`Create a question`;  
    },  
    shortcut: ["c q"],  
    shortcutGroup: "global" as...
```

### **frontend/src/metabase/palette/shortcuts/index.ts**

```
import { adminShortcuts } from "../admin";  
import { collectionShortcuts } from "../collection";  
import { dashboardShortcuts } from "../dashboard";  
import { globalShortcuts } from "../global";  
import {...
```

### **frontend/src/metabase/palette/shortcuts/question.ts**

```
import { t } from "ttag";  
  
export const questionShortcuts = {  
  "query-builder-toggle-notebook-editor": {  
    get name() {  
      return t`Switch to editor`;
```

## Analyst Base

```
},  
  shortcut: ["e"],  
  ...  
}
```

### **frontend/src/metabase/palette/shortcuts/shortcuts.unit.spec.ts**

```
import { type KeyboardShortcutId, shortcuts } from ".";  
  
const getShortcutsWithPrefix = (prefix: string) => {  
  const allIds = Object.keys(shortcuts) as KeyboardShortcutId[];  
  return allIds  
}
```

### **frontend/src/metabase/palette/types.ts**

```
import type { LocationDescriptor } from "history";  
import type { Action, ActionImpl } from "kbar";  
import type React from "react";  
  
import type { IconName } from "metabase/ui";  
import type { ...
```

### **frontend/src/metabase/palette/utils.ts**

```
import type { LocationDescriptor } from "history";  
import type { MouseEvent } from "react";  
import { t } from "ttag";  
import _ from "underscore";  
  
import { color } from "metabase/lib/colors";  
import ...
```

### **frontend/src/metabase/palette/utils.unit.spec.ts**

```
import type { LocationDescriptor } from "history";  
  
import type { PaletteActionImpl } from "../types";  
import {  
  locationDescriptorToURL,  
  navigateActionIndex,  
  processResults,  
  processSection,  
} ...
```

### **frontend/src/metabase/parameters/actions.ts**

```
import { CardApi, DashboardApi, ParameterApi } from "metabase/services";  
import { getNonVirtualFields } from "metabase-lib/v1/parameters/utils/parameter-fields";  
import { normalizeParameter } from ...
```

## Analyst Base

### **frontend/src/metabase/parameters/components/FilterApplyToast/index.ts**

```
export { FilterApplyToast } from "./FilterApplyToast";
```

### **frontend/src/metabase/parameters/components/FilterApplyToast/utils.ts**

```
import { msgid, ngettext } from "ttag";
import _ from "underscore";
```

```
function isParameterValueEmpty(value: unknown): boolean {
  return value == null || (Array.isArray(value) && value.length ===...
```

### **frontend/src/metabase/parameters/components/FilterApplyToast/utils.unit.spec.ts**

```
import { getFilterChangeDescription } from "./utils";
```

```
describe("getFilterChangeDescription", () => {
  describe("single filter changes", () => {
    it("should describe a single filter added", () =>...
```

### **frontend/src/metabase/parameters/components/FormattedParameterValue/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/parameters/components/ParameterLinkedFilters/index.ts**

```
export { ParameterLinkedFilters } from "./ParameterLinkedFilters";
```

### **frontend/src/metabase/parameters/components/ParameterLinkedFilters/types.ts**

```
import type { UiParameter } from "metabase-lib/v1/parameters/types";
import type { FieldId } from "metabase-types/api";
```

```
export type ParameterInfo = {
  parameter: UiParameter;
  filteredIds:...
```

### **frontend/src/metabase/parameters/components/ParameterLinkedFilters/utils.ts**

```
import type { UiParameter } from "metabase-lib/v1/parameters/types";
import { getFields } from "metabase-lib/v1/parameters/utils/parameter-fields";
import type {
  FieldId,
  ...
```

### **frontend/src/metabase/parameters/components/ParameterLinkedFilters/utils.unit.spec.ts**

## Analyst Base

```
import { checkNotNull } from "metabase/lib/types";
import { SAMPLE_METADATA } from "metabase-lib/test-helpers";
import { createMockUiParameter } from "metabase-lib/v1/parameters/mock";
import type {...
```

### **frontend/src/metabase/parameters/components/ParameterSettings/MoveParameterMenu/index.ts**

```
export * from "../MoveParameterMenu";
```

### **frontend/src/metabase/parameters/components/ParameterSettings/TemporalUnitSettings/index.ts**

```
export * from "../TemporalUnitSettings";
```

### **frontend/src/metabase/parameters/components/ParameterSettings/index.ts**

```
export { ParameterSettings } from "../ParameterSettings";
```

### **frontend/src/metabase/parameters/components/ParameterSidebar/index.ts**

```
export { ParameterSidebar } from "../ParameterSidebar";
```

### **frontend/src/metabase/parameters/components/ParameterWidget/index.ts**

```
export * from "../ParameterWidget";
```

### **frontend/src/metabase/parameters/components/ParametersList/index.ts**

```
export * from "../ParametersList";
```

```
export * from "../types";
```

### **frontend/src/metabase/parameters/components/ParametersList/types.ts**

```
import type {
  DashboardFullscreenControls,
  DashboardNightModeControls,
  EmbedHideParametersControls,
} from "metabase/dashboard/types";
import type Question from...
```

### **frontend/src/metabase/parameters/components/RequiredParamToggle/index.ts**

```
export { RequiredParamToggle } from "../RequiredParamToggle";
```

### **frontend/src/metabase/parameters/components/UpdateFilterButton/getUpdateButtonProps.ts**

```
import { t } from "ttag";
```

```
import { areParameterValuesIdentical } from "metabase-lib/v1/parameters/parameter-values";
```

```
const getUpdateLabel = () => t`Update filter`;
```

```
const getAddLabel = () =>...
```



## **frontend/src/metabase/parameters/components/UpdateFilterButton/getUpdateButtonProps.unit.spec.ts**

```
import { getUpdateButtonProps } from "../getUpdateButtonProps";
```

```
describe("getUpdateButtonProps", () => {  
  describe("non-required parameters", () => {  
    it("without both value and unsaved, shows...
```

## **frontend/src/metabase/parameters/components/UpdateFilterButton/index.ts**

```
export { UpdateFilterButton } from "../UpdateFilterButton";
```

## **frontend/src/metabase/parameters/components/ValuesSourceModal/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

## **frontend/src/metabase/parameters/components/ValuesSourceModal/utils.ts**

```
import { parse } from "csv-parse/browser/esm/sync";  
import { stringify } from "csv-stringify/browser/esm/sync";
```

```
import type { ParameterValue } from "metabase-types/api";
```

```
export const getValuesText...
```

## **frontend/src/metabase/parameters/components/ValuesSourceModal/utils.unit.spec.ts**

```
import { getStaticValues, getValuesText } from "../utils";
```

```
describe("getValuesText", () => {  
  it("should stringify just values correctly", () => {  
    expect(getValuesText(["Foo", "Bar", ...
```

## **frontend/src/metabase/parameters/components/ValuesSourceSettings/index.ts**

```
export { ValuesSourceSettings } from "../ValuesSourceSettings";
```

## **frontend/src/metabase/parameters/components/WidgetStatus/index.ts**

```
export * from "../WidgetStatus";
```

## **frontend/src/metabase/parameters/components/WidgetStatus/types.ts**

```
*
```

## **frontend/src/metabase/parameters/components/widgets/NumberInputWidget/index.ts**

```
export { NumberInputWidget } from "../NumberInputWidget";
```

## **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/FieldValuesWidget/ListField/index.ts**

## Analyst Base

```
export { ListField } from "../ListField";
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/FieldValuesWidget/ListField/test-constants.ts**

```
import _ from "underscore";
```

```
import type { ContentTranslationFunction } from "metabase/i18n/types";
```

```
import type { DictionaryResponse } from "metabase-types/api";
```

```
export const translateToGerman:...
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/FieldValuesWidget/ListField/types.ts**

```
import type { JSX } from "react";
```

```
import type { FieldValue, RowValue } from "metabase-types/api";
```

```
export type Option = FieldValue;
```

```
export const getOptionDisplayName = (option: Option | RowValue[])...
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/FieldValuesWidget/ListField/utils.ts**

```
import type { ContentTranslationFunction } from "metabase/i18n/types";
```

```
import type { FieldValue, RowValue } from "metabase-types/api";
```

```
import type { Option } from...
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/FieldValuesWidget/ListField/utils.unit.spec.ts**

```
import type { ContentTranslationFunction } from "metabase/i18n/types";
```

```
import type { FieldValue } from "metabase-types/api";
```

```
import { translateToGerman, translateToJapanese } from...
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/FieldValuesWidget/SingleSelectListField/SingleSelectListField.unit.spec.js**

```
import { waitForElementToBeRemoved } from "@testing-library/react";
```

```
import userEvent from "@testing-library/user-event";
```

```
import { render, screen } from "__support__/ui";
```

```
import { Value as...
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/**

## Analyst Base

### **FieldValuesWidget/SingleSelectListField/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/ FieldValuesWidget/SingleSelectListField/types.ts**

```
import type { FieldValue, RowValue } from "metabase-types/api";
```

```
export type Option = FieldValue;
```

```
export interface SingleSelectListFieldProps {  
  onChange: (value: RowValue[]) => void;  
  value:...
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/ FieldValuesWidget/SingleSelectListField/utils.ts**

```
export function optionItemEqualsFilter(  
  optionItem: any,  
  filter: string,  
)> boolean {  
  return String(optionItem) === filter;  
}
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/ FieldValuesWidget/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/ FieldValuesWidget/testMocks.ts**

```
import { createMockEntitiesState } from "__support__/store";  
import { getMetadata } from "metabase/selectors/metadata";  
import { createMockField } from "metabase-types/api/mocks";  
import {  
  ...
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/ FieldValuesWidget/types.ts**

```
export type ValuesMode = "search" | "list" | "none";  
export type LoadingStateType = "LOADING" | "LOADED" | "INIT";
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/ FieldValuesWidget/utils.ts**

```
import { t } from "ttag";  
import _ from "underscore";
```

## Analyst Base

```
import { isTransientId } from "metabase/dashboard/utils";
import { stripId } from "metabase/lib/formatting";
import type { ComboboxItem } from...
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/FieldValuesWidget/utils.unit.spec.js**

```
import { ORDERS, PEOPLE, PRODUCTS } from "metabase-types/api/mocks/presets";
```

```
import {
  LISTABLE_FIELD_WITH_MANY_VALUES_ID,
  STRING_PK_FIELD_ID,
  metadata,
} from "../testMocks";
import {...
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/ParameterFieldWidgetValue/ParameterFieldWidgetValue.unit.spec.js**

```
import { render, screen } from "___support___/ui";
```

```
import { ParameterFieldWidgetValue } from "../ParameterFieldWidgetValue";
```

```
const value = "A value";
```

```
describe("when fields is empty array", () => {
  ...
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/normalizeValue.ts**

```
export function normalizeValue(value: unknown) {
  if (Array.isArray(value)) {
    return value;
  }
```

```
  return value || value === 0 ? [value] : [];
}
```

### **frontend/src/metabase/parameters/components/widgets/ParameterFieldWidget/normalizeValue.unit.spec.js**

```
import { normalizeValue } from "../normalizeValue";
```

```
describe("normalizeValue", () => {
  it("returns empty array if value is null", () => {
```

## Analyst Base

```
const value = null;  
const expected = [];
```

```
const...
```

### **frontend/src/metabase/parameters/components/widgets/StringInputWidget/index.ts**

```
export { StringInputWidget } from "./StringInputWidget";
```

### **frontend/src/metabase/parameters/components/widgets/TemporalUnitWidget/index.ts**

```
export * from "./TemporalUnitWidget";
```

### **frontend/src/metabase/parameters/components/widgets/TextWidget/index.ts**

```
export { TextWidget } from "./TextWidget";
```

### **frontend/src/metabase/parameters/components/widgets/constants.ts**

```
import type { ComboboxProps } from "metabase/ui";
```

```
export const MIN_WIDTH = 300;
```

```
export const WIDTH = 380;
```

```
export const COMBOBOX_PROPS: Partial<ComboboxProps> = {  
  width: 314,  
  position:...
```

### **frontend/src/metabase/parameters/reducers.ts**

```
import { handleActions } from "redux-actions";
```

```
import {  
  INITIALIZE,  
  RESET,  
  UPDATE_DASHBOARD_AND_CARDS,  
} from "metabase/dashboard/actions";  
import {  
  API_UPDATE_QUESTION,  
  INITIALIZE_QB,  
  ...
```

### **frontend/src/metabase/parameters/selectors.ts**

```
import type { State } from "metabase-types/store";
```

```
export const getParameterValuesCache = (state: State) => {  
  return state.parameters.parameterValuesCache;  
};
```

## Analyst Base

### **frontend/src/metabase/parameters/utils/dashboard-options.ts**

```
import { t } from "ttag";
```

```
import {  
  BOOLEAN_OPTION,  
  ID_OPTION,  
} from "metabase-lib/v1/parameters/constants";  
import type { ParameterSectionId } from...
```

### **frontend/src/metabase/parameters/utils/dashboard-options.unit.spec.js**

```
import _ from "underscore";
```

```
import { getDashboardParameterSections } from "../dashboard-options";
```

```
describe("parameters/utils/dashboard-options", () => {  
  describe("getDashboardParameterSections", ...
```

### **frontend/src/metabase/parameters/utils/dashboards.ts**

```
import _ from "underscore";
```

```
import { isQuestionCard, isQuestionDashCard } from "metabase/dashboard/utils";  
import { slugify } from "metabase/lib/formatting";  
import { isNotNull } from...
```

### **frontend/src/metabase/parameters/utils/dashboards.unit.spec.js**

```
import { createMockMetadata } from "__support__/metadata";  
import {  
  createParameter,  
  getFilteringParameterValuesMap,  
  getUnsavedDashboardUiParameters,  
  hasMapping,  
  hasMatchingParameters,  
  ...
```

### **frontend/src/metabase/parameters/utils/date-formatting.ts**

```
import { getDateFilterDisplayName } from "metabase/querying/filters/utils/dates";  
import { deserializeDateParameterValue } from "metabase/querying/parameters/utils/parsing";  
import type { Parameter } from...
```

### **frontend/src/metabase/parameters/utils/formatting.ts**

```
import { msgid, ngettext } from "ttag";
```

```
import { formatValue } from "metabase/lib/formatting";  
import * as Lib from "metabase-lib";
```

## Analyst Base

```
import Field from "metabase-lib/v1/metadata/Field";
import type {...
```

### **frontend/src/metabase/parameters/utils/formatting.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import { checkNotNull } from "metabase/lib/types";
import { createMockUiParameter } from "metabase-lib/v1/parameters/mock";
import {
  ...
```

### **frontend/src/metabase/parameters/utils/linked-filters.js**

```
import {
  FIELD_FILTER_PARAMETER_TYPES,
  TYPE_SUPPORTS_LINKED_FILTERS,
} from "metabase-lib/v1/parameters/constants";
import { getParameterType } from...
```

### **frontend/src/metabase/parameters/utils/mapping-options.ts**

```
import { t } from "ttag";
import _ from "underscore";

import { tag_names } from "cljs/metabase.parameters.shared";
import { getColumnIcon } from "metabase/common/utils/columns";
import {...
```

### **frontend/src/metabase/parameters/utils/mapping-options.unit.spec.js**

```
import { createMockMetadata } from "__support__/metadata";
import * as Lib from "metabase-lib";
import {
  SAMPLE_METADATA,
  createQueryWithClauses,
} from "metabase-lib/test-helpers";
import...
```

### **frontend/src/metabase/parameters/utils/parameter-id.ts**

```
export function generateParameterId() {
  const num = Math.floor(Math.random() * Math.pow(2, 32));
  return num.toString(16);
}
```

### **frontend/src/metabase/parameters/utils/parameter-id.unit.spec.ts**

```
import { generateParameterId } from "../parameter-id";

describe("parameters/utils/parameter-id", () => {
  it("should generate a random parameter id", () => {
```

## Analyst Base

...

### **frontend/src/metabase/parameters/utils/parameter-type.ts**

```
import { SINGLE_OR_MULTI_SELECTABLE_TYPES } from "metabase-lib/v1/parameters/constants";
import type { ParameterWithTemplateTagTarget } from "metabase-lib/v1/parameters/types";
import {
```

...

### **frontend/src/metabase/parameters/utils/parameter-type.unit.spec.ts**

```
import { createMockUiParameter } from "metabase-lib/v1/parameters/mock";
import { createMockParameter } from "metabase-types/api/mocks";
```

```
import { isSingleOrMultiSelectable } from...
```

### **frontend/src/metabase/parameters/utils/ui.ts**

```
import _ from "underscore";
```

```
import type { IconName } from "metabase/ui";
import { isEqualOperator } from "metabase-lib/v1/operators/utils";
import type { UiParameter } from...
```

### **frontend/src/metabase/parameters/utils/ui.unit.spec.ts**

```
import { createMockUiParameter } from "metabase-lib/v1/parameters/mock";
```

```
import {
  buildHiddenParametersSlugSet,
  getParameterIconName,
  getParameterWidgetTitle,
  getVisibleParameters,
} from...
```

### **frontend/src/metabase/plugins/builtin/auth/google.js**

```
import MetabaseSettings from "metabase/lib/settings";
import {
  PLUGIN_AUTH_PROVIDERS,
  PLUGIN_IS_PASSWORD_USER,
} from "metabase/plugins";
```

```
PLUGIN_AUTH_PROVIDERS.providers.push((providers) => {
```

...

### **frontend/src/metabase/plugins/builtin/auth/jwt.js**

```
import { PLUGIN_IS_PASSWORD_USER } from "metabase/plugins";
```

```
PLUGIN_IS_PASSWORD_USER.push((user) => user.sso_source !== "jwt");
```



## Analyst Base

### **frontend/src/metabase/plugins/builtin/auth/ldap.js**

```
import { PLUGIN_IS_PASSWORD_USER } from "metabase/plugins";

PLUGIN_IS_PASSWORD_USER.push((user) => user.sso_source !== "ldap");
```

### **frontend/src/metabase/plugins/builtin/auth/password.js**

```
import { PLUGIN_AUTH_PROVIDERS } from "metabase/plugins";

PLUGIN_AUTH_PROVIDERS.providers.push((providers) => {
  const passwordProvider = {
    name: "password",
    // circular dependencies
    ...
  }
});
```

### **frontend/src/metabase/plugins/builtin/auth/saml.js**

```
import { PLUGIN_IS_PASSWORD_USER } from "metabase/plugins";

PLUGIN_IS_PASSWORD_USER.push((user) => user.sso_source !== "saml");
```

### **frontend/src/metabase/plugins/builtin.js**

```
import "metabase/plugins/builtin/auth/password";
import "metabase/plugins/builtin/auth/google";
import "metabase/plugins/builtin/auth/ldap";
import "metabase/plugins/builtin/auth/jwt";
import...
```

### **frontend/src/metabase/plugins/components/PluginPlaceholder/index.ts**

```
export * from "./PluginPlaceholder";
```

### **frontend/src/metabase/plugins/index.ts**

```
import type { Middleware } from "@reduxjs/toolkit";
import React, {
  type ComponentType,
  type Dispatch,
  type HTMLAttributes,
  type ReactNode,
  type SetStateAction,
  useCallback,
  useMemo,
}...
```

### **frontend/src/metabase/plugins/types.ts**

```
import type { ComponentType } from "react";
```

## Analyst Base

```
import type { ConfirmationState } from "metabase/common/hooks/use-confirmation";
import type { Member, Membership, User } from...
```

### **frontend/src/metabase/public/components/EmbedFrame/LogoBadge/index.ts**

```
export { LogoBadge } from "./LogoBadge";
```

### **frontend/src/metabase/public/components/EmbedFrame/index.ts**

```
export { EmbedFrame } from "./EmbedFrame";
export { SyncedEmbedFrame } from "./SyncedEmbedFrame";
```

### **frontend/src/metabase/public/components/EmbedModal/EmbedModalContent/index.ts**

```
export { EmbedModalContent } from "./EmbedModalContent";
```

### **frontend/src/metabase/public/components/EmbedModal/SelectEmbedTypePane/index.ts**

```
export { SelectEmbedTypePane } from "./SelectEmbedTypePane";
```

### **frontend/src/metabase/public/components/EmbedModal/StaticEmbedSetupPane/PreviewPane/index.ts**

```
export * from "./PreviewPane";
```

### **frontend/src/metabase/public/components/EmbedModal/StaticEmbedSetupPane/PreviewPane/utils.ts**

```
import { match } from "ts-pattern";
```

```
import type { PreviewBackgroundType } from "./PreviewPane";
```

```
export function getCheckerBoardDataUri(
  theme: Extract<
    PreviewBackgroundType,
    ...

```

### **frontend/src/metabase/public/components/EmbedModal/StaticEmbedSetupPane/config.ts**

```
import type { EmbeddingDisplayOptions } from "metabase/public/lib/types";
```

```
export function getDefaultDisplayOptions(
  shouldShownDownloadData: boolean,
): EmbeddingDisplayOptions {
  return {
    ...

```

### **frontend/src/metabase/public/components/EmbedModal/StaticEmbedSetupPane/index.ts**

## Analyst Base

```
export { StaticEmbedSetupPane } from "../StaticEmbedSetupPane";
```

### **frontend/src/metabase/public/components/EmbedModal/StaticEmbedSetupPane/tabs.ts**

```
export const EMBED_MODAL_TABS = {  
  Overview: "overview" as const,  
  Parameters: "parameters" as const,  
  LookAndFeel: "lookAndFeel" as const,  
};
```

### **frontend/src/metabase/public/components/EmbedModal/StaticEmbedSetupPane/types.ts**

```
import type { EmbedResourceParameter } from "metabase/public/lib/types";
```

```
export type ActivePreviewPane = "preview" | "code";
```

```
export type EmbedResourceParameterWithValue = EmbedResourceParameter &...
```

### **frontend/src/metabase/public/components/EmbedModal/StaticEmbedSetupPane/utils.ts**

```
export function getHighlightedRanges(  
  source: string,  
  highlightedTexts: string[] = [],  
) {  
  return highlightedTexts.flatMap((highlightedText) =>  
    getHighlightedRangesForText(source,...
```

### **frontend/src/metabase/public/components/EmbedModal/StaticEmbedSetupPane/utils.unit.spec.ts**

```
import { getHighlightedRanges } from "../utils";
```

```
describe("getHighlightedRanges", () => {  
  it("returns an empty array if no highlighted text is provided", () => {  
    ...
```

### **frontend/src/metabase/public/components/EmbedModal/index.ts**

```
export { EmbedModal } from "../EmbedModal";  
export { EmbedModalContent } from "../EmbedModalContent";
```

### **frontend/src/metabase/public/constants.ts**

```
import { DEFAULT_DASHBOARD_DISPLAY_OPTIONS } from "metabase/dashboard/constants";
```

```
export const DEFAULT_EMBED_DISPLAY_PARAMS = {  
  titled: DEFAULT_DASHBOARD_DISPLAY_OPTIONS.titled,
```

## Analyst Base

theme:...

**frontend/src/metabase/public/containers/PublicAction/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/public/containers/PublicApp/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/public/containers/PublicOrEmbeddedDashboard/PublicOrEmbeddedDashboardPage/index.ts**

export { PublicOrEmbeddedDashboardPage } from "./PublicOrEmbeddedDashboardPage";

**frontend/src/metabase/public/containers/PublicOrEmbeddedDashboard/index.ts**

export { PublicOrEmbeddedDashboardPage } from "./PublicOrEmbeddedDashboardPage";

**frontend/src/metabase/public/containers/PublicOrEmbeddedDashboard/types.ts**

import type { Dashboard } from "metabase-types/api";

export type PublicOrEmbeddedDashboardEventHandlersProps = {

/\*\*

\* Callback that is called when the dashboard is loaded.

\*/

onLoad?:...

**frontend/src/metabase/public/containers/PublicOrEmbeddedDashboard/use-dashboard-load-handlers.ts**

import { useCallback, useEffect, useRef } from "react";

/\* eslint-disable-next-line no-restricted-imports -- deprecated sdk import \*/

import { getEventHandlers } from...

**frontend/src/metabase/public/containers/PublicOrEmbeddedQuestion/PublicOrEmbeddedQuestion/index.ts**

export { PublicOrEmbeddedQuestion } from "./PublicOrEmbeddedQuestion";

**frontend/src/metabase/public/containers/PublicOrEmbeddedQuestion/index.ts**

export { PublicOrEmbeddedQuestion } from "./PublicOrEmbeddedQuestion";

**frontend/src/metabase/public/hooks/index.ts**

export \* from "./use-embed-frame-options";

export \* from "./use-set-embed-font";

**frontend/src/metabase/public/hooks/use-embed-frame-options.ts**

## Analyst Base

```
import type { Location } from "history";
import { useEffect } from "react";
```

```
import { useDocsUrl } from "metabase/common/hooks";
import { parseHashOptions } from "metabase/lib/browser";
import {...
```

### **frontend/src/metabase/public/hooks/use-set-embed-font.ts**

```
import type { Location } from "history";
import { useEffect } from "react";
```

```
import { parseHashOptions } from "metabase/lib/browser";
import { useDispatch } from "metabase/lib/redux";
import type {...
```

### **frontend/src/metabase/public/lib/analytics.ts**

```
import type { ExportFormatType } from "metabase/embedding/components/PublicLinkPopover/types";
import { trackSchemaEvent } from "metabase/lib/analytics";
```

```
import type {
  DisplayTheme,
  ...
```

### **frontend/src/metabase/public/lib/code-templates.ts**

```
import { match } from "ts-pattern";
```

```
import { optionsToHashParams } from "../embed";
import type {
  CodeSampleParameters,
  EmbeddingDisplayOptions,
  EmbeddingHashOptions,
  ...
```

### **frontend/src/metabase/public/lib/code.ts**

```
import {
  clojure,
  getHtmlSource,
  getJsxSource,
  getPugSource,
  node,
  python,
  ruby,
} from "../code-templates";
import type {
```

## Analyst Base

```
ClientCodeSampleConfig,  
CodeSampleParameters,  
...
```

### **frontend/src/metabase/public/lib/embed.ts**

```
import { CompactSign } from "jose"; // using jose because jsonwebtoken doesn't work on the web :-/  
import querystring from "querystring";
```

```
import type {  
  EmbedResource,  
  EmbedResourceType,  
  ...
```

### **frontend/src/metabase/public/lib/types.ts**

```
import type { CodeLanguage } from "metabase/common/components/CodeEditor";  
import type { Card, Dashboard } from "metabase-types/api";
```

```
export type DisplayTheme = "light" | "night" | ...
```

### **frontend/src/metabase/query\_builder/actions/core/card.ts**

```
import Questions from "metabase/entities/questions";  
import type { Dispatch, GetState } from "metabase-types/store";
```

```
// load a card either by ID or from a base64 serialization. if both are present...
```

### **frontend/src/metabase/query\_builder/actions/core/core.ts**

```
import { createAction } from "redux-actions";  
import _ from "underscore";
```

```
import { invalidateNotificationsApiCache } from "metabase/api";  
import Databases from "metabase/entities/databases";  
import...
```

### **frontend/src/metabase/query\_builder/actions/core/index.ts**

```
export * from "./core";  
export * from "./initializeQB";  
export * from "./updateQuestion";  
export * from "./native";
```

### **frontend/src/metabase/query\_builder/actions/core/initializeQB.ts**

```
import type { LocationDescriptorObject } from "history";  
import querystring from "querystring";
```

```
import Questions from "metabase/entities/questions";  
import Snippets from...
```

## Analyst Base

### **frontend/src/metabase/query\_builder/actions/core/initializeQB.unit.spec.ts**

```
import fetchMock from "fetch-mock";
import type { LocationDescriptorObject } from "history";

import { createMockEntitiesState } from "__support__/store";
import Databases from...
```

### **frontend/src/metabase/query\_builder/actions/core/native.ts**

```
export const SET_IS_SHOWING_TEMPLATE_TAGS_EDITOR =
  "metabase/qb/SET_IS_SHOWING_TEMPLATE_TAGS_EDITOR";
export const setIsShowingTemplateTagsEditor = (
  isShowingTemplateTagsEditor: boolean,
) =>...
```

### **frontend/src/metabase/query\_builder/actions/core/parameterUtils.ts**

```
import { hasMatchingParameters } from "metabase/parameters/utils/dashboards";
import { setErrorPage } from "metabase/redux/app";
import { DashboardApi } from "metabase/services";
import type Metadata...
```

### **frontend/src/metabase/query\_builder/actions/core/pivot-table.ts**

```
import { assocIn } from "icepick";
import _ from "underscore";

import * as Lib from "metabase-lib";
import type Question from "metabase-lib/v1/Question";
import type { Series } from...
```

### **frontend/src/metabase/query\_builder/actions/core/types.ts**

```
export const SOFT_RELOAD_CARD = "metabase/qb/SOFT_RELOAD_CARD";
export const API_UPDATE_QUESTION = "metabase/qb/API_UPDATE_QUESTION";
```

### **frontend/src/metabase/query\_builder/actions/core/updateQuestion.ts**

```
import _ from "underscore";

import { getTrashUndoMessage } from "metabase/archive/utils";
import Questions from "metabase/entities/questions";
import { createThunkAction } from...
```

### **frontend/src/metabase/query\_builder/actions/core/updateQuestion.unit.spec.ts**

```
import { createMockEntitiesState } from "__support__/store";
import { checkNotNull } from "metabase/lib/types";
import * as questionActions from "metabase/questions/actions";
import { getMetadata }...
```

## Analyst Base

### **frontend/src/metabase/query\_builder/actions/core/utils.ts**

```
import _ from "underscore";

import { syncVizSettingsWithQuery } from "metabase/querying/viz-settings/utils/sync-viz-settings";
import { getPersistableDefaultSettingsForSeries } from...
```

### **frontend/src/metabase/query\_builder/actions/index.ts**

```
export * from "./core";
export * from "./models";
export * from "./native";
export * from "./navigation";
export * from "./object-detail";
export * from "./query-updates";
export * from...
```

### **frontend/src/metabase/query\_builder/actions/modal.ts**

```
import { createThunkAction } from "metabase/lib/redux";
import { checkNotNull } from "metabase/lib/types";
import { UserApi } from "metabase/services";
```

```
export const CLOSE_QB_NEWB_MODAL =...
```

### **frontend/src/metabase/query\_builder/actions/models.ts**

```
import { push } from "react-router-redux";
import { createAction } from "redux-actions";
import { t } from "ttag";
```

```
import { addUndo } from "metabase/redux/undo";
import type { Dispatch, GetState }...
```

### **frontend/src/metabase/query\_builder/actions/native.ts**

```
import { createAction } from "redux-actions";

import Questions from "metabase/entities/questions";
import { createThunkAction } from "metabase/lib/redux";
import { updateUserSetting } from...
```

### **frontend/src/metabase/query\_builder/actions/navigation.ts**

```
import type { Location } from "history";

import { createThunkAction } from "metabase/lib/redux";
import { equals } from "metabase/lib/utils";
import { getLocation } from...
```



## Analyst Base

### **frontend/src/metabase/query\_builder/actions/object-detail.ts**

```
import _ from "underscore";

import { createThunkAction } from "metabase/lib/redux";
import { getMetadata } from "metabase/selectors/metadata";
import { MetabaseApi } from "metabase/services";
import...
```

### **frontend/src/metabase/query\_builder/actions/query-updates.ts**

```
import type { Limit } from "metabase-lib";
import * as Lib from "metabase-lib";
import type { Dispatch, GetState } from "metabase-types/store";

import { getQuestion } from "../selectors";

import {...
```

### **frontend/src/metabase/query\_builder/actions/querying.ts**

```
import { createAction } from "redux-actions";
import { t } from "ttag";

import { defer } from "metabase/lib/promise";
import { createThunkAction } from "metabase/lib/redux";
import {...
```

### **frontend/src/metabase/query\_builder/actions/state.ts**

```
import { createAction } from "redux-actions";

export const SET_CURRENT_STATE = "metabase/qb/SET_CURRENT_STATE";
export const setCurrentState = createAction(SET_CURRENT_STATE);
```

### **frontend/src/metabase/query\_builder/actions/timelines.ts**

```
import { createAction } from "redux-actions";

import type { CollectionId, Timeline } from "metabase-types/api";
import type { Dispatch, GetState } from "metabase-types/store";

import {...
```

### **frontend/src/metabase/query\_builder/actions/ui.ts**

```
import { createAction } from "redux-actions";

import { updateSetting } from "metabase/admin/settings/settings";
import { getOriginalCard } from "metabase/query_builder/selectors";
```

## Analyst Base

```
import {...
```

### **frontend/src/metabase/query\_builder/actions/url.ts**

```
import type { LocationDescriptor } from "history";
import { push, replace } from "react-router-redux";
import { parse as parseUrl } from "url";
```

```
import { isEqualCard } from...
```

### **frontend/src/metabase/query\_builder/actions/visualization-settings.ts**

```
import * as Lib from "metabase-lib";
import type Question from "metabase-lib/v1/Question";
import type { VisualizationSettings } from "metabase-types/api";
import type { Dispatch, GetState } from...
```

### **frontend/src/metabase/query\_builder/actions/zoom.ts**

```
import type { ObjectId } from "metabase/visualizations/components/ObjectDetail/types";
import type { Dispatch, GetState } from "metabase-types/store";
```

```
import { getPKColumnIndex } from...
```

### **frontend/src/metabase/query\_builder/analytics.js**

```
import { trackSchemaEvent, trackSimpleEvent } from "metabase/lib/analytics";
import * as Lib from "metabase-lib";
```

```
export const trackNewQuestionSaved = (
  draftQuestion,
  createdQuestion,
  ...

```

### **frontend/src/metabase/query\_builder/components/AddToDashSelectDashModal/hooks.ts**

```
import { useAsync } from "react-use";
```

```
import { ActivityApi } from "metabase/services";
import type { Dashboard } from "metabase-types/api";
```

```
export const useMostRecentlyViewedDashboard = () => {
  ...

```

### **frontend/src/metabase/query\_builder/components/AddToDashSelectDashModal/index.ts**

```
export { AddToDashSelectDashModal } from "../AddToDashSelectDashModal";
```

### **frontend/src/metabase/query\_builder/components/AddToDashSelectDashModal**

## **/utils.ts**

```
import {
  coerceCollectionId,
  isPublicCollection,
} from "metabase/collections/utils";
import type { DashboardPickerItem } from "metabase/common/components/Pickers/DashboardPicker";
import {...
```

## **frontend/src/metabase/query\_builder/components/DataSelector/DataSelector.unit.spec.js**

```
import userEvent from "@testing-library/user-event";

import { createMockMetadata } from "__support__/metadata";
import { getIcon, render, renderWithProviders, screen } from "__support__/ui";
import...
```

## **frontend/src/metabase/query\_builder/components/DataSelector/DataSelectorDataBucketPicker/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/query\_builder/components/DataSelector/DataSelectorDatabasePicker/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/query\_builder/components/DataSelector/DataSelectorDatabaseSchemaPicker/DataSelectorDatabaseSchemaPicker.unit.spec.js**

```
import { createMockEntitiesState } from "__support__/store";
import { render, renderWithProviders, screen } from "__support__/ui";
import { checkNotNull } from "metabase/lib/types";
import {...
```

## **frontend/src/metabase/query\_builder/components/DataSelector/DataSelectorDatabaseSchemaPicker/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/query\_builder/components/DataSelector/DataSelectorFieldPicker/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/query\_builder/components/DataSelector/DataSelectorLoading/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/query\_builder/components/DataSelector/DataSelectorSc**

## Analyst Base

### **hemaPicker/DataSelectorSchemaPicker.unit.spec.js**

```
import { render, screen } from "___support___/ui";

import DataSelectorSchemaPicker from "../DataSelectorSchemaPicker";

describe("DataSelectorSchemaPicker", () => {
  it("displays schema name", () =>...
```

### **frontend/src/metabase/query\_builder/components/DataSelector/DataSelectorSchemaPicker/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/query\_builder/components/DataSelector/DataSelectorSectionHeader/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/query\_builder/components/DataSelector/DataSelectorTablePicker/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/query\_builder/components/DataSelector/constants.ts**

```
import { t } from "ttag";

import type { DataPickerDataType, DataTypeInfoItem } from "../types";

export const CONTAINER_WIDTH = 300;

export const DATA_BUCKET: Record<string, DataPickerDataType> = {
  ...
```

### **frontend/src/metabase/query\_builder/components/DataSelector/saved-entity-picker/constants.ts**

```
import { t } from "ttag";

export const CARD_INFO = {
  question: {
    get title() {
      return t`Saved Questions`;
    },
  },
  model: "card",
  icon: "table2",
},
model: {
```

## Analyst Base

```
get title() {
```

```
...
```

### **frontend/src/metabase/query\_builder/components/DataSelector/saved-entity-picker/utils.js**

```
export const findCollectionById = (collections, collectionId) => {  
  if (!collections || collections.length === 0) {  
    return null;  
  }
```

```
  const collection = collections.find((c) => c.id ===...
```

### **frontend/src/metabase/query\_builder/components/DataSelector/types.ts**

```
import type { IconName } from "metabase/ui";
```

```
export type DataPickerDataType =  
  | "models"  
  | "raw-data"  
  | "questions"  
  | "metrics";
```

```
export type DataTypeInfoItem = {  
  id: DataPickerDataType;  
  ...
```

### **frontend/src/metabase/query\_builder/components/DataSelector/utils.ts**

```
import {  
  METRICS_INFO_ITEM,  
  MODELS_INFO_ITEM,  
  RAW_DATA_INFO_ITEM,  
  SAVED_QUESTIONS_INFO_ITEM,  
} from "../constants";  
import type { DataTypeInfoItem } from "../types";
```

```
export function...
```

### **frontend/src/metabase/query\_builder/components/DatasetEditor/DatasetFieldMetadataSidebar/MappedFieldPicker/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/query\_builder/components/DatasetEditor/DatasetFieldMetadataSidebar/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/query\_builder/components/DatasetEditor/DatasetNotebook/index.ts**

```
export * from "./DatasetNotebook";
```

**frontend/src/metabase/query\_builder/components/DatasetEditor/EditorTabs/index.ts**

```
export * from "./EditorTabs";
```

**frontend/src/metabase/query\_builder/components/DatasetEditor/TabHintToast/index.ts**

```
export * from "./TabHintToast";
```

**frontend/src/metabase/query\_builder/components/DatasetEditor/constants.ts**

```
*
```

**frontend/src/metabase/query\_builder/components/DatasetEditor/index.ts**

```
export { DatasetEditor } from "./DatasetEditor";
```

**frontend/src/metabase/query\_builder/components/ImpossibleToCreateModelModal/index.ts**

```
export * from "./ImpossibleToCreateModelModal";
```

**frontend/src/metabase/query\_builder/components/ImpossibleToCreateModelModal/tests/common.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import { setup } from "./setup";
```

```
describe("ImpossibleToCreateModelModal (OSS)", () => {  
  it("should show a help link when `show-metabase-links: true`", ()...
```

**frontend/src/metabase/query\_builder/components/ImpossibleToCreateModelModal/tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import type { SetupOpts } from "./setup";
```

```
import { setup as baseSetup } from "./setup";
```

```
function setup(opts: SetupOpts) {  
  baseSetup({...
```

**frontend/src/metabase/query\_builder/components/ImpossibleToCreateModelModal/tests/premium.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

## Analyst Base

```
import type { SetupOpts } from "./setup";
import { setup as baseSetup } from "./setup";
```

```
function setup(opts: SetupOpts) {
  baseSetup({
    ...
```

### **frontend/src/metabase/query\_builder/components/NativeQueryEditor/CodeMirrorEditor/language.ts**

```
import { json } from "@codemirror/lang-json";
import {
  MySQL,
  PLSQL,
  PostgreSQL,
  SQLDialect,
  StandardSQL,
  sql,
} from "@codemirror/lang-sql";
import {
  type LanguageSupport,
  ...
```

### **frontend/src/metabase/query\_builder/components/NativeQueryEditor/DataSourceSelectors/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/query\_builder/components/NativeQueryEditor/NativeQueryEditorActionButtons/index.ts**

```
export * from "./NativeQueryEditorActionButtons";
```

### **frontend/src/metabase/query\_builder/components/NativeQueryEditor/RightClickPopover/index.ts**

```
export * from "./RightClickPopover";
```

### **frontend/src/metabase/query\_builder/components/NativeQueryEditor/VisibilityToggle/index.ts**

```
export * from "./VisibilityToggle";
```

### **frontend/src/metabase/query\_builder/components/NativeQueryEditor/\_\_mocks\_\_/index.ts**

eslint-disable-next-line import/no-default-export

### **frontend/src/metabase/query\_builder/components/NativeQueryEditor/constants**

**.ts**

```
export const SCROLL_MARGIN = 8;
export const MIN_HEIGHT_LINES = 15;
export const MIN_EDITOR_HEIGHT_AFTER_DRAGGING = 0;
export const THRESHOLD_FOR_AUTO_CLOSE = 50;
```

**frontend/src/metabase/query\_builder/components/NativeQueryEditor/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/query\_builder/components/NativeQueryEditor/utils.ts**

```
import type { FormatOptionsWithLanguage, SqlLanguage } from "sql-formatter";
```

```
import { getEngineNativeType } from "metabase/lib/engine";
```

```
import type NativeQuery from...
```

**frontend/src/metabase/query\_builder/components/NativeQueryEditor/utils.unit.spec.ts**

```
import {
  canFormatForEngine,
  formatQuery,
} from "metabase/query_builder/components/NativeQueryEditor/utils";
```

```
const formattingTestCases = [
  {
    engine: "mysql",
    input:
      "select...
```

**frontend/src/metabase/query\_builder/components/NewDatasetModal/index.ts**

```
export * from "../NewDatasetModal";
```

**frontend/src/metabase/query\_builder/components/QueryModals/index.ts**

```
export { QueryModals } from "../QueryModals";
```

**frontend/src/metabase/query\_builder/components/QuestionActivityTimeline.unit.spec.js**

```
import {
  setupRevisionsEndpoints,
  setupUsersEndpoints,
} from "__support__/server-mocks";
import {
  renderWithProviders,
  screen,
  waitForLoaderToBeRemoved,
```



## Analyst Base

```
} from "__support__/ui";  
import {...
```

### **frontend/src/metabase/query\_builder/components/QuestionDownloadPopover/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/query\_builder/components/QuestionDownloadWidget/index.ts**

```
export { QuestionDownloadWidget } from "./QuestionDownloadWidget";
```

### **frontend/src/metabase/query\_builder/components/QuestionDownloadWidget/use-download-data.ts**

```
import { useAsyncFn } from "react-use";  
import type { AsyncFnReturn } from "react-use/lib/useAsyncFn";
```

```
import { useDispatch } from "metabase/lib/redux";  
import {  
  type DownloadQueryResultsOpts,  
  ...
```

### **frontend/src/metabase/query\_builder/components/QuestionEmbedWidget/index.ts**

```
export * from "./QuestionEmbedWidget";
```

### **frontend/src/metabase/query\_builder/components/SavedQuestionHeaderButton/SavedQuestionHeaderButton.unit.spec.js**

```
import userEvent from "@testing-library/user-event";
```

```
import { setupEnterpriseTest } from "__support__/enterprise";  
import { createMockMetadata } from "__support__/metadata";  
import { getIcon,...
```

### **frontend/src/metabase/query\_builder/components/SidebarContent/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/query\_builder/components/SidebarHeader/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/query\_builder/components/VisualizationError/components/AdminEmail/index.ts**

```
export * from "./AdminEmail";
```

### **frontend/src/metabase/query\_builder/components/VisualizationError/components/index.ts**

## Analyst Base

```
export * from "../AdminEmail";
```

### **frontend/src/metabase/query\_builder/components/VisualizationError/index.ts**

```
export { VisualizationError } from "../VisualizationError";
```

### **frontend/src/metabase/query\_builder/components/VisualizationError/tests/common.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import { createMockCard, createMockDatabase } from "metabase-types/api/mocks";
```

```
import { setup } from "../setup";
```

```
describe("VisualizationError (OSS)", () =>...
```

### **frontend/src/metabase/query\_builder/components/VisualizationError/tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import { createMockCard, createMockDatabase } from "metabase-types/api/mocks";
```

```
import type { SetupOpts } from "../setup";
```

```
import { setup as baseSetup } from...
```

### **frontend/src/metabase/query\_builder/components/VisualizationError/tests/premium.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import { createMockCard, createMockDatabase } from "metabase-types/api/mocks";
```

```
import type { SetupOpts } from "../setup";
```

```
import { setup as baseSetup } from...
```

### **frontend/src/metabase/query\_builder/components/VisualizationError/tests/Utils.unit.spec.ts**

```
import { adjustPositions, stripRemarks } from "../utils";
```

```
describe("adjustPositions", () => {
```

```
  const remarkedQuery =
```

```
    "-- Metabase:: userID: 1 queryType: native queryHash:...
```

### **frontend/src/metabase/query\_builder/components/Warnings.unit.spec.js**

```
import userEvent from "@testing-library/user-event";
```

```
import { render, screen } from "__support__/ui";
```

```
import Warnings from "metabase/query_builder/components/Warnings";
```

## Analyst Base

```
describe("Warnings", () =>...
```

### **frontend/src/metabase/query\_builder/components/chart-type-selector/ChartTypeList/index.ts**

```
export * from "../ChartTypeList";
```

### **frontend/src/metabase/query\_builder/components/chart-type-selector/ChartTypeOption/index.ts**

```
export * from "../ChartTypeOption";
```

### **frontend/src/metabase/query\_builder/components/chart-type-selector/ChartTypeSettings/index.ts**

```
export * from "../ChartTypeSettings";
```

### **frontend/src/metabase/query\_builder/components/chart-type-selector/index.ts**

```
export * from "../ChartTypeList";
```

```
export * from "../ChartTypeOption";
```

```
export * from "../ChartTypeSettings";
```

```
export * from "../use-question-visualization-state";
```

```
export * from "../viz-order";
```

### **frontend/src/metabase/query\_builder/components/chart-type-selector/use-question-visualization-state.ts**

```
import { useCallback } from "react";
```

```
import _ from "underscore";
```

```
import visualizations from "metabase/visualizations";
```

```
import { sanitizeResultData } from...
```

### **frontend/src/metabase/query\_builder/components/chart-type-selector/viz-order.ts**

```
import type { CardDisplayType } from "metabase-types/api";
```

```
export const DEFAULT_VIZ_ORDER: CardDisplayType[] = [
```

```
  "table",
```

```
  "bar",
```

```
  "line",
```

```
  "pie",
```

```
  "scalar",
```

```
  "row",
```

```
  "area",
```

```
  "combo",
```

```
  ...
```

## **frontend/src/metabase/query\_builder/components/dataref/QuestionPane/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/query\_builder/components/expressions/CombineColumns/util.ts**

```
import { t } from "ttag";
```

```
import { isNotNull } from "metabase/lib/types";
```

```
import * as Lib from "metabase-lib";
```

```
export type ColumnAndSeparator = {  
  separator: string | null;  
  column:...
```

## **frontend/src/metabase/query\_builder/components/expressions/CombineColumns/util.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
```

```
import * as Lib from "metabase-lib";
```

```
import { columnFinder, createQuery } from "metabase-lib/test-helpers";
```

```
import {  
  createMockDatabase,  
  ...
```

## **frontend/src/metabase/query\_builder/components/expressions/Editor/CloseModal/index.ts**

```
export * from "./CloseModal";
```

```
export * from "./utils";
```

## **frontend/src/metabase/query\_builder/components/expressions/Editor/CloseModal/utils.ts**

```
import { useCallback, useEffect, useState } from "react";
```

```
import { usePreventPopoverExit } from "metabase/ui/components/utils/PreventPopoverExit";
```

```
/**
```

```
 * useCloseModal sets up click handlers that...
```

## **frontend/src/metabase/query\_builder/components/expressions/Editor/Errors/index.ts**

```
export * from "./Errors";
```

## **frontend/src/metabase/query\_builder/components/expressions/Editor/constant**

### **s.ts**

```
export const DEBOUNCE_VALIDATION_MS = 1000;
```

### **frontend/src/metabase/query\_builder/components/expressions/Editor/extensions.ts**

```
import type { Extension } from "@codemirror/state";
import { EditorView, tooltips } from "@codemirror/view";
import { useMemo } from "react";
```

```
import { isNotNull } from "metabase/lib/types";
import {...
```

### **frontend/src/metabase/query\_builder/components/expressions/Editor/language.ts**

```
import { LRLanguage, LanguageSupport } from "@codemirror/language";
import { type Diagnostic, linter } from "@codemirror/lint";
import type { EditorView } from "@codemirror/view";
```

```
import {
  type...
```

### **frontend/src/metabase/query\_builder/components/expressions/Editor/utils.ts**

```
import {
  hasNextSnippetField,
  hasPrevSnippetField,
  snippet,
} from "@codemirror/autocomplete";
import type { EditorState } from "@codemirror/state";
import type { EditorView } from...
```

### **frontend/src/metabase/query\_builder/components/expressions/ExpressionWidget/index.ts**

```
export * from "./ExpressionWidget";
export * from "./ExpressionWidgetHeader";
```

### **frontend/src/metabase/query\_builder/components/expressions/ExtractColumn/util.ts**

```
import * as Lib from "metabase-lib";
```

```
function getNextName(names: string[], name: string, index: number): string {
  const suffixed = index === 0 ? name : `${name} (${index})`;
  if...
```

### **frontend/src/metabase/query\_builder/components/expressions/ExtractColumn/**

### **util.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import type * as Lib from "metabase-lib";
import { createQuery } from "metabase-lib/test-helpers";
import {
  createMockDatabase,
  ...
```

### **frontend/src/metabase/query\_builder/components/expressions/FunctionBrowser/Utils.ts**

```
import { t } from "ttag";

import { isNotNull } from "metabase/lib/types";
import {
  type HelpText,
  getHelpText,
  getSupportedClauses,
} from "metabase/querying/expressions";
import * as Lib from...
```

### **frontend/src/metabase/query\_builder/components/expressions/FunctionBrowser/Utils.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import { getHelpText } from "metabase/querying/expressions";
import type * as Lib from "metabase-lib";
import { createSampleDatabase } from...
```

### **frontend/src/metabase/query\_builder/components/expressions/HighlightExpression/Utils.unit.spec.ts**

```
import { highlight } from "../utils";
```

```
// By default css modules dont return classes in unit tests.
```

```
// Since we need the classnames for this test, we mock the...
```

### **frontend/src/metabase/query\_builder/components/expressions/HighlightExpression/Utils.ts**

```
import { type Highlighter, type Tag, highlightCode } from "@lezer/highlight";
```

```
import { parser } from "metabase/querying/expressions/tokenizer/parser";
import { classNameForTag } from...
```

### **frontend/src/metabase/query\_builder/components/expressions/NameInput/index.ts**

## Analyst Base

```
export * from "../NameInput";
```

**frontend/src/metabase/query\_builder/components/expressions/NameInput/utils.ts**

```
import { t } from "ttag";
```

```
import type * as Lib from "metabase-lib";
```

```
export function getPlaceholder(expressionMode: Lib.ExpressionMode) {  
  if (expressionMode === "expression") {  
    return t`Give...`  
  }  
}
```

**frontend/src/metabase/query\_builder/components/expressions/index.ts**

```
export * from "../ExpressionWidget";  
export * from "../CombineColumns";  
export * from "../ExtractColumn";
```

**frontend/src/metabase/query\_builder/components/template\_tags/SnippetForm/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/query\_builder/components/template\_tags/SnippetFormModal/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/query\_builder/components/template\_tags/SnippetRow/index.ts**

```
export * from "../SnippetRow";
```

**frontend/src/metabase/query\_builder/components/template\_tags/SnippetSidebar/index.ts**

```
export * from "../SnippetSidebar";
```

**frontend/src/metabase/query\_builder/components/template\_tags/TagEditorHelp/index.ts**

```
export { TagEditorHelp } from "../TagEditorHelp";
```

**frontend/src/metabase/query\_builder/components/template\_tags/TagEditorHelp/tests/common.unit.spec.ts**

```
import { screen } from "___support___/ui";
```

```
import { setup } from "../setup";
```

```
describe("TagEditorHelp (OSS)", () => {
```

## Analyst Base

```
it("should show a help link when `show-metabase-links: true`, () => {  
  ...
```

### **frontend/src/metabase/query\_builder/components/template\_tags/TagEditorHelp /tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";  
  
import type { SetupOpts } from "../setup";  
import { setup as baseSetup } from "../setup";
```

```
function setup(opts: SetupOpts) {  
  baseSetup({  
    ...
```

### **frontend/src/metabase/query\_builder/components/template\_tags/TagEditorHelp /tests/premium.unit.spec.ts**

```
import { screen } from "__support__/ui";  
  
import type { SetupOpts } from "../setup";  
import { setup as baseSetup } from "../setup";
```

```
function setup(opts: SetupOpts) {  
  baseSetup({  
    ...
```

### **frontend/src/metabase/query\_builder/components/template\_tags/types.ts**

```
export interface WidgetOption {  
  name?: string;  
  menuName?: string;  
  type: string;  
}
```

### **frontend/src/metabase/query\_builder/components/view/DataReferenceButton/index.ts**

```
export * from "../DataReferenceButton";
```

### **frontend/src/metabase/query\_builder/components/view/ExecutionTime/index.ts**

```
export * from "../ExecutionTime";
```

### **frontend/src/metabase/query\_builder/components/view/ExecutionTime/utils.ts**

```
import { t } from "ttag";  
  
export const formatDuration = (time: number): string => {  
  if (time < 1000) {
```



## Analyst Base

```
return t`${time}ms`;  
}
```

```
return t`${(time / 1000).toFixed(1)}s`;  
};
```

### **frontend/src/metabase/query\_builder/components/view/ExecutionTime/utils.unit.spec.ts**

```
import { formatDuration } from "./utils";
```

```
describe("formatDuration", () => {  
  it("formats duration correctly", () => {  
    expect(formatDuration(100)).toBe("100ms");  
    ...  
  });  
});
```

### **frontend/src/metabase/query\_builder/components/view/NativeCodePanel/index.ts**

```
export { NativeCodePanel } from "./NativeCodePanel";
```

### **frontend/src/metabase/query\_builder/components/view/NativeQueryPreview/index.ts**

```
export { NativeQueryPreview } from "./NativeQueryPreview";
```

### **frontend/src/metabase/query\_builder/components/view/NativeVariablesButton/index.ts**

```
export * from "./NativeVariablesButton";
```

### **frontend/src/metabase/query\_builder/components/view/PreviewQueryButton/index.ts**

```
export * from "./PreviewQueryButton";
```

### **frontend/src/metabase/query\_builder/components/view/PreviewQueryModal/index.ts**

```
export { PreviewQueryModal } from "./PreviewQueryModal";
```

### **frontend/src/metabase/query\_builder/components/view/QuestionRowCount/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/query\_builder/components/view/QuestionTimelineWidget/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/query\_builder/components/view/SnippetSidebarButton/index.ts**

## **index.ts**

```
export * from "../SnippetSidebarButton";
```

## **frontend/src/metabase/query\_builder/components/view/View/NotebookContainer/index.ts**

```
export { NotebookContainer } from "../NotebookContainer";
```

## **frontend/src/metabase/query\_builder/components/view/View/View/index.ts**

```
export * from "../View";
```

## **frontend/src/metabase/query\_builder/components/view/View/ViewHeaderContainer/index.ts**

```
export * from "../ViewHeaderContainer";
```

## **frontend/src/metabase/query\_builder/components/view/View/ViewLeftSidebarContainer/index.ts**

```
export * from "../ViewLeftSidebarContainer";
```

## **frontend/src/metabase/query\_builder/components/view/View/ViewMainContainer/index.ts**

```
export * from "../ViewMainContainer";
```

## **frontend/src/metabase/query\_builder/components/view/View/ViewNativeQueryEditor/index.ts**

```
export * from "../ViewNativeQueryEditor";
```

## **frontend/src/metabase/query\_builder/components/view/View/ViewRightSidebarContainer/index.ts**

```
export * from "../ViewRightSidebarContainer";
```

## **frontend/src/metabase/query\_builder/components/view/View/index.ts**

```
export { View } from "../View";
```

## **frontend/src/metabase/query\_builder/components/view/ViewFooter/index.ts**

```
export * from "../ViewFooter";
```

## **frontend/src/metabase/query\_builder/components/view/ViewHeader/ViewTitleHeader.unit.spec.js**

```
import userEvent from "@testing-library/user-event";
```

```
import fetchMock from "fetch-mock";
```

```
import { Route } from "react-router";
```

```
import _ from "underscore";
```

```
import { setupGetUserKeyValueEndpoint }...
```

## Analyst Base

**frontend/src/metabase/query\_builder/components/view/ViewHeader/components/AdHocQuestionDescription/index.ts**

```
export { AdHocQuestionDescription } from "./AdHocQuestionDescription";
```

**frontend/src/metabase/query\_builder/components/view/ViewHeader/components/AdHocQuestionLeftSide/index.ts**

```
export * from "./AdHocQuestionLeftSide";
```

**frontend/src/metabase/query\_builder/components/view/ViewHeader/components/DashboardBackButton/index.ts**

```
export * from "./DashboardBackButton";
```

**frontend/src/metabase/query\_builder/components/view/ViewHeader/components/QuestionActions/QuestionMoreActionsMenu/index.ts**

```
export * from "./QuestionMoreActionsMenu";
```

**frontend/src/metabase/query\_builder/components/view/ViewHeader/components/QuestionActions/index.ts**

```
export { QuestionActions } from "./QuestionActions";
```

**frontend/src/metabase/query\_builder/components/view/ViewHeader/components/QuestionDataSource/QuestionDataSource.unit.spec.js**

```
eslint-disable react/display-name, react/prop-types
```

**frontend/src/metabase/query\_builder/components/view/ViewHeader/components/QuestionDataSource/index.ts**

```
export { QuestionDataSource } from "./QuestionDataSource";
```

**frontend/src/metabase/query\_builder/components/view/ViewHeader/components/QuestionDataSource/utils.unit.spec.ts**

```
import { isValidElement } from "react";
```

```
import { createMockMetadata } from "__support__/metadata";
```

```
import Question from "metabase-lib/v1/Question";
```

```
import type { Card } from...
```

**frontend/src/metabase/query\_builder/components/view/ViewHeader/components/QuestionFiltersHeader/index.ts**

```
export * from "./QuestionFiltersHeader";
```

**frontend/src/metabase/query\_builder/components/view/ViewHeader/components/QuestionNotebookButton/index.ts**

```
export * from "./QuestionNotebookButton";
```

## **frontend/src/metabase/query\_builder/components/view/ViewHeader/components/SavedQuestionLeftSide/hooks.ts**

```
import { useSelector } from "metabase/lib/redux";
import { getMetadataUnfiltered } from "metabase/selectors/metadata";
import * as Lib from "metabase-lib";
import type Question from...
```

## **frontend/src/metabase/query\_builder/components/view/ViewHeader/components/SavedQuestionLeftSide/index.ts**

```
export * from "./SavedQuestionLeftSide";
```

## **frontend/src/metabase/query\_builder/components/view/ViewHeader/components/ToggleNativeQueryPreview/index.ts**

```
export { ToggleNativeQueryPreview } from "./ToggleNativeQueryPreview";
```

## **frontend/src/metabase/query\_builder/components/view/ViewHeader/components/ViewTitleHeaderRightSide/index.ts**

```
export * from "./ViewTitleHeaderRightSide";
```

## **frontend/src/metabase/query\_builder/components/view/ViewHeader/components/index.ts**

```
export * from "./ToggleNativeQueryPreview";
export * from "./HeaderBreadcrumbs/HeaderBreadcrumbs";
export * from "./QuestionActions";
export * from "./QuestionDataSource";
export * from...
```

## **frontend/src/metabase/query\_builder/components/view/ViewHeader/index.ts**

```
export { ViewTitleHeader } from "./ViewTitleHeader";
```

## **frontend/src/metabase/query\_builder/components/view/ViewHeader/utils.ts**

```
import * as Lib from "metabase-lib";
import type Question from "metabase-lib/v1/Question";
```

```
/**
```

```
 * We can only "explore results" (i.e. create new questions based on this one)
```

```
 * when question is a...
```

## **frontend/src/metabase/query\_builder/components/view/ViewHeader/units/Spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import Question from "metabase-lib/v1/Question";
import {
```

## Analyst Base

```
createMockCard,  
createMockNativeDatasetQuery,  
createMockParameter,  
...
```

### **frontend/src/metabase/query\_builder/components/view/ViewSidebar/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/query\_builder/components/view/sidebars/ChartSettings Sidebar.unit.spec.js**

```
import { createMockMetadata } from "__support__/metadata";  
import { fireEvent, renderWithProviders, screen } from "__support__/ui";  
import registerVisualizations from...
```

### **frontend/src/metabase/query\_builder/components/view/sidebars/ChartTypeSide bar/index.ts**

```
export * from "./ChartTypeSidebar";
```

### **frontend/src/metabase/query\_builder/components/view/sidebars/DatasetManag ementSection/DatasetMetadataStrengthIndicator/DatasetMetadataStrenghtIndic ator.unit.spec.js**

```
import { render, screen } from "__support__/ui";
```

```
import DatasetMetadataStrengthIndicator from "./DatasetMetadataStrengthIndicator";
```

```
function setup({ resultMetadata } = {}) {  
  const mockDataset =...
```

### **frontend/src/metabase/query\_builder/components/view/sidebars/DatasetManag ementSection/DatasetMetadataStrengthIndicator/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/query\_builder/components/view/sidebars/ModelCacheM anagementSection/index.ts**

```
export * from "./ModelCacheManagementSection";
```

### **frontend/src/metabase/query\_builder/components/view/sidebars/QuestionInfoS idebar/components/hooks.ts**

```
import { useState } from "react";
```

```
export const DEFAULT_LIST_LIMIT = 5;
```

```
export const useExpandableList = (arr: any[], limit = DEFAULT_LIST_LIMIT) => {  
  const [isExpanded, setIsExpanded] =...
```

## **frontend/src/metabase/query\_builder/components/view/sidebar/QuestionInfoSidebar/components/types.ts**

```
import type { IconData } from "metabase/lib/icon";
```

```
export interface QuestionSource {  
  href: string;  
  name: string;  
  model?: string;  
  iconProps?: IconData;  
}
```

## **frontend/src/metabase/query\_builder/components/view/sidebar/QuestionInfoSidebar/components/utils.unit.spec.ts**

```
import { createMockMetadata } from "___support___/metadata";  
import Question from "metabase-lib/v1/Question";  
import { createMockCard } from "metabase-types/api/mocks";  
import {  
  SAMPLE_DB_ID,  
  ...
```

## **frontend/src/metabase/query\_builder/components/view/sidebar/QuestionInfoSidebar/index.ts**

```
export { QuestionInfoSidebar } from "./QuestionInfoSidebar";
```

## **frontend/src/metabase/query\_builder/components/view/sidebar/QuestionSettingsSidebar/index.ts**

```
export * from "./QuestionSettingsSidebar";
```

## **frontend/src/metabase/query\_builder/components/view/sidebar/SummarizeSidebar/AddAggregationButton/index.ts**

```
export * from "./AddAggregationButton";
```

## **frontend/src/metabase/query\_builder/components/view/sidebar/SummarizeSidebar/AggregationItem/index.ts**

```
export * from "./AggregationItem";
```

## **frontend/src/metabase/query\_builder/components/view/sidebar/SummarizeSidebar/BreakoutColumnList/BreakoutColumnListItem/index.ts**

```
export * from "./BreakoutColumnListItem";
```

## **frontend/src/metabase/query\_builder/components/view/sidebar/SummarizeSidebar/BreakoutColumnList/index.ts**

```
export * from "./BreakoutColumnList";
```

## Analyst Base

```
export * from "./util";
export * from "./types";
```

### **frontend/src/metabase/query\_builder/components/view/sidebars/SummarizeSidebar/BreakoutColumnList/types.ts**

```
import type * as Lib from "metabase-lib";

export type ListItem = Lib.ColumnDisplayInfo & {
  column: Lib.ColumnMetadata;
  breakout?: Lib.BreakoutClause;
};
```

```
export type ListSection = {
  name:...
```

### **frontend/src/metabase/query\_builder/components/view/sidebars/SummarizeSidebar/BreakoutColumnList/util.ts**

```
import * as Lib from "metabase-lib";

import type { ListItem, ListSection } from "./types";

export function getBreakoutListItem(
  query: Lib.Query,
  stageIndex: number,
  breakout:...
```

### **frontend/src/metabase/query\_builder/components/view/sidebars/SummarizeSidebar/SummarizeContent/index.ts**

```
export * from "./SummarizeAggregationItemList";
export * from "./SummarizeBreakoutColumnList";
```

### **frontend/src/metabase/query\_builder/components/view/sidebars/SummarizeSidebar/index.ts**

```
export * from "./SummarizeSidebar";
```

### **frontend/src/metabase/query\_builder/components/view/sidebars/TimelineSidebar/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/query\_builder/constants.ts**

```
export const MODAL_TYPES = {
  SAVE: "save",
  ADD_TO_DASHBOARD: "add-to-dashboard",
  MOVE: "move",
```

## Analyst Base

```
CLONE: "clone",  
ARCHIVE: "archive",  
SAVED: "saved",  
ADD_TO_DASHBOARD_SAVE:...
```

### **frontend/src/metabase/query\_builder/containers/use-create-question.ts**

```
import { useCallback } from "react";  
  
import type { ScheduleCallback } from "metabase/common/hooks/use-callback-effect";  
import { useDispatch } from "metabase/lib/redux";  
import {  
  ...
```

### **frontend/src/metabase/query\_builder/containers/use-save-question.ts**

```
import { useCallback } from "react";  
  
import type { ScheduleCallback } from "metabase/common/hooks/use-callback-effect";  
import { useDispatch } from "metabase/lib/redux";  
import type Question from...
```

### **frontend/src/metabase/query\_builder/defaults.ts**

```
import type {  
  QueryBuilderLoadingControls,  
  QueryBuilderQueryStatus,  
  QueryBuilderUIControls,  
} from "metabase-types/store";  
  
export const DEFAULT_UI_CONTROLS: QueryBuilderUIControls = {  
  ...
```

### **frontend/src/metabase/query\_builder/hooks/index.ts**

```
export * from "./use-breakout-query-handlers";  
export * from "./use-default-query-aggregation";  
export * from "./types";
```

### **frontend/src/metabase/query\_builder/hooks/types.ts**

```
import type * as Lib from "metabase-lib";  
  
export type UpdateQueryHookProps = {  
  query: Lib.Query;  
  onQueryChange: (nextQuery: Lib.Query) => void;  
  stageIndex: number;  
};
```

### **frontend/src/metabase/query\_builder/hooks/use-breakout-query-handlers.ts**



## Analyst Base

```
import { useCallback } from "react";
```

```
import * as Lib from "metabase-lib";
```

```
import type { UpdateQueryHookProps } from "../types";
```

```
export const useBreakoutQueryHandlers = ({  
  query,  
  onQueryChange,  
  ...  
})
```

### **frontend/src/metabase/query\_builder/hooks/use-default-query-aggregation.ts**

```
import { useCallback, useMemo, useState } from "react";
```

```
import * as Lib from "metabase-lib";
```

```
import type { UpdateQueryHookProps } from "../types";
```

```
export const useDefaultQueryAggregation = ({  
  ...  
})
```

### **frontend/src/metabase/query\_builder/hooks/use-notebook-screen-size.ts**

```
import { useWindowSize } from "react-use";
```

```
const INITIAL_WINDOW_WIDTH = Infinity;
```

```
const BREAKPOINT = 1280;
```

```
const NOT_MOUNTED_YET = undefined;
```

```
type NotMountedYet = typeof NOT_MOUNTED_YET;
```

```
type...
```

### **frontend/src/metabase/query\_builder/hooks/use-sync-url-parameters.ts**

```
import querystring from "querystring";
```

```
import { useEffect, useMemo } from "react";
```

```
import { IS_EMBED_PREVIEW } from "metabase/lib/embed";
```

```
import { getParameterValuesBySlug } from...
```

### **frontend/src/metabase/query\_builder/reducers-typed.ts**

```
import { createReducer } from "@reduxjs/toolkit";
```

## Analyst Base

```
import {
  createCardPublicLink,
  deleteCardPublicLink,
  updateCardEmbeddingParams,
  updateCardEnableEmbedding,
} from "metabase/api";
import...
```

### **frontend/src/metabase/query\_builder/reducers.js**

```
import { assoc, merge } from "icepick";
import { handleActions } from "redux-actions";
import _ from "underscore";
```

```
import {
  EDIT_QUESTION,
  NAVIGATE_TO_NEW_CARD,
} from...
```

### **frontend/src/metabase/query\_builder/selectors/mode.js**

```
import { createSelector } from "@reduxjs/toolkit";

import { getMetadata } from "metabase/selectors/metadata";
import { getMode as getQuestionMode } from...
```

### **frontend/src/metabase/query\_builder/selectors.js**

```
eslint no-use-before-define: "error"
```

### **frontend/src/metabase/query\_builder/selectors.unit.spec.js**

```
import { assoc, assocIn } from "icepick";

import { createMockEntitiesState } from "__support__/store";
import {
  getIsResultDirty,
  getIsVisualized,
  getNativeEditorCursorOffset,
  ...
```

### **frontend/src/metabase/query\_builder/typed-utils.ts**

```
import type { LocationDescriptorObject } from "history";

import type { DatasetEditorTab, QueryBuilderMode } from "metabase-types/store";

type LocationQBModeResult = {
  queryBuilderMode:...
```

## Analyst Base

### **frontend/src/metabase/query\_builder/utils/get-aggregation-items.ts**

```
import * as Lib from "metabase-lib";

import type { UpdateQueryHookProps } from "../hooks/types";

export type AggregationItem = {
  aggregation: Lib.AggregationClause;
  aggregationIndex: number;
  ...
```

### **frontend/src/metabase/query\_builder/utils/index.ts**

```
import type { Location } from "history";
import querystring from "querystring";
import _ from "underscore";

import { serializeCardForUrl } from "metabase/lib/card";
import * as Urls from...
```

### **frontend/src/metabase/query\_builder/utils/question.ts**

```
import * as Lib from "metabase-lib";
import type Question from "metabase-lib/v1/Question";
import { isAdHocModelOrMetricQuestion } from "metabase-lib/v1/metadata/utils/models";

export function...
```

### **frontend/src/metabase/query\_builder/utils/units.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import { getNextId } from "__support__/utils";
import { serializeCardForUrl } from "metabase/lib/card";
import { checkNotNull } from...
```

### **frontend/src/metabase/querying/analytics.ts**

```
import { trackSchemaEvent } from "metabase/lib/analytics";
import * as Lib from "metabase-lib";
import type Question from "metabase-lib/v1/Question";
```

```
export const trackColumnCombineViaColumnHeader =...
```

### **frontend/src/metabase/querying/drills/utils/constants.ts**

```
import type { Drill } from "metabase/visualizations/types";
import type * as Lib from "metabase-lib";

import { automaticInsightsDrill } from "../automatic-insights-drill";
import { columnExtractDrill...
```

## Analyst Base

### **frontend/src/metabase/querying/drills/utils/distribution-drill.ts**

```
import { t } from "ttag";

import type { Drill } from "metabase/visualizations/types/click-actions";
import type * as Lib from "metabase-lib";

export const distributionDrill:...
```

### **frontend/src/metabase/querying/drills/utils/fk-details-drill.ts**

```
import { t } from "ttag";

import type { Drill } from "metabase/visualizations/types/click-actions";
import type * as Lib from "metabase-lib";

export const fkDetailsDrill:...
```

### **frontend/src/metabase/querying/drills/utils/fk-filter-drill.ts**

```
import dayjs from "dayjs";
import { t } from "ttag";

import {
  pluralize,
  singularize,
  stripId,
} from "metabase/lib/formatting/strings";
import type { Drill } from...
```

### **frontend/src/metabase/querying/drills/utils/pk-drill.ts**

```
import { t } from "ttag";

import type { Drill } from "metabase/visualizations/types/click-actions";
import type * as Lib from "metabase-lib";

export const pkDrill: Drill<Lib.PKDrillThruInfo> = ({
  ...
```

### **frontend/src/metabase/querying/drills/utils/query-drill.ts**

```
import { isNotNull } from "metabase/lib/types";
import type { ClickAction } from "metabase/visualizations/types";
import type { DrillThruDisplayInfo } from "metabase-lib";
import * as Lib from...
```

### **frontend/src/metabase/querying/drills/utils/sort-drill.ts**

```
import { t } from "ttag";
```

## Analyst Base

```
import type {
  ClickActionBase,
  Drill,
} from "metabase/visualizations/types/click-actions";
import type * as Lib from "metabase-lib";
```

```
const ACTIONS: Record<string,...
```

### **frontend/src/metabase/querying/drills/utils/summarize-column-by-time-drill.ts**

```
import { t } from "ttag";
```

```
import type { Drill } from "metabase/visualizations/types/click-actions";
import type * as Lib from "metabase-lib";
```

```
export const summarizeColumnByTimeDrill: Drill<
...

```

### **frontend/src/metabase/querying/drills/utils/summarize-column-drill.ts**

```
import { t } from "ttag";
```

```
import type {
  ClickActionBase,
  Drill,
} from "metabase/visualizations/types/click-actions";
import type * as Lib from "metabase-lib";
import type { Dispatch } from...
```

### **frontend/src/metabase/querying/drills/utils/underlying-records-drill.ts**

```
import dayjs from "dayjs";
import { msgid, ngettext } from "ttag";
```

```
import { inflect } from "metabase/lib/formatting/strings";
import type {
  Drill,
  QuestionChangeClickAction,
} from...
```

### **frontend/src/metabase/querying/drills/utils/zoom-drill.ts**

```
import { t } from "ttag";
```

```
import { zoomInRow } from "metabase/query_builder/actions";
import type { Drill } from "metabase/visualizations/types/click-actions";
import type * as Lib from...
```

## Analyst Base

### **frontend/src/metabase/querying/drills/utils/zoom-in-binning-drill.ts**

```
import { t } from "ttag";

import type { Drill } from "metabase/visualizations/types";
import type * as Lib from "metabase-lib";

export const zoomInBinningDrill: Drill<Lib.ZoomDrillThruInfo> = ({
  ...
```

### **frontend/src/metabase/querying/drills/utils/zoom-in-geographic-drill.ts**

```
import { t } from "ttag";

import type { Drill } from "metabase/visualizations/types";
import type * as Lib from "metabase-lib";

export const zoomInGeographicDrill: Drill<Lib.ZoomDrillThruInfo> = ...
```

### **frontend/src/metabase/querying/drills/utils/zoom-in-timeseries-drill.ts**

```
import type { Drill } from "metabase/visualizations/types";
import type * as Lib from "metabase-lib";

export const zoomInTimeseriesDrill: Drill<Lib.ZoomTimeseriesDrillThruInfo> = ({
  drill,
  ...
```

### **frontend/src/metabase/querying/expressions/clause.ts**

```
import _ from "underscore";

import { isNotNull } from "metabase/lib/types";
import type * as Lib from "metabase-lib";
import type Database from "metabase-lib/v1/metadata/Database";

import {
  ...
```

### **frontend/src/metabase/querying/expressions/compile-expression.ts**

```
import * as Lib from "metabase-lib";

import { type ExpressionError, renderError } from "./errors";
import { compile, lexify, parse } from "./pratt";
import { type Resolver, resolver as ...
```

### **frontend/src/metabase/querying/expressions/compile-expression.unit.spec.ts**

```
import * as Lib from "metabase-lib";
```

## Analyst Base

```
import { compileExpression } from "../compile-expression";
import {
  expressions,
  fields,
  query,
  segments,
  stageIndex,
} from "../test/shared";
import {...
```

### **frontend/src/metabase/querying/expressions/config.ts**

```
import dayjs from "dayjs";
import { t } from "tag";
```

```
import type * as Lib from "metabase-lib";
```

```
import { defineClauses, dimension, op } from "../define";
import { MBQLClauseCategory as CATEGORY }...
```

### **frontend/src/metabase/querying/expressions/define.ts**

```
import type { MBQLClauseDefinition } from "../types";
```

```
const names = new Set();
```

```
export function defineClauses<
  const T extends Record<string, MBQLClauseDefinition>,
>(
  options:...
```

### **frontend/src/metabase/querying/expressions/diagnostics/diagnostics.ts**

```
import type * as Lib from "metabase-lib";
import type Metadata from "metabase-lib/v1/metadata/Metadata";
```

```
import { compileExpression } from "../compile-expression";
import type { ExpressionError }...
```

### **frontend/src/metabase/querying/expressions/diagnostics/diagnostics.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import * as Lib from "metabase-lib";
import type Metadata from "metabase-lib/v1/metadata/Metadata";
import { createSampleDatabase } from...
```

## **frontend/src/metabase/querying/expressions/diagnostics/expression/check-arg-count.ts**

```
import { msgid, ngettext } from "ttag";
```

```
import * as Lib from "metabase-lib";
```

```
import { getClauseDefinition } from "../../clause";
```

```
import { visit } from "../../visitor";
```

```
import { error } from...
```

## **frontend/src/metabase/querying/expressions/diagnostics/expression/check-arg-validators.ts**

```
import * as Lib from "metabase-lib";
```

```
import { getClauseDefinition } from "../../clause";
```

```
import { visit } from "../../visitor";
```

```
import { error } from "../../utils";
```

```
export function...
```

## **frontend/src/metabase/querying/expressions/diagnostics/expression/check-case-or-if-arg-count.ts**

```
import { t } from "ttag";
```

```
import * as Lib from "metabase-lib";
```

```
import { visit } from "../../visitor";
```

```
import { error } from "../../utils";
```

```
export function checkCaseOrIfArgCount({
```

```
...
```

## **frontend/src/metabase/querying/expressions/diagnostics/expression/check-comparison-operator-args.ts**

```
import { t } from "ttag";
```

```
import * as Lib from "metabase-lib";
```

```
import { COMPARISON, EQUALITY } from "../../pratt";
```

```
import { parsePunctuator } from "../../punctuator";
```

```
import { visit } from...
```

## **frontend/src/metabase/querying/expressions/diagnostics/expression/check-function-args.ts**



## **ctions-for-expression-mode.ts**

```
import * as Lib from "metabase-lib";

import { checkExpressionModeSupportsClause } from "../mode";
import { visit } from "../visitor";

export function checkFunctionsForExpressionMode({
  ...
```

## **frontend/src/metabase/querying/expressions/diagnostics/expression/check-known-functions.ts**

```
import { t } from "ttag";

import * as Lib from "metabase-lib";

import { getClauseDefinition } from "../clause";
import { visit } from "../visitor";
import { error } from "../utils";

export...
```

## **frontend/src/metabase/querying/expressions/diagnostics/expression/check-lib-diagnostics.ts**

```
import * as Lib from "metabase-lib";

import { DiagnosticError } from "../errors";

export function checkLibDiagnostics({
  query,
  stageIndex,
  expressionMode,
  expressionClause,
  ...
```

## **frontend/src/metabase/querying/expressions/diagnostics/expression/check-supported-functions.ts**

```
import { t } from "ttag";

import * as Lib from "metabase-lib";
import type Metadata from "metabase-lib/v1/metadata/Metadata";

import { getClauseDefinition } from "../clause";
import {...
```

## **frontend/src/metabase/querying/expressions/diagnostics/expression/index.ts**

```
import type * as Lib from "metabase-lib";
import type Metadata from "metabase-lib/v1/metadata/Metadata";
```

```
import { checkArgCount } from "../check-arg-count";
import { checkArgValidators } from...
```

## **frontend/src/metabase/querying/expressions/diagnostics/index.ts**

```
export * from "../diagnostics";
```

## **frontend/src/metabase/querying/expressions/diagnostics/syntax/check-bad-tokens.ts**

```
import { t } from "ttag";

import { BAD_TOKEN, type Token } from "../../pratt";
import { error } from "../../utils";

export function checkBadTokens({ tokens }: { tokens: Token[] }) {
  for (const token...
```

## **frontend/src/metabase/querying/expressions/diagnostics/syntax/check-double-commas.ts**

```
import { t } from "ttag";

import { COMMA, type Token } from "../../pratt";
import { error } from "../../utils";

export function checkDoubleCommas({ tokens }: { tokens: Token[] }) {
  for (let i = 1; i...
```

## **frontend/src/metabase/querying/expressions/diagnostics/syntax/check-field-quotes.ts**

```
import { t } from "ttag";

import { FIELD, type Token } from "../../pratt";
import { quoteString } from "../../string";
import { error } from "../../utils";

export function checkFieldQuotes({ tokens }:
```

## **frontend/src/metabase/querying/expressions/diagnostics/syntax/check-matching-parenthesis.ts**

```
import { t } from "ttag";
```

## Analyst Base

```
import type { Token } from "../pratt";
import { GROUP, GROUP_CLOSE } from "../pratt";
import { error } from "../utils";
```

```
export function checkMatchingParentheses({...
```

### **frontend/src/metabase/querying/expressions/diagnostics/syntax/check-mismatched-siblings.ts**

```
import { t } from "ttag";
```

```
import {
  BAD_TOKEN,
  BOOLEAN,
  CALL,
  END_OF_INPUT,
  FIELD,
  GROUP,
  GROUP_CLOSE,
  IDENTIFIER,
  NUMBER,
  STRING,
  type Token,
} from "../pratt";
import {...
```

### **frontend/src/metabase/querying/expressions/diagnostics/syntax/check-number-exponent.ts**

```
import { t } from "ttag";
```

```
import { NUMBER, type Token } from "../pratt";
import { error } from "../utils";
```

```
export function checkNumberExponent({ tokens }: { tokens: Token[] }) {
  for (const...
```

### **frontend/src/metabase/querying/expressions/diagnostics/syntax/check-open-parenthesis-after-function.ts**

```
import { t } from "ttag";
```

```
import { getClauseDefinition, getMBQLName } from "../clause";
import { GROUP, IDENTIFIER, type Token } from "../pratt";
import { error } from "../utils";
```

export...

### **frontend/src/metabase/querying/expressions/diagnostics/syntax/check-string-quotes.ts**

import { t } from "ttag";

import { STRING, type Token } from "../../pratt";

import { quoteString } from "../../string";

import { error } from "../../utils";

export function checkStringQuotes({ tokens...

### **frontend/src/metabase/querying/expressions/diagnostics/syntax/index.ts**

import type { Token } from "../../pratt";

import { checkBadTokens } from "../check-bad-tokens";

import { checkDoubleCommas } from "../check-double-commas";

import { checkFieldQuotes } from...

### **frontend/src/metabase/querying/expressions/diagnostics/utils.ts**

import type \* as Lib from "metabase-lib";

import { DiagnosticError } from "../errors";

import { type Node, Token } from "../../pratt";

export function error(message: string): never;

export function...

### **frontend/src/metabase/querying/expressions/diagnostics/utils.unit.spec.ts**

import { CALL, Token } from "../../pratt";

import { position } from "../utils";

describe("position", () => {

it("returns the correct position", () => {

const token = new Token({

type:...

### **frontend/src/metabase/querying/expressions/errors.ts**

import { t } from "ttag";

import type { Node } from "../../pratt";

## Analyst Base

```
/*  
 * This class helps anything that handles parser errors to use instanceof to  
 * easily distinguish between compilation error...
```

### **frontend/src/metabase/querying/expressions/formatter/formatter.ts**

```
import type { AstPath, Doc, ParserOptions, Plugin } from "prettier";  
import { builders } from "prettier/doc";  
import { format as pformat } from "prettier/standalone";
```

```
import { parseNumber } from...
```

### **frontend/src/metabase/querying/expressions/formatter/formatter.unit.spec.ts**

```
eslint-disable jest/expect-expect
```

### **frontend/src/metabase/querying/expressions/formatter/index.ts**

```
export * from "../formatter";
```

### **frontend/src/metabase/querying/expressions/formatter/utils.ts**

```
import type { AstPath } from "prettier";
```

```
import * as Lib from "metabase-lib";
```

```
import { EXPRESSION_OPERATORS } from "../config";  
import * as literal from "../literal";
```

```
type Assertion<T> = T extends...
```

### **frontend/src/metabase/querying/expressions/fuzz.compiler.unit.spec.ts**

```
import * as Lib from "metabase-lib";
```

```
import { compileExpression } from "../compile-expression";  
import { fuzz } from "../test/fuzz";  
import { generateExpression } from "../test/generator";  
import {...
```

### **frontend/src/metabase/querying/expressions/fuzz.string.unit.spec.ts**

```
import { quoteString, unquoteString } from "../string";  
import { fuzz } from "../test/fuzz";  
import { createRandom } from "../test/generator";
```

```
const MAX_SEED = 1000;
```

```
const simple = [  
  ...
```

## Analyst Base

### **frontend/src/metabase/querying/expressions/help-text.ts**

```
import { isNotNull } from "metabase/lib/types";
import type * as Lib from "metabase-lib";
import type Database from "metabase-lib/v1/metadata/Database";
```

```
import { getClauseDefinition } from...
```

### **frontend/src/metabase/querying/expressions/help-text.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import { checkNotNull } from "metabase/lib/types";
import type { Database } from "metabase-types/api";
import { createMockDatabase } from...
```

### **frontend/src/metabase/querying/expressions/identifier.ts**

```
import { FK_SYMBOL } from "metabase/lib/formatting/constants";
```

```
import { quoteString } from "./string";
```

```
const IDENTIFIER_QUOTE = "[";
```

```
export const EDITOR_FK_SYMBOLS = {
  // specifies which symbols...
```

### **frontend/src/metabase/querying/expressions/index.ts**

```
export * from "./clause";
export * from "./diagnostics";
export * from "./errors";
export * from "./formatter";
export * from "./help-text";
export * from "./mode";
export * from "./position";
export...
```

### **frontend/src/metabase/querying/expressions/literal.ts**

```
import type {
  BooleanLiteral,
  NumericLiteral,
  StringLiteral,
} from "metabase-types/api";
```

```
export function isLiteral(
  x: unknown,
): x is StringLiteral | NumericLiteral | BooleanLiteral {
```

...

### **frontend/src/metabase/querying/expressions/mode.ts**

```
import { t } from "ttag";
```

```
import * as Lib from "metabase-lib";
```

```
import { getClauseDefinition, isDefinedClause } from "../clause";
```

```
import { DiagnosticError } from "../errors";
```

```
const...
```

### **frontend/src/metabase/querying/expressions/position.ts**

```
import { getMBQLName } from "../clause";
```

```
import { END_OF_INPUT, STRING, type Token, lexify } from "../pratt";
```

```
import { parser } from "../tokenizer/parser";
```

```
export function enclosingFunction(doc:...
```

### **frontend/src/metabase/querying/expressions/position.unit.spec.ts**

```
import expression from "ts-dedent";
```

```
import { enclosingFunction } from "../position";
```

```
describe("enclosingFunction", () => {
```

```
  function setup(doc: string) {
```

```
    const pos = doc.indexOf("|");
```

```
    if...
```

### **frontend/src/metabase/querying/expressions/pratt/compiler.ts**

```
import { t } from "ttag";
```

```
import { type NumberValue, parseNumber } from "metabase/lib/number";
```

```
import * as Lib from "metabase-lib";
```

```
import { getClauseDefinition, getMBQLName, isDefinedClause } from...
```

### **frontend/src/metabase/querying/expressions/pratt/compiler.unit.spec.ts**

```
import * as Lib from "metabase-lib";
```

```
import { compileExpression } from "../compile-expression";
```

```
import { query } from "../test/shared";
```

```
import { value } from "../test/utils";
```

```
import type {...
```

## Analyst Base

### **frontend/src/metabase/querying/expressions/pratt/fuzz.lexifier.unit.spec.ts**

```
import { lexify } from "../pratt/lexifier";
import { fuzz } from "../test/fuzz";
import { generateExpression } from "../test/generator";

describe("metabase/querying/expressions/tokenizer", () => {
  ...
```

### **frontend/src/metabase/querying/expressions/pratt/index.ts**

```
export * from "../lexifier";
export * from "../parser";
export * from "../compiler";
export * from "../syntax";
export * from "../node";
export * from "../token";
```

### **frontend/src/metabase/querying/expressions/pratt/lexifier.ts**

```
import type { SyntaxNodeRef } from "@lezer/common";

import { parsePunctuator } from "../punctuator";
import { unquoteString } from "../string";
import { tokenize } from "../tokenizer";

import type...
```

### **frontend/src/metabase/querying/expressions/pratt/lexifier.unit.spec.ts**

```
import { lexify } from "../lexifier";
import type { NodeType } from "../node";
import {
  ADD,
  BAD_TOKEN,
  BOOLEAN,
  CALL,
  COMMA,
  COMPARISON,
  END_OF_INPUT,
  FIELD,
  GROUP,
  GROUP_CLOSE,
  ...
```

### **frontend/src/metabase/querying/expressions/pratt/node.ts**

```
import type { Token } from "../token";
```



## Analyst Base

```
export type NodeType = {
  name?: string;

  // Number of operands to expect for this node on the left side
  leftOperands: number;
  // Number of operands to...
```

### **frontend/src/metabase/querying/expressions/pratt/parser.ts**

```
import { t } from "ttag";

import { CompileError } from "../errors";
import type { Hooks } from "../types";
import { assert } from "../utils";

import type { Node, NodeType } from "../node";
import {
  ...
```

### **frontend/src/metabase/querying/expressions/pratt/parser.unit.spec.ts**

```
import { lexify, parse } from ".";

describe("pratt/parser", () => {
  function parseExpression(source: string) {
    const tokens = lexify(source);
    return parse(tokens, {
      hooks: {
        ...
```

### **frontend/src/metabase/querying/expressions/pratt/syntax.ts**

```
import type { Node, NodeType } from "../node";

/*
 * This file specifies most of the syntax for the Metabase handwritten custom
 * expression parser. The rest is contained in the parser special...
```

### **frontend/src/metabase/querying/expressions/pratt/token.ts**

```
import type { NodeType } from "../node";

export class Token {
  type: NodeType;
  text: string;
  value?: string;
```

## Analyst Base

```
start: number;  
end: number;
```

```
constructor({  
  type,  
  start,  
  end,  
  ...  
})
```

### **frontend/src/metabase/querying/expressions/punctuator.ts**

```
import type { NodeType } from "../pratt";  
import { NODE_TYPE as t } from "../pratt/syntax";  
  
export type Punctuator = keyof typeof PUNCTUATOR_TO_TYPE;
```

```
const PUNCTUATOR_TO_TYPE = {  
  ",": t.COMMA,  
  ...  
}
```

### **frontend/src/metabase/querying/expressions/resolver.ts**

```
import { c, t } from "ttag";  
import _ from "underscore";  
  
import * as Lib from "metabase-lib";  
  
import { CompileError } from "../errors";  
import { EDITOR_FK_SYMBOLS, getDisplayNameWithSeparator } from...
```

### **frontend/src/metabase/querying/expressions/resolver.unit.spec.ts**

```
import * as Lib from "metabase-lib";  
  
import { columnsForExpressionMode } from "../mode";  
import { resolver as makeResolver } from "../resolver";  
import {  
  expressions,  
  fields,  
  findDimensions,  
  ...  
}
```

### **frontend/src/metabase/querying/expressions/string.ts**

```
const DOUBLE_QUOTE = '"';  
const SINGLE_QUOTE = "'";  
const OPEN_BRACKET = "[";  
const CLOSE_BRACKET = "]";
```

## Analyst Base

```
const BACKSLASH = "\\";
```

```
export const STRING_LITERAL_DEFAULT_QUOTE = "";
```

```
export type...
```

### **frontend/src/metabase/querying/expressions/string.unit.spec.ts**

```
import { quoteString, unquoteString } from "./string";
```

```
const dq = (str: string) => quoteString(str, "");
```

```
const sq = (str: string) => quoteString(str, "");
```

```
const bq = (str: string) =>...
```

### **frontend/src/metabase/querying/expressions/suggestions/\_\_support\_\_.ts**

```
import {  
  CompletionContext,  
  type CompletionSource,  
} from "@codemirror/autocomplete";  
import { EditorState } from "@codemirror/state";
```

```
export function complete(source: CompletionSource | null,...
```

### **frontend/src/metabase/querying/expressions/suggestions/aggregations.ts**

```
import type { CompletionContext } from "@codemirror/autocomplete";
```

```
import type * as Lib from "metabase-lib";
```

```
import type Metadata from "metabase-lib/v1/metadata/Metadata";
```

```
import {...
```

### **frontend/src/metabase/querying/expressions/suggestions/aggregations.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";  
import { createQuery } from "metabase-lib/test-helpers";  
import type { DatabaseFeature } from "metabase-types/api";  
import {...
```

### **frontend/src/metabase/querying/expressions/suggestions/fields.ts**

```
import type { CompletionContext } from "@codemirror/autocomplete";
```

```
import { getColumnIcon } from "metabase/common/utils/columns";
```

```
import * as Lib from "metabase-lib";
```

```
import { formatIdentifier }...
```

## Analyst Base

### **frontend/src/metabase/querying/expressions/suggestions/fields.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import * as Lib from "metabase-lib";
import { SAMPLE_DATABASE, createQuery } from "metabase-lib/test-helpers";
import type { DatasetQuery,...
```

### **frontend/src/metabase/querying/expressions/suggestions/functions.ts**

```
import type { CompletionContext } from "@codemirror/autocomplete";

import type * as Lib from "metabase-lib";
import type Metadata from "metabase-lib/v1/metadata/Metadata";
```

```
import {...
```

### **frontend/src/metabase/querying/expressions/suggestions/functions.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import { createQuery } from "metabase-lib/test-helpers";
import type { DatabaseFeature } from "metabase-types/api";
import {...
```

### **frontend/src/metabase/querying/expressions/suggestions/index.ts**

```
export * from "./suggest";
export * from "./util";
export { type ExpressionSuggestion } from "./types";
```

### **frontend/src/metabase/querying/expressions/suggestions/literals.ts**

```
import type { CompletionContext } from "@codemirror/autocomplete";

import { tokenAtPos } from "../position";

import { isFieldReference, isIdentifier } from "../util";
```

```
export function...
```

### **frontend/src/metabase/querying/expressions/suggestions/literals.unit.spec.ts**

```
import { complete } from "../__support__";
import { suggestLiterals } from "../literals";

describe("suggestLiterals", () => {
  it("should suggest True and False", () => {
    const results =...
```

### **frontend/src/metabase/querying/expressions/suggestions/metrics.ts**

## Analyst Base

```
import type { CompletionContext } from "@codemirror/autocomplete";
```

```
import * as Lib from "metabase-lib";
```

```
import { formatIdentifier } from "../identifier";
```

```
import { tokenAtPos } from...
```

### **frontend/src/metabase/querying/expressions/suggestions/metrics.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
```

```
import { SAMPLE_DATABASE, createQuery } from "metabase-lib/test-helpers";
```

```
import {
```

```
  createMockCard,
```

```
  createMockField,
```

```
  ...
```

### **frontend/src/metabase/querying/expressions/suggestions/segments.ts**

```
import type { CompletionContext } from "@codemirror/autocomplete";
```

```
import * as Lib from "metabase-lib";
```

```
import { formatIdentifier } from "../identifier";
```

```
import { tokenAtPos } from...
```

### **frontend/src/metabase/querying/expressions/suggestions/segments.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
```

```
import { SAMPLE_DATABASE, createQuery } from "metabase-lib/test-helpers";
```

```
import { createMockSegment, createMockTable } from...
```

### **frontend/src/metabase/querying/expressions/suggestions/suggest.ts**

```
import { autocomplete } from "@codemirror/autocomplete";
```

```
import { isNotNull } from "metabase/lib/types";
```

```
import type * as Lib from "metabase-lib";
```

```
import type Metadata from...
```

### **frontend/src/metabase/querying/expressions/suggestions/types.ts**

```
import type {
```

```
  Completion as CodeMirrorCompletion,
```

```
  CompletionResult as CodeMirrorCompletionResult,
```

```
} from "@codemirror/autocomplete";
```

```
import type { IconName } from "metabase/ui";
```

```
import type * as...
```

## Analyst Base

### frontend/src/metabase/querying/expressions/suggestions/util.ts

```
import { snippetCompletion } from "@codemirror/autocomplete";
import Fuse from "fuse.js";
```

```
import { CALL, FIELD, IDENTIFIER, type Token } from "../pratt";
import type { MBQLClauseFunctionConfig }...
```

### frontend/src/metabase/querying/expressions/test/fuzz.ts

```
import _ from "underscore";
```

```
export const fuzz = process.env.MB_FUZZ ? describe : _.noop;
```

### frontend/src/metabase/querying/expressions/test/generator.ts

```
type Generator<T> = () => T;
```

```
type Node = {
  type: number;
  value?: number | string;
  params?: Node[];
  left?: Node;
  right?: Node;
  child?: Node;
  op?: string;
};
```

```
function assert<T>(x: T |...
```

### frontend/src/metabase/querying/expressions/test/shared.ts

```
import { createMockMetadata } from "__support__/metadata";
import { getNextId } from "__support__/utils";
import * as Lib from "metabase-lib";
import {
  type ExpressionClauseOpts,
  createQuery,
  ...
```

### frontend/src/metabase/querying/expressions/test/utils.ts

```
import type * as Lib from "metabase-lib";
```

```
export function op(
  operator: string,
  ...args: (Lib.ExpressionParts | Lib.ExpressionArg)[]
): Lib.ExpressionParts {
```

## Analyst Base

```
return {  
  operator: operator as...
```

### **frontend/src/metabase/querying/expressions/tokenizer/index.ts**

```
export * from "./tokenize";
```

### **frontend/src/metabase/querying/expressions/tokenizer/lezer.js**

This file was generated by lezer-generator. You probably shouldn't edit it.

### **frontend/src/metabase/querying/expressions/tokenizer/lezer.terms.js**

This file was generated by lezer-generator. You probably shouldn't edit it.

### **frontend/src/metabase/querying/expressions/tokenizer/parser.ts**

```
import { styleTags, tags as t } from "@lezer/highlight";
```

```
import { parser as baseParser } from "./lezer";
```

```
export const tags = styleTags({  
  Identifier: t.variableName,  
  Boolean: t.bool,  
  True:...
```

### **frontend/src/metabase/querying/expressions/tokenizer/tokenize.ts**

```
import { parser } from "./parser";
```

```
export function tokenize(source: string) {  
  const tree = parser.parse(source);  
  return tree.cursor();  
}
```

### **frontend/src/metabase/querying/expressions/tokenizer/tokens.ts**

```
import { ExternalTokenizer } from "@lezer/lr";
```

```
import { Field } from "./lezer.terms";
```

```
function char(char: string): number {  
  if (char.length !== 1) {  
    throw new Error(`Expected a single...
```

### **frontend/src/metabase/querying/expressions/types.ts**

```
import type * as Lib from "metabase-lib";  
import type Database from "metabase-lib/v1/metadata/Database";  
import type { DatabaseFeature } from "metabase-types/api";  
  
import type { DefinedClauseName }...
```

## Analyst Base

### **frontend/src/metabase/querying/expressions/utils.ts**

```
import * as Lib from "metabase-lib";
import type Database from "metabase-lib/v1/metadata/Database";
import type Metadata from "metabase-lib/v1/metadata/Metadata";
```

```
import { CompileError } from...
```

### **frontend/src/metabase/querying/expressions/visitor.ts**

```
import * as Lib from "metabase-lib";
```

```
type Visitor = (expression: Lib.ExpressionParts | Lib.ExpressionArg) => void;
```

```
export function visit(
  node: Lib.ExpressionParts | Lib.ExpressionArg,
  visitor:...
```

### **frontend/src/metabase/querying/fields/components/FieldPanel/index.ts**

```
export * from "./FieldPanel";
```

### **frontend/src/metabase/querying/fields/components/FieldPanel/types.ts**

```
import type * as Lib from "metabase-lib";
```

```
export type ColumnItem = {
  column: Lib.ColumnMetadata;
  displayName: string;
  isSelected: boolean;
  isDisabled: boolean;
  isHidden?:...
```

### **frontend/src/metabase/querying/fields/components/FieldPanel/utils.ts**

```
import * as Lib from "metabase-lib";
```

```
import type { ColumnGroupItem, ColumnItem } from "./types";
```

```
function getColumns(query: Lib.Query, stageIndex: number) {
  const aggregations =...
```

### **frontend/src/metabase/querying/filters/components/BooleanPicker/index.ts**

```
export * from "./BooleanPicker";
```

### **frontend/src/metabase/querying/filters/components/DatePicker/DateOperatorPicker/constants.ts**

```
import { t } from "ttag";
```



## Analyst Base

```
import type { OperatorOption } from "../types";

export const OPERATOR_OPTIONS: OperatorOption[] = [
  {
    get label() {
      return t`All time`;
    },
    value:...
```

### **frontend/src/metabase/querying/filters/components/DatePicker/DateOperatorPicker/index.ts**

```
export * from "../DateOperatorPicker";
```

### **frontend/src/metabase/querying/filters/components/DatePicker/DateOperatorPicker/types.ts**

```
import type {
  DatePickerOperator,
  RelativeIntervalDirection,
} from "metabase/querying/filters/types";
```

```
export type OptionType =
  | "none"
  | DatePickerOperator
  | ...
```

### **frontend/src/metabase/querying/filters/components/DatePicker/DateOperatorPicker/utils.ts**

```
import type {
  DatePickerOperator,
  DatePickerValue,
} from "metabase/querying/filters/types";
```

```
import { getExcludeOperatorValue } from "../ExcludeDatePicker/utils";
import {
  ...
```

### **frontend/src/metabase/querying/filters/components/DatePicker/DateOperatorPicker/utils.unit.spec.ts**

```
import { DATE_PICKER_OPERATORS } from "metabase/querying/filters/constants";
import type {
  DatePickerTruncationUnit,
  DatePickerValue,
  ExcludeDatePickerValue,
```

## Analyst Base

RelativeDatePickerValue,

...

### **frontend/src/metabase/querying/filters/components/DatePicker/DateShortcutPicker/constants.ts**

import { t } from "ttag";

import type { ShortcutOption } from "metabase/querying/filters/types";

import type { TypeOption } from "../types";

const DAY\_WEEK\_SHORTCUT\_OPTIONS: ShortcutOption[] = [

...

### **frontend/src/metabase/querying/filters/components/DatePicker/DateShortcutPicker/index.ts**

export \* from "../DateShortcutPicker";

export { getShortcutOptions } from "../utils";

### **frontend/src/metabase/querying/filters/components/DatePicker/DateShortcutPicker/types.ts**

import type {  
 DatePickerOperator,  
 DatePickerValueType,  
} from "metabase/querying/filters/types";

export interface TypeOption {  
 label: string;  
 type: DatePickerValueType;  
 operators:...

### **frontend/src/metabase/querying/filters/components/DatePicker/DateShortcutPicker/utils.ts**

import type {  
 DatePickerOperator,  
 DatePickerShortcut,  
 ShortcutOption,  
} from "metabase/querying/filters/types";

import { SHORTCUT\_OPTION\_GROUPS, TYPE\_OPTIONS } from "../constants";  
import type...

### **frontend/src/metabase/querying/filters/components/DatePicker/ExcludeDatePicker/constants.ts**

## Analyst Base

```
import { t } from "ttag";
```

```
import type { ExcludeOperatorOption, ExcludeUnitOption } from "../types";
```

```
export const EXCLUDE_UNIT_OPTIONS: ExcludeUnitOption[] = [
  {
    unit: "day-of-week",
    get...
```

### **frontend/src/metabase/querying/filters/components/DatePicker/ExcludeDatePicker/index.ts**

```
export * from "../ExcludeDatePicker";
```

### **frontend/src/metabase/querying/filters/components/DatePicker/ExcludeDatePicker/types.ts**

```
import type {
  DatePickerExtractionUnit,
  ExcludeDatePickerOperator,
} from "metabase/querying/filters/types";
```

```
export interface ExcludeUnitOption {
  unit: DatePickerExtractionUnit;
  label:...
```

### **frontend/src/metabase/querying/filters/components/DatePicker/ExcludeDatePicker/utils.ts**

```
import dayjs from "dayjs";
import { t } from "ttag";
import _ from "underscore";
```

```
import type {
  DatePickerExtractionUnit,
  DatePickerOperator,
  DatePickerUnit,
  ExcludeDatePickerOperator,
  ...
```

### **frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/CurrentDatePicker/constants.ts**

```
import type { DatePickerTruncationUnit } from "metabase/querying/filters/types";
```

```
export const UNIT_GROUPS: DatePickerTruncationUnit[][] = [
  ["day", "week", "month"],
```

## Analyst Base

```
["quarter", "year"],  
];
```

### **frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/CurrentDatePicker/index.ts**

```
export * from "./CurrentDatePicker";
```

### **frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/CurrentDatePicker/utils.ts**

```
import type {  
  DatePickerTruncationUnit,  
  DatePickerUnit,  
  RelativeDatePickerValue,  
} from "metabase/querying/filters/types";
```

```
import { UNIT_GROUPS } from "./constants";
```

```
export function...
```

### **frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/DateIntervalPicker/SimpleDateIntervalPicker/index.ts**

```
export * from "./SimpleDateIntervalPicker";
```

### **frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/DateIntervalPicker/constants.ts**

```
import type { DatePickerTruncationUnit } from "metabase/querying/filters/types";
```

```
export const DEFAULT_OFFSETS: Record<DatePickerTruncationUnit, number> = {  
  minute: 60,  
  hour: 24,  
  day: 7,  
  ...  
};
```

### **frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/DateIntervalPicker/index.ts**

```
export * from "./DateIntervalPicker";
```

### **frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/DateIntervalPicker/utils.ts**

```
import type {  
  DatePickerTruncationUnit,  
  RelativeDatePickerValue,  
} from "metabase/querying/filters/types";  
import * as Lib from "metabase-lib";
```

## Analyst Base

```
import { DEFAULT_OFFSETS } from...
```

**frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/DateOffsetIntervalPicker/index.ts**

```
export * from "../DateOffsetIntervalPicker";
```

**frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/DateOffsetIntervalPicker/utils.ts**

```
import { t } from "ttag";
```

```
import type {  
  DatePickerTruncationUnit,  
  DatePickerUnit,  
  RelativeDatePickerValue,  
  RelativeIntervalDirection,  
} from "metabase/querying/filters/types";  
import * as...
```

**frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/IncludeCurrentSwitch/index.ts**

```
export * from "../IncludeCurrentSwitch";
```

**frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/constants.ts**

```
import { t } from "ttag";
```

```
import type { Tab } from "../types";
```

```
export const TABS: Tab[] = [  
  {  
    get label() {  
      return t`Previous`;  
    },  
    direction: "last",  
  },  
  {  
    get label() {  
      ...  
    }  
  }  
];
```

**frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/index.ts**

```
export * from "../RelativeDatePicker";
```

## **frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/types.ts**

```
import type { RelativeIntervalDirection } from "metabase/querying/filters/types";
```

```
export interface Tab {  
  label: string;  
  direction: RelativeIntervalDirection;  
}
```

## **frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/utils.ts**

```
import { DATE_PICKER_TRUNCATION_UNITS } from "metabase/querying/filters/constants";
```

```
import type {  
  DatePickerTruncationUnit,  
  DatePickerUnit,  
  DatePickerValue,  
  RelativeDatePickerValue,  
  ...
```

## **frontend/src/metabase/querying/filters/components/DatePicker/RelativeDatePicker/utils.unit.spec.ts**

```
import type {  
  DatePickerTruncationUnit,  
  RelativeDatePickerValue,  
} from "metabase/querying/filters/types";
```

```
import { setDirection } from "../utils";
```

```
describe("setDirection", () => {  
  ...
```

## **frontend/src/metabase/querying/filters/components/DatePicker/SimpleDatePicker/index.ts**

```
export * from "../SimpleDatePicker";
```

## **frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/DateRangePicker/DateRangePickerBody/index.ts**

```
export * from "../DateRangePickerBody";
```

## **frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/DateRangePicker/SimpleDateRangePicker/index.ts**

```
export * from "../SimpleDateRangePicker";
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/DateRangePicker/index.ts**

```
export * from "./DateRangePicker";  
export * from "./types";
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/DateRangePicker/types.ts**

```
export interface DateRangePickerValue {  
  dateRange: [Date, Date];  
  hasTime: boolean;  
}
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/SimpleSpecificDatePicker/index.ts**

```
export * from "./SimpleSpecificDatePicker";
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/SingleDatePicker/SimpleSingleDatePicker/index.ts**

```
export * from "./SimpleSingleDatePicker";
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/SingleDatePicker/SingleDatePickerBody/index.ts**

```
export * from "./SingleDatePickerBody";
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/SingleDatePicker/index.ts**

```
export * from "./SingleDatePicker";  
export * from "./types";
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/SingleDatePicker/types.ts**

```
export interface SingleDatePickerValue {  
  date: Date;  
  hasTime: boolean;  
}
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/TimeToggle/index.ts**

```
export * from "./TimeToggle";
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/constants.ts**

```
import { t } from "ttag";
```

## Analyst Base

```
import type { Tab } from "./types";
```

```
export const TABS: Tab[] = [
```

```
{
  get label() {
    return t`Between`;
  },
  operator: "between",
},
{
  get label() {
```

```
...
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/index.ts**

```
export * from "./SpecificDatePicker";
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/types.ts**

```
import type { SpecificDatePickerOperator } from "metabase/querying/filters/types";
```

```
export interface Tab {
  label: string;
  operator: SpecificDatePickerOperator;
}
```

**frontend/src/metabase/querying/filters/components/DatePicker/SpecificDatePicker/utils.ts**

```
import dayjs from "dayjs";
```

```
import type {
  DatePickerOperator,
  DatePickerUnit,
  DatePickerValue,
  SpecificDatePickerOperator,
  SpecificDatePickerValue,
} from...
```

**frontend/src/metabase/querying/filters/components/DatePicker/constants.ts**

```
export const MIN_WIDTH = 300;
```

**frontend/src/metabase/querying/filters/components/DatePicker/index.ts**

```
export * from "./DatePicker";
```



## Analyst Base

### **frontend/src/metabase/querying/filters/components/DatePicker/types.ts**

```
import type { DatePickerValue } from "metabase/querying/filters/types";
```

```
export type DatePickerSubmitButtonProps = {  
  value: DatePickerValue;  
  isDisabled?: boolean;  
};
```

### **frontend/src/metabase/querying/filters/components/FilterPanel/FilterPanelPopover/index.ts**

```
export * from "./FilterPanelPopover";
```

### **frontend/src/metabase/querying/filters/components/FilterPanel/FilterPill/index.ts**

```
export * from "./FilterPill";
```

### **frontend/src/metabase/querying/filters/components/FilterPanel/index.ts**

```
export * from "./FilterPanel";
```

### **frontend/src/metabase/querying/filters/components/FilterPanel/types.ts**

```
import type * as Lib from "metabase-lib";
```

```
export interface FilterItem {  
  filter: Lib.FilterClause;  
  filterIndex: number;  
  stageIndex: number;  
}
```

### **frontend/src/metabase/querying/filters/components/FilterPanel/utils.ts**

```
import * as Lib from "metabase-lib";
```

```
import type { FilterItem } from "./types";
```

```
export function getFilterItems(query: Lib.Query): FilterItem[] {  
  const stageIndexes = Lib.stageIndexes(query);  
  ...
```

### **frontend/src/metabase/querying/filters/components/FilterPanel/utils.unit.spec.ts**

```
import * as Lib from "metabase-lib";
```

```
import { createQuery } from "metabase-lib/test-helpers";
```

```
import { getFilterItems } from "./utils";
```

```
const STAGE_COUNT = 4;
```

```
function createFilteredQuery(query:...
```

**frontend/src/metabase/querying/filters/components/FilterPicker/BooleanFilterPicker/index.ts**

```
export * from "./BooleanFilterPicker";
```

**frontend/src/metabase/querying/filters/components/FilterPicker/CoordinateFilterPicker/CoordinateColumnPicker/index.ts**

```
export * from "./CoordinateColumnPicker";
```

**frontend/src/metabase/querying/filters/components/FilterPicker/CoordinateFilterPicker/CoordinateColumnPicker/types.ts**

```
import type * as Lib from "metabase-lib";
```

```
export interface ColumnOption {  
  value: string;  
  label: string;  
  column: Lib.ColumnMetadata;  
}
```

**frontend/src/metabase/querying/filters/components/FilterPicker/CoordinateFilterPicker/CoordinateColumnPicker/utils.ts**

```
import { t } from "ttag";
```

```
import * as Lib from "metabase-lib";
```

```
import type { ColumnOption } from "./types";
```

```
export function getColumnOptions(  
  query: Lib.Query,  
  stageIndex: number,  
  columns:...
```

**frontend/src/metabase/querying/filters/components/FilterPicker/CoordinateFilterPicker/index.ts**

```
export * from "./CoordinateFilterPicker";
```

**frontend/src/metabase/querying/filters/components/FilterPicker/DateFilterPicker/SimpleDateFilterPicker/index.ts**

```
export * from "./SimpleDateFilterPicker";
```

**frontend/src/metabase/querying/filters/components/FilterPicker/DateFilterPicker/index.ts**

## Analyst Base

export \* from "../DateFilterPicker";

**frontend/src/metabase/querying/filters/components/FilterPicker/DefaultFilterPicker/index.ts**

export \* from "../DefaultFilterPicker";

**frontend/src/metabase/querying/filters/components/FilterPicker/FilterColumnPicker/index.ts**

export \* from "../FilterColumnPicker";

**frontend/src/metabase/querying/filters/components/FilterPicker/FilterOperatorPicker/index.ts**

export \* from "../FilterOperatorPicker";

**frontend/src/metabase/querying/filters/components/FilterPicker/FilterPickerBody/index.ts**

export \* from "../FilterPickerBody";

**frontend/src/metabase/querying/filters/components/FilterPicker/FilterPickerFooter/index.ts**

export \* from "../FilterPickerFooter";

**frontend/src/metabase/querying/filters/components/FilterPicker/FilterPickerHeader/index.ts**

export \* from "../FilterPickerHeader";

**frontend/src/metabase/querying/filters/components/FilterPicker/FilterSubmitButton/index.ts**

export \* from "../FilterSubmitButton";

**frontend/src/metabase/querying/filters/components/FilterPicker/MultiStageFilterPicker/index.ts**

export \* from "../MultiStageFilterPicker";

**frontend/src/metabase/querying/filters/components/FilterPicker/NumberFilterPicker/index.ts**

export \* from "../NumberFilterPicker";

**frontend/src/metabase/querying/filters/components/FilterPicker/StringFilterPicker/index.ts**

export \* from "../StringFilterPicker";

**frontend/src/metabase/querying/filters/components/FilterPicker/TimeFilterPicker/index.ts**

## Analyst Base

```
export * from "../TimeFilterPicker";
```

### **frontend/src/metabase/querying/filters/components/FilterPicker/constants.ts**

```
import type { ComboboxProps } from "metabase/ui";
```

```
export const WIDTH = 380;
```

```
export const COMBOBOX_PROPS: Partial<ComboboxProps> = {  
  width: 298,  
  position: "bottom-start",  
};
```

### **frontend/src/metabase/querying/filters/components/FilterPicker/index.ts**

```
export { FilterPicker, type FilterPickerProps } from "../FilterPicker";
```

### **frontend/src/metabase/querying/filters/components/FilterPicker/test-utils.ts**

```
istanbul ignore file
```

### **frontend/src/metabase/querying/filters/components/FilterPicker/types.ts**

```
import type { DefinedClauseName } from "metabase/querying/expressions";  
import type * as Lib from "metabase-lib";
```

```
export type FilterPickerWidgetProps = {  
  autoFocus: boolean;  
  query: Lib.Query;  
  ...
```

### **frontend/src/metabase/querying/filters/components/FilterValuePicker/ListViewPicker/index.ts**

```
export * from "../ListViewPicker";
```

### **frontend/src/metabase/querying/filters/components/FilterValuePicker/ListViewPicker/Utils.ts**

```
import type { ComboboxItem } from "metabase/ui";  
import type { FieldValue } from "metabase-types/api";
```

```
import { getFieldOptions } from "../utils";
```

```
export function searchOptions(  
  options:...
```

### **frontend/src/metabase/querying/filters/components/FilterValuePicker/SearchValuePicker/constants.ts**

```
export const SEARCH_DEBOUNCE = 500;  
export const SEARCH_LIMIT = 100;
```

## **frontend/src/metabase/querying/filters/components/FilterValuePicker/SearchValuePicker/index.ts**

```
export * from "./SearchValuePicker";
```

## **frontend/src/metabase/querying/filters/components/FilterValuePicker/SearchValuePicker/utils.ts**

```
import { t } from "ttag";
```

```
import type { FieldValue } from "metabase-types/api";
```

```
import { SEARCH_LIMIT } from "./constants";
```

```
export function shouldSearch(  
  searchValue: string,  
  searchQuery:...
```

## **frontend/src/metabase/querying/filters/components/FilterValuePicker/StaticValuePicker/index.ts**

```
export * from "./StaticValuePicker";
```

## **frontend/src/metabase/querying/filters/components/FilterValuePicker/index.ts**

```
export * from "./FilterValuePicker";
```

## **frontend/src/metabase/querying/filters/components/FilterValuePicker/utils.ts**

```
import { t } from "ttag";
```

```
import type { ComboboxItem } from "metabase/ui";
```

```
import type { FieldValuesSearchInfo } from "metabase-lib";
```

```
import * as Lib from "metabase-lib";
```

```
import type { FieldValue,...
```

## **frontend/src/metabase/querying/filters/components/MonthYearPicker/index.ts**

```
export * from "./MonthYearPicker";
```

## **frontend/src/metabase/querying/filters/components/NumberFilterInput/index.ts**

```
export * from "./NumberFilterInput";
```

## **frontend/src/metabase/querying/filters/components/QuarterYearPicker/index.ts**

```
export * from "./QuarterYearPicker";
```

## **frontend/src/metabase/querying/filters/components/RelativeDateShortcutPicker/index.ts**

```
export * from "./RelativeDateShortcutPicker";
```

## **frontend/src/metabase/querying/filters/components/RelativeDateShortcutPicker**

### **/types.ts**

```
import type { RelativeDatePickerValue } from "metabase/querying/filters/types";
```

```
export type ShortcutGroup = {  
  label?: string;  
  columns: number;  
  shortcuts: Shortcut[];  
};
```

```
export type Shortcut =...
```

### **frontend/src/metabase/querying/filters/components/RelativeDateShortcutPicker /utils.ts**

```
import { msgid, ngettext, t } from "ttag";
```

```
import type { ShortcutGroup } from "../types";
```

```
export function getShortcutGroups(): ShortcutGroup[] {  
  return [  
    {  
      columns: 2,  
      shortcuts:...
```

### **frontend/src/metabase/querying/filters/components/TemporalUnitPicker/index.t s**

```
export * from "../TemporalUnitPicker";
```

### **frontend/src/metabase/querying/filters/components/TimeseriesChrome/Timeser iesBucketPicker/index.ts**

```
export * from "../TimeseriesBucketPicker";
```

### **frontend/src/metabase/querying/filters/components/TimeseriesChrome/Timeser iesFilterPicker/index.ts**

```
export * from "../TimeseriesFilterPicker";
```

### **frontend/src/metabase/querying/filters/components/TimeseriesChrome/index.ts**

```
export * from "../TimeseriesChrome";
```

### **frontend/src/metabase/querying/filters/components/TimeseriesChrome/utils.ts**

```
import * as Lib from "metabase-lib";
```

```
export function findFilterColumn(  
  query: Lib.Query,
```

## Analyst Base

```
stageIndex: number,  
breakoutColumn: Lib.ColumnMetadata,  
): Lib.ColumnMetadata | undefined {  
  const...
```

### **frontend/src/metabase/querying/filters/constants.ts**

```
export const SPECIFIC_DATE_PICKER_OPERATORS = [  
  "=" as const,  
  "<" as const,  
  ">" as const,  
  "between" as const,  
];
```

```
export const EXCLUDE_DATE_PICKER_OPERATORS = [  
  "!=" as const,  
  "is-null"...
```

### **frontend/src/metabase/querying/filters/hooks/use-boolean-filter/index.ts**

```
export * from "./use-boolean-filter";
```

### **frontend/src/metabase/querying/filters/hooks/use-boolean-filter/use-boolean-filter.ts**

```
import { useState } from "react";
```

```
import type * as Lib from "metabase-lib";
```

```
import { getFilterClause, getFilterValue } from "./utils";
```

```
type UseBooleanFilterProps = {  
  query: Lib.Query;  
  ...
```

### **frontend/src/metabase/querying/filters/hooks/use-boolean-filter/use-boolean-filter.unit.spec.ts**

```
import { act, renderHook } from "@testing-library/react";
```

```
import { createMockMetadata } from "__support__/metadata";  
import type { BooleanFilterValue } from "metabase/querying/filters/types";  
import...
```

### **frontend/src/metabase/querying/filters/hooks/use-boolean-filter/utils.ts**

```
import type { BooleanFilterValue } from "metabase/querying/filters/types";  
import * as Lib from "metabase-lib";
```

## Analyst Base

```
export function getFilterValue(  
  query: Lib.Query,  
  stageIndex: number,  
  ...  

```

### **frontend/src/metabase/querying/filters/hooks/use-coordinate-filter/constants.ts**

```
import type * as Lib from "metabase-lib";  
  
import type { OperatorOption } from "../types";  
  
export const OPERATOR_OPTIONS: Record<  
  Lib.CoordinateFilterOperator,  
  OperatorOption  
> = {  
  "=": {  
    ...  

```

### **frontend/src/metabase/querying/filters/hooks/use-coordinate-filter/index.ts**

```
export * from "../use-coordinate-filter";  
export * from "../types";
```

### **frontend/src/metabase/querying/filters/hooks/use-coordinate-filter/types.ts**

```
import type { FilterOperatorOption } from "metabase/querying/filters/types";  
import type * as Lib from "metabase-lib";
```

```
type CoordinatePickerOperator =  
  | "="  
  | "!="  
  | ">"  
  | "<"  
  | "between"  
  ...
```

### **frontend/src/metabase/querying/filters/hooks/use-coordinate-filter/use-coordinate-filter.ts**

```
import { useMemo, useState } from "react";  
  
import * as Lib from "metabase-lib";  
  
import type { NumberOrEmptyValue } from "../types";  
import {  
  canPickColumns,  
  getAvailableColumns,  
  ...  

```



## **frontend/src/metabase/querying/filters/hooks/use-coordinate-filter/use-coordinate-filter.unit.spec.ts**

```
import { act, renderHook } from "@testing-library/react";
```

```
import { checkNotNull } from "metabase/lib/types";  
import * as Lib from "metabase-lib";  
import { columnFinder, createQuery } from...
```

## **frontend/src/metabase/querying/filters/hooks/use-coordinate-filter/utils.ts**

```
import { isNotNull } from "metabase/lib/types";  
import {  
  getAvailableOperatorOptions,  
  getDefaultAvailableOperator,  
} from "metabase/querying/filters/utils/operators";  
import * as Lib from...
```

## **frontend/src/metabase/querying/filters/hooks/use-date-filter/index.ts**

```
export * from "./use-date-filter";
```

## **frontend/src/metabase/querying/filters/hooks/use-date-filter/use-date-filter.ts**

```
import { useMemo } from "react";  
  
import type { DatePickerValue } from "metabase/querying/filters/types";  
import {  
  getDateFilterClause,  
  getDatePickerOperators,  
  getDatePickerUnits,  
  ...  
}
```

## **frontend/src/metabase/querying/filters/hooks/use-date-filter/use-date-filter.unit.spec.ts**

```
import { renderHook } from "@testing-library/react";  
  
import type { DatePickerValue } from "metabase/querying/filters/types";  
import * as Lib from "metabase-lib";  
import {  
  columnFinder,  
  ...  
}
```

## **frontend/src/metabase/querying/filters/hooks/use-default-filter/constants.ts**

```
import type * as Lib from "metabase-lib";  
  
import type { OperatorOption } from "./types";
```

## Analyst Base

```
export const OPERATOR_OPTIONS: Record<
  Lib.DefaultFilterOperator,
  OperatorOption
> = {
  "is-null": {
    ...
```

### **frontend/src/metabase/querying/filters/hooks/use-default-filter/index.ts**

```
export * from "./use-default-filter";
export * from "./types";
```

### **frontend/src/metabase/querying/filters/hooks/use-default-filter/types.ts**

```
import type { FilterOperatorOption } from "metabase/querying/filters/types";
import type * as Lib from "metabase-lib";
```

```
export type OperatorOption = FilterOperatorOption<Lib.DefaultFilterOperator>;
```

### **frontend/src/metabase/querying/filters/hooks/use-default-filter/use-default-filter.ts**

```
import { useMemo, useState } from "react";
```

```
import * as Lib from "metabase-lib";
```

```
import {
  getAvailableOptions,
  getDefaultOperator,
  getFilterClause,
} from "./utils";
```

```
interface...
```

### **frontend/src/metabase/querying/filters/hooks/use-default-filter/use-default-filter.unit.spec.ts**

```
import { act, renderHook } from "@testing-library/react";
```

```
import { createMockMetadata } from "__support__/metadata";
import { checkNotNull } from "metabase/lib/types";
import * as Lib from...
```

### **frontend/src/metabase/querying/filters/hooks/use-default-filter/utils.ts**

```
import {
  getAvailableOperatorOptions,
  getDefaultAvailableOperator,
```

## Analyst Base

```
} from "metabase/querying/filters/utils/operators";
import * as Lib from "metabase-lib";
```

```
import { OPERATOR_OPTIONS } from...
```

### **frontend/src/metabase/querying/filters/hooks/use-number-filter/constants.ts**

```
import type * as Lib from "metabase-lib";
```

```
import type { OperatorOption } from "./types";
```

```
export const OPERATOR_OPTIONS: Record<
  Lib.NumberFilterOperator,
  OperatorOption
> = {
  "=": {
    ...
```

### **frontend/src/metabase/querying/filters/hooks/use-number-filter/index.ts**

```
export * from "./use-number-filter";
export * from "./types";
```

### **frontend/src/metabase/querying/filters/hooks/use-number-filter/types.ts**

```
import type { FilterOperatorOption } from "metabase/querying/filters/types";
import type * as Lib from "metabase-lib";
```

```
export interface OperatorOption
  extends...
```

### **frontend/src/metabase/querying/filters/hooks/use-number-filter/use-number-filter.ts**

```
import { useMemo, useState } from "react";
```

```
import * as Lib from "metabase-lib";
```

```
import type { NumberOrEmptyValue } from "./types";
import {
  getAvailableOptions,
  getDefaultOperator,
  ...
```

### **frontend/src/metabase/querying/filters/hooks/use-number-filter/use-number-filter.unit.spec.ts**

```
import { act, renderHook } from "@testing-library/react";
```

## Analyst Base

```
import { createMockMetadata } from "__support__/metadata";
import { checkNotNull } from "metabase/lib/types";
import * as Lib from...
```

### **frontend/src/metabase/querying/filters/hooks/use-number-filter/utils.ts**

```
import { isNotNull } from "metabase/lib/types";
import {
  getAvailableOperatorOptions,
  getDefaultAvailableOperator,
} from "metabase/querying/filters/utils/operators";
import * as Lib from...
```

### **frontend/src/metabase/querying/filters/hooks/use-string-filter/constants.ts**

```
import type * as Lib from "metabase-lib";

import type { OperatorOption } from "./types";

export const OPERATOR_OPTIONS: Record<
  Lib.StringFilterOperator,
  OperatorOption
> = {
  "=": {
    ...
```

### **frontend/src/metabase/querying/filters/hooks/use-string-filter/index.ts**

```
export * from "./use-string-filter";
export * from "./types";
```

### **frontend/src/metabase/querying/filters/hooks/use-string-filter/types.ts**

```
import type { FilterOperatorOption } from "metabase/querying/filters/types";
import type * as Lib from "metabase-lib";
```

```
export type OperatorType = "exact" | "partial" | "empty";
```

```
export interface...
```

### **frontend/src/metabase/querying/filters/hooks/use-string-filter/use-string-filter.ts**

```
import { useMemo, useState } from "react";
```

```
import * as Lib from "metabase-lib";
```

```
import {
  getAvailableOptions,
  getDefaultOperator,
```

## Analyst Base

```
getDefaultValues,  
getFilterClause,  
getOptionByOperator,  
...
```

### **frontend/src/metabase/querying/filters/hooks/use-string-filter/use-string-filter.unit.spec.ts**

```
import { act, renderHook } from "@testing-library/react";  
  
import { createMockMetadata } from "__support__/metadata";  
import { checkNotNull } from "metabase/lib/types";  
import * as Lib from...
```

### **frontend/src/metabase/querying/filters/hooks/use-string-filter/utils.ts**

```
import {  
  getAvailableOperatorOptions,  
  getDefaultAvailableOperator,  
} from "metabase/querying/filters/utils/operators";  
import * as Lib from "metabase-lib";
```

```
import { OPERATOR_OPTIONS } from...
```

### **frontend/src/metabase/querying/filters/hooks/use-time-filter/constants.ts**

```
import type * as Lib from "metabase-lib";
```

```
import type { OperatorOption } from "./types";
```

```
export const OPERATOR_OPTIONS: Record<Lib.TimeFilterOperator, OperatorOption> =  
{  
  "<": {  
    ...
```

### **frontend/src/metabase/querying/filters/hooks/use-time-filter/index.ts**

```
export * from "./use-time-filter";  
export * from "./types";
```

### **frontend/src/metabase/querying/filters/hooks/use-time-filter/types.ts**

```
import type { FilterOperatorOption } from "metabase/querying/filters/types";  
import type * as Lib from "metabase-lib";
```

```
export interface OperatorOption  
  extends...
```

### **frontend/src/metabase/querying/filters/hooks/use-time-filter/use-time-filter.ts**

## Analyst Base

```
import { useMemo, useState } from "react";
```

```
import * as Lib from "metabase-lib";
```

```
import type { TimeValue } from "./types";
```

```
import {  
  getAvailableOptions,  
  getDefaultOperator,  
  getDefaultValues,  
  ...
```

### **frontend/src/metabase/querying/filters/hooks/use-time-filter/use-time-filter.unit.spec.ts**

```
import { act, renderHook } from "@testing-library/react";
```

```
import { createMockMetadata } from "__support__/metadata";  
import { checkNotNull } from "metabase/lib/types";  
import * as Lib from...
```

### **frontend/src/metabase/querying/filters/hooks/use-time-filter/utils.ts**

```
import dayjs from "dayjs";
```

```
import { isNotNull } from "metabase/lib/types";  
import {  
  getAvailableOperatorOptions,  
  getDefaultAvailableOperator,  
} from...
```

### **frontend/src/metabase/querying/filters/types.ts**

```
import type { IconName } from "metabase/ui";  
import type * as Lib from "metabase-lib";
```

```
import type {  
  DATE_PICKER_EXTRACTION_UNITS,  
  DATE_PICKER_SHORTCUTS,  
  DATE_PICKER_TRUNCATION_UNITS,  
  ...
```

### **frontend/src/metabase/querying/filters/utils/dates.ts**

```
import dayjs from "dayjs";  
import { match } from "ts-pattern";  
import { c, msgid, ngettext, t } from "ttag";
```

```
import {
```

## Analyst Base

```
DATE_PICKER_EXTRACTION_UNITS,  
DATE_PICKER_OPERATORS,  
...
```

### **frontend/src/metabase/querying/filters/utils/dates.unit.spec.ts**

```
import type { DateFilterValue } from "metabase/querying/filters/types";  
import * as Lib from "metabase-lib";  
import { columnFinder, createQuery } from "metabase-lib/test-helpers";  
  
import {...
```

### **frontend/src/metabase/querying/filters/utils/groups.ts**

```
import type * as Lib from "metabase-lib";  
  
export function getGroupName(  
  groupInfo: Lib.ColumnGroupDisplayInfo,  
  stageIndex: number,  
) {  
  return groupInfo.isMainGroup && stageIndex > 1  
    ?...
```

### **frontend/src/metabase/querying/filters/utils/operators.ts**

```
import type { FilterOperatorOption } from "metabase/querying/filters/types";  
import * as Lib from "metabase-lib";  
  
export function getAvailableOperatorOptions<  
  T extends...
```

### **frontend/src/metabase/querying/metrics/components/MetricEditor/MetricEditor Body/MetricEditorSidebar/index.ts**

```
export * from "./MetricEditorSidebar";
```

### **frontend/src/metabase/querying/metrics/components/MetricEditor/MetricEditor Body/ResizableBoxHandle/index.ts**

```
export * from "./ResizableBoxHandle";
```

### **frontend/src/metabase/querying/metrics/components/MetricEditor/MetricEditor Body/index.ts**

```
export * from "./MetricEditorBody";
```

### **frontend/src/metabase/querying/metrics/components/MetricEditor/MetricEditor Footer/MetricEmptyState/index.ts**

```
export * from "./MetricEmptyState";
```

### **frontend/src/metabase/querying/metrics/components/MetricEditor/MetricEditor**

## **Footer/index.ts**

```
export * from "../MetricEditorFooter";
```

## **frontend/src/metabase/querying/metrics/components/MetricEditor/MetricEditorHeader/index.ts**

```
export * from "../MetricEditorHeader";
```

## **frontend/src/metabase/querying/metrics/components/MetricEditor/index.ts**

```
export * from "../MetricEditor";
```

## **frontend/src/metabase/querying/metrics/components/MetricEditor/types.ts**

```
export type MetricModalType = "create" | "leave";
```

## **frontend/src/metabase/querying/notebook/components/AggregateStep/index.ts**

```
export * from "../AggregateStep";
```

## **frontend/src/metabase/querying/notebook/components/BreakoutStep/index.ts**

```
export * from "../BreakoutStep";
```

## **frontend/src/metabase/querying/notebook/components/ClauseStep/index.ts**

```
export * from "../ClauseStep";
```

## **frontend/src/metabase/querying/notebook/components/DataStep/index.ts**

```
export * from "../DataStep";
```

## **frontend/src/metabase/querying/notebook/components/FieldPicker/index.ts**

```
export * from "../FieldPicker";
```

## **frontend/src/metabase/querying/notebook/components/FilterStep/index.ts**

```
export * from "../FilterStep";
```

## **frontend/src/metabase/querying/notebook/components/JoinStep/Join/index.ts**

```
export * from "../Join";
```

## **frontend/src/metabase/querying/notebook/components/JoinStep/JoinComplete/index.ts**

```
export * from "../JoinComplete";
```

## **frontend/src/metabase/querying/notebook/components/JoinStep/JoinCondition/index.ts**

```
export * from "../JoinCondition";
```

## **frontend/src/metabase/querying/notebook/components/JoinStep/JoinConditionColumnPicker/JoinColumnButton/index.ts**

```
export * from "../JoinColumnButton";
```



**frontend/src/metabase/querying/notebook/components/JoinStep/JoinCondition  
ColumnPicker/JoinColumnDropdown/index.ts**

```
export * from "../JoinColumnDropdown";
```

**frontend/src/metabase/querying/notebook/components/JoinStep/JoinCondition  
ColumnPicker/index.ts**

```
export * from "../JoinConditionColumnPicker";
```

**frontend/src/metabase/querying/notebook/components/JoinStep/JoinCondition  
Draft/index.ts**

```
export * from "../JoinConditionDraft";
```

**frontend/src/metabase/querying/notebook/components/JoinStep/JoinCondition  
Draft/utils.ts**

```
import * as Lib from "metabase-lib";
```

```
export function getDefaultJoinConditionOperator(  
  query: Lib.Query,  
  stageIndex: number,  
) : Lib.JoinConditionOperator {  
  const operators =...
```

**frontend/src/metabase/querying/notebook/components/JoinStep/JoinCondition  
OperatorPicker/index.ts**

```
export * from "../JoinConditionOperatorPicker";
```

**frontend/src/metabase/querying/notebook/components/JoinStep/JoinCondition  
RemoveButton/index.ts**

```
export * from "../JoinConditionRemoveButton";
```

**frontend/src/metabase/querying/notebook/components/JoinStep/JoinDraft/inde  
x.ts**

```
export * from "../JoinDraft";
```

**frontend/src/metabase/querying/notebook/components/JoinStep/JoinDraft/utils.  
ts**

```
import * as Lib from "metabase-lib";
```

```
export function getDefaultJoinStrategy(  
  query: Lib.Query,  
  stageIndex: number,  
) : Lib.JoinStrategy {  
  const strategies = Lib.availableJoinStrategies(query,...
```

## **frontend/src/metabase/querying/notebook/components/JoinStep/JoinStrategyPicker/index.ts**

```
export * from "./JoinStrategyPicker";
```

## **frontend/src/metabase/querying/notebook/components/JoinStep/JoinTableColumnDraftPicker/index.ts**

```
export * from "./JoinTableColumnDraftPicker";
```

## **frontend/src/metabase/querying/notebook/components/JoinStep/JoinTableColumnPicker/index.ts**

```
export * from "./JoinTableColumnPicker";
```

## **frontend/src/metabase/querying/notebook/components/JoinStep/JoinTablePicker/index.ts**

```
export * from "./JoinTablePicker";
```

## **frontend/src/metabase/querying/notebook/components/JoinStep/index.ts**

```
export * from "./JoinStep";
```

## **frontend/src/metabase/querying/notebook/components/JoinStep/Utils.ts**

```
import type { IconName } from "metabase/ui";
```

```
import type * as Lib from "metabase-lib";
```

```
const JOIN_ICONS: Record<string, IconName> = {  
  "left-join": "join_left_outer",  
  "right-join": ...
```

## **frontend/src/metabase/querying/notebook/components/LimitStep/index.ts**

```
export { LimitStep } from "./LimitStep";
```

## **frontend/src/metabase/querying/notebook/components/LimitStep/Util.ts**

```
export function isLimitValid(number: number) {  
  return !Number.isNaN(number) && Number.isInteger(number) && number > 0;  
}
```

```
export function parseLimit(value: string) {  
  return parseInt(value, 10);  
}
```

## **frontend/src/metabase/querying/notebook/components/Notebook/index.ts**

```
export * from "./Notebook";
```

```
export * from "./use-run-visualization";
```

```
export * from "./VisualizationButton";
```

## **frontend/src/metabase/querying/notebook/components/Notebook/use-run-visualization**

### lization.ts

```
import { cleanQuestion } from "metabase/query_builder/utils/question";
import type Question from "metabase-lib/v1/Question";
```

```
type UseVisualizationProps = {
  question?: Question;
  isDirty:...
```

### frontend/src/metabase/querying/notebook/components/NotebookCell/constants.ts

```
export const CONTAINER_PADDING = "10px";
```

### frontend/src/metabase/querying/notebook/components/NotebookCell/index.ts

```
export * from "./NotebookCell";
```

### frontend/src/metabase/querying/notebook/components/NotebookDataPicker/EmbeddingDataPicker/index.ts

```
export { EmbeddingDataPicker } from "./EmbeddingDataPicker";
```

### frontend/src/metabase/querying/notebook/components/NotebookDataPicker/index.ts

```
export * from "./NotebookDataPicker";
```

### frontend/src/metabase/querying/notebook/components/NotebookDataPicker/utils.ts

```
import * as Urls from "metabase/lib/urls";
import * as Lib from "metabase-lib";
```

```
type Props = {
  query: Lib.Query;
  table?: Lib.TableMetadata | Lib.CardMetadata;
  stageIndex: number;
};
```

```
export...
```

### frontend/src/metabase/querying/notebook/components/NotebookNativePreview/index.ts

```
export { NotebookNativePreview } from "./NotebookNativePreview";
```

### frontend/src/metabase/querying/notebook/components/NotebookNativePreview/utils.ts

```
import { formatNativeQuery } from "metabase/lib/engine";
```

## Analyst Base

```
import type Question from "metabase-lib/v1/Question";
import type { NativeDatasetResponse } from "metabase-types/api";
```

```
export function...
```

**frontend/src/metabase/querying/notebook/components/NotebookStep/NotebookActionButton/index.ts**

```
export * from "../NotebookActionButton";
```

**frontend/src/metabase/querying/notebook/components/NotebookStep/NotebookStepHeader/index.ts**

```
export * from "../NotebookStepHeader";
```

**frontend/src/metabase/querying/notebook/components/NotebookStep/NotebookStepPreview/index.ts**

```
export * from "../NotebookStepPreview";
```

**frontend/src/metabase/querying/notebook/components/NotebookStep/index.ts**

```
export * from "../NotebookStep";
```

**frontend/src/metabase/querying/notebook/components/NotebookStep/utils.ts**

```
import type { ComponentType } from "react";
import { t } from "ttag";
```

```
import { color } from "metabase/lib/colors";
import type { IconName } from "metabase/ui";
```

```
import type {
  ...
```

**frontend/src/metabase/querying/notebook/components/NotebookStepList/index.ts**

```
export { NotebookStepList } from "../NotebookStepList";
```

**frontend/src/metabase/querying/notebook/components/SortStep/index.ts**

```
export * from "../SortStep";
```

**frontend/src/metabase/querying/notebook/components/SummarizeStep/SummarizeStepHeader/index.ts**

```
export * from "../SummarizeStepHeader";
```

**frontend/src/metabase/querying/notebook/components/SummarizeStep/index.ts**

```
export * from "../SummarizeStep";
```

**frontend/src/metabase/querying/notebook/test-utils.ts**

## Analyst Base

istanbul ignore file

### **frontend/src/metabase/querying/notebook/types.ts**

```
import type * as Lib from "metabase-lib";
import type Question from "metabase-lib/v1/Question";
```

```
export type NotebookStepType =
  | "data"
  | "join"
  | "expression"
  | "filter"
  | "summarize"
  | ...
```

### **frontend/src/metabase/querying/notebook/utils/steps.ts**

```
import _ from "underscore";
```

```
import type { Query } from "metabase-lib";
import * as Lib from "metabase-lib";
import type Question from "metabase-lib/v1/Question";
import type Metadata from ...
```

### **frontend/src/metabase/querying/notebook/utils/steps.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import { checkNotNull } from "metabase/lib/types";
import * as Lib from "metabase-lib";
import { createQueryWithClauses } from ...
```

### **frontend/src/metabase/querying/parameters/components/BooleanWidget/index.ts**

```
export * from "./BooleanWidget";
```

### **frontend/src/metabase/querying/parameters/components/DateAllOptionsWidget/index.ts**

```
export * from "./DateAllOptionsWidget";
```

### **frontend/src/metabase/querying/parameters/components/DateMonthYearWidget/index.ts**

```
export * from "./DateMonthYearWidget";
```

### **frontend/src/metabase/querying/parameters/components/DateQuarterYearWidget/index.ts**

```
export * from "./DateQuarterYearWidget";
```

### **frontend/src/metabase/querying/parameters/components/DateRangeWidget/index.ts**

**ex.ts**

```
export * from "../DateRangeWidget";
```

**frontend/src/metabase/querying/parameters/components/DateRelativeWidget/index.ts**

```
export * from "../DateRelativeWidget";
```

**frontend/src/metabase/querying/parameters/components/DateSingleWidget/index.ts**

```
export * from "../DateSingleWidget";
```

**frontend/src/metabase/querying/parameters/components/WidgetFooter/index.ts**

```
export * from "../WidgetFooter";
```

**frontend/src/metabase/querying/parameters/constants.ts**

```
export const MIN_WIDTH = 300;
```

**frontend/src/metabase/querying/parameters/utils/parsing.ts**

```
import dayjs from "dayjs";
```

```
import { type NumberValue, parseNumber } from "metabase/lib/number";
```

```
import type { DateFilterValue } from "metabase/querying/filters/types";
```

```
import {...
```

**frontend/src/metabase/querying/parameters/utils/parsing.unit.spec.ts**

```
import type { DateFilterValue } from "metabase/querying/filters/types";
```

```
import type { ParameterValueOrArray } from "metabase-types/api";
```

```
import {  
  deserializeBooleanParameterValue,  
  ...
```

**frontend/src/metabase/querying/parameters/utils/query.ts**

```
import { P, match } from "ts-pattern";
```

```
import { isNotNull } from "metabase/lib/types";
```

```
import { getDateFilterClause } from "metabase/querying/filters/utils/dates";
```

```
import * as Lib from...
```

**frontend/src/metabase/querying/parameters/utils/query.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
```

```
import { isNotNull } from "metabase/lib/types";
```

```
import * as Lib from "metabase-lib";
```

```
import {
```

## Analyst Base

```
columnFinder,  
createQuery,  
...
```

### **frontend/src/metabase/querying/segments/components/SegmentEditor/ClauseStep/index.ts**

```
export * from "./ClauseStep";
```

### **frontend/src/metabase/querying/segments/components/SegmentEditor/DataStep/index.ts**

```
export * from "./DataStep";
```

### **frontend/src/metabase/querying/segments/components/SegmentEditor/FilterStep/index.ts**

```
export * from "./FilterStep";
```

### **frontend/src/metabase/querying/segments/components/SegmentEditor/PreviewStep/index.ts**

```
export * from "./PreviewStep";
```

### **frontend/src/metabase/querying/segments/components/SegmentEditor/index.ts**

```
export * from "./SegmentEditor";
```

### **frontend/src/metabase/querying/viz-settings/utils/sync-viz-settings.ts**

```
import * as Lib from "metabase-lib";  
import {  
  getColumnKey,  
  getColumnNameFromKey,  
} from "metabase-lib/v1/queries/utils/column-key";  
import {  
  isColumnNameCollapsedRowsSetting,  
  ...  
}
```

### **frontend/src/metabase/questions/actions.ts**

```
import Questions from "metabase/entities/questions";  
import Tables from "metabase/entities/tables";  
import type { Card, TableId, UnsavedCard } from "metabase-types/api";  
import { isSavedCard } from...
```

### **frontend/src/metabase/questions/components/QuestionMoveToast/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/questions/constants.ts**

```
export const QUESTION_NAME_MAX_LENGTH = 254;
```

### **frontend/src/metabase/reducers-common.ts**

## Analyst Base

Reducers shared between "main" and "public" apps

### **frontend/src/metabase/reducers-main.ts**

Reducers needed for main application

### **frontend/src/metabase/reducers-public.ts**

Reducers needed for public questions and dashboards

### **frontend/src/metabase/redux/app.ts**

```
import {  
  type PayloadAction,  
  createAction,  
  createSlice,  
} from "@reduxjs/toolkit";  
import { LOCATION_CHANGE, push } from "react-router-redux";
```

```
import {  
  isSmallScreen,  
  openInBlankWindow,  
  ...
```

### **frontend/src/metabase/redux/auth.ts**

```
import { createReducer } from "@reduxjs/toolkit";  
  
import { login, loginGoogle, pauseRedirect } from "metabase/auth/actions";  
  
const initialState = {  
  loginPending: false,  
  redirect:...
```

### **frontend/src/metabase/redux/downloads-analytics.ts**

```
import { trackSchemaEvent } from "metabase/lib/analytics";  
  
import type { ResourceAccessedVia, ResourceType } from "../downloads";  
  
const SCHEMA = "downloads";  
  
export const trackDownloadResults = ({  
  ...
```

### **frontend/src/metabase/redux/downloads.ts**

```
import { createSlice } from "@reduxjs/toolkit";  
import { t } from "ttag";  
import _ from "underscore";
```



## Analyst Base

```
import api, { GET, POST } from "metabase/lib/api";
import { isWithinIframe, openSaveDialog }...
```

### **frontend/src/metabase/redux/downloads.unit.spec.ts**

```
import api from "metabase/lib/api";
import Question from "metabase-lib/v1/Question";
import { createMockCard } from "metabase-types/api/mocks";
```

```
import { getChartFileName, getDatasetDownloadUrl }...
```

### **frontend/src/metabase/redux/embed/embed.ts**

```
import {
  type PayloadAction,
  asyncThunkCreator,
  buildCreateSlice,
} from "@reduxjs/toolkit";
import { compose, pick } from "underscore";
```

```
import { parseSearchOptions } from...
```

### **frontend/src/metabase/redux/embed/embed.unit.spec.ts**

```
import { type Dispatch, configureStore } from "@reduxjs/toolkit";

import { reducer as embeddingDataPickerReducer } from "../embedding-data-picker";

import {
  ...
```

### **frontend/src/metabase/redux/embed/index.ts**

```
export * from "../embed";
```

### **frontend/src/metabase/redux/embedding-data-picker.ts**

```
import type { PayloadAction } from "@reduxjs/toolkit";
import { createSlice } from "@reduxjs/toolkit";
import _ from "underscore";
```

```
import type {
  EmbeddingDataPickerState,
  EmbeddingEntityType,
}...
```

### **frontend/src/metabase/redux/entities.js**

```
import * as entitiesMap from "metabase/entities";
import { combineEntities } from "metabase/lib/entities";
```

## Analyst Base

```
const entitiesArray = Object.values(entitiesMap);
```

```
export const { entities, reducer,...
```

### **frontend/src/metabase/redux/metadata.js**

```
import { getIn } from "icepick";
```

```
import _ from "underscore";
```

```
import { cardApi, dashboardApi, datasetApi } from "metabase/api";
```

```
import Databases from "metabase/entities/databases";
```

```
import Fields from...
```

### **frontend/src/metabase/redux/metadata.unit.spec.js**

```
import Fields from "metabase/entities/fields";
```

```
import { fetchField } from "../metadata";
```

```
describe("deprecated metadata actions", () => {
```

```
  let dispatch;
```

```
  beforeEach(() => {
```

```
    ...
```

### **frontend/src/metabase/redux/requests.js**

```
import { assoc, getIn, updateIn } from "icepick";
```

```
import { createAction, handleActions } from "redux-actions";
```

```
export const setRequestLoading = createAction(
```

```
  ...
```

### **frontend/src/metabase/redux/revisions.js**

```
import { assocIn } from "icepick";
```

```
import _ from "underscore";
```

```
import { FETCH_REVISIONS } from "../metadata";
```

```
// NOTE: actions are still in metabase/redux/metadata
```

```
export default (state = {},...
```

### **frontend/src/metabase/redux/settings.ts**

```
import { createAction, createReducer } from "@reduxjs/toolkit";
```

```
import { sessionApi } from "metabase/api";
```

## Analyst Base

```
import { createAsyncThunk } from "metabase/lib/redux";
import { SettingsApi } from...
```

### frontend/src/metabase/redux/ui.ts

```
import { createAction, handleActions } from "metabase/lib/redux";

export const SET_OPEN_MODAL = "metabase/ui/SET_OPEN_MODAL";
export const CLOSE_MODAL = "metabase/ui/CLOSE_MODAL";

export const...
```

### frontend/src/metabase/redux/undo.ts

```
import { createSelector } from "@reduxjs/toolkit";
import { createRef } from "react";
import type { Action } from "redux-actions";
import _ from "underscore";
```

```
import { createAction,...
```

### frontend/src/metabase/redux/undo.unit.spec.ts

```
import { configureStore } from "@reduxjs/toolkit";

import { act } from "__support__/ui";
import type { Dispatch } from "metabase-types/store";

import {
  addUndo,
  dismissAllUndo,
  dismissUndo,
  ...
```

### frontend/src/metabase/redux/uploads.ts

```
import { assocIn, dissocIn, updateIn } from "icepick";
import { t } from "ttag";

import { cardApi } from "metabase/api";
import Collections from "metabase/entities/collections";
import {...
```

### frontend/src/metabase/redux/uploads.unit.spec.js

```
import fetchMock from "fetch-mock";

import { getStore } from "__support__/entities-store";
import { Api } from "metabase/api";
```

## Analyst Base

```
import { mainReducers } from "metabase/reducers-main";
import {...
```

### **frontend/src/metabase/redux/user.ts**

```
import { createAction, createReducer } from "@reduxjs/toolkit";

import Dashboards from "metabase/entities/dashboards";
import { createAsyncThunk } from "metabase/lib/redux";
import {...
```

### **frontend/src/metabase/reference/reference.js**

```
import { assoc } from "icepick";

import { createAction, handleActions } from "metabase/lib/redux";

import { filterUntouchedFields, isEmptyObject } from "../utils.js";
```

```
export const SET_ERROR =...
```

### **frontend/src/metabase/reference/segments/SegmentList/index.ts**

```
export { SegmentList } from "../SegmentList";
```

### **frontend/src/metabase/reference/segments/SegmentList/tests/common.unit.spec.ts**

```
import { screen } from "__support__/ui";
import { createMockUser } from "metabase-types/api/mocks";

import { setup } from "../setup";
```

```
describe("SegmentList (OSS)", () => {
  describe("Admins", () =>...
```

### **frontend/src/metabase/reference/segments/SegmentList/tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";
import { createMockUser } from "metabase-types/api/mocks";
```

```
import type { SetupOpts } from "../setup";
import { setup as baseSetup } from "../setup";
```

```
function...
```

### **frontend/src/metabase/reference/segments/SegmentList/tests/premium.unit.spec.ts**

## Analyst Base

```
import { screen } from "__support__/ui";
import { createMockUser } from "metabase-types/api/mocks";
```

```
import type { SetupOpts } from "./setup";
import { setup as baseSetup } from "./setup";
```

function...

### **frontend/src/metabase/reference/selectors.js**

```
import { createSelector } from "@reduxjs/toolkit";
import { getIn } from "icepick";
```

```
import Dashboards from "metabase/entities/dashboards";
import { resourceListToMap } from...
```

### **frontend/src/metabase/reference/utils.js**

```
import dayjs from "dayjs";
import { assoc } from "icepick";
import { t } from "ttag";
```

```
import { humanize, titleize } from "metabase/lib/formatting";
import * as Urls from "metabase/lib/urls";
import...
```

### **frontend/src/metabase/reference/utils.unit.spec.js**

```
import { createMockMetadata } from "__support__/metadata";
import { separateTablesBySchema } from "metabase/reference/databases/TableList";
import { databaseToForeignKeys, getQuestion } from...
```

### **frontend/src/metabase/schema.js**

normalizr schema for use in actions/reducers

### **frontend/src/metabase/search/analytics.ts**

```
import { trackSchemaEvent } from "metabase/lib/analytics";
import type { SearchRequest, SearchResponse } from "metabase-types/api";
```

```
type SearchRequestFilter = Pick<
  SearchRequest,
  |...
```

### **frontend/src/metabase/search/components/DropdownSidebarFilter/index.ts**

```
export { DropdownSidebarFilter } from "./DropdownSidebarFilter";
```

### **frontend/src/metabase/search/components/InfoText/index.ts**

```
export * from "./InfoText";
```

### **frontend/src/metabase/search/components/SearchFilterDateDisplay/index.ts**

```
export * from "./SearchFilterDateDisplay";
```

### **frontend/src/metabase/search/components/SearchFilterDatePicker/index.ts**

```
export * from "./SearchFilterDatePicker";
```

### **frontend/src/metabase/search/components/SearchFilterPopoverWrapper/index.ts**

```
export { SearchFilterPopoverWrapper } from "./SearchFilterPopoverWrapper";
```

### **frontend/src/metabase/search/components/SearchResult/components/constants.ts**

```
export const DEFAULT_ICON_SIZE = 16;
```

```
export const LARGE_ICON_SIZE = 20;
```

### **frontend/src/metabase/search/components/SearchResult/components/index.ts**

```
export { Context } from "./Context";
```

```
export { ItemIcon } from "./ItemIcon";
```

```
export * from "./constants";
```

### **frontend/src/metabase/search/components/SearchResult/components/utils.ts**

```
import type { WrappedResult } from "metabase/search/types";
```

```
import type { IconComponentProps } from "./ItemIcon";
```

```
export const isWrappedResult = (
  item: IconComponentProps["item"],
): item is...
```

### **frontend/src/metabase/search/components/SearchResult/index.ts**

```
export { SearchResult } from "./SearchResult";
```

```
export * from "./components";
```

```
export * from "./SearchResult.styled";
```

### **frontend/src/metabase/search/components/SearchResult/tests/util.ts**

```
import type { WrappedResult } from "metabase/search/types";
```

```
import { createMockSearchResult } from "metabase-types/api/mocks";
```

```
export const createWrappedSearchResult = (
  options:...
```

### **frontend/src/metabase/search/components/SearchResultLink/index.ts**

```
export * from "./SearchResultLink";
```

### **frontend/src/metabase/search/components/SearchSidebar/index.ts**

## Analyst Base

```
export { SearchSidebar } from "./SearchSidebar";
```

### **frontend/src/metabase/search/components/SearchUserPicker/index.ts**

```
export { SearchUserPicker } from "./SearchUserPicker";  
export * from "./SearchUserPicker.styled";
```

### **frontend/src/metabase/search/components/ToggleSidebarFilter/index.ts**

```
export { ToggleSidebarFilter } from "./ToggleSidebarFilter";  
export type { ToggleSidebarFilterProps } from "./ToggleSidebarFilter";
```

### **frontend/src/metabase/search/components/UserListElement/index.ts**

```
export { UserListElement } from "./UserListElement";
```

### **frontend/src/metabase/search/components/UserNameDisplay/index.ts**

```
export { UserNameDisplay } from "./UserNameDisplay";  
export type { UserNameDisplayProps } from "./UserNameDisplay";
```

### **frontend/src/metabase/search/components/filters/CreatedByFilter/index.ts**

```
export * from "./CreatedByFilter";
```

### **frontend/src/metabase/search/components/filters/NativeQueryFilter/index.ts**

```
export { NativeQueryFilter } from "./NativeQueryFilter";
```

### **frontend/src/metabase/search/components/filters/TypeFilter/index.ts**

```
export * from "./TypeFilter";
```

### **frontend/src/metabase/search/constants.ts**

```
import type { EnabledSearchModel } from "metabase-types/api";
```

```
export const SearchFilterKeys = {  
  Type: "type",  
  Verified: "verified",  
  CreatedBy: "created_by",  
  CreatedAt: "created_at",  
  ...  
};
```

### **frontend/src/metabase/search/containers/constants.ts**

```
export const PAGE_SIZE = 50;
```

### **frontend/src/metabase/search/types.ts**

```
import type { Location } from "history";  
import type { ComponentType } from "react";
```

```
import type { SearchFilterKeys } from "metabase/search/constants";  
import type { IconName } from...
```

### **frontend/src/metabase/search/utils/enabled-search-type/enabled-search-type.ts**

## Analyst Base

```
import { enabledSearchTypes } from "metabase/search/constants";
import type { EnabledSearchModel } from "metabase-types/api";
```

```
export function isEnabledSearchModelType(
  value: unknown,
): value is...
```

### **frontend/src/metabase/search/utils/enabled-search-type/enabled-search-type.unit.spec.ts**

```
import {
  filterEnabledSearchTypes,
  isEnabledSearchModelType,
} from "metabase/search/utils";
```

```
const TEST_VALID_VALUES = [
  "collection",
  "dashboard",
  "card",
  "database",
  "table",
  ...
```

### **frontend/src/metabase/search/utils/enabled-search-type/index.ts**

```
export {
  isEnabledSearchModelType,
  filterEnabledSearchTypes,
} from "./enabled-search-type";
```

### **frontend/src/metabase/search/utils/index.ts**

```
export * from "./search-location";
export * from "./user-search-params";
export * from "./enabled-search-type";
```

### **frontend/src/metabase/search/utils/search-location/index.ts**

```
export * from "./search-location";
```

### **frontend/src/metabase/search/utils/search-location/search-location.ts**

```
import _ from "underscore";
```

```
import { SearchFilterKeys } from "metabase/search/constants";
import type {
  SearchAwareLocation,
  URLSearchFilterQueryParams,
} from "metabase/search/types";
```



## Analyst Base

export...

### **frontend/src/metabase/search/utils/search-location/search-location.unit.spec.ts**

```
import { SearchFilterKeys } from "metabase/search/constants";
import type { SearchAwareLocation } from "metabase/search/types";
import {
  getFiltersFromLocation,
  getSearchTextFromLocation,
  ...
```

### **frontend/src/metabase/search/utils/user-search-params/index.ts**

```
export { parseUserIdArray, stringifyUserIdArray } from "./user-search-params";
```

### **frontend/src/metabase/search/utils/user-search-params/user-search-params.ts**

```
import { isNotNull } from "metabase/lib/types";
import type { SearchQueryParamValue } from "metabase/search/types";
import type { UserId } from "metabase-types/api";
```

```
export const parseUserIdArray =...
```

### **frontend/src/metabase/search/utils/user-search-params/user-search-params.unit.spec.ts**

```
import {
  parseUserId,
  parseUserIdArray,
  stringifyUserIdArray,
} from "./user-search-params";
```

```
describe("parseUserIdArray", () => {
  it("should return a UserId array when value is a string", ()...
```

### **frontend/src/metabase/selectors/app.ts**

```
import type { Selector } from "@reduxjs/toolkit";
import { createSelector } from "@reduxjs/toolkit";
import type { Location } from "history";
```

```
import {
  getDashboard,
  getDashboardId,
  ...
```

### **frontend/src/metabase/selectors/data.ts**

```
import { getEngineNativeType } from "metabase/lib/engine";
```

## Analyst Base

```
import type DatabaseEntity from "metabase-lib/v1/metadata/Database";
import type { Database } from "metabase-types/api";
```

```
export const...
```

### **frontend/src/metabase/selectors/data.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import { checkNotNull } from "metabase/lib/types";
import type { Database } from "metabase-types/api";
import { createMockDatabase } from...
```

### **frontend/src/metabase/selectors/embed.ts**

```
import { createSelector } from "@reduxjs/toolkit";

import { isEmbeddingSdk } from "metabase/embedding-sdk/config";
import { isWithinIframe } from "metabase/lib/dom";
import type {...
```

### **frontend/src/metabase/selectors/embedding-data-picker.ts**

```
import type { State } from "metabase-types/store";
import type { EmbeddingEntityType } from "metabase-types/store/embedding-data-picker";
```

```
export const getEntityTypes = (state: State):...
```

### **frontend/src/metabase/selectors/metadata.ts**

```
import { createSelector } from "@reduxjs/toolkit";
import { normalize } from "normalizr";

import { FieldSchema } from "metabase/schema";
import Question from "metabase-lib/v1/Question";
import...
```

### **frontend/src/metabase/selectors/metadata.unit.spec.ts**

```
import { createMockEntitiesState } from "__support__/store";
import { checkNotNull } from "metabase/lib/types";
import { getMetadata } from "metabase/selectors/metadata";
import Metadata from...
```

### **frontend/src/metabase/selectors/routing.ts**

```
import type { State } from "metabase-types/store";

export const getLocation = (state: State) =>
  state.routing?.locationBeforeTransitions ?? { pathname: "" };
```

### **frontend/src/metabase/selectors/selectors.unit.spec.ts**

## Analyst Base

```
import {
  createMockTokenFeatures,
  createMockTokenStatus,
} from "metabase-types/api/mocks";
import {
  createMockSettingsState,
  createMockState,
} from "metabase-types/store/mocks";

import {
  ...
```

### **frontend/src/metabase/selectors/settings.ts**

```
import { createSelector } from "@reduxjs/toolkit";

import { getPlan } from "metabase/common/utils/plan";
import type { TokenStatus, Version } from "metabase-types/api";
import type { State } from...
```

### **frontend/src/metabase/selectors/ui.ts**

```
import type { State } from "metabase-types/store";

export const currentOpenModal = (state: State) => state.modal;
```

### **frontend/src/metabase/selectors/user.ts**

```
import { createSelector } from "@reduxjs/toolkit";

import { PLUGIN_APPLICATION_PERMISSIONS } from "metabase/plugins";
import type { State } from "metabase-types/store";

export const getUser =...
```

### **frontend/src/metabase/selectors/user.unit.spec.ts**

```
import { createMockUser } from "metabase-types/api/mocks";
import { createMockState } from "metabase-types/store/mocks";

import { getUserIsAdmin } from "../user";
```

```
describe("metabase/selectors/user",...
```

### **frontend/src/metabase/selectors/web-app.ts**

```
import { createSelector } from "reselect";

import { isEmbeddingSdk } from "metabase/embedding-sdk/config";
```

## Analyst Base

```
import { getIsEmbeddingIframe } from "../embed";
import { getSetting } from...
```

### **frontend/src/metabase/selectors/whitelabel/index.ts**

```
import { PLUGIN_SELECTORS } from "metabase/plugins";
import type { State } from "metabase-types/store";

export function getWhiteLabeledLoadingMessageFactory(state: State) {
  return...
```

### **frontend/src/metabase/selectors/whitelabel/tests/common.unit.spec.ts**

```
import {
  getApplicationName,
  getCanWhitelabel,
  getIsWhiteLabeling,
  getShowMetabaseLinks,
  getWhiteLabeledLoadingMessageFactory,
} from "..";

import { setup } from...
```

### **frontend/src/metabase/selectors/whitelabel/tests/enterprise.unit.spec.ts**

```
import {
  getApplicationName,
  getCanWhitelabel,
  getIsWhiteLabeling,
  getShowMetabaseLinks,
  getWhiteLabeledLoadingMessageFactory,
} from "..";

import type { SetupOpts } from "../setup";
import...
```

### **frontend/src/metabase/selectors/whitelabel/tests/premium.unit.spec.ts**

```
import {
  getApplicationName,
  getCanWhitelabel,
  getIsWhiteLabeling,
  getShowMetabaseLinks,
  getWhiteLabeledLoadingMessageFactory,
} from "..";

import type { SetupOpts } from "../setup";
```

## Analyst Base

import...

### **frontend/src/metabase/services.js**

import \_ from "underscore";

import api, { DELETE, GET, POST, PUT } from "metabase/lib/api";

import { IS\_EMBED\_PREVIEW } from "metabase/lib/embed";

import { PLUGIN\_API, PLUGIN\_CONTENT\_TRANSLATION }...

### **frontend/src/metabase/services.unit.spec.ts**

import fetchMock from "fetch-mock";

import { createMockEntitiesState } from "\_\_support\_\_/store";

import { defer } from "metabase/lib/promise";

import { getMetadata } from...

### **frontend/src/metabase/setup/actions.ts**

import { createAction } from "@reduxjs/toolkit";

import { t } from "ttag";

import { createDatabase } from "metabase/admin/databases/database";

import {

    initializeSettings,

    updateSetting,

...

### **frontend/src/metabase/setup/analytics.ts**

import { trackSchemaEvent } from "metabase/lib/analytics";

import type { SetupVersion } from "metabase-types/analytics/setup";

import type { UsageReason } from "metabase-types/api";

import type {...

### **frontend/src/metabase/setup/components/ActiveStep/index.ts**

export \* from "./ActiveStep";

### **frontend/src/metabase/setup/components/CloudMigrationHelp/index.ts**

export \* from "./CloudMigrationHelp";

### **frontend/src/metabase/setup/components/CompletedStep/index.ts**

export \* from "./CompletedStep";

### **frontend/src/metabase/setup/components/DataUsageStep/index.ts**

export \* from "./DataUsageStep";

### **frontend/src/metabase/setup/components/DatabaseHelp/index.ts**

## Analyst Base

```
export * from "./DatabaseHelp";
```

### **frontend/src/metabase/setup/components/DatabaseStep/index.ts**

```
export * from "./DatabaseStep";
```

### **frontend/src/metabase/setup/components/EmbeddingSetup/index.ts**

```
export { WelcomeStep } from "./steps/WelcomeStep";
```

```
export { DataConnectionStep } from "./steps/DataConnectionStep";
```

```
export { TableSelectionStep } from "./steps/TableSelectionStep";
```

```
export {...
```

### **frontend/src/metabase/setup/components/EmbeddingSetup/steps/embeddingSetupSteps.ts**

```
import { t } from "ttag";
```

```
import type { IconName } from "metabase/ui";
```

```
export interface StepDefinition {
```

```
  key: string;
```

```
  title: string;
```

```
  icon: IconName;
```

```
  visibleInSidebar: boolean;
```

```
}
```

```
export...
```

### **frontend/src/metabase/setup/components/InactiveStep/index.ts**

```
export * from "./InactiveStep";
```

### **frontend/src/metabase/setup/components/InviteUserForm/index.ts**

```
export * from "./InviteUserForm";
```

### **frontend/src/metabase/setup/components/LanguageSelector/index.ts**

```
export { LanguageSelector } from "./LanguageSelector";
```

### **frontend/src/metabase/setup/components/LanguageStep/index.ts**

```
export * from "./LanguageStep";
```

### **frontend/src/metabase/setup/components/LicenseTokenStep/index.ts**

```
export { LicenseTokenStep } from "./LicenseTokenStep";
```

### **frontend/src/metabase/setup/components/SettingsPage/index.ts**

```
export * from "./SettingsPage";
```

### **frontend/src/metabase/setup/components/Setup/index.ts**

```
export * from "./Setup";
```

## Analyst Base

### **frontend/src/metabase/setup/components/SetupCardContainer/index.ts**

```
export * from "./SetupCardContainer.styled";
```

### **frontend/src/metabase/setup/components/SetupHelp/index.ts**

```
export * from "./SetupHelp";
```

### **frontend/src/metabase/setup/components/SetupSection/index.ts**

```
export * from "./SetupSection";
```

### **frontend/src/metabase/setup/components/UsageQuestionStep/index.ts**

```
export { UsageQuestionStep } from "./UsageQuestionStep";
```

### **frontend/src/metabase/setup/components/UserForm/index.ts**

```
export * from "./UserForm";
```

### **frontend/src/metabase/setup/components/UserStep/index.ts**

```
export * from "./UserStep";
```

### **frontend/src/metabase/setup/components/WelcomePage/index.ts**

```
export * from "./WelcomePage";
```

### **frontend/src/metabase/setup/components/types.ts**

```
export type NumberedStepProps = { stepLabel: number };
```

### **frontend/src/metabase/setup/constants.ts**

```
export const LOCALE_TIMEOUT = 300;
```

```
export const SUBSCRIBE_URL =
```

```
  "https://metabase.us10.list-manage.com/subscribe/post?u=869fec0e4689e8fd1db91e795&id=b9664113a8";
```

```
export const SUBSCRIBE_TOKEN =...
```

### **frontend/src/metabase/setup/index.ts**

```
export * from "./actions";
```

```
export * from "./selectors";
```

### **frontend/src/metabase/setup/reducers.ts**

```
import { createReducer } from "@reduxjs/toolkit";
```

```
import type { SetupState } from "metabase-types/store";
```

```
import {  
  loadLocaleDefaults,  
  loadUserDefaults,  
  selectStep,  
  skipDatabase,
```

...

### **frontend/src/metabase/setup/selectors.ts**

```
import { createSelector } from "@reduxjs/toolkit";

import { isEEBuild } from "metabase/lib/utils";
import { getSetting } from "metabase/selectors/settings";
import type {
  DatabaseData,
  ...
```

### **frontend/src/metabase/setup/types.ts**

```
export type SetupStep =
  | "welcome"
  | "language"
  | "user_info"
  | "usage_question"
  | "db_connection"
  | "license_token"
  | "data_usage"
  | "completed";
```

### **frontend/src/metabase/setup/useStep.ts**

```
import { useDispatch, useSelector } from "metabase/lib/redux";

import { selectStep } from "../actions";
import {
  getIsSetupCompleted,
  getIsStepActive,
  getIsStepCompleted,
} from...
```

### **frontend/src/metabase/setup/utils.ts**

```
import { getIn } from "icepick";
import _ from "underscore";

import MetabaseSettings from "metabase/lib/settings";
import { UtilApi } from "metabase/services";
import type { LocaleData } from...
```

### **frontend/src/metabase/static-viz/components/ComboChart/index.ts**

```
export * from "../ComboChart";
```

### **frontend/src/metabase/static-viz/components/ComboChart/stories-data/index.ts**

```
import twoBarsTwoAreasOneLineLinear from "../2-bars-2-areas-1-line-linear.json";
```



## Analyst Base

```
import twoBarsTwoAreasOneLineLog from "./2-bars-2-areas-1-line-log.json";
import twoBarsTwoAreasOneLinePower from...
```

### **frontend/src/metabase/static-viz/components/FunnelBarChart/index.ts**

```
export * from "./FunnelBarChart";
```

### **frontend/src/metabase/static-viz/components/FunnelBarChart/stories-data/index.ts**

```
import funnelBarCategorical from "./funnel-bar-categorical.json";
import funnelBarOrderedRows from "./funnel-bar-ordered-rows.json";
import funnelBarUnorderedRows from...
```

### **frontend/src/metabase/static-viz/components/FunnelChart/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/static-viz/components/FunnelChart/stories-data.ts**

```
import type { FunnelProps } from "metabase/static-viz/components/FunnelChart/FunnelChart";
```

```
export const DEFAULT: FunnelProps = {
  data: [
    ["Visitors", 1000],
    ["Started sign up", 300],
    ...
  ]
}
```

### **frontend/src/metabase/static-viz/components/FunnelChart/types.ts**

```
import type { NumberFormatOptions } from "metabase/static-viz/lib/numbers";
import type { VisualizationSettings } from "metabase-types/api";
```

```
export type Step = string | number;
export type Measure =...
```

### **frontend/src/metabase/static-viz/components/FunnelChart/utils/funnel.ts**

```
import type { PolygonProps } from "@visx/shape/lib/shapes/Polygon";
```

```
import { isNotNull } from "metabase/lib/types";
import { CHAR_SIZES_FONT_WEIGHT } from...
```

### **frontend/src/metabase/static-viz/components/FunnelChart/utils/funnel.unit.spec.ts**

```
import { merge } from "icepick";
```

```
import type { FunnelDatum, FunnelSettings } from "../types";
```

```
import {
  calculateFunnelPolygonPoints,
}
```

## Analyst Base

```
calculateFunnelSteps,  
reorderData,  
} from...
```

### **frontend/src/metabase/static-viz/components/FunnelChart/utils/index.ts**

```
export * from "../funnel";  
export * from "../margin";
```

### **frontend/src/metabase/static-viz/components/FunnelChart/utils/margin.ts**

```
import { formatNumber } from "metabase/static-viz/lib/numbers";  
import {  
  measureTextHeight,  
  measureTextWidth,  
} from "metabase/static-viz/lib/text";
```

```
import type { FunnelDatum, FunnelSettings }...
```

### **frontend/src/metabase/static-viz/components/Gauge/stories-data.ts**

```
import type { GaugeContainerProps } from "metabase/static-viz/components/Gauge/GaugeContainer";
```

```
export const DEFAULT: Omit<GaugeContainerProps, "getColor"> = {  
  card: {  
    visualization_settings:...
```

### **frontend/src/metabase/static-viz/components/Gauge/units.spec.ts**

```
import type { NumberFormatOptions } from "metabase/static-viz/lib/numbers";
```

```
import {  
  GAUGE_ARC_ANGLE,  
  SEGMENT_LABEL_ANCHOR_THRESHOLD_ANGLE,  
} from "../constants";  
import {  
  ...
```

### **frontend/src/metabase/static-viz/components/Legend/constants.ts**

```
export const LEGEND_CIRCLE_SIZE = 10;  
export const LEGEND_CIRCLE_MARGIN_RIGHT = 4;  
export const LEGEND_ITEM_MARGIN_RIGHT = 24;  
export const LEGEND_ITEM_MARGIN_RIGHT_GRID = 16;
```

```
export const...
```

### **frontend/src/metabase/static-viz/components/Legend/index.ts**

```
export * from "../Legend";
```

## Analyst Base

### **frontend/src/metabase/static-viz/components/Legend/types.ts**

```
import type { LegendItem } from "metabase/visualizations/echarts/cartesian/model/types";

export type PositionedLegendItem = LegendItem & {
  left: number;
  top: number;
  width?: number;
};
```

### **frontend/src/metabase/static-viz/components/Legend/utils.ts**

```
import { measureTextWidth } from "metabase/static-viz/lib/text";
import type { LegendItem } from "metabase/visualizations/echarts/cartesian/model/types";
import { truncateText } from...
```

### **frontend/src/metabase/static-viz/components/PieChart/stories-data/index.ts**

```
import allNegativeWithOther from "./all-negative-with-other.json";
import allNegative from "./all-negative.json";
import allSettings from "./all-settings.json";
import allZeroMetric44847 from...
```

### **frontend/src/metabase/static-viz/components/ProgressBar/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/static-viz/components/ProgressBar/stories-data.ts**

```
export const ZERO = {
  data: {
    value: 0,
    goal: 100000,
  },
  settings: {
    format: {
      number_style: "currency",
      currency: "USD",
      currency_style: "symbol",
      decimals:...
```

### **frontend/src/metabase/static-viz/components/ProgressBar/types.ts**

```
export type ProgressBarData = {
  value: number;
  goal: number;
};
```

### **frontend/src/metabase/static-viz/components/ProgressBar/utils.ts**

```
import Color from "color";
```

## Analyst Base

```
import { t } from "ttag";
```

```
import { measureTextWidth } from "metabase/static-viz/lib/text";
```

```
import type { ProgressBarData } from "./types";
```

```
const createPalette = (color:...
```

### **frontend/src/metabase/static-viz/components/RowChart/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/static-viz/components/RowChart/stories-data.ts**

```
import type { StaticRowChartProps } from "metabase/static-viz/components/RowChart/RowChart";
```

```
export const MULTIPLE_SERIES: StaticRowChartProps = {  
  settings: {  
    "graph.dimensions":...
```

### **frontend/src/metabase/static-viz/components/RowChart/theme.ts**

```
import type { RowChartTheme } from "metabase/visualizations/shared/components/RowChart/types";  
import type { ColorGetter } from "metabase/visualizations/types";
```

```
export const getStaticChartTheme = (  
  ...
```

### **frontend/src/metabase/static-viz/components/RowChart/utils/labels.ts**

```
import type { CartesianChartColumns } from "metabase/visualizations/lib/graph/columns";  
import type { VisualizationSettings } from "metabase-types/api";
```

```
// Uses inverse axis settings to have...
```

### **frontend/src/metabase/static-viz/components/SankeyChart/index.ts**

```
export * from "./SankeyChart";
```

### **frontend/src/metabase/static-viz/components/SankeyChart/stories-data/index.ts**

```
import defaultSettings from "./default-settings.json";
```

```
export const data = { defaultSettings };
```

### **frontend/src/metabase/static-viz/components/ScalarChart/index.ts**

```
export * from "./ScalarChart";
```

### **frontend/src/metabase/static-viz/components/ScalarChart/stories-data/index.ts**

```
import twoScalars from "./two-scalars.json";
```

## Analyst Base

```
export const data = {  
  twoScalars,  
};
```

### **frontend/src/metabase/static-viz/components/ScatterPlot/stories-data/index.ts**

```
import autoYAxisExcludeZeroWithGoal from "./auto-y-axis-exclude-zero-with-goal.json";  
import bubbleSize from "./bubble-size.json";  
import customYAxisRangeWithColumnScaling from...
```

### **frontend/src/metabase/static-viz/components/SmartScalar/index.ts**

```
export * from "./SmartScalar";
```

### **frontend/src/metabase/static-viz/components/StaticVisualization/index.ts**

```
export * from "./StaticVisualization";  
export * from "./types";
```

### **frontend/src/metabase/static-viz/components/StaticVisualization/types.ts**

```
import type {  
  ComputedVisualizationSettings,  
  RenderingContext,  
} from "metabase/visualizations/types";  
import type { RawSeries } from "metabase-types/api";
```

```
export interface StaticChartProps {  
  ...
```

### **frontend/src/metabase/static-viz/components/Text/index.ts**

```
export * from "./Text";
```

### **frontend/src/metabase/static-viz/components/WaterfallChart/stories-data/index.ts**

```
import customColors from "./custom-colors.json";  
import customYAxisRangeWithColumnScaling from "./custom-y-axis-range-with-column-scaling.json";  
import dataLabelsColumnFormatting from...
```

### **frontend/src/metabase/static-viz/constants/accessors.ts**

```
export const DIMENSION_ACCESSORS = {  
  dimension: (row: any[]) => row[0],  
  metric: (row: any[]) => row[1],  
};
```

### **frontend/src/metabase/static-viz/constants/char-sizes.ts**

```
export const CHAR_SIZES_FONT_SIZE = 10;  
export const CHAR_SIZES_FONT_WEIGHT = 400;
```

## Analyst Base

```
// Generated with the mapping tool from here https://github.com/Evgenus/js-server-text-width
export const...
```

### **frontend/src/metabase/static-viz/containers/LegacyStaticChart/index.ts**

```
export * from "./LegacyStaticChart";
```

### **frontend/src/metabase/static-viz/index.js**

```
import "../mock-environment";
import "fast-text-encoding";
```

```
import { setPlatformAPI } from "echarts/core";
import ReactDOMServer from "react-dom/server";
```

```
// eslint-disable-next-line...
```

### **frontend/src/metabase/static-viz/lib/colors.ts**

```
import { colors } from "metabase/lib/colors/colors";
import { color } from "metabase/lib/colors/palette";
import type { ColorPalette } from "metabase/lib/colors/types";
import type { ColorGetter }...
```

### **frontend/src/metabase/static-viz/lib/format.ts**

```
import type { NumberLike, StringLike } from "@visx/scale";

import { formatValue } from "metabase/lib/formatting";
import { getFormattingOptionsWithoutScaling } from...
```

### **frontend/src/metabase/static-viz/lib/numbers.ts**

```
import { formatNumber as appFormatNumber } from "metabase/lib/formatting/numbers";

export type NumberFormatOptions = {
  number_style?: "currency" | "decimal" | "scientific" | "percentage";
  ...
}
```

### **frontend/src/metabase/static-viz/lib/numbers.unit.spec.js**

```
import { formatNumber, formatPercent } from "../numbers";

describe("formatNumber", () => {
  it("should format a number with default options", () => {
    const number = 1500;

    const text =...
```

### **frontend/src/metabase/static-viz/lib/rendering-context.ts**

```
import type { ColorPalette } from "metabase/lib/colors/types";
```

## Analyst Base

```
import { DEFAULT_VISUALIZATION_THEME } from "metabase/visualizations/shared/utils/theme";
import type { RenderingContext } from...
```

### **frontend/src/metabase/static-viz/lib/svg.ts**

```
import type { Element, ElementContent } from "hast";
import { fromHtml } from "hast-util-from-html";
import { toHtml } from "hast-util-to-html";
```

```
// FIXME: instead of Regex parse svg, update,...
```

### **frontend/src/metabase/static-viz/lib/svg.unit.spec.ts**

```
import type { Element, ElementContent } from "hast";
import { fromHtml } from "hast-util-from-html";
```

```
import { patchDominantBaseline } from "../svg";
```

```
const OUTER_TEXT = "outer-text";
const G_ELEM =...
```

### **frontend/src/metabase/static-viz/lib/text.ts**

```
import { init } from "server-text-width";
```

```
import {
  CHAR_SIZES,
  CHAR_SIZES_FONT_SIZE,
  CHAR_SIZES_FONT_WEIGHT,
} from "../constants/char-sizes";
```

```
const FONT_WEIGHT_WIDTH_FACTOR = 0.039;
```

```
export...
```

### **frontend/src/metabase/static-viz/lib/text.unit.spec.js**

```
import { measureTextWidth } from "../text";
```

```
const fontSize = 11;
```

```
describe("measureTextWidth", () => {
  it("should measure text", () => {
    expect(Math.round(measureTextWidth("abc",...
```

### **frontend/src/metabase/static-viz/mock-environment.js**

```
eslint-disable no-prototype-builtins
```

### **frontend/src/metabase/static-viz/register.js**

## Analyst Base

```
import {
  registerVisualization,
  setDefaultVisualization,
} from "metabase/visualizations";
import { AreaChart } from "metabase/visualizations/visualizations/AreaChart";
import { BarChart } from...
```

### **frontend/src/metabase/status/components/DatabaseStatus/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/status/components/DatabaseStatusLarge/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/status/components/DatabaseStatusSmall/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/status/components/DownloadsStatus/index.ts**

```
export * from "./DownloadsStatus";
```

### **frontend/src/metabase/status/components/DownloadsStatusLarge/index.ts**

```
export * from "./DownloadsStatusLarge";
```

### **frontend/src/metabase/status/components/DownloadsStatusSmall/index.ts**

```
export * from "./DownloadsStatusSmall";
```

### **frontend/src/metabase/status/components/FileUploadStatus/index.ts**

```
export * from "./FileUploadStatus";
```

### **frontend/src/metabase/status/components/FileUploadStatusLarge/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/status/components/PublicStatusListing/index.ts**

```
export * from "./PublicStatusListing";
```

### **frontend/src/metabase/status/components/StatusLarge/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/status/components/StatusListing/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/status/components/StatusSmall/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/status/components/utils/downloads.ts**

```
import type { Download } from "metabase-types/store";
```

```
export const isCompleted = (download: Download) =>
```



## Analyst Base

```
download.status === "complete";  
export const isErrorred = (download: Download) =>...
```

### **frontend/src/metabase/status/components/utils/status.ts**

```
import { isReducedMotionPreferred } from "metabase/lib/dom";  
import type { IconName } from "metabase/ui";  
import type { LongTaskStatus } from "metabase-types/api";
```

```
export const getIconName =...
```

### **frontend/src/metabase/status/containers/DatabaseStatus/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/status/hooks/use-check-active-downloads-before-unload.ts**

```
import { useBeforeUnload } from "react-use";  
import { t } from "ttag";
```

```
import { useSelector } from "metabase/lib/redux";  
import { hasActiveDownloads } from "metabase/redux/downloads";
```

```
export const...
```

### **frontend/src/metabase/status/hooks/use-status-visibility.ts**

```
import { useEffect, useState } from "react";
```

```
export const HIDE_DELAY = 6000;
```

```
const useStatusVisibility = (isActive: boolean) => {  
  const [isVisible, setIsVisible] = useState(isActive);
```

```
...
```

### **frontend/src/metabase/store.js**

```
import { combineReducers, configureStore } from "@reduxjs/toolkit";  
import { routerMiddleware, routerReducer as routing } from "react-router-redux";  
import promise from "redux-promise";
```

```
import { Api...
```

### **frontend/src/metabase/styled-components/components/EmotionCacheProvider/index.ts**

```
export * from "../EmotionCacheProvider";
```

### **frontend/src/metabase/styled-components/containers/GlobalStyles/index.ts**

## Analyst Base

```
export { GlobalStyles } from "./GlobalStyles";
```

### **frontend/src/metabase/styled-components/selectors.ts**

```
import { createSelector } from "@reduxjs/toolkit";  
import _ from "underscore";
```

```
import { getEmbedOptions } from "metabase/selectors/embed";  
import { getSettings } from...
```

### **frontend/src/metabase/styled-components/theme/constants.ts**

```
export const SPACE_LEVELS = ["4px", "8px", "16px", "32px", "64px", "128px"];
```

### **frontend/src/metabase/styled-components/theme/css-variables.ts**

```
eslint-disable-next-line no-restricted-imports
```

### **frontend/src/metabase/styled-components/theme/css-variables.unit.spec.ts**

```
import type { MantineTheme } from "metabase/ui";
```

```
import { getThemeSpecificCssVariables } from "../css-variables";
```

```
describe("getThemeSpecificCssVariables", () => {  
  it("returns the correct CSS...
```

### **frontend/src/metabase/styled-components/theme/index.ts**

```
export * from "../media-queries";  
export * from "../space";  
export * from "../typography";
```

### **frontend/src/metabase/styled-components/theme/media-queries.ts**

```
export const breakpointMinExtraSmall = "@media screen and (min-width: 23em)";  
export const breakpointMinSmall = "@media screen and (min-width: 40em)";  
export const breakpointMinMedium = "@media...
```

### **frontend/src/metabase/styled-components/theme/space.ts**

```
import { SPACE_LEVELS as levels } from "../constants";
```

```
/**
```

```
 * Returns a pixel amount: 4px, 8px, 16px, on to 128px  
 * @param {number} level must be an integer between 0 and 5  
 * @returns {string}
```

```
...
```

### **frontend/src/metabase/styled-components/theme/space.unit.spec.ts**

```
import { space } from "../space";
```

## Analyst Base

```
it("returns pixel amount for acceptable levels", () => {  
  expect(space(0)).toBe("4px");  
  expect(space(1)).toBe("8px");  
  expect(space(2)).toBe("16px");  
  ...  
})
```

### **frontend/src/metabase/styled-components/theme/typography.ts**

```
export const monospaceFontFamily = `Monaco, Menlo, Ubuntu Mono, Consolas, Source Code Pro,  
source-code-pro, monospace`;
```

### **frontend/src/metabase/timelines/collections/actions.ts**

```
import { push } from "react-router-redux";  
  
import Timelines from "metabase/entities/timelines";  
import * as Urls from "metabase/lib/urls";  
import type { Timeline } from "metabase-types/api";  
import...
```

### **frontend/src/metabase/timelines/collections/components/EventCard/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/timelines/collections/components/EventList/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/timelines/collections/components>LoadingAndErrorWrapper/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/timelines/collections/components/SearchEmptyState/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/timelines/collections/components/TimelineCard/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/timelines/collections/components/TimelineDetailsModal/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/timelines/collections/components/TimelineEmptyState/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/timelines/collections/components/TimelineIndexModal/index.ts**

## Analyst Base

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/components/TimelineList/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/components/TimelineListModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/containers/DeleteEventModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/containers/DeleteTimelineModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/containers/EditEventModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/containers/EditTimelineModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/containers/MoveEventModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/containers/MoveTimelineModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/containers/NewEventModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/containers/NewEventWithTimelineModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/containers/NewTimelineModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/collections/containers/TimelineArchiveModal/i**

## **ndex.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/timelines/collections/containers/TimelineDetailsModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/timelines/collections/containers/TimelineIndexModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/timelines/collections/containers/TimelineListArchiveModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/timelines/collections/containers/TimelineListModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/timelines/collections/types.ts**

```
export interface MenuItem {  
  title: string;  
  link?: string;  
  action?: () => void;  
}
```

```
export interface ModalParams {  
  slug: string;  
  timelineId?: string;  
  timelineEventId?: string;  
}
```

## **frontend/src/metabase/timelines/common/components/DeleteEventModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/timelines/common/components/DeleteTimelineModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/timelines/common/components/EditEventModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## **frontend/src/metabase/timelines/common/components/EditTimelineModal/index**

**.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/components/EventForm/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/components/FormArchiveButton/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/components/ModalBody/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/components/ModalFooter/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/components/ModalHeader/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/components/MoveEventModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/components/MoveTimelineModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/components/NewEventModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/components/NewTimelineModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/components/TimelineForm/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/components/TimelinePicker/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/common/containers/EventForm/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/questions/components/EventCard/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

## Analyst Base

**frontend/src/metabase/timelines/questions/components/TimelineEmptyState/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/questions/components/TimelineList/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/questions/components/TimelinePanel/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/questions/containers/EditEventModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/questions/containers/MoveEventModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/questions/containers/NewEventModal/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/timelines/questions/containers/TimelinePanel/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/ui/components/buttons/ActionIcon/index.ts**

export { actionIconOverrides } from "./ActionIcon.config";

export { ActionIcon } from "@mantine/core";

export type { ActionIconProps } from "@mantine/core";

**frontend/src/metabase/ui/components/buttons/Button/index.ts**

import type { ButtonProps as MantineButtonProps } from "@mantine/core";

import type { HTMLAttributes } from "react";

export { Button } from "@mantine/core";

export type { ButtonGroupProps } from...

**frontend/src/metabase/ui/components/buttons/CopyButton.ts**

export { CopyButton, type CopyButtonProps } from "@mantine/core";

**frontend/src/metabase/ui/components/buttons/PopoverBackButton/index.ts**

export \* from "./PopoverBackButton";

**frontend/src/metabase/ui/components/buttons/UnstyledButton/index.ts**

export { UnstyledButton } from "@mantine/core";

export type { UnstyledButtonProps } from "@mantine/core";

**frontend/src/metabase/ui/components/buttons/index.ts**

## Analyst Base

```
export * from "./ActionIcon";
export * from "./Button";
export * from "./PopoverBackButton";
export * from "./UnstyledButton";
export * from "./CopyButton";
```

### **frontend/src/metabase/ui/components/data-display/Accordion/Accordion.config.ts**

```
import { Accordion } from "@mantine/core";

import AccordionStyles from "./Accordion.module.css";

export const accordionOverrides = {
  Accordion: Accordion.extend({
    classNames: {
      control:...
```

### **frontend/src/metabase/ui/components/data-display/Accordion/index.ts**

```
export { Accordion } from "@mantine/core";
export type { AccordionProps } from "@mantine/core";
export { accordionOverrides } from "./Accordion.config";
```

### **frontend/src/metabase/ui/components/data-display/Badge/index.ts**

```
export { Badge } from "@mantine/core";
export * from "./Badge.config";
```

### **frontend/src/metabase/ui/components/data-display/Card/Card.config.ts**

```
import { Card } from "@mantine/core";

import CardStyles from "./Card.module.css";

export const cardOverrides = {
  Card: Card.extend({
    classNames: {
      section: CardStyles.section,
    },
  },
  ...
```

### **frontend/src/metabase/ui/components/data-display/Card/index.ts**

```
export { Card } from "@mantine/core";
export type { CardProps } from "@mantine/core";
export { cardOverrides } from "./Card.config";
```

### **frontend/src/metabase/ui/components/data-display/Indicator/index.ts**



## Analyst Base

```
export { Indicator } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/data-display/Kbd/Kbd.config.ts**

```
import { Kbd } from "@mantine/core";
```

```
import S from "./Kbd.module.css";
```

```
export const kbdOverrides = {  
  Kbd: Kbd.extend({  
    classNames: {  
      root: S.root,  
    },  
  }),  
};
```

### **frontend/src/metabase/ui/components/data-display/Kbd/index.ts**

```
export { Kbd } from "@mantine/core";
```

```
export * from "./Kbd.config";
```

### **frontend/src/metabase/ui/components/data-display/index.ts**

```
export * from "./Accordion";  
export * from "./Badge";  
export * from "./Card";  
export * from "./Image";  
export * from "./Indicator";  
export * from "./Kbd";
```

### **frontend/src/metabase/ui/components/feedback/Alert/index.ts**

```
export { Alert } from "@mantine/core";  
export type { AlertProps } from "@mantine/core";  
export { alertOverrides } from "./Alert.config";
```

### **frontend/src/metabase/ui/components/feedback/Loader/index.ts**

```
export { Loader } from "./Loader";  
export type { LoaderProps } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/feedback/Progress/index.ts**

```
export { Progress } from "@mantine/core";  
export * from "./Progress.config";
```

### **frontend/src/metabase/ui/components/feedback/Skeleton/Skeleton.config.ts**

```
import { Skeleton } from "@mantine/core";
```

```
import S from "./Skeleton.module.css";
```

## Analyst Base

```
export const skeletonOverrides = {  
  Skeleton: Skeleton.extend({  
    classNames: {  
      root: S.Skeleton,  
    },  
  },  
  ...  
}
```

### **frontend/src/metabase/ui/components/feedback/Skeleton/index.ts**

```
export { Repeat } from "./Repeat";  
export { Skeleton } from "./Skeleton";  
export * from "./Skeleton.config";
```

### **frontend/src/metabase/ui/components/feedback/index.ts**

```
export * from "./Loader";  
export * from "./Alert";  
export * from "./Progress";  
export * from "./Skeleton";
```

### **frontend/src/metabase/ui/components/icons/Icon/icons/index.ts**

```
import ten_thousand_component from "./10k.svg?component";  
import ten_thousand_source from "./10k.svg?source";  
import one_million_component from "./1m.svg?component";  
import one_million_source from...
```

### **frontend/src/metabase/ui/components/icons/Icon/index.ts**

```
export * from "./Icon";  
export * from "./icons";
```

### **frontend/src/metabase/ui/components/icons/index.ts**

```
export * from "./Icon";
```

### **frontend/src/metabase/ui/components/index.ts**

```
export * from "./buttons";  
export * from "./data-display";  
export * from "./feedback";  
export * from "./icons";  
export * from "./inputs";  
export * from "./layout";  
export * from...
```

### **frontend/src/metabase/ui/components/inputs/Autocomplete/index.ts**

```
export { Autocomplete } from "@mantine/core";  
export type { AutocompleteProps } from "@mantine/core";  
export { autocompleteOverrides } from "./Autocomplete.config";
```

## Analyst Base

### **frontend/src/metabase/ui/components/inputs/Calendar/index.ts**

```
export type { DateValue, DatesRangeValue } from "@mantine/dates";
export { calendarOverrides } from "./Calendar.config";
```

### **frontend/src/metabase/ui/components/inputs/Checkbox/Checkbox.config.ts**

```
import { Checkbox, getSize, rem } from "@mantine/core";
```

```
import CheckboxStyles from "./Checkbox.module.css";
import { CheckboxIcon } from "./CheckboxIcon";
```

```
const SIZES: Record<string, string> = {
  ...
```

### **frontend/src/metabase/ui/components/inputs/Checkbox/index.ts**

```
export { Checkbox } from "@mantine/core";
export type { CheckboxProps, CheckboxGroupProps } from "@mantine/core";
export { checkboxOverrides } from "./Checkbox.config";
```

### **frontend/src/metabase/ui/components/inputs/Chip/Chip.config.ts**

```
import { Chip, type MantineThemeOverride } from "@mantine/core";
```

```
import S from "./Chip.module.css";
```

```
export const chipOverrides: MantineThemeOverride["components"] = {
  Chip: Chip.extend({
    ...
```

### **frontend/src/metabase/ui/components/inputs/Chip/index.ts**

```
export { Chip } from "@mantine/core";
export type { ChipProps, ChipGroupProps } from "@mantine/core";
export { chipOverrides } from "./Chip.config";
```

### **frontend/src/metabase/ui/components/inputs/Combobox/Combobox.config.ts**

```
import {
  Combobox,
  ComboboxChevron,
  type MantineThemeOverride,
} from "@mantine/core";
```

```
import S from "./Combobox.module.css";
```

```
export const comboboxOverrides:...
```

### **frontend/src/metabase/ui/components/inputs/Combobox/index.ts**

## Analyst Base

```
export { Combobox } from "@mantine/core";
export type {
  ComboboxGroupProps,
  ComboboxProps,
  ComboboxItem,
} from "@mantine/core";
export { comboboxOverrides } from "../Combobox.config";
```

### **frontend/src/metabase/ui/components/inputs/DateInput/index.ts**

```
export { DateInput } from "@mantine/dates";
export type { DateInputProps } from "@mantine/dates";
export { dateInputOverrides } from "../DateInput.config";
```

### **frontend/src/metabase/ui/components/inputs/DatePicker/index.ts**

```
export { DatePicker } from "@mantine/dates";
export type { DatePickerProps } from "@mantine/dates";
export { datePickerOverrides } from "../DatePicker.config";
```

### **frontend/src/metabase/ui/components/inputs/FileInput/index.ts**

```
export { FileInput } from "@mantine/core";
export type { FileInputProps } from "@mantine/core";
export { fileInputOverrides } from "../FileInput.config";
```

### **frontend/src/metabase/ui/components/inputs/Input/index.ts**

```
export { Input } from "@mantine/core";
export { inputOverrides } from "../Input.config";
```

### **frontend/src/metabase/ui/components/inputs/MonthPicker/index.ts**

```
export { MonthPicker } from "@mantine/dates";
export type { MonthPickerProps } from "@mantine/dates";
export { monthPickerOverrides } from "../MonthPicker.config";
```

### **frontend/src/metabase/ui/components/inputs/MultiAutocomplete/MultiAutocompleteOption/index.ts**

```
export * from "../MultiAutocompleteOption";
```

### **frontend/src/metabase/ui/components/inputs/MultiAutocomplete/MultiAutocompleteValue/index.ts**

```
export * from "../MultiAutocompleteValue";
```

### **frontend/src/metabase/ui/components/inputs/MultiAutocomplete/index.ts**

```
export * from "../MultiAutocomplete";
export * from "../MultiAutocompleteOption";
export * from "../MultiAutocompleteValue";
```

### **frontend/src/metabase/ui/components/inputs/MultiAutocomplete/use-multi-auto**

## **complete/index.ts**

```
export * from "../use-multi-autocomplete";
```

## **frontend/src/metabase/ui/components/inputs/MultiAutocomplete/use-multi-autocomplete/use-multi-autocomplete.ts**

```
import {
  type ComboboxData,
  type ComboboxItem,
  type ComboboxParsedItem,
  getParsedComboboxData,
  isOptionsGroup,
  useCombobox,
} from "@mantine/core";
import { useWindowEvent } from...
```

## **frontend/src/metabase/ui/components/inputs/MultiSelect/index.ts**

```
export { MultiSelect, type MultiSelectProps } from "@mantine/core";
export { multiSelectOverrides } from "../MultiSelect.config";
```

## **frontend/src/metabase/ui/components/inputs/NumberInput/index.ts**

```
export { NumberInput } from "../NumberInput";
export type { NumberInputProps } from "../NumberInput";
```

## **frontend/src/metabase/ui/components/inputs/Pill/Pill.config.ts**

```
import { type MantineThemeOverride, Pill } from "@mantine/core";

import S from "../Pill.module.css";

export const pillOverrides: MantineThemeOverride["components"] = {
  Pill: Pill.extend({
    ...
```

## **frontend/src/metabase/ui/components/inputs/Pill/index.ts**

```
export { Pill } from "@mantine/core";
export { pillOverrides } from "../Pill.config";
```

## **frontend/src/metabase/ui/components/inputs/PillsInput/PillsInput.config.ts**

```
import {
  type MantineThemeOverride,
  PillsInput,
  PillsInputField,
} from "@mantine/core";

import S from "../PillsInput.module.css";
```

## Analyst Base

```
export const pillsInputOverrides:...
```

### **frontend/src/metabase/ui/components/inputs/PillsInput/index.ts**

```
export { pillsInputOverrides } from "./PillsInput.config";
```

### **frontend/src/metabase/ui/components/inputs/QuarterPicker/index.ts**

```
export * from "./QuarterPicker";
```

### **frontend/src/metabase/ui/components/inputs/Radio/Radio.config.ts**

```
import { Radio, getSize, rem } from "@mantine/core";
```

```
import RadioStyles from "./Radio.module.css";
```

```
const SIZES: Record<string, string> = {  
  md: rem(20),  
};
```

```
export const radioOverrides = {  
  ...
```

### **frontend/src/metabase/ui/components/inputs/SegmentedControl/SegmentedControl.config.ts**

```
import {  
  type MantineTheme,  
  type MantineThemeOverride,  
  SegmentedControl,  
  type SegmentedControlProps,  
  rem,  
} from "@mantine/core";
```

```
import S from "./SegmentedControl.module.css";
```

```
export...
```

### **frontend/src/metabase/ui/components/inputs/Select/SelectItem/index.ts**

```
export * from "./DefaultSelectItem";  
export * from "./SelectItem";  
export * from "./utils";
```

### **frontend/src/metabase/ui/components/inputs/Select/SelectItem/utils.ts**

```
import { type MantineSize, getSize, rem } from "@mantine/core";
```

```
const FONT_SIZES = {
```

## Analyst Base

```
xs: rem(12),  
md: rem(14),  
};
```

```
const LINE_HEIGHTS = {  
  xs: rem(16),  
  md: rem(24),  
};
```

export function...

### **frontend/src/metabase/ui/components/inputs/Select/index.ts**

```
export { Select, type SelectOption, type SelectProps } from "./Select";  
export { selectOverrides } from "./Select.config";  
export * from "./SelectItem";
```

### **frontend/src/metabase/ui/components/inputs/Switch/Switch.config.ts**

```
import { Switch, getSize, rem } from "@mantine/core";
```

```
import SwitchStyles from "./Switch.module.css";
```

```
const LABEL_FONT_SIZES: Record<string, string> = {  
  xs: rem(12),  
  sm: rem(14),  
  md:...
```

### **frontend/src/metabase/ui/components/inputs/Switch/index.ts**

```
export { Switch } from "@mantine/core";  
export type { SwitchProps, SwitchGroupProps } from "@mantine/core";  
export { switchOverrides } from "./Switch.config";
```

### **frontend/src/metabase/ui/components/inputs/TextInput/index.ts**

```
export { TextInput } from "@mantine/core";  
export type { TextInputProps } from "@mantine/core";  
export { textInputOverrides } from "./TextInput.config";
```

### **frontend/src/metabase/ui/components/inputs/Textarea/index.ts**

```
export { Textarea } from "@mantine/core";  
export type { TextareaProps } from "@mantine/core";  
export { textareaOverrides } from "./Textarea.config";
```

### **frontend/src/metabase/ui/components/inputs/TimeInput/index.ts**

```
export * from "./TimeInput";  
export { timeInputOverrides } from "./TimeInput.config";
```

## Analyst Base

### **frontend/src/metabase/ui/components/inputs/index.ts**

```
export * from "./Autocomplete";
export * from "./Calendar";
export * from "./Checkbox";
export * from "./Chip";
export * from "./Combobox";
export * from "./DateInput";
export * from...
```

### **frontend/src/metabase/ui/components/layout/Center/index.ts**

```
export { Center } from "@mantine/core";
export type { CenterProps } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/layout/Collapse/index.ts**

```
export { Collapse } from "@mantine/core";
export type { CollapseProps } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/layout/Flex/index.ts**

```
export { Flex } from "@mantine/core";
export type { FlexProps } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/layout/Grid/index.ts**

```
export { Grid } from "@mantine/core";
export type { GridProps } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/layout/Group/index.ts**

```
export { Group } from "@mantine/core";
export type { GroupProps } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/layout/ScrollArea/index.ts**

```
export { ScrollArea } from "@mantine/core";
export type { ScrollAreaProps } from "@mantine/core";
```

```
export { scrollAreaOverrides } from "./ScrollArea.config";
```

### **frontend/src/metabase/ui/components/layout/SimpleGrid/index.ts**

```
export { SimpleGrid } from "@mantine/core";
export type { SimpleGridProps } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/layout/Stack/index.ts**

```
export { Stack } from "@mantine/core";
export type { StackProps } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/layout/index.ts**

```
export * from "./Center";
```



## Analyst Base

```
export * from "./Collapse";
export * from "./Flex";
export * from "./Grid";
export * from "./Group";
export * from "./SimpleGrid";
export * from "./ScrollArea";
export *...
```

### **frontend/src/metabase/ui/components/navigation/NavLink/NavLink.config.ts**

```
import { NavLink } from "@mantine/core";
```

```
import S from "./NavLink.module.css";
```

```
export const navLinkOverrides = {
  NavLink: NavLink.extend({
    defaultProps: {
      // @ts-expect-error - this does...
```

### **frontend/src/metabase/ui/components/navigation/NavLink/index.ts**

```
export { NavLink, type NavLinkProps } from "@mantine/core";
```

```
export { navLinkOverrides } from "./NavLink.config";
```

### **frontend/src/metabase/ui/components/navigation/Tabs/index.ts**

```
export { Tabs } from "@mantine/core";
```

```
export type {
  TabsProps,
  TabsTabProps,
  TabsListProps,
  TabsPanelProps,
} from "@mantine/core";
export { tabsOverrides } from "./Tabs.config";
```

### **frontend/src/metabase/ui/components/navigation/index.ts**

```
export * from "./NavLink";
export * from "./Tabs";
```

### **frontend/src/metabase/ui/components/overlays/Drawer/index.ts**

```
export { Drawer } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/overlays/Menu/Menu.config.ts**

```
import { Menu } from "@mantine/core";
```

```
import MenuStyles from "./Menu.module.css";
```

## Analyst Base

```
export const menuOverrides = {  
  Menu: Menu.extend({  
    defaultProps: {  
      radius: "sm",  
      shadow: "md",  
    },  
  })  
}
```

### **frontend/src/metabase/ui/components/overlays/Menu/MenuDropdown/index.ts**

```
export * from "./MenuDropdown";
```

### **frontend/src/metabase/ui/components/overlays/Menu/MenuItem/index.ts**

```
export * from "./MenuItem";
```

### **frontend/src/metabase/ui/components/overlays/Menu/index.ts**

```
export type { MenuProps, MenuItemProps } from "@mantine/core";  
export { Menu } from "./Menu";  
export { menuOverrides } from "./Menu.config";
```

### **frontend/src/metabase/ui/components/overlays/Overlay/Overlay.config.ts**

```
import { type MantineThemeOverride, Overlay } from "@mantine/core";
```

```
import OverlayStyles from "./Overlay.module.css";
```

```
export const overlayOverrides: MantineThemeOverride["components"] = {  
  ...  
}
```

### **frontend/src/metabase/ui/components/overlays/Popover/Popover.config.ts**

```
import { Popover } from "@mantine/core";
```

```
import PopoverStyles from "./Popover.module.css";
```

```
export const DEFAULT_POPOVER_Z_INDEX = 300;
```

```
export const popoverOverrides = {  
  Popover: ...  
}
```

### **frontend/src/metabase/ui/components/overlays/Tooltip/index.ts**

```
export { Tooltip } from "@mantine/core";  
export type { TooltipProps } from "@mantine/core";  
export { tooltipOverrides } from "./Tooltip.config";
```

### **frontend/src/metabase/ui/components/overlays/index.ts**

```
export * from "./Drawer";  
export * from "./HoverCard";  
export * from "./Menu";
```

## Analyst Base

```
export * from "./Modal";
export * from "./Popover";
export * from "./Overlay";
export * from "./Tooltip";
```

### **frontend/src/metabase/ui/components/theme/DatesProvider/index.ts**

```
export * from "./DatesProvider";
```

### **frontend/src/metabase/ui/components/theme/ThemeProvider/context.ts**

```
import { createContext } from "react";
```

```
interface ThemeContext {
  withCssVariables?: boolean;
  withGlobalClasses?: boolean;
}
```

```
export const ThemeProviderContext = createContext<ThemeContext>({});
```

### **frontend/src/metabase/ui/components/theme/ThemeProvider/index.ts**

```
export * from "./ThemeProvider";
```

### **frontend/src/metabase/ui/components/theme/index.ts**

```
export * from "./ThemeProvider";
```

### **frontend/src/metabase/ui/components/typography/Anchor/index.ts**

```
export { Anchor } from "@mantine/core";
export type { AnchorProps } from "@mantine/core";
export { anchorOverrides } from "./Anchor.config";
```

### **frontend/src/metabase/ui/components/typography/Code/Code.config.ts**

```
import { Code } from "@mantine/core";
```

```
import CodeStyles from "./Code.module.css";
```

```
export const codeOverrides = {
  Code: Code.extend({
    classNames: {
      root: CodeStyles.root,
    },
  }),
};
```

### **frontend/src/metabase/ui/components/typography/Code/index.ts**

```
export { Code } from "@mantine/core";
export type { CodeProps } from "@mantine/core";
```

## Analyst Base

```
export { codeOverrides } from "./Code.config";
```

### **frontend/src/metabase/ui/components/typography/List/index.ts**

```
export { List } from "@mantine/core";  
export type { ListProps } from "@mantine/core";  
export { listOverrides } from "./List.config";
```

### **frontend/src/metabase/ui/components/typography/Text/index.ts**

```
export { Text } from "@mantine/core";  
export type { TextProps } from "@mantine/core";  
export { textOverrides } from "./Text.config";
```

### **frontend/src/metabase/ui/components/typography/Title/index.ts**

```
export { Title } from "@mantine/core";  
export type { TitleProps } from "@mantine/core";  
export { titleOverrides } from "./Title.config";
```

### **frontend/src/metabase/ui/components/typography/index.ts**

```
export * from "./Text";  
export * from "./Title";  
export * from "./Anchor";  
export * from "./List";  
export * from "./Code";
```

### **frontend/src/metabase/ui/components/utils/Box/index.ts**

```
export { Box } from "@mantine/core";  
export type { BoxProps } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/utils/Divider/index.ts**

```
export { Divider } from "@mantine/core";  
export type { DividerProps } from "@mantine/core";  
export { dividerOverrides } from "./Divider.config";
```

### **frontend/src/metabase/ui/components/utils/FocusTrap/index.ts**

```
export { FocusTrap } from "@mantine/core";  
export type { FocusTrapProps } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/utils/Paper/index.ts**

```
export { Paper } from "@mantine/core";  
export type { PaperProps } from "@mantine/core";  
export { paperOverrides } from "./Paper.config";
```

### **frontend/src/metabase/ui/components/utils/PreventPopoverExit/index.ts**

```
export * from "./PreventPopoverExit";
```

### **frontend/src/metabase/ui/components/utils/Space/index.ts**

## Analyst Base

```
export { Space } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/utils/Transition/index.ts**

```
export { Transition } from "@mantine/core";
```

```
export type { TransitionProps } from "@mantine/core";
```

### **frontend/src/metabase/ui/components/utils/index.ts**

```
export * from "./Box";
```

```
export * from "./DelayGroup";
```

```
export * from "./Divider";
```

```
export * from "./FocusTrap";
```

```
export * from "./PreventEagerPortal";
```

```
export * from "./Paper";
```

```
export * from...
```

### **frontend/src/metabase/ui/index.ts**

```
export { rem, useCombobox, useMantineTheme } from "@mantine/core";
```

```
export type {
```

```
  FloatingPosition,
```

```
  MantineSize,
```

```
  MantineStyleProps,
```

```
  MantineTheme,
```

```
  MantineThemeOther,
```

```
  MantineThemeOverride,
```

```
}...
```

### **frontend/src/metabase/ui/syntax/highlight.ts**

```
import { HighlightStyle } from "@codemirror/language";
```

```
import type { Tag } from "@lezer/highlight";
```

```
import { tags } from "@lezer/highlight";
```

```
import S from "./highlight.module.css";
```

```
const styledTags...
```

### **frontend/src/metabase/ui/syntax/index.ts**

```
export * from "./highlight";
```

### **frontend/src/metabase/ui/theme.ts**

```
import type { MantineThemeOverride } from "@mantine/core";
```

```
import { rem } from "@mantine/core";
```

```
import { DEFAULT_METABASE_COMPONENT_THEME } from "metabase/embedding-sdk/theme";
```

```
import Styles from...
```

## Analyst Base

### **frontend/src/metabase/ui/utils/colors.ts**

```
import type { MantineTheme } from "@mantine/core";

import { color as legacyColor } from "metabase/lib/colors";
type ColorShades = MantineTheme["colors"]["dark"];

const ORIGINAL_COLORS = [
  ...
```

### **frontend/src/metabase/visualizations/click-actions/Mode/Mode.ts**

```
import type { MetabasePluginsConfig } from "metabase/embedding-sdk/types/plugins";
import { queryDrill } from "metabase/querying/drills/utils/query-drill";
import type { DrillThruDisplayInfo } from...
```

### **frontend/src/metabase/visualizations/click-actions/Mode/index.ts**

```
export { Mode } from "./Mode";
```

### **frontend/src/metabase/visualizations/click-actions/actions/ColumnFormattingAction/index.ts**

```
export { ColumnFormattingAction } from "./ColumnFormattingAction";
```

### **frontend/src/metabase/visualizations/click-actions/actions/CombineColumnsAction/index.ts**

```
export { CombineColumnsAction } from "./CombineColumnsAction";
```

### **frontend/src/metabase/visualizations/click-actions/actions/ExtractColumnAction/index.ts**

```
export { ExtractColumnAction } from "./ExtractColumnAction";
```

### **frontend/src/metabase/visualizations/click-actions/actions/HideColumnAction/index.ts**

```
export { HideColumnAction } from "./HideColumnAction";
```

### **frontend/src/metabase/visualizations/click-actions/lib/modes.ts**

```
import type { MetabasePluginsConfig } from "metabase/embedding-sdk/types/plugins";
import type { QueryClickActionsMode } from "metabase/visualizations/types";
import type Question from...
```

### **frontend/src/metabase/visualizations/click-actions/modes/ArchivedMode.ts**

```
import type { QueryClickActionsMode } from "../../types";

export const ArchivedMode: QueryClickActionsMode = {
  name: "archived",
  hasDrills: false,
```

## Analyst Base

```
clickActions: [],  
};
```

### **frontend/src/metabase/visualizations/click-actions/modes/DefaultMode.ts**

```
import type { QueryClickActionsMode } from "../../types";  
import { ColumnFormattingAction } from "../../actions/ColumnFormattingAction";  
import { CombineColumnsAction } from...
```

### **frontend/src/metabase/visualizations/click-actions/modes/EmbeddingSdkMode.ts**

```
import type { QueryClickActionsMode } from "../../types";  
import { CombineColumnsAction } from "../../actions/CombineColumnsAction";  
import { DashboardClickAction } from...
```

### **frontend/src/metabase/visualizations/click-actions/modes/PublicMode.ts**

```
import type { QueryClickActionsMode } from "../../types";  
import { DashboardClickAction } from "../../actions/DashboardClickAction";
```

```
export const PublicMode: QueryClickActionsMode = {  
  name:...
```

### **frontend/src/metabase/visualizations/components/ChartRenderingErrorBoundary/index.ts**

```
export * from "./ChartRenderingErrorBoundary";
```

### **frontend/src/metabase/visualizations/components/ChartSettings/BaseChartSettings/hooks.ts**

```
import { useMemo, useState } from "react";  
import { t } from "ttag";  
import _ from "underscore";
```

```
import type { Widget } from "../../types";
```

```
import type { BaseChartSettingsProps } from "../../types";
```

```
//...
```

### **frontend/src/metabase/visualizations/components/ChartSettings/BaseChartSettings/index.ts**

```
export * from "./BaseChartSettings";
```

### **frontend/src/metabase/visualizations/components/ChartSettings/BaseChartSettings/types.ts**

```
import type { StackProps } from "metabase/ui";
```

## Analyst Base

```
import type { ComputedVisualizationSettings } from "metabase/visualizations/types";
import type Question from "metabase-lib/v1/Question";
import type {...
```

### **frontend/src/metabase/visualizations/components/ChartSettings/ChartSettingsFooter/index.ts**

```
export * from "./ChartSettingsFooter";
```

### **frontend/src/metabase/visualizations/components/ChartSettings/ChartSettingsVisualization/index.ts**

```
export * from "./ChartSettingsVisualization";
```

### **frontend/src/metabase/visualizations/components/ChartSettings/ChartSettingsVisualization/types.ts**

```
import type { StackProps } from "metabase/ui";
import type Question from "metabase-lib/v1/Question";
import type {
  Dashboard,
  DashboardCard,
  RawSeries,
  VisualizationSettings,
} from...
```

### **frontend/src/metabase/visualizations/components/ChartSettings/DashboardChartSettings/index.ts**

```
export * from "./DashboardChartSettings";
```

### **frontend/src/metabase/visualizations/components/ChartSettings/DashboardChartSettings/types.ts**

```
import type { Dashboard, DashboardCard } from "metabase-types/api";
```

```
import type { CommonChartSettingsProps, Widget } from "../types";
```

```
export type DashboardChartSettingsProps = {
  className?:...
```

### **frontend/src/metabase/visualizations/components/ChartSettings/QuestionChartSettings/index.ts**

```
export * from "./QuestionChartSettings";
```

### **frontend/src/metabase/visualizations/components/ChartSettings/QuestionChartSettings/types.ts**

```
import type { BaseChartSettingsProps } from "../BaseChartSettings/types";
import type { CommonChartSettingsProps, Widget } from "../types";
```



## Analyst Base

```
export type QuestionChartSettingsProps = {  
  widgets?:...
```

### **frontend/src/metabase/visualizations/components/ChartSettings/hooks.ts**

```
import { assocIn } from "icepick";  
import { useCallback, useMemo } from "react";
```

```
import {  
  extractRemappings,  
  getVisualizationTransformed,  
} from "metabase/visualizations";  
import {...
```

### **frontend/src/metabase/visualizations/components/ChartSettings/index.ts**

```
export { QuestionChartSettings } from "./QuestionChartSettings";  
export { DashboardChartSettings } from "./DashboardChartSettings";  
export { BaseChartSettings } from "./BaseChartSettings";  
export *...
```

### **frontend/src/metabase/visualizations/components/ChartSettings/types.ts**

```
import type Question from "metabase-lib/v1/Question";  
import type { Series, VisualizationSettings } from "metabase-types/api";
```

```
// this type is not full, we need to extend it later  
export type Widget...
```

### **frontend/src/metabase/visualizations/components/ChartTooltip/EChartsTooltip/index.ts**

```
export * from "./EChartsTooltip";
```

### **frontend/src/metabase/visualizations/components/ChartTooltip/KeyValuePairChartTooltip/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/visualizations/components/ChartTooltip/StackedDataTooltip/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/visualizations/components/ChartTooltip/StackedDataTooltip/Utils.ts**

```
import { t } from "ttag";
```

```
import type { TooltipRowModel } from "metabase/visualizations/types";
```

## Analyst Base

```
export const getTotalValue = (  
  headerRows: TooltipRowModel[] = [],  
  bodyRows: TooltipRowModel[] =...
```

### **frontend/src/metabase/visualizations/components/ChartTooltip/TooltipRow/index.ts**

```
export * from "../TooltipRow";
```

### **frontend/src/metabase/visualizations/components/ChartTooltip/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/visualizations/components/ChartTooltip/utils.ts**

```
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";  
import { formatValue } from "metabase/lib/formatting";  
import type { OptionsType } from "metabase/lib/formatting/types";  
import type {  
  ...
```

### **frontend/src/metabase/visualizations/components/ChoroplethMap.unit.spec.js**

```
import { mockIsEmbeddingSdk } from "metabase/embedding-sdk/mocks/config-mock";  
import {  
  getLegendTitles,  
  getMapUrl,  
} from...
```

### **frontend/src/metabase/visualizations/components/ClickActions/index.ts**

```
export { ClickActionsView } from "../ClickActionsView";  
export { ConnectedClickActionsPopover } from "../ClickActionsPopover";
```

### **frontend/src/metabase/visualizations/components/ClickActions/utils.ts**

```
import { t } from "ttag";  
import _ from "underscore";  
  
import type {  
  ClickActionSection,  
  RegularClickAction,  
} from "metabase/visualizations/types";
```

```
type Section = {  
  index?: number;  
};
```

```
export...
```

## Analyst Base

### **frontend/src/metabase/visualizations/components/EChartsRenderer/index.ts**

```
export * from "./EChartsRenderer";
export type { ResponsiveEChartsRendererProps } from "./ResponsiveEChartsRenderer";
export { ResponsiveEChartsRenderer } from "./ResponsiveEChartsRenderer.styled";
```

### **frontend/src/metabase/visualizations/components/EmptyVizState/index.ts**

```
export { EmptyVizState } from "./EmptyVizState";
```

### **frontend/src/metabase/visualizations/components/EmptyVizState/utils.ts**

```
import { t } from "ttag";

import { getSubpathSafeUrl } from "metabase/lib/urls";
import type { CardDisplayType } from "metabase-types/api";
```

```
/**
 * The "table" and the "object" (detail) charts can...
```

### **frontend/src/metabase/visualizations/components/LegendVertical.unit.spec.js**

```
import { render, screen } from "__support__/ui";
import LegendVertical from "metabase/visualizations/components/LegendVertical";
```

```
describe("LegendVertical", () => {
  it("should render string titles...
```

### **frontend/src/metabase/visualizations/components/ObjectDetail/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/visualizations/components/ObjectDetail/types.ts**

```
import type Question from "metabase-lib/v1/Question";
import type ForeignKey from "metabase-lib/v1/metadata/ForeignKey";
import type Metadata from "metabase-lib/v1/metadata/Metadata";
import type...
```

### **frontend/src/metabase/visualizations/components/ObjectDetail/utils.ts**

```
import { t } from "ttag";

import {
  isImplicitDeleteAction,
  isImplicitUpdateAction,
} from "metabase/actions/utils";
import { formatValue, singularize } from "metabase/lib/formatting";
import...
```

### **frontend/src/metabase/visualizations/components/ObjectDetail/utils.unit.spec.t**

**s**

## Analyst Base

```
import { createMockMetadata } from "__support__/metadata";
import Question from "metabase-lib/v1/Question";
import {
  createMockColumn,
  createMockDatasetData,
  createMockImplicitQueryAction,
  ...
```

### **frontend/src/metabase/visualizations/components/PinMap.unit.spec.js**

```
import PinMap from "metabase/visualizations/components/PinMap";

describe("PinMap", () => {
  it("should filter out rows with null values in either the lat, long, or metric column", () => {
    const...
```

### **frontend/src/metabase/visualizations/components/ScalarValue/utils.ts**

```
import { DEFAULT_CARD_SIZE, GRID_WIDTH } from "metabase/lib/dashboard_grid";
import { measureText } from "metabase/lib/measure-text";

interface FindSizeInput {
  text: string;
  targetHeight:...
```

### **frontend/src/metabase/visualizations/components/ScalarValue/utils.unit.spec.ts**

```
import * as measureText from "metabase/lib/measure-text";
import type { FontStyle } from "metabase/visualizations/shared/types/measure-text";

import { findSize } from...
```

### **frontend/src/metabase/visualizations/components/TableInteractive/hooks/use-object-detail.ts**

```
import { useCallback, useMemo } from "react";

import { useDispatch, useSelector } from "metabase/lib/redux";
import { zoomInRow } from "metabase/query_builder/actions";
import { getRowIndexToPKMap }...
```

### **frontend/src/metabase/visualizations/components/TableInteractive/hooks/use-reset-widths-on-columns-change.ts**

```
import { useEffect, useRef } from "react";
import _ from "underscore";

import type Question from "metabase-lib/v1/Question";
import { isAdHocModelOrMetricQuestion } from...
```

## Analyst Base

**frontend/src/metabase/visualizations/components/TableInteractive/index.ts**

```
export * from "./TableInteractive";
```

**frontend/src/metabase/visualizations/components/TransformedVisualization/index.ts**

```
export * from "./TransformedVisualization";
```

**frontend/src/metabase/visualizations/components/Visualization/ChartSettingsErrorButton/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/visualizations/components/Visualization/ErrorView/index.ts**

```
export { ErrorView } from "./ErrorView";
```

**frontend/src/metabase/visualizations/components/Visualization/LoadingView/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/visualizations/components/Visualization/NoResultsView/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/visualizations/components/Visualization/Visualization.unit.spec.js**

```
import PropTypes from "prop-types";
```

```
import { mockSettings } from "__support__/settings";
```

```
import { renderWithProviders, screen } from "__support__/ui";
```

```
import { delay } from...
```

**frontend/src/metabase/visualizations/components/Visualization/Watermark/index.ts**

```
export * from "./Watermark";
```

**frontend/src/metabase/visualizations/components/Visualization/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

**frontend/src/metabase/visualizations/components/legend/LegendCaption/index.ts**

```
export { LegendCaption } from "./LegendCaption";
```

```
export {
```

```
  LEGEND_LABEL_FONT_SIZE,
```

```
  LEGEND_LABEL_FONT_WEIGHT,
```

## Analyst Base

```
} from "../LegendCaption.styled";
```

### **frontend/src/metabase/visualizations/components/settings/ChartNestedSettingSeriesMultiple.unit.spec.js**

these tests use ChartSettings directly, but logic we're testing lives in ChartNestedSettingSeries

### **frontend/src/metabase/visualizations/components/settings/ChartSettingColorPicker/index.ts**

```
export * from "../ChartSettingColorPicker";
```

### **frontend/src/metabase/visualizations/components/settings/ChartSettingFieldPicker.unit.spec.js**

these tests use QuestionChartSettings directly, but logic we're testing logic in ChartSettingFieldPicker

### **frontend/src/metabase/visualizations/components/settings/ChartSettingFieldsPartition.unit.spec.js**

```
import { render, screen } from "__support__/ui";  
import { createMockColumn } from "metabase-types/api/mocks";
```

```
import { ChartSettingFieldsPartition } from...
```

### **frontend/src/metabase/visualizations/components/settings/ChartSettingFieldsPicker.unit.spec.js**

```
import userEvent from "@testing-library/user-event";
```

```
import { renderWithProviders, screen } from "__support__/ui";  
import ChartSettingFieldsPicker from...
```

### **frontend/src/metabase/visualizations/components/settings/ChartSettingOrderedItems/index.ts**

```
export * from "../ChartSettingOrderedItems";
```

### **frontend/src/metabase/visualizations/components/settings/ChartSettingSegmentedControl/index.ts**

```
export * from "../ChartSettingSegmentedControl";
```

### **frontend/src/metabase/visualizations/components/settings/ChartSettingStacked.unit.spec.js**

these tests use QuestionChartSettings directly, but logic we're testing lives in ChartNestedSettingSeries

### **frontend/src/metabase/visualizations/components/settings/ChartSettingTableColumns/TableColumnPanel/index.ts**

```
export * from "../TableColumnPanel";
```

### **frontend/src/metabase/visualizations/components/settings/ChartSettingTableC**

## **olumns/TableColumnPanel/types.ts**

```
import type { IconName } from "metabase/ui";
import type {
  DatasetColumn,
  TableColumnOrderSetting,
} from "metabase-types/api";
```

```
export type ColumnItem = {
  name: string;
  enabled: boolean;
  ...
```

## **frontend/src/metabase/visualizations/components/settings/ChartSettingTableColumns/TableColumnPanel/utils.ts**

```
import _ from "underscore";

import type { ContentTranslationFunction } from "metabase/i18n/types";
import type { IconName } from "metabase/ui";
import { getIconForField } from...
```

## **frontend/src/metabase/visualizations/components/settings/ChartSettingTableColumns/index.ts**

```
export * from "../ChartSettingTableColumns";
```

## **frontend/src/metabase/visualizations/components/settings/ChartSettingTableColumns/types.ts**

```
export type EditWidgetData = {
  id: string;
  props: EditWidgetProps;
};
```

```
export type EditWidgetProps = {
  initialKey: string;
};
```

## **frontend/src/metabase/visualizations/components/settings/ChartSettingTableColumns/utils.ts**

```
import * as Lib from "metabase-lib";

export function canEditQuery(query?: Lib.Query) {
  if (!query) {
    return false;
  }
}
```

## Analyst Base

```
const { isNative, isEditable } = Lib.queryDisplayInfo(query);  
return...
```

### **frontend/src/metabase/visualizations/components/settings/ChartSettingsTableFormatting/constants.ts**

```
import {  
  getAccentColors,  
  getStatusColorRanges,  
} from "metabase/lib/colors/groups";  
import type {  
  ColumnRangeFormattingSetting,  
  ColumnSingleFormattingSetting,  
} from...
```

### **frontend/src/metabase/visualizations/components/settings/ChartSettingsTableFormatting/get-operators-for-columns.ts**

```
import { t } from "ttag";  
  
import {  
  isBoolean,  
  isFK,  
  isNumeric,  
  isPK,  
  isString,  
} from "metabase-lib/v1/types/utils/isa";  
import type {  
  ColumnFormattingOperator,  
  DatasetColumn,  
} from...
```

### **frontend/src/metabase/visualizations/components/settings/ChartSettingsTableFormatting/get-operators-for-columns.unit.spec.ts**

```
import { createMockColumn } from "metabase-types/api/mocks";  
  
import { getOperatorsForColumns } from "../get-operators-for-columns";  
  
const STRING_COLUMN = createMockColumn({  
  base_type:...
```

### **frontend/src/metabase/visualizations/components/settings/ChartSettingsTableFormatting/index.ts**

```
export { ChartSettingsTableFormatting } from "../ChartSettingsTableFormatting";  
export { isFormattable } from "../util";
```



## Analyst Base

```
export { ALL_OPERATOR_NAMES } from "../get-operators-for-columns";
```

### **frontend/src/metabase/visualizations/components/settings/ChartSettingsTableFormatting/util.ts**

```
import {
  isBoolean,
  isNumeric,
  isString,
} from "metabase-lib/v1/types/utils/isa";
import type { DatasetColumn } from "metabase-types/api";
```

```
// predicate for columns that can be...
```

### **frontend/src/metabase/visualizations/components/settings/ColumnItem/index.ts**

```
export * from "../ColumnItem";
```

### **frontend/src/metabase/visualizations/components/settings/types.ts**

```
import type { VisualizationSettings } from "metabase-types/api";
```

```
export interface ChartSettingWidgetProps<TValue> {
  value: TValue | undefined;
  onChange: (value?: TValue | null) => void;
  ...
```

### **frontend/src/metabase/visualizations/components/skeletons/AreaSkeleton/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/visualizations/components/skeletons/BarSkeleton/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/visualizations/components/skeletons/ChartSkeleton/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/visualizations/components/skeletons/FunnelSkeleton/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/visualizations/components/skeletons/GaugeSkeleton/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/visualizations/components/skeletons/LineSkeleton/index.ts**

## Analyst Base

**x.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/visualizations/components/skeletons/MapSkeleton/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/visualizations/components/skeletons/PieSkeleton/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/visualizations/components/skeletons/ProgressSkeleton/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/visualizations/components/skeletons/RowSkeleton/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/visualizations/components/skeletons/SankeySkeleton/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/visualizations/components/skeletons/ScatterSkeleton/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/visualizations/components/skeletons/SkeletonCaption/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/visualizations/components/skeletons/SkeletonCaption/types.ts**

export type SkeletonCaptionSize = "medium" | "large";

**frontend/src/metabase/visualizations/components/skeletons/StaticSkeleton/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/visualizations/components/skeletons/TableSkeleton/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

**frontend/src/metabase/visualizations/components/skeletons/WaterfallSkeleton/i**

### **ndex.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/visualizations/echarts/cartesian/chart-measurements/index.ts**

```
import _ from "underscore";
```

```
import { isNotNull } from "metabase/lib/types";
```

```
import { X_AXIS_DATA_KEY } from "metabase/visualizations/echarts/cartesian/constants/dataset";
```

```
import { CHART_STYLE } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/chart-measurements/types.ts**

```
import type {  
  ComputedVisualizationSettings,  
  Padding,  
} from "metabase/visualizations/types";
```

```
export interface TicksDimensions {
```

```
  yTicksWidthLeft: number;
```

```
  yTicksWidthRight: number;
```

```
  ...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/constants/dataset.ts**

We prefix misc data keys we add to avoid possible collisions with series data keys

### **frontend/src/metabase/visualizations/echarts/cartesian/constants/style.ts**

```
import type { LineSize } from "metabase-types/api";
```

```
export const LINE_SIZE: Record<LineSize, number> = {
```

```
  S: 1,
```

```
  M: 2,
```

```
  L: 3,
```

```
};
```

```
export const Z_INDEXES = {
```

```
  // Note: timeline events use...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/axis.ts**

```
import * as d3 from "d3";
```

```
import dayjs from "dayjs";
```

```
import _ from "underscore";
```

## Analyst Base

```
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";
import { formatValue } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/axis.unit.spec.ts**

```
import { computeSplit } from "./axis";
import type { SeriesExtents } from "./types";
```

```
describe("computeSplit", () => {
  const extents: SeriesExtents = {
    "1": [6, 8],
    "2": [9, 13],
    "3":...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/dataset.ts**

```
import { t } from "ttag";
```

```
import { getObjectKeys } from "metabase/lib/objects";
import { parseTimestamp } from "metabase/lib/time-dayjs";
import { checkNumber, isNotNull } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/dataset.unit.spec.ts**

```
import dayjs from "dayjs";
```

```
import { createMockSeriesModel } from "__support__/echarts";
import { checkNumber } from "metabase/lib/types";
import {
  ECHARTS_CATEGORY_AXIS_NULL_VALUE,
  INDEX_KEY,
  ...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/guards.ts**

```
import type {
  BaseSeriesModel,
  BreakoutSeriesModel,
  CategoryXAxisModel,
  NumericXAxisModel,
  TimeSeriesInterval,
  TimeSeriesXAxisModel,
  XAxisModel,
} from "./types";
```

```
export const...
```

## Analyst Base

### **frontend/src/metabase/visualizations/echarts/cartesian/model/index.ts**

```
import { OTHER_DATA_KEY } from "metabase/visualizations/echarts/cartesian/constants/dataset";
import {
  getXAxisModel,
  getYAxesModels,
} from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/legend.ts**

```
import { isBreakoutSeries } from "../guards";
import type { BaseSeriesModel, LegendItem } from "../types";
```

```
export const getLegendItems = (
  seriesModels: BaseSeriesModel[],
  showAllLegendItems:...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/legend.unit.spec.ts**

```
import {
  createMockBreakoutSeriesModel,
  createMockSeriesModel,
} from "__support__/echarts";

import { getLegendItems } from "../legend";

describe("getLegendItems", () => {
  it("should return an...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/other-series.ts**

```
import { t } from "ttag";

import { checkNumber } from "metabase/lib/types";
import { isEmpty } from "metabase/lib/validate";
import { SERIES_SETTING_KEY } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/series.ts**

```
import { memoize } from "metabase/common/hooks/use-memoized-callback";
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";
import { formatValue } from "metabase/lib/formatting";
import type...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/series.unit.spec.ts**

```
import type {
```

## Analyst Base

```
BreakoutChartColumns,  
CartesianChartColumns,  
} from "metabase/visualizations/lib/graph/columns";  
import { SERIES_COLORS_SETTING_KEY } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/stack.ts**

```
import _ from "underscore";  
  
import { getObjectKeys } from "metabase/lib/objects";  
import type { ComputedVisualizationSettings } from "metabase/visualizations/types";  
  
import type { SeriesModel,...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/transforms.ts**

```
import { isNumber } from "metabase/lib/types";  
import type { NumericScale } from "metabase-types/api";  
  
import type { NumericAxisScaleTransforms } from "../types";
```

```
function getSign(value: number) {  
  ...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/trend-line.ts**

```
import Color from "color";  
  
import { checkNumber, isNotNull } from "metabase/lib/types";  
import { X_AXIS_DATA_KEY } from "metabase/visualizations/echarts/cartesian/constants/dataset";  
import {...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/types.ts**

```
import type { Dayjs } from "dayjs";  
import type { OptionAxisType } from "echarts/types/src/coord/axisCommonTypes";  
  
import type {  
  INDEX_KEY,  
  NEGATIVE_STACK_TOTAL_DATA_KEY,  
  ...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/model/util.ts**

```
import type { OptionsType } from "metabase/lib/formatting/types";  
import type {  
  ComputedVisualizationSettings,  
  RemappingHydratedDatasetColumn,  
} from "metabase/visualizations/types";
```

## Analyst Base

```
import {...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/option/axis.ts**

```
import type { XAxisOption, YAxisOption } from "echarts/types/dist/shared";
```

```
import type { AxisBaseOptionCommon } from "echarts/types/src/coord/axisCommonTypes";
```

```
import { parseNumberValue } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/option/goal-line.ts**

```
import type { CustomSeriesOption } from "echarts/charts";
```

```
import type {
```

```
  ComputedVisualizationSettings,
```

```
  RenderingContext,
```

```
} from "metabase/visualizations/types";
```

```
import type {...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/option/index.ts**

```
import type { EChartsCoreOption } from "echarts/core";
```

```
import type { OptionSourceData } from "echarts/types/src/util/types";
```

```
import {
```

```
  NEGATIVE_STACK_TOTAL_DATA_KEY,
```

```
  OTHER_DATA_KEY,
```

```
...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/option/index.unit.spec.ts**

```
import type { XAxisOption, YAxisOption } from "echarts/types/dist/shared";
```

```
import { DEFAULT_VISUALIZATION_THEME } from "metabase/visualizations/shared/utils/theme";
```

```
import type { RenderingContext }...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/option/series.ts**

```
import type { BarSeriesOption, LineSeriesOption } from "echarts/charts";
```

```
import type { CallbackDataParams } from "echarts/types/dist/shared";
```

```
import type {
```

```
  LabelLayoutOptionCallbackParams,
```

```
...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/option/ticks.ts**

```
import type { Dayjs } from "dayjs";
```

```
import dayjs from "dayjs";
```

## Analyst Base

```
import type { ContinuousDomain } from "metabase/visualizations/shared/types/scale";
```

```
import type {  
  TimeSeriesAxisFormatter,  
  ...  
}
```

### **frontend/src/metabase/visualizations/echarts/cartesian/option/trend-line.ts**

```
import type { LineSeriesOption } from "echarts/charts";
```

```
import { X_AXIS_DATA_KEY } from "metabase/visualizations/echarts/cartesian/constants/dataset";
```

```
import { Z_INDEXES } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/option/types.ts**

```
import type {  
  BarSeriesOption,  
  CustomSeriesOption,  
  LineSeriesOption,  
  ScatterSeriesOption,  
} from "echarts/charts";
```

```
export type EChartsSeriesOption =  
  | LineSeriesOption  
  | BarSeriesOption  
  ...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/option/utils.ts**

```
import type { BaseCartesianChartModel, DataKey } from "../model/types";
```

```
export function getSeriesYAxisIndex(  
  dataKey: DataKey,  
  chartModel: BaseCartesianChartModel,  
): number {  
  const {...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/scatter/model/dataset.ts**

```
import {  
  INDEX_KEY,  
  X_AXIS_DATA_KEY,  
} from "metabase/visualizations/echarts/cartesian/constants/dataset";  
import type { CartesianChartColumns } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/scatter/model/index.ts**



## Analyst Base

```
import { getObjectValues } from "metabase/lib/objects";
import { isNotNull } from "metabase/lib/types";
import type { ShowWarning } from "metabase/visualizations/echarts/types";
import type {
  ...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/scatter/option/index.ts**

```
import type { EChartsCoreOption } from "echarts/core";
import type { OptionSourceData } from "echarts/types/src/util/types";

import type {
  ComputedVisualizationSettings,
  RenderingContext,
} from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/scatter/option/series.ts**

```
import * as d3 from "d3";
import type { ScatterSeriesOption } from "echarts/charts";

import { X_AXIS_DATA_KEY } from "metabase/visualizations/echarts/cartesian/constants/dataset";
import type { ...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/timeline-events/model.ts**

```
import type { OpUnitType } from "dayjs";
import dayjs from "dayjs";
import _ from "underscore";
```

```
import { CHART_STYLE } from "metabase/visualizations/echarts/cartesian/constants/style";
import type...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/timeline-events/model.unit.spec.ts**

```
import type { TimeSeriesInterval } from "metabase/visualizations/echarts/cartesian/model/types";
import type { TimelineEventGroup } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/timeline-events/option.ts**

```
import type { LineSeriesOption } from "echarts/charts";
import type { MarkLine1DDatItemOption } from "echarts/types/src/component/marker/MarkLineModel";

import type { IconName } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/timeline-events/types.ts**

```
import type { TimelineEvent } from "metabase-types/api";
```

## Analyst Base

```
export type TimelineEventGroup = {  
  date: string;  
  events: TimelineEvent[];  
};
```

```
export type TimelineEventsModel = TimelineEventGroup[];
```

### **frontend/src/metabase/visualizations/echarts/cartesian/utils/timeseries.ts**

```
import type { Dayjs } from "dayjs";  
import dayjs from "dayjs";  
import _ from "underscore";
```

```
import { parseTimestamp } from "metabase/lib/time-dayjs";  
import { isNotNull } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/utils/timeseries.tz.unit.spec.js**

```
import dayjs from "dayjs";
```

```
import testAcrossTimezones from "__support__/timezones";  
import { computeTimeseriesDataInterval } from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/utils/timeseries.unit.spec.js**

```
import dayjs from "dayjs";  
import timezone from "dayjs/plugin/timezone";  
import utc from "dayjs/plugin/utc";
```

```
// Enable timezone and UTC plugins  
dayjs.extend(utc);  
dayjs.extend(timezone);
```

```
import {...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/waterfall/constants.ts**

```
import { NULL_CHAR } from "../constants/dataset";
```

```
// Start of a waterfall bar  
export const WATERFALL_START_KEY = "start";  
// End of a waterfall bar  
export const WATERFALL_END_KEY = "end";  
// Value...
```

## Analyst Base

### **frontend/src/metabase/visualizations/echarts/cartesian/waterfall/model/axis.ts**

```
import { t } from "ttag";
```

```
import type {  
  ChartDataset,  
  DateRange,  
  DimensionModel,  
  Extent,  
  TimeSeriesXAxisModel,  
  WaterfallXAxisModel,  
} from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/waterfall/model/dataset.ts**

```
import { t } from "ttag";
```

```
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";  
import {  
  INDEX_KEY,  
  IS_WATERFALL_TOTAL_DATA_KEY,  
  X_AXIS_DATA_KEY,  
} from...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/waterfall/model/index.ts**

```
import { getYAxisModel } from "metabase/visualizations/echarts/cartesian/model/axis";  
import {  
  filterNullDimensionValues,  
  getCardsColumnByDataKeyMap,  
  getJoinedCardsDataset,  
  scaleDataset,  
  ...
```

### **frontend/src/metabase/visualizations/echarts/cartesian/waterfall/option/index.ts**

```
import type { EChartsCoreOption } from "echarts/core";  
import type { LabelLayoutOptionCallback } from "echarts/types/src/util/types";  
  
import { X_AXIS_DATA_KEY } from...
```

### **frontend/src/metabase/visualizations/echarts/graph/sankey/constants/style.ts**

```
export const SANKEY_CHART_STYLE = {  
  nodeWidth: 20,  
  nodeLabels: {  
    weight: 400,
```

## Analyst Base

```
size: 13,  
textBorderWidth: 3,  
},  
edgeLabels: {  
  weight: 400,  
  size: 12,  
  textBorderWidth:...
```

### **frontend/src/metabase/visualizations/echarts/graph/sankey/layout/index.ts**

```
import type {  
  ComputedVisualizationSettings,  
  RenderingContext,  
} from "metabase/visualizations/types";
```

```
import { SANKEY_CHART_STYLE } from "../../constants/style";  
import type { SankeyChartModel,...
```

### **frontend/src/metabase/visualizations/echarts/graph/sankey/layout/types.ts**

```
import type { Padding } from "metabase/visualizations/types";
```

```
export type SankeyChartLayout = {  
  padding: Padding;  
  nodeIndicesWithTruncatedLabels: Set<number> | null;  
  truncateLabelWidth:...
```

### **frontend/src/metabase/visualizations/echarts/graph/sankey/model/dataset.ts**

```
import { sumMetric } from "metabase/visualizations/lib/dataset";  
import { getColumnDescriptors } from "metabase/visualizations/lib/graph/columns";  
import type { ComputedVisualizationSettings } from...
```

### **frontend/src/metabase/visualizations/echarts/graph/sankey/model/dataset.unit.spec.ts**

```
import { getColumnKey } from "metabase-lib/v1/queries/utils/column-key";  
import type { DatasetColumn } from "metabase-types/api";  
import { createMockCard } from...
```

### **frontend/src/metabase/visualizations/echarts/graph/sankey/model/formatters.ts**

```
import { memoize } from "metabase/common/hooks/use-memoized-callback";  
import { formatValue } from "metabase/lib/formatting";  
import type { ComputedVisualizationSettings } from...
```

### **frontend/src/metabase/visualizations/echarts/graph/sankey/model/index.ts**

```
import { t } from "ttag";
```

## Analyst Base

```
import { getColorsForValues } from "metabase/lib/colors/charts";
import type { ComputedVisualizationSettings } from "metabase/visualizations/types";
import type { RawSeries...
```

### **frontend/src/metabase/visualizations/echarts/graph/sankey/model/types.ts**

```
import type { ColumnDescriptor } from "metabase/visualizations/lib/graph/columns";
import type { RowValue } from "metabase-types/api";
```

```
export type ColumnKey = string;
```

```
export interface...
```

### **frontend/src/metabase/visualizations/echarts/graph/sankey/option/index.ts**

```
import type { SankeySeriesOption } from "echarts/charts";
import type { EChartsCoreOption } from "echarts/core";
```

```
import { truncateText } from "metabase/visualizations/lib/text";
import type {
  ...
```

### **frontend/src/metabase/visualizations/echarts/index.ts**

```
import {
  BarChart,
  CustomChart,
  LineChart,
  SankeyChart,
  ScatterChart,
  SunburstChart,
} from "echarts/charts";
import {
  BrushComponent,
  DataZoomComponent,
  DatasetComponent,
  ...
```

### **frontend/src/metabase/visualizations/echarts/pie/constants.ts**

```
import { t } from "ttag";

import { NULL_CHAR } from "../cartesian/constants/dataset";

export const DIMENSIONS = {
  maxSideLength: 550,
  padding: {
    legend: 16,
```

## Analyst Base

```
side: 12,  
},  
slice: {  
...  
}
```

### **frontend/src/metabase/visualizations/echarts/pie/format.ts**

```
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";  
import { formatValue } from "metabase/lib/formatting";  
import { computeMaxDecimalsForValues } from...
```

### **frontend/src/metabase/visualizations/echarts/pie/model/index.ts**

```
import { pie } from "d3";  
import _ from "underscore";  
  
import { findWithIndex } from "metabase/lib/arrays";  
import { checkNotNull } from "metabase/lib/types";  
import type { ColumnDescriptor } from...
```

### **frontend/src/metabase/visualizations/echarts/pie/model/index.unit.spec.ts**

```
import type { RenderingContext } from "metabase/visualizations/types";  
import type { VisualizationSettings } from "metabase-types/api";  
import {  
  createMockCard,  
  createMockColumn,  
  ...  
}
```

### **frontend/src/metabase/visualizations/echarts/pie/model/types.ts**

```
import type { ColumnDescriptor } from "metabase/visualizations/lib/graph/columns";  
import type { RemappingHydratedDatasetColumn } from "metabase/visualizations/types";  
  
export interface PieRow {  
  ...  
}
```

### **frontend/src/metabase/visualizations/echarts/pie/option.ts**

```
import Color from "color";  
import type { EChartsOption, SunburstSeriesOption } from "echarts";  
  
import { getTextColorForBackground } from "metabase/lib/colors";  
import { checkNotNull } from...
```

### **frontend/src/metabase/visualizations/echarts/pie/types.ts**

```
import type { EChartsSeriesMouseEvent } from "../types";  
  
type TreePathInfo = {  
  name: string;  
}
```

## Analyst Base

```
dataIndex: number;  
value: number;  
};
```

```
export type EChartsSunburstSeriesMouseEvent =...
```

### **frontend/src/metabase/visualizations/echarts/pie/util/colors.ts**

```
import { match } from "ts-pattern";
```

```
import { aliases, colors } from "metabase/lib/colors";  
import { isEmpty } from "metabase/lib/validate";
```

```
const ACCENT_KEY_PREFIX = "accent";
```

```
type AccentKey =...
```

### **frontend/src/metabase/visualizations/echarts/pie/util/colors.unit.spec.ts**

```
import { getPickerColorAlias, getRingColorAlias } from "../colors";
```

```
describe("getRingColorAlias", () => {  
  it("should return the correct color alias for inner ring", () => {  
    ...
```

### **frontend/src/metabase/visualizations/echarts/pie/util/index.ts**

```
import { checkNotNull } from "metabase/lib/types";
```

```
import { OPTION_NAME_SEPERATOR } from "../constants";  
import type { SliceTree, SliceTreeNode } from "../model/types";  
import type {...
```

### **frontend/src/metabase/visualizations/echarts/pie/util/label.ts**

```
export type Point = [number, number];
```

```
/**  
 * Calculates the length of a chord given the radius of a circle and the central angle.  
 * @param radius - The radius of the circle.  
 * @param angle - The...
```

### **frontend/src/metabase/visualizations/echarts/pie/util/label.unit.spec.ts**

```
import {  
  calcAvailableDonutSliceLabelLength,  
  calcChordLength,  
  calcCircleIntersectionByHorizontalLine,  
  calcInnerOuterRadiusesForRing,
```

## Analyst Base

```
  getCoordOnCircle,  
} from...
```

### **frontend/src/metabase/visualizations/echarts/tooltip/index.unit.spec.ts**

```
import { renderHook } from "@testing-library/react";  
import type { EChartsType } from "echarts/core";  
import type { MutableRefObject } from "react";
```

```
import {  
  TOOLTIP_POINTER_MARGIN,  
  ...
```

### **frontend/src/metabase/visualizations/echarts/types.ts**

```
import type {  
  BarSeriesOption,  
  CustomSeriesOption,  
  LineSeriesOption,  
  ScatterSeriesOption,  
} from "echarts/charts";  
import type { ElementEvent } from "echarts/core";  
import type {...
```

### **frontend/src/metabase/visualizations/hooks/use-browser-rendering-context.ts**

```
import { useMemo } from "react";  
  
import { usePalette } from "metabase/common/hooks/use-palette";  
import { getIsNightMode } from "metabase/dashboard/selectors";  
import { color } from...
```

### **frontend/src/metabase/visualizations/index.ts**

```
import { t } from "ttag";  
import _ from "underscore";  
  
import { isStorybookActive } from "metabase/env";  
import type {  
  DatasetData,  
  RawSeries,  
  Series,  
  TransformedSeries,  
  ...
```

### **frontend/src/metabase/visualizations/lib/action.js**

```
import { push } from "react-router-redux";  
import _ from "underscore";
```



## Analyst Base

```
import { setParameterValuesFromQueryParams } from "metabase/dashboard/actions/parameters";
import { isEmbeddingSdk } from...
```

### **frontend/src/metabase/visualizations/lib/action.unit.spec.ts**

```
import MetabaseSettings from "metabase/lib/settings";
import type {
  QuestionChangeClickAction,
  UrlClickAction,
} from "metabase/visualizations/types";
import Question from...
```

### **frontend/src/metabase/visualizations/lib/color.ts**

```
import Color from "color";

export function getHexColor(color: string) {
  // Convert color values to hex format since Apache Batik (SVG renderer used in static visualizations)
  // doesn't support...
```

### **frontend/src/metabase/visualizations/lib/dataset.ts**

```
import { isNotNull } from "metabase/lib/types";
import { isMetric } from "metabase-lib/v1/types/utils/isa";
import type {
  DatasetData,
  RawSeries,
  RowValue,
  RowValues,
} from...
```

### **frontend/src/metabase/visualizations/lib/dataset.unit.spec.ts**

```
import {
  createMockColumn,
  createMockDatasetData,
} from "metabase-types/api/mocks";

import { groupDatasetMetrics, sumMetric } from "../dataset";

describe("sumMetric", () => {
  it("should...
```

### **frontend/src/metabase/visualizations/lib/errors.js**

```
import { msgid, ngettext, t } from "ttag";

export class MinColumnsError extends Error {
```

## Analyst Base

```
constructor(minColumns, actualColumns) {  
  super(  
    t`Doh! The data from your query doesn't fit the...
```

### frontend/src/metabase/visualizations/lib/errors.unit.spec.js

```
import { MinRowsError } from "metabase/visualizations/lib/errors";
```

```
describe("MinRowsError", () => {  
  it("should be an instanceof Error", () => {  
    expect(new MinRowsError(1, 0) instanceof...
```

### frontend/src/metabase/visualizations/lib/exports-branding-utils.unit.spec.ts

```
import {  
  createBrandingElement,  
  getBrandingConfig,  
  getBrandingSize,  
} from "../exports-branding-utils";
```

```
describe("getBrandingSize", () => {  
  it("should return correct size for different...
```

### frontend/src/metabase/visualizations/lib/get-dashboard-image.ts

```
import { DASHBOARD_HEADER_PARAMETERS_PDF_EXPORT_NODE_ID } from  
"metabase/dashboard/constants";
```

```
import { SAVING_DOM_IMAGE_CLASS } from "../image-exports";  
const PARAMETERS_MARGIN_BOTTOM = 12;
```

```
export...
```

### frontend/src/metabase/visualizations/lib/graph/columns.ts

```
import _ from "underscore";
```

```
import { isNotNull } from "metabase/lib/types";  
import type { RemappingHydratedDatasetColumn } from "metabase/visualizations/types";  
import type {  
  DatasetColumn,  
  ...
```

### frontend/src/metabase/visualizations/lib/graph/columns.unit.spec.ts

```
import { createMockColumn } from "metabase-types/api/mocks";
```

```
import { getCartesianChartColumns } from "../columns";
```

## Analyst Base

```
describe("getCartesianChartColumns", () => {  
  it("should ignore duplicated...
```

### **frontend/src/metabase/visualizations/lib/image-exports.ts**

```
eslint-disable-next-line no-restricted-imports
```

### **frontend/src/metabase/visualizations/lib/map.ts**

```
import type { Dataset, JsonQuery } from "metabase-types/api";
```

```
interface TileCoordinate {  
  x: number | string;  
  y: number | string;  
}
```

```
interface TileUrlParams {  
  cardId?: number;  
  dashboardId?:...
```

### **frontend/src/metabase/visualizations/lib/map.unit.spec.ts**

```
import type { JsonQuery } from "metabase-types/api";  
import { createMockDataset } from "metabase-types/api/mocks/dataset";
```

```
import { getTileUrl } from "../map";
```

```
describe("map", () => {  
  ...
```

### **frontend/src/metabase/visualizations/lib/mapping.js**

```
import * as d3 from "d3";  
import L from "leaflet/dist/leaflet-src.js";
```

```
import { COUNTRY_NAME_TO_CODE, STATE_CODES } from "../mapping_codes";
```

```
export function computeMinimalBounds(features) {  
  const...
```

### **frontend/src/metabase/visualizations/lib/mapping.unit.spec.ts**

```
import { getCanonicalRowKey } from "../mapping";
```

```
describe("getCanonicalRowKey", () => {  
  it("should convert US state names to their iso2 codes", () => {  
    expect(getCanonicalRowKey("Alabama",...
```

### **frontend/src/metabase/visualizations/lib/mapping\_codes.ts**

```
export const STATE_CODES = [
```

## Analyst Base

```
["AL", "Alabama"],
["AK", "Alaska"],
["AS", "American Samoa"],
["AZ", "Arizona"],
["AR", "Arkansas"],
["CA", "California"],
["CO", "Colorado"],
["CT", ...
```

### frontend/src/metabase/visualizations/lib/numeric.js

```
import { isNumeric } from "metabase-lib/v1/types/utils/isa";

export function dimensionIsNumeric({ cols, rows }, i = 0) {
  if (isNumeric(cols[i])) {
    return true;
  }
}
```

```
const hasAtLeastOneNumber...
```

### frontend/src/metabase/visualizations/lib/numeric.unit.spec.js

```
import {
  computeChange,
  computeNumericDataInterval,
  isMultipleOf,
  precision,
} from "metabase/visualizations/lib/numeric";
```

```
describe("visualization.lib.numeric", () => {
  ...
```

### frontend/src/metabase/visualizations/lib/renderer\_utils.js

```
import { getIn } from "icepick";
import _ from "underscore";

import { formatNullable } from "metabase/lib/formatting/nullable";
import { parseTimestamp } from "metabase/lib/time";
import {...
```

### frontend/src/metabase/visualizations/lib/renderer\_utils.unit.spec.js

```
import moment from "moment-timezone"; // eslint-disable-line no-restricted-imports -- deprecated usage

import {
  getXValues,
  parseXValue,
```

## Analyst Base

```
} from...
```

### **frontend/src/metabase/visualizations/lib/save-chart-image.ts**

```
eslint-disable-next-line no-restricted-imports
```

### **frontend/src/metabase/visualizations/lib/save-dashboard-pdf.ts**

```
import { t } from "ttag";
```

```
import { DASHBOARD_HEADER_PARAMETERS_PDF_EXPORT_NODE_ID } from
"metabase/dashboard/constants";
import type { Dashboard } from "metabase-types/api";
```

```
import {
```

```
...
```

### **frontend/src/metabase/visualizations/lib/save-dashboard-pdf.unit.spec.ts**

```
import { findPageBreakCandidates, getPageBreaks } from "./save-dashboard-pdf";
```

```
describe("save-dashboard-pdf", () => {
  describe("findPageBreakCandidates", () => {
    it("should find a potential...
```

### **frontend/src/metabase/visualizations/lib/scalar\_utils.ts**

```
import { formatValue } from "metabase/lib/formatting";
import type { OptionsType } from "metabase/lib/formatting/types";
import type { RowValue } from "metabase-types/api";
```

```
export const...
```

### **frontend/src/metabase/visualizations/lib/scalar\_utils.unit.spec.ts**

```
import { TYPE } from "metabase-lib/v1/types/constants";
```

```
import {
  COMPACT_MAX_WIDTH,
  COMPACT_MIN_LENGTH,
  COMPACT_WIDTH_PER_DIGIT,
  compactifyValue,
} from "./scalar_utils";
```

```
describe("scalar...
```

### **frontend/src/metabase/visualizations/lib/series.ts**

```
import { assocIn } from "icepick";
```

```
import { SERIES_SETTING_KEY } from "metabase/visualizations/shared/settings/series";
```

## Analyst Base

```
import type { Card, VisualizationSettings } from...
```

### **frontend/src/metabase/visualizations/lib/settings/analytics.ts**

```
import { trackSchemaEvent } from "metabase/lib/analytics";
import type { DashboardId } from "metabase-types/api";
```

```
export const trackCardSetToHideWhenNoResults = (dashboardId: DashboardId) => {
  ...
}
```

### **frontend/src/metabase/visualizations/lib/settings/column.js**

```
import { t } from "ttag";
import _ from "underscore";
```

```
import { currency } from "cljs/metabase.util.currency";
import {
  displayNameForColumn,
  getCurrency,
  getCurrencyStyleOptions,
  ...
}
```

### **frontend/src/metabase/visualizations/lib/settings/column.unit.spec.js**

```
import { getComputedSettings } from "metabase/visualizations/lib/settings";
import registerVisualizations from "metabase/visualizations/register";
```

```
import { NUMBER_COLUMN_SETTINGS, columnSettings }...
```

### **frontend/src/metabase/visualizations/lib/settings/goal.ts**

```
import { t } from "ttag";
```

```
import { getDefaultGoalLabel } from "metabase/visualizations/shared/settings/cartesian-chart";
import type { ChartGoal } from...
```

### **frontend/src/metabase/visualizations/lib/settings/graph.js**

```
import { t } from "ttag";
import _ from "underscore";
```

```
import { color } from "metabase/lib/colors";
import {
  getMaxDimensionsSupported,
  getMaxMetricsSupported,
} from...
```

### **frontend/src/metabase/visualizations/lib/settings/graph.unit.spec.js**

```
import {
  createMockCard,
```

## Analyst Base

```
createMockColumn,  
createMockDataset,  
createMockDatasetData,  
createMockSingleSeries,  
} from "metabase-types/api/mocks";
```

```
import {  
  GRAPH_AXIS_SETTINGS,  
  ...
```

### **frontend/src/metabase/visualizations/lib/settings/nested.js**

```
import { t } from "ttag";  
import _ from "underscore";
```

```
import                                chartSettingNestedSettings                                from  
"metabase/visualizations/components/settings/ChartSettingNestedSettings";
```

```
import { getComputedSettings,...
```

### **frontend/src/metabase/visualizations/lib/settings/nested.unit.spec.js**

```
import { getComputedSettings } from "metabase/visualizations/lib/settings";  
import { nestedSettings } from "metabase/visualizations/lib/settings/nested";
```

```
describe("nestedSettings", () => {  
  ...
```

### **frontend/src/metabase/visualizations/lib/settings/row-values.ts**

```
export const getNumberOr = <TValue, TReplacement>(  
  value: TValue,  
  replacement: TReplacement,  
) : number | TReplacement => {  
  if (typeof value === "number") {  
    return value;  
  }
```

```
  return...
```

### **frontend/src/metabase/visualizations/lib/settings/series.js**

```
import { getIn } from "icepick";  
import { t } from "ttag";
```

```
import                                ChartNestedSettingSeries                                from  
"metabase/visualizations/components/settings/ChartNestedSettingSeries";  
import { OTHER_DATA_KEY } from...
```

## Analyst Base

### **frontend/src/metabase/visualizations/lib/settings/series.unit.spec.ts**

```
import {
  createMockCard,
  createMockVisualizationSettings,
} from "metabase-types/api/mocks";

import { getColors } from "../series";

describe("Series unit settings", () => {
  ...
```

### **frontend/src/metabase/visualizations/lib/settings/stacking.ts**

```
import type { StackOffset } from "metabase/visualizations/shared/components/RowChart/types";
import type { VisualizationSettings } from "metabase-types/api";
```

```
export const getStackOffset = (
  ...
```

### **frontend/src/metabase/visualizations/lib/settings/typed-utils.ts**

```
import { getIn } from "icepick";
import _ from "underscore";

import { getVisualization } from "metabase/visualizations";
import type { VisualizationSettingDefinition } from...
```

### **frontend/src/metabase/visualizations/lib/settings/typed-utils.unit.spec.ts**

```
import { registerVisualization } from "metabase/visualizations";
import { BarChart } from "metabase/visualizations/visualizations/BarChart";
import {
  createMockCard,
  ...
```

### **frontend/src/metabase/visualizations/lib/settings/utils.js**

```
import {
  columnsAreValid,
  getDefaultDimensionAndMetric,
} from "metabase/visualizations/lib/utils";
import { isDimension, isMetric } from "metabase-lib/v1/types/utils/isa";

export function...
```

### **frontend/src/metabase/visualizations/lib/settings/validation.js**

```
import { t } from "ttag";
import _ from "underscore";
```



## Analyst Base

```
import { isNotNull } from "metabase/lib/types";
import {
  ChartSettingsError,
  MinRowsError,
} from...
```

### **frontend/src/metabase/visualizations/lib/settings/visualization.js**

```
import { assocIn } from "icepick";
import { t } from "ttag";
```

```
import { isVirtualDashCard } from "metabase/dashboard/utils";
import { getVisualizationRaw } from "metabase/visualizations";
import {...
```

### **frontend/src/metabase/visualizations/lib/settings/visualization.unit.spec.js**

```
import icepick from "icepick";
```

```
import { DateTimeColumn, NumberColumn } from "___support___/visualizations";
import {
  getComputedSettingsForSeries,
  getStoredSettingsForSeries,
} from...
```

### **frontend/src/metabase/visualizations/lib/settings/widgets.ts**

```
const PREFIX = "\0_";
```

```
// Encode boolean/null values to strings for Mantine form widgets (Select, Radio, etc), , needed for settings
like "graph.x_axis.axis_enabled"
const toWidgetValue = new...
```

### **frontend/src/metabase/visualizations/lib/settings.js**

```
import _ from "underscore";
```

```
import {
  ChartSettingColorPicker
} from "metabase/visualizations/components/settings/ChartSettingColorPicker";
import ChartSettingColorsPicker from...
```

### **frontend/src/metabase/visualizations/lib/settings.unit.spec.js**

NOTE: need to load visualizations first for getSettings to work

### **frontend/src/metabase/visualizations/lib/table.js**

```
import { isCoordinate, isNumber } from "metabase-lib/v1/types/utils/isa";
```

```
/**
```

## Analyst Base

```
* @param {import("metabase-types/api").Series} series
* @param {number} rowIndex
* @param {number} columnIndex
* ...
```

### frontend/src/metabase/visualizations/lib/table.unit.spec.js

```
import {
  getTableCellClickedObject,
  getTableClickedObjectRowData,
  isColumnRightAligned,
} from "metabase/visualizations/lib/table";
import { TYPE } from...
```

### frontend/src/metabase/visualizations/lib/table\_format.js

NOTE: this file is used on the frontend and backend and there are some limitations. See frontend/src/metabase-shared/color\_selector for details

### frontend/src/metabase/visualizations/lib/table\_format.unit.spec.js

```
import { ALL_OPERATOR_NAMES } from
"metabase/visualizations/components/settings/ChartSettingsTableFormatting";
import {
  OPERATOR_FORMATTER_FACTORIES,
  canCompareSubstrings,
  compileFormatter,
  ...
```

### frontend/src/metabase/visualizations/lib/text.ts

```
import type {
  FontStyle,
  TextWidthMeasurer,
} from "../shared/types/measure-text";

export const CHAR_ELLIPSES = "...";

export function truncateText(
  text: string,
  width: number,
  measurer:...
```

### frontend/src/metabase/visualizations/lib/text.unit.spec.ts

```
import { measureTextWidth as measureDynamic } from "metabase/lib/measure-text";
import { measureTextWidth as measureStatic } from "metabase/static-viz/lib/text";

import type { TextWidthMeasurer }...
```

## Analyst Base

### **frontend/src/metabase/visualizations/lib/timeseries.js**

```
import moment from "moment-timezone"; // eslint-disable-line no-restricted-imports -- deprecated usage
```

```
import { isDate } from "metabase-lib/v1/types/utils/isa";
```

```
const TIMESERIES_UNITS = new Set([
```

```
...
```

### **frontend/src/metabase/visualizations/lib/timeseries.unit.spec.js**

```
import { dimensionIsTimeseries } from "metabase/visualizations/lib/timeseries";
```

```
import registerVisualizations from "metabase/visualizations/register";
```

```
import { TYPE } from...
```

### **frontend/src/metabase/visualizations/lib/tooltip.ts**

```
import { formatValue } from "metabase/lib/formatting";
```

```
import { formatNullable } from "metabase/lib/formatting/nullable";
```

```
import type { DatasetColumn, VisualizationSettings } from...
```

### **frontend/src/metabase/visualizations/lib/trends.js**

```
mappings of allowed operators
```

### **frontend/src/metabase/visualizations/lib/utils.js**

```
import crossfilter from "crossfilter";
```

```
import * as d3 from "d3";
```

```
import _ from "underscore";
```

```
import { isNotNull } from "metabase/lib/types";
```

```
import { getColumnKey } from...
```

### **frontend/src/metabase/visualizations/lib/utils.unit.spec.js**

```
import _ from "underscore";
```

```
import {
```

```
  cardHasBecomeDirty,
```

```
  computeMaxDecimalsForValues,
```

```
  computeSplit,
```

```
  findSensibleSankeyColumns,
```

```
  getCardAfterVisualizationClick,
```

```
  getColumnCardinality,
```

```
...
```

### **frontend/src/metabase/visualizations/lib/warnings.js**

```
import { t } from "ttag";
```

## Analyst Base

```
const NULL_DIMENSION_WARNING = "NULL_DIMENSION_WARNING";
export function nullDimensionWarning() {
  return {
    key: NULL_DIMENSION_WARNING,
    text: `Data includes...
```

### frontend/src/metabase/visualizations/register.js

```
import ActionViz from "metabase/actions/components/ActionViz";
import {
  registerVisualization,
  setDefaultVisualization,
} from "metabase/visualizations";
```

```
import { AreaChart } from...
```

### frontend/src/metabase/visualizations/shared/components/RowChart/constants.ts

```
export const LABEL_PADDING = 4;
```

### frontend/src/metabase/visualizations/shared/components/RowChart/index.ts

```
export * from "../RowChart";
```

### frontend/src/metabase/visualizations/shared/components/RowChart/types.ts

```
import type { StringLike } from "@visx/scale";
```

```
import type { AxisStyle, ChartFont, GoalStyle } from "../../types/style";
```

```
export type XValue = number | null;
```

```
export type YValue = string | number |...
```

### frontend/src/metabase/visualizations/shared/components/RowChart/utils/data.ts

```
import type { Series as D3Series } from "d3";
```

```
import * as d3 from "d3";
```

```
import { stack, stackOffsetDiverging, stackOffsetExpand } from "d3";
```

```
import _ from "underscore";
```

```
import { formatNullable }...
```

### frontend/src/metabase/visualizations/shared/components/RowChart/utils/domain.ts

```
import { extent } from "d3";
```

```
import { isNotNull } from "metabase/lib/types";
```

## Analyst Base

```
import type {  
  ContinuousDomain,  
  ContinuousScaleType,  
} from "metabase/visualizations/shared/types/scale";
```

```
import...
```

**frontend/src/metabase/visualizations/shared/components/RowChart/utils/layout.ts**

```
import type { ScaleContinuousNumeric } from "d3-scale";
```

```
import type { Margin } from "metabase/visualizations/shared/types/layout";  
import type { TextWidthMeasurer } from...
```

**frontend/src/metabase/visualizations/shared/components/RowChart/utils/scale.ts**

```
import type { ContinuousDomain } from "@visx/scale";  
import { scaleBand, scaleLinear, scaleLog, scalePower } from "@visx/scale";  
import type { ScaleContinuousNumeric } from "d3-scale";
```

```
import type {...
```

**frontend/src/metabase/visualizations/shared/components/RowChart/utils/ticks.ts**

```
import type { ScaleContinuousNumeric } from "d3-scale";  
import _ from "underscore";
```

```
import type { ValueFormatter } from "metabase/visualizations/shared/types/format";  
import type { TextWidthMeasurer...
```

**frontend/src/metabase/visualizations/shared/components/RowChartView/constants.ts**

```
export const DATA_LABEL_OFFSET = 4;
```

**frontend/src/metabase/visualizations/shared/components/RowChartView/index.ts**

```
export * from "./RowChartView";  
export * from "./constants";
```

**frontend/src/metabase/visualizations/shared/components/RowChartView/utils/data-labels.ts**

```
import type { ScaleContinuousNumeric } from "d3-scale";
```

```
import type { BarData } from "../../RowChart/types";
```

## Analyst Base

```
export const getDataLabel = <TDatum>(  
  bar: BarData<TDatum>,  
  xScale:...
```

### **frontend/src/metabase/visualizations/shared/components/RowChartView/utils/data-labels.unit.spec.ts**

```
import { scaleLinear } from "d3-scale";  
  
import type { BarData } from "../../RowChart/types";  
  
import { getDataLabel } from "../data-labels";
```

```
describe("getDataLabel", () => {  
  const mockXScale =...
```

### **frontend/src/metabase/visualizations/shared/settings/cartesian-chart.ts**

```
import { t } from "ttag";  
import _ from "underscore";  
  
import { isNotNull } from "metabase/lib/types";  
import {  
  getMaxDimensionsSupported,  
  getMaxMetricsSupported,  
} from...
```

### **frontend/src/metabase/visualizations/shared/settings/column.ts**

```
import { isCurrency, isPercentage } from "metabase-lib/v1/types/utils/isa";  
import type { ColumnSettings, DatasetColumn } from "metabase-types/api";  
  
export function getDefaultNumberStyle(  
  column:...
```

### **frontend/src/metabase/visualizations/shared/settings/pie.ts**

```
import Color from "color";  
import _ from "underscore";  
  
import { getColorsForValues } from "metabase/lib/colors/charts";  
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";  
import {...
```

### **frontend/src/metabase/visualizations/shared/settings/series.ts**

```
import { getIn } from "icepick";  
import _ from "underscore";
```

## Analyst Base

```
import { getColorsForValues } from "metabase/lib/colors/charts";
import type { ComputedVisualizationSettings } from...
```

### **frontend/src/metabase/visualizations/shared/types/data.ts**

```
import type { DatasetColumn, RowValue, RowValues } from "metabase-types/api";
```

```
export type SeriesInfo = {
  metricColumn: DatasetColumn;
  dimensionColumn: DatasetColumn;
  breakoutValue?:...
```

### **frontend/src/metabase/visualizations/shared/types/events.ts**

```
export type HoveredData = {
  seriesIndex: number;
  datumIndex?: number;
};
```

### **frontend/src/metabase/visualizations/shared/types/format.ts**

```
import type { DatasetColumn } from "metabase-types/api";
```

```
export type ValueFormatter = (value: any) => string;
```

```
export type ColumnFormatter = (value: any, column: DatasetColumn) => string;
```

```
export...
```

### **frontend/src/metabase/visualizations/shared/types/layout.ts**

```
export type Margin = {
  top: number;
  bottom: number;
  right: number;
  left: number;
};
```

### **frontend/src/metabase/visualizations/shared/types/measure-text.ts**

```
export type FontStyle = {
  size: string | number;
  family: string;
  weight: string | number;
};
```

```
export interface TextSize {
  width: number;
```

## Analyst Base

```
height: number;  
}
```

```
export type TextWidthMeasurer =...
```

### **frontend/src/metabase/visualizations/shared/types/scale.ts**

```
export type ContinuousScaleType = "linear" | "pow" | "log";  
export type ContinuousDomain = [number, number];  
export type Range = [number, number];
```

### **frontend/src/metabase/visualizations/shared/types/settings.ts**

```
export type ChartGoal = {  
  label: string;  
  value: number;  
};
```

### **frontend/src/metabase/visualizations/shared/types/style.ts**

```
export type ChartFont = {  
  size: number;  
  family: string;  
  weight: number;  
  color: string;  
};
```

```
export type GoalStyle = {  
  lineStroke: string;  
  label: ChartFont;  
};
```

```
export type AxisStyle = {  
  ...
```

### **frontend/src/metabase/visualizations/shared/utils/colors.ts**

```
import { getColorsForValues } from "metabase/lib/colors/charts";  
import type { VisualizationSettings } from "metabase-types/api";
```

```
import type { Series } from "../components/RowChart/types";
```

```
export...
```

### **frontend/src/metabase/visualizations/shared/utils/data.ts**

```
import { t } from "ttag";
```

```
import { formatNullable } from "metabase/lib/formatting/nullable";
```



## Analyst Base

```
import { getColumnScaling } from "metabase/visualizations/echarts/cartesian/model/util";
import {...
```

### **frontend/src/metabase/visualizations/shared/utils/data.unit.spec.ts**

```
import type {
  BreakoutChartColumns,
  MultipleMetricsChartColumns,
} from "metabase/visualizations/lib/graph/columns";
import type { ColumnFormatter } from...
```

### **frontend/src/metabase/visualizations/shared/utils/parameter-substitution.js**

```
import _ from "underscore";

import { substitute_tags } from "cljs/metabase.parameters.shared";
import { siteLocale, withInstanceLanguage } from "metabase/lib/i18n";

export function...
```

### **frontend/src/metabase/visualizations/shared/utils/series.ts**

```
import type { CartesianChartColumns } from "metabase/visualizations/lib/graph/columns";
import { getCartesianChartColumns } from "metabase/visualizations/lib/graph/columns";
import type {...
```

### **frontend/src/metabase/visualizations/shared/utils/size-in-px.ts**

```
*
```

### **frontend/src/metabase/visualizations/shared/utils/size-in-px.unit.spec.ts**

```
import { getSizeInPx } from "../size-in-px";

describe("getSizeInPx", () => {
  it("returns the number value if it's not a string", () => {
    expect(getSizeInPx(12)).toBe(12);
  });

  it("returns...
```

### **frontend/src/metabase/visualizations/shared/utils/sizes.ts**

```
import _ from "underscore";

import { CARD_SIZE_DEFAULTS_JSON } from "cljs/metabase.dashboards.constants";
import { DEFAULT_CARD_SIZE } from "metabase/lib/dashboard_grid";
import type {...
```

### **frontend/src/metabase/visualizations/shared/utils/theme.ts**

```
import { DEFAULT_METABASE_COMPONENT_THEME } from "metabase/embedding-sdk/theme";
```

## Analyst Base

```
import { color } from "metabase/lib/colors";
import type { MantineThemeOther } from "metabase/ui";
import {...
```

### **frontend/src/metabase/visualizations/types/click-actions.ts**

```
import type React from "react";

import type { IconName } from "metabase/ui";
import type { Mode } from "metabase/visualizations/click-actions/Mode";
import type * as Lib from "metabase-lib";
import...
```

### **frontend/src/metabase/visualizations/types/columns.ts**

```
import type { DatasetColumn, DatasetData } from "metabase-types/api";

export type RemappingHydratedDatasetColumn = DatasetColumn & {
  remapped_from_index?: number;
  remapped_to_column?:...
```

### **frontend/src/metabase/visualizations/types/echarts.ts**

```
export type EChartsEventHandler = {
  eventName: string;
  query?: string;
  handler: (event: any) => void;
};
```

```
export type ZREventHandler = {
  eventName: string;
  handler: (event: any) => void;
};
```

### **frontend/src/metabase/visualizations/types/hover.ts**

```
import type { ClickObjectDataRow } from "metabase-lib";
import type { RowValue, TimelineEvent } from "metabase-types/api";

import type { RemappingHydratedDatasetColumn } from "../columns";
import...
```

### **frontend/src/metabase/visualizations/types/index.ts**

```
export * from "../hover";
export * from "../columns";
export * from "../visualization";
export * from "../click-actions";
```

### **frontend/src/metabase/visualizations/types/visualization.ts**

## Analyst Base

```
import type { ReactNode } from "react";
import type React from "react";
```

```
import type { OptionsType } from "metabase/lib/formatting/types";
import type { IconName, IconProps } from...
```

### **frontend/src/metabase/visualizations/visualizations/AreaChart/index.ts**

```
export * from "./AreaChart";
```

### **frontend/src/metabase/visualizations/visualizations/BarChart/index.ts**

```
export * from "./BarChart";
```

### **frontend/src/metabase/visualizations/visualizations/CartesianChart/chart-definition-legacy.js**

```
import _ from "underscore";
```

```
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";
import { formatValue } from "metabase/lib/formatting";
import { isEmpty } from...
```

### **frontend/src/metabase/visualizations/visualizations/CartesianChart/chart-definition.ts**

```
import _ from "underscore";
```

```
import { GRAPH_GOAL_SETTINGS } from "metabase/visualizations/lib/settings/goal";
import {
  GRAPH_AXIS_SETTINGS,
  GRAPH_COLORS_SETTINGS,
  GRAPH_DATA_SETTINGS,
  ...
```

### **frontend/src/metabase/visualizations/visualizations/CartesianChart/chart-definition.unit.spec.ts**

```
import { SERIES_SETTING_KEY } from "metabase/visualizations/shared/settings/series";
```

```
import { getCartesianChartDefinition } from "./chart-definition";
```

```
describe("chart-definition", () => {
  ...
```

### **frontend/src/metabase/visualizations/visualizations/CartesianChart/events.ts**

```
import { t } from "ttag";
import _ from "underscore";
```

## Analyst Base

```
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";
import { formatChangeWithSign } from "metabase/lib/formatting";
import {...
```

### **frontend/src/metabase/visualizations/visualizations/CartesianChart/index.ts**

```
export * from "../CartesianChart";
```

### **frontend/src/metabase/visualizations/visualizations/CartesianChart/padding.ts**

```
import type { MantineTheme } from "metabase/ui";
```

```
export const getChartPadding = ({
  theme,
  isQueryBuilder,
}): {
  isQueryBuilder?: boolean;
  theme: MantineTheme;
}) => {
  const { padding } =...
```

### **frontend/src/metabase/visualizations/visualizations/CartesianChart/use-chart-debug.ts**

```
eslint-disable no-console
```

### **frontend/src/metabase/visualizations/visualizations/CartesianChart/use-chart-events.ts**

```
import type { EChartsCoreOption, EChartsType } from "echarts/core";
import type * as React from "react";
import {
  useCallback,
  useEffect,
  useLayoutEffect,
  useMemo,
  useRef,
} from...
```

### **frontend/src/metabase/visualizations/visualizations/CartesianChart/use-models-and-option.ts**

```
import { useCallback, useMemo } from "react";
```

```
import { useTranslateContent } from "metabase/i18n/hooks";
import { isReducedMotionPreferred } from "metabase/lib/dom";
import { extractRemappings }...
```

### **frontend/src/metabase/visualizations/visualizations/CartesianChart/use-tooltip-**

### **mouse-leave.ts**

```
import type { EChartsType } from "echarts/core";
import { useEffect, useRef } from "react";
import _ from "underscore";
```

```
import { ECHARTS_TOOLTIP_CONTAINER_CLASS } from...
```

### **frontend/src/metabase/visualizations/visualizations/CartesianChart/utils.ts**

```
import type { EChartsCoreOption } from "echarts/core";
import { t } from "ttag";
```

```
import { isNotNull } from "metabase/lib/types";
import type {
  BaseCartesianChartModel,
  DataKey,
  SeriesModel,
}...
```

### **frontend/src/metabase/visualizations/visualizations/ComboChart/index.ts**

```
export * from "./ComboChart";
```

### **frontend/src/metabase/visualizations/visualizations/Funnel/funnel-bar-transform.ts**

```
import { formatValue } from "metabase/lib/formatting";
import { isNotNull } from "metabase/lib/types";
import type { TransformSeries } from...
```

### **frontend/src/metabase/visualizations/visualizations/Funnel/index.ts**

```
export * from "./Funnel";
```

### **frontend/src/metabase/visualizations/visualizations/Funnel/types.ts**

```
import type { RowValue } from "metabase-types/api";
```

```
export type FunnelRow = {
  key: RowValue;
  name: RowValue;
  enabled: boolean;
};
```

### **frontend/src/metabase/visualizations/visualizations/Gauge/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/visualizations/visualizations/Gauge/utils.ts**

```
export const getValue = (rows: unknown[][]) => {
```

## Analyst Base

```
const rawValue = rows[0] && rows[0][0];
```

```
if (rawValue === "Infinity") {  
  return Infinity;  
}
```

```
if (typeof rawValue !== "number") {  
  return...
```

### **frontend/src/metabase/visualizations/visualizations/Gauge/utils.unit.spec.ts**

```
import { getValue } from "./utils";
```

```
describe("Visualizations > Gauge > utils", () => {  
  const valueTestCases = [  
    [[[null]], 0],  
    [[[undefined]], 0],  
    [[[ "foo" ]], 0],  
    [[[ " " ]], 0],  
    ...
```

### **frontend/src/metabase/visualizations/visualizations/Heading/index.ts**

```
import { t } from "ttag";
```

```
import {  
  getDefaultSize,  
  getMinSize,  
} from "metabase/visualizations/shared/utils/sizes";
```

```
import { Heading } from "../Heading";
```

```
const HeadingWrapper =...
```

### **frontend/src/metabase/visualizations/visualizations/IframeViz/IframeVizSettings.ts**

```
import { t } from "ttag";
```

```
import {  
  getDefaultSize,  
  getMinSize,  
} from "metabase/visualizations/shared/utils/sizes";
```

```
export const settings = {  
  getUiName: () => "iframe",
```

## Analyst Base

```
canSavePng: false,
```

```
...
```

### **frontend/src/metabase/visualizations/visualizations/IFrameViz/index.ts**

```
export * from "./IFrameViz";
```

### **frontend/src/metabase/visualizations/visualizations/IFrameViz/utils.ts**

```
import { isSafeUrl } from "metabase/lib/formatting/link";
```

```
/**
```

```
 * Reconstructs a URL from its parts while preserving parameter placeholders (e.g. {{param}}).
```

```
 * Unlike URL.toString(), this won't...
```

### **frontend/src/metabase/visualizations/visualizations/IFrameViz/utils.unit.spec.ts**

```
import {
  getAllowedIframeAttributes,
  getIframeDomainName,
  isAllowedIframeUrl,
} from "./utils";
```

```
describe("getAllowedIframeAttributes", () => {
  describe("share to embed link transformation", ...
```

### **frontend/src/metabase/visualizations/visualizations/LineChart/index.ts**

```
export * from "./LineChart";
```

### **frontend/src/metabase/visualizations/visualizations/LinkViz/LinkVizSettings.ts**

```
import { t } from "ttag";
```

```
import {
  getDefaultSize,
  getMinSize,
} from "metabase/visualizations/shared/utils/sizes";
```

```
export const settings = {
  getUiName: () => "Link",
  canSavePng: false,
```

```
...
```

### **frontend/src/metabase/visualizations/visualizations/LinkViz/index.ts**

```
export { LinkViz } from "./LinkViz";
```

### **frontend/src/metabase/visualizations/visualizations/LinkViz/types.ts**

```
import type { IconName } from "metabase/ui";
import type { UnrestrictedLinkEntity } from "metabase-types/api";
```

## Analyst Base

```
type WrappedEntity = {  
  getIcon: () => { name: IconName };  
  getUrl: () =>...
```

### **frontend/src/metabase/visualizations/visualizations/LinkViz/utils.ts**

```
export const isUrlString = (str?: string) => str && /^http/i.test(str);
```

### **frontend/src/metabase/visualizations/visualizations/Map/CustomMapFooter/index.ts**

```
export { CustomMapFooter } from "./CustomMapFooter";
```

### **frontend/src/metabase/visualizations/visualizations/Map/CustomMapFooter/tests/common.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import { setup } from "./setup";
```

```
describe("CustomMapFooter (OSS)", () => {  
  describe("admin users", () => {  
    it("should show an admin settings link...
```

### **frontend/src/metabase/visualizations/visualizations/Map/CustomMapFooter/tests/enterprise.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import type { SetupOpts } from "./setup";  
import { setup as baseSetup } from "./setup";
```

```
function setup(opts: SetupOpts) {  
  baseSetup({...
```

### **frontend/src/metabase/visualizations/visualizations/Map/CustomMapFooter/tests/premium.unit.spec.ts**

```
import { screen } from "__support__/ui";
```

```
import type { SetupOpts } from "./setup";  
import { setup as baseSetup } from "./setup";
```

```
function setup(opts: SetupOpts) {  
  baseSetup({  
    ...
```

### **frontend/src/metabase/visualizations/visualizations/Map/index.ts**



## Analyst Base

```
export { Map } from "../Map";
```

### **frontend/src/metabase/visualizations/visualizations/PieChart/chart-definition.ts**

```
import { t } from "ttag";
```

```
import _ from "underscore";
```

```
import { formatValue } from "metabase/lib/formatting";
```

```
import {
```

```
  ChartSettingsError,
```

```
  MinRowsError,
```

```
} from...
```

### **frontend/src/metabase/visualizations/visualizations/PieChart/index.ts**

```
export { PieChart } from "../PieChart";
```

### **frontend/src/metabase/visualizations/visualizations/PieChart/use-chart-events.ts**

```
import type { EChartsType } from "echarts/core";
```

```
import { type MutableRefObject, useEffect, useMemo } from "react";
```

```
import { t } from "ttag";
```

```
import { checkNotNull } from...
```

### **frontend/src/metabase/visualizations/visualizations/PivotTable/PivotTable.unit.spec.js**

```
import userEvent from "@testing-library/user-event";
```

```
import { thaw } from "icepick";
```

```
import { useState } from "react";
```

```
import { createMockMetadata } from "___support___/metadata";
```

```
import { render,...
```

### **frontend/src/metabase/visualizations/visualizations/PivotTable/constants.ts**

```
cell width and height for normal body cells
```

### **frontend/src/metabase/visualizations/visualizations/PivotTable/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase/visualizations/visualizations/PivotTable/settings.ts**

```
import { getIn } from "icepick";
```

```
import { t } from "ttag";
```

```
import _ from "underscore";
```

```
import {
```

```
  COLLAPSED_ROWS_SETTING,
```

## Analyst Base

```
COLUMN_FORMATTING_SETTING,  
COLUMN_SHOW_TOTALS,  
COLUMN_SORT_ORDER,  
...
```

### **frontend/src/metabase/visualizations/visualizations/PivotTable/stories-data.ts**

```
const cols = [  
  {  
    name: "field-123",  
    display_name: "field-123",  
    source: "breakout",  
    field_ref: ["field", 123, null],  
  },  
  {  
    name: "field-456",  
    display_name: "field-456",  
    ...  
  },  
]
```

### **frontend/src/metabase/visualizations/visualizations/PivotTable/types.ts**

```
import type { ClickObjectDataRow, ClickObjectDimension } from "metabase-lib";  
import type { DatasetColumn } from "metabase-types/api";  
  
type PivotTableClickDimension = ClickObjectDimension & {  
  ...  
}
```

### **frontend/src/metabase/visualizations/visualizations/PivotTable/utils.ts**

```
import { t } from "ttag";  
import _ from "underscore";  
  
import { DEFAULT_METABASE_COMPONENT_THEME } from "metabase/embedding-sdk/theme";  
import { sumArray } from "metabase/lib/arrays";  
import { ...  
}
```

### **frontend/src/metabase/visualizations/visualizations/PivotTable/utils.unit.spec.ts**

```
import type { PivotTableColumnSplitSetting } from "metabase-types/api";  
import { createMockColumn } from "metabase-types/api/mocks";  
  
import {  
  CELL_PADDING,  
  MAX_HEADER_CELL_WIDTH,  
  ...  
}
```

### **frontend/src/metabase/visualizations/visualizations/Progress/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

## Analyst Base

### **frontend/src/metabase/visualizations/visualizations/Progress/utils.ts**

```
export const getValue = (rows: unknown[][]) => {
  const rawValue = rows[0] && rows[0][0];

  if (rawValue === "Infinity") {
    return Infinity;
  }

  if (typeof rawValue !== "number") {
    return...
```

### **frontend/src/metabase/visualizations/visualizations/Progress/utils.unit.spec.ts**

```
import { getValue } from "./utils";

describe("Visualizations > Progress > utils", () => {
  const valueTestCases = [
    [[[null]], 0],
    [[[undefined]], 0],
    [[["foo"]], 0],
    [[[""]], 0],
    ...
```

### **frontend/src/metabase/visualizations/visualizations/RowChart/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase/visualizations/visualizations/RowChart/utils/events.ts**

```
import { getIn } from "icepick";
import _ from "underscore";

import { formatNullable } from "metabase/lib/formatting/nullable";
import { isNotNull } from "metabase/lib/types";
import { sumMetric }...
```

### **frontend/src/metabase/visualizations/visualizations/RowChart/utils/events.unit.spec.ts**

```
import type { MultipleMetricsChartColumns } from "metabase/visualizations/lib/graph/columns";
import type {
  BarData,
  Series,
} from...
```

### **frontend/src/metabase/visualizations/visualizations/RowChart/utils/format.ts**

```
import type { NumberLike, StringLike } from "@visx/scale";
```

## Analyst Base

```
import { NULL_DISPLAY_VALUE } from "metabase/lib/constants";
import { formatValue } from "metabase/lib/formatting";
import { isEmpty } from...
```

### **frontend/src/metabase/visualizations/visualizations/RowChart/utils/legend.ts**

```
import type { LegendItem } from "metabase/visualizations/echarts/cartesian/model/types";
import type { Series } from "metabase/visualizations/shared/components/RowChart/types";
import type {...
```

### **frontend/src/metabase/visualizations/visualizations/RowChart/utils/settings-definitions.js**

```
import { t } from "ttag";
```

```
import { GRAPH_GOAL_SETTINGS } from "metabase/visualizations/lib/settings/goal";
import { getDefaultDimensionLabel } from...
```

### **frontend/src/metabase/visualizations/visualizations/RowChart/utils/settings.ts**

```
import { getStackOffset } from "metabase/visualizations/lib/settings/stacking";
import type { Series } from "metabase/visualizations/shared/components/RowChart/types";
import type { Range } from...
```

### **frontend/src/metabase/visualizations/visualizations/RowChart/utils/theme.ts**

```
import { useMemo } from "react";
```

```
import { getIsNightMode } from "metabase/dashboard/selectors";
import { color } from "metabase/lib/colors";
import { useSelector } from "metabase/lib/redux";
import...
```

### **frontend/src/metabase/visualizations/visualizations/RowChart/utils/warnings.ts**

```
import type { CartesianChartColumns } from "metabase/visualizations/lib/graph/columns";
import { unaggregatedDataWarning } from "metabase/visualizations/lib/warnings";
import type { RowValues } from...
```

### **frontend/src/metabase/visualizations/visualizations/SankeyChart/chart-definition.ts**

```
import { t } from "ttag";
```

```
import { getSankeyChartColumns } from "metabase/visualizations/echarts/graph/sankey/model/dataset";
import { ChartSettingsError } from...
```

### **frontend/src/metabase/visualizations/visualizations/SankeyChart/chart-definition.unit.spec.ts**

```
import { ChartSettingsError } from "metabase/visualizations/lib/errors";
```

## Analyst Base

```
import { createMockCard } from "metabase-types/api/mocks/card";
import {
  createMockColumn,
  createMockDatasetData,
} from...
```

### **frontend/src/metabase/visualizations/visualizations/SankeyChart/events.ts**

```
import type { EChartsType } from "echarts/core";
import { useMemo } from "react";
```

```
import type {
  ColumnKey,
  SankeyChartColumns,
  SankeyLink,
  SankeyNode,
} from...
```

### **frontend/src/metabase/visualizations/visualizations/SankeyChart/events.unit.spec.ts**

```
import type { EChartsSeriesMouseEvent } from "metabase/visualizations/echarts/types";
import { getColumnKey } from "metabase-lib/v1/queries/utils/column-key";
import { createMockCard } from...
```

### **frontend/src/metabase/visualizations/visualizations/SankeyChart/index.ts**

```
export * from "./SankeyChart";
```

### **frontend/src/metabase/visualizations/visualizations/SankeyChart/stories-data/index.ts**

```
import sankeyDisconnectedGraphs from "./sankey-disconnected-graphs.json";
import sankeyEdgeLabelsAuto from "./sankey-edge-labels-auto.json";
import sankeyEdgeLabelsCompact from...
```

### **frontend/src/metabase/visualizations/visualizations/SankeyChart/utils/cycle-detection.ts**

```
import type { RowValue, RowValues } from "metabase-types/api";
```

```
type Graph = Map<RowValue, Set<RowValue>>;
```

```
const buildGraph = (
  rows: RowValues[],
  sourceIndex: number,
  targetIndex: number,
):...
```

## **frontend/src/metabase/visualizations/visualizations/SankeyChart/utils/cycle-detection.unit.spec.ts**

```
import { hasCyclicFlow } from "../cycle-detection";
```

```
describe("hasCyclicFlow", () => {
  it.each([
    [
      [
        ["A", "B"],
        ["B", "C"],
        ["C", "A"],
      ],
    ],
    [
      ...
```

## **frontend/src/metabase/visualizations/visualizations/Scalar/constants.ts**

```
export const PADDING = 32;
```

```
export const TITLE_ICON_SIZE = 10;
```

```
export const SCALAR_TITLE_LINE_HEIGHT = 23;
```

```
export const TITLE_2_LINES_HEIGHT_THRESHOLD = 120; // determined empirically
```

## **frontend/src/metabase/visualizations/visualizations/Scalar/index.ts**

```
export { Scalar } from "../Scalar";
```

## **frontend/src/metabase/visualizations/visualizations/Scalar/scalars-bar-transform.ts**

```
import { t } from "ttag";
```

```
import type { TransformSeries } from "metabase/visualizations/components/TransformedVisualization";
```

```
import { TYPE } from "metabase-lib/v1/types/constants";
```

```
import type {...
```

## **frontend/src/metabase/visualizations/visualizations/Scalar/utils.ts**

```
import {
  PADDING,
  SCALAR_TITLE_LINE_HEIGHT,
  TITLE_2_LINES_HEIGHT_THRESHOLD,
  TITLE_ICON_SIZE,
} from "../constants";
```

## Analyst Base

```
export const getTitleLinesCount = (height: number) =>
  height >...
```

### **frontend/src/metabase/visualizations/visualizations/Scalar/utils.unit.spec.ts**

```
import { TITLE_2_LINES_HEIGHT_THRESHOLD } from "../constants";
import { getValueHeight, getValueWidth } from "../utils";
```

```
describe("Scalar > utils", () => {
  describe("getValueHeight", () => {
    ...
```

### **frontend/src/metabase/visualizations/visualizations/ScatterPlot/index.ts**

```
export * from "./ScatterPlot";
```

### **frontend/src/metabase/visualizations/visualizations/SmartScalar/compute.ts**

```
import dayjs from "dayjs";
import { t } from "ttag";
import _ from "underscore";
```

```
import { formatValue } from "metabase/lib/formatting";
import { formatDateTimeRangeWithUnit } from...
```

### **frontend/src/metabase/visualizations/visualizations/SmartScalar/compute.unit.spec.js**

```
import { color, colors } from "metabase/lib/colors";
import { formatValue } from "metabase/lib/formatting/value";
import { computeChange } from "metabase/visualizations/lib/numeric";
import {
  ...
```

### **frontend/src/metabase/visualizations/visualizations/SmartScalar/constants.ts**

```
export const SPACING = 8;
```

```
export const ICON_SIZE = 13;
```

```
export const TOOLTIP_ICON_SIZE = 11;
```

```
export const ICON_MARGIN_RIGHT = SPACING;
```

```
export const SCALAR_TITLE_LINE_HEIGHT = 23;
```

```
export const...
```

### **frontend/src/metabase/visualizations/visualizations/SmartScalar/index.ts**

## Analyst Base

```
export { SmartScalar } from "./SmartScalar";
```

### **frontend/src/metabase/visualizations/visualizations/SmartScalar/tests/test-mocks.ts**

```
import { DateTimeColumn, NumberColumn } from "__support__/visualizations";
import { COMPARISON_TYPES } from "metabase/visualizations/visualizations/SmartScalar/constants";
import type {
  RowValues,
  ...
```

### **frontend/src/metabase/visualizations/visualizations/SmartScalar/types.ts**

```
import type { COMPARISON_TYPES } from "./constants";
```

```
type AnotherColumnMenuOption = {
  type: typeof COMPARISON_TYPES.ANOTHER_COLUMN;
  name: string;
};
type PreviousValueMenuOption = {
  type:...
```

### **frontend/src/metabase/visualizations/visualizations/SmartScalar/utils.ts**

```
import dayjs from "dayjs";
import { t } from "ttag";
import _ from "underscore";
```

```
import { formatNumber } from "metabase/lib/formatting/numbers";
import { measureText } from...
```

### **frontend/src/metabase/visualizations/visualizations/SmartScalar/utils.unit.spec.ts**

```
import { DateTimeColumn, NumberColumn } from "__support__/visualizations";
import * as measureText from "metabase/lib/measure-text";
import type { FontStyle } from...
```

### **frontend/src/metabase/visualizations/visualizations/Table/Table.unit.spec.js**

```
import userEvent from "@testing-library/user-event";
import { thaw } from "icepick";
import { useState } from "react";
```

```
import { createMockMetadata } from "__support__/metadata";
import {...
```

### **frontend/src/metabase/visualizations/visualizations/Table/stories-data/index.ts**

```
import images from "./images.json";
```



## Analyst Base

```
import ordersWithPeople from "./orders-with-people.json";
import variousColumnSettings from "./various-column-settings.json";
import wrappedLinks from...
```

### **frontend/src/metabase/visualizations/visualizations/Text/index.ts**

```
import { t } from "ttag";
```

```
import {
  getDefaultSize,
  getMinSize,
} from "metabase/visualizations/shared/utils/sizes";
```

```
import { Text } from "./Text";
```

```
const TextWrapper = Object.assign(Text, {
  ...
```

### **frontend/src/metabase/visualizations/visualizations/WaterfallChart/index.ts**

```
export * from "./WaterfallChart";
```

### **frontend/src/metabase/visualizer/components/DataImporter/ColumnsList/index.ts**

```
export * from "./ColumnsListItem";
export * from "./ColumnsList";
```

### **frontend/src/metabase/visualizer/components/DataImporter/DatasetsList/getIsCompatible.ts**

```
import type { ComputedVisualizationSettings } from "metabase/visualizations/types";
import { groupColumnsBySuitableVizSettings } from "metabase/visualizer/visualizations/compat";
import type {
  ...
```

### **frontend/src/metabase/visualizer/components/DataImporter/DatasetsList/getIsCompatible.unit.spec.ts**

```
import registerVisualizations from "metabase/visualizations/register";
import type { Field } from "metabase-types/api";
import {
  createMockCategoryColumn,
  createMockDataset,
  ...
```

### **frontend/src/metabase/visualizer/components/DataImporter/index.ts**

```
export * from "./DataImporter";
```

### **frontend/src/metabase/visualizer/components/DragOverlay/index.ts**

## Analyst Base

```
export * from "./DragOverlay";
```

**frontend/src/metabase/visualizer/components/Footer/index.ts**

```
export { Footer } from "./Footer";
```

**frontend/src/metabase/visualizer/components/Header/index.ts**

```
export * from "./Header";
```

**frontend/src/metabase/visualizer/components/TabularPreviewModal/index.ts**

```
export * from "./TabularPreviewModal";
```

**frontend/src/metabase/visualizer/components/VisualizationCanvas/index.ts**

```
export * from "./VisualizationCanvas";
```

**frontend/src/metabase/visualizer/components/VisualizationCanvas/wells/HorizontalWell/index.ts**

```
export * from "./HorizontalWell";
```

**frontend/src/metabase/visualizer/components/VisualizationCanvas/wells/ScatterFloatingWell/index.ts**

```
export * from "./ScatterFloatingWell";
```

**frontend/src/metabase/visualizer/components/VisualizationCanvas/wells/VerticalWell/index.ts**

```
export * from "./VerticalWell";
```

**frontend/src/metabase/visualizer/components/VisualizationPicker/index.ts**

```
export * from "./VisualizationPicker";
```

**frontend/src/metabase/visualizer/components/Visualizer/index.ts**

```
export * from "./Visualizer";
```

**frontend/src/metabase/visualizer/components/VisualizerModal/index.ts**

```
export * from "./VisualizerModal";
```

**frontend/src/metabase/visualizer/components/VisualizerUiContext/index.ts**

```
export * from "./VisualizerUiContext";
```

**frontend/src/metabase/visualizer/constants.ts**

```
export const DRAGGABLE_ID = {  
  COLUMN: "COLUMN",  
  WELL_ITEM: "WELL_ITEM",  
};
```

```
export const DROPPABLE_ID = {  
  CANVAS_MAIN: "CANVAS_MAIN",
```

## Analyst Base

```
X_AXIS_WELL: "X_AXIS_WELL",  
Y_AXIS_WELL:...
```

### **frontend/src/metabase/visualizer/hooks/use-boolean-map.ts**

```
import { useCallback, useState } from "react";
```

```
/**  
 * A hook to manage one boolean state per key in an object.  
 */  
export const useBooleanMap = () => {  
  const [values, setValues] =...
```

### **frontend/src/metabase/visualizer/hooks/use-can-handle-active-item.ts**

```
import type { Active } from "@dnd-kit/core";  
import { useMemo } from "react";
```

```
import { useSelector } from "metabase/lib/redux";  
import {  
  getHoveredItems,  
  getReferencedColumns,  
} from...
```

### **frontend/src/metabase/visualizer/hooks/use-is-card-pristine.ts**

```
import { useMemo } from "react";  
import _ from "underscore";  
  
import { useSelector } from "metabase/lib/redux";  
import type { VisualizerDataSource } from "metabase-types/api";
```

```
import {  
  getCards,  
  ...
```

### **frontend/src/metabase/visualizer/hooks/use-visualizer-history.ts**

```
import { useDispatch, useSelector } from "metabase/lib/redux";  
  
import { getCanRedo, getCanUndo } from "../selectors";  
import { redo, undo } from "../visualizer.slice";
```

```
export function...
```

### **frontend/src/metabase/visualizer/selectors.ts**

```
import { createSelector } from "@reduxjs/toolkit";
```

## Analyst Base

```
import _ from "underscore";
```

```
import {  
  extractRemappings,  
  getVisualization,  
  getVisualizationTransformed,  
  isCartesianChart,  
} from...
```

### **frontend/src/metabase/visualizer/utils/click-actions.ts**

```
import type { ClickObject } from "metabase/visualizations/types";  
import type {  
  DatasetColumn,  
  RawSeries,  
  VisualizerColumnValueSource,  
} from "metabase-types/api";
```

```
import {...
```

### **frontend/src/metabase/visualizer/utils/column.ts**

```
import _ from "underscore";
```

```
import { isPivotGroupColumn } from "metabase/lib/data_grid";  
import { isDate, isDimension, isMetric } from "metabase-lib/v1/types/utils/isa";  
import type {  
  ...
```

### **frontend/src/metabase/visualizer/utils/dashboard-card-supports-visualizer.ts**

```
import visualizations from "metabase/visualizations";  
import type { DashboardCard, VisualizationDisplay } from "metabase-types/api";
```

```
import { isVisualizerDashboardCard } from...
```

### **frontend/src/metabase/visualizer/utils/data-source.ts**

```
import type {  
  VisualizerColumnValueSource,  
  VisualizerDataSource,  
  VisualizerDataSourceId,  
  VisualizerDataSourceNameReference,  
  VisualizerDataSourceType,  
} from "metabase-types/api";
```

```
import {...
```

### **frontend/src/metabase/visualizer/utils/drag-and-drop.ts**

## Analyst Base

```
import type { Active } from "@dnd-kit/core";
```

```
import type {  
  DraggedColumn,  
  DraggedItem,  
  DraggedWellItem,  
} from "metabase-types/store/visualizer";
```

```
import { DRAGGABLE_ID } from...
```

### **frontend/src/metabase/visualizer/utils/get-initial-state-for-card-data-source.ts**

```
import { isPivotGroupColumn } from "metabase/lib/data_grid";  
import { isNotNull } from "metabase/lib/types";  
import { isCartesianChart } from "metabase/visualizations";  
import {...
```

### **frontend/src/metabase/visualizer/utils/get-initial-state-for-card-data-source.unit.spec.ts**

```
import registerVisualizations from "metabase/visualizations/register";  
import type { CardDisplayType } from "metabase-types/api";  
import {  
  createMockCard,  
  createMockColumn,  
  ...
```

### **frontend/src/metabase/visualizer/utils/get-initial-state-for-multiple-series.ts**

```
import { isNotNull } from "metabase/lib/types";  
import type {  
  Card,  
  Dataset,  
  DatasetColumn,  
  RawSeries,  
  VisualizerColumnReference,  
  VisualizerDataSource,  
} from "metabase-types/api";  
import...
```

### **frontend/src/metabase/visualizer/utils/get-initial-state-for-multiple-series.unit.spec.ts**

```
import { registerVisualization } from "metabase/visualizations";  
import { LineChart } from "metabase/visualizations/visualizations/LineChart";  
import {  
  createMockCard,  
  createMockColumn,
```

...

### **frontend/src/metabase/visualizer/utils/get-initial-state-for-visualizer-card.ts**

```
import type {
  Dataset,
  VisualizerDashboardCard,
  VisualizerDataSourceId,
} from "metabase-types/api";

import { createDataSource } from "../data-source";
import { getVisualizationColumns } from...
```

### **frontend/src/metabase/visualizer/utils/get-updated-settings-for-display.ts**

```
import _ from "underscore";

import { isNotNull } from "metabase/lib/types";
import {
  getMaxDimensionsSupported,
  isCartesianChart,
} from "metabase/visualizations";
import type {
  ...
```

### **frontend/src/metabase/visualizer/utils/get-updated-settings-for-display.unit.spec.ts**

```
import registerVisualizations from "metabase/visualizations/register";
import type { VisualizerColumnValueSource } from "metabase-types/api";
import {
  createMockCategoryColumn,
  ...
```

### **frontend/src/metabase/visualizer/utils/get-visualization-columns.ts**

```
import type {
  Dataset,
  DatasetColumn,
  VisualizerDataSource,
  VisualizerDataSourceId,
  VisualizerVizDefinition,
} from "metabase-types/api";

import {
  createDimensionColumn,
  ...
```

## Analyst Base

### **frontend/src/metabase/visualizer/utils/get-visualization-columns.unit.spec.ts**

```
import type {
  Dataset,
  VisualizerDataSource,
  VisualizerVizDefinition,
} from "metabase-types/api";
import {
  createMockColumn,
  createMockDataset,
  createMockDatasetData,
} from...
```

### **frontend/src/metabase/visualizer/utils/index.ts**

```
export * from "./viz-settings";
export * from "./click-actions";
export * from "./column";
export * from "./dashboard-card-supports-visualizer";
export * from "./data-source";
export * from...
```

### **frontend/src/metabase/visualizer/utils/is-visualizer-dashboard-card.ts**

```
import type {
  BaseDashboardCard,
  VisualizerDashboardCard,
} from "metabase-types/api";

export function isVisualizerDashboardCard(
  dashboard?: BaseDashboardCard,
): dashboard is...
```

### **frontend/src/metabase/visualizer/utils/merge-data.ts**

```
import _ from "underscore";

import type {
  Dataset,
  DatasetColumn,
  RowValues,
  VisualizerColumnValueSource,
  VisualizerDataSource,
  VisualizerDataSourceId,
} from...
```

### **frontend/src/metabase/visualizer/utils/split-series.ts**

## Analyst Base

```
import _ from "underscore";
```

```
import { isNotNull } from "metabase/lib/types";
import { isCartesianChart } from "metabase/visualizations";
import type {
  RawSeries,
  VisualizerColumnValueSource,
}...
```

### **frontend/src/metabase/visualizer/utils/update-viz-settings-with-refs.ts**

```
import {
  getColumnKey,
  getColumnNameFromKey,
} from "metabase-lib/v1/queries/utils/column-key";
import type { VisualizationSettings } from "metabase-types/api";
```

```
/**
 * Recursively converts...
```

### **frontend/src/metabase/visualizer/utils/update-viz-settings-with-refs.unit.spec.ts**

```
import { getColumnKey } from "metabase-lib/v1/queries/utils/column-key";
import type { VisualizationSettings } from "metabase-types/api";

import {
  updateVizSettingsKeysWithRefs,
  ...
```

### **frontend/src/metabase/visualizer/utils/viz-settings.ts**

```
import { getVisualization } from "metabase/visualizations";
import type { VisualizationSettingDefinition } from "metabase/visualizations/types";
import type { VisualizationDisplay } from...
```

### **frontend/src/metabase/visualizer/utils/viz-settings.unit.spec.ts**

```
import { registerVisualization } from "metabase/visualizations";
import { BarChart } from "metabase/visualizations/visualizations/BarChart";
import { Map } from...
```

### **frontend/src/metabase/visualizer/visualizations/cartesian.ts**

```
import type { DragEndEvent } from "@dnd-kit/core";
import type { Draft } from "immer";
import _ from "underscore";
```

```
import { isNotNull } from "metabase/lib/types";
import { isCartesianChart } from...
```



## Analyst Base

### **frontend/src/metabase/visualizer/visualizations/cartesian.unit.spec.ts**

```
import _ from "underscore";
```

```
import {  
  createMockCategoryColumn,  
  createMockColumn,  
  createMockDataset,  
  createMockDatetimeColumn,  
  createMockNumericColumn,  
} from...
```

### **frontend/src/metabase/visualizer/visualizations/compat.ts**

```
import _ from "underscore";
```

```
import { isCartesianChart } from "metabase/visualizations";  
import type { ComputedVisualizationSettings } from "metabase/visualizations/types";  
import type {  
  Dataset,  
  ...
```

### **frontend/src/metabase/visualizer/visualizations/funnel.ts**

```
import type { DragEndEvent } from "@dnd-kit/core";  
import type { Draft } from "immer";  
import _ from "underscore";
```

```
import type { ComputedVisualizationSettings } from...
```

### **frontend/src/metabase/visualizer/visualizations/funnel.unit.spec.ts**

```
import _ from "underscore";
```

```
import {  
  createMockCategoryColumn,  
  createMockDataset,  
  createMockNumericColumn,  
} from "metabase-types/api/mocks";  
import type { VisualizerVizDefinitionWithColumns }...
```

### **frontend/src/metabase/visualizer/visualizations/pie.ts**

```
import type { DragEndEvent } from "@dnd-kit/core";  
import _ from "underscore";
```

```
import { isNotNull } from "metabase/lib/types";  
import type { ComputedVisualizationSettings } from...
```

## Analyst Base

### frontend/src/metabase/visualizer/visualizations/pie.unit.spec.ts

```
import {
  createMockCategoryColumn,
  createMockDataset,
  createMockNumericColumn,
} from "metabase-types/api/mocks";
import type { VisualizerVizDefinitionWithColumns } from...
```

### frontend/src/metabase/visualizer/visualizations/utils.ts

```
import type { VisualizerVizDefinitionWithColumns } from "metabase-types/store/visualizer";
```

```
/**
 * Ensures that the column is removed from the state if it is not used in any settings.
 *
 * @param...
```

### frontend/src/metabase/visualizer/visualizer.slice.ts

```
import type { DragEndEvent } from "@dnd-kit/core";
import {
  type PayloadAction,
  createAction,
  createSlice,
} from "@reduxjs/toolkit";
import { shallowEqual } from "react-redux";
import...
```

### frontend/src/metabase-lib/.eslintrc

```
{
  "rules": {
    // Note: adding this rule to a eslint config file in a subfolder will remove
    // *not* carry over the restricted imports from parent folders, you will
    // need to copy them...
```

### frontend/src/metabase-lib/aggregation.ts

```
import * as ML from "cljs/metabase.lib.js";

import { displayInfo } from "../metadata";
import type {
  Aggregable,
  AggregationClause,
  AggregationOperator,
  ColumnMetadata,
  Query,
```

```
} from...
```

### **frontend/src/metabase-lib/binning.ts**

```
import * as ML from "cljs/metabase.lib.js";
```

```
import { displayInfo } from "./metadata";
```

```
import type { Bucket, Clause, ColumnMetadata, Query } from "./types";
```

```
export function binning(clause: Clause |...
```

### **frontend/src/metabase-lib/breakout.ts**

```
import * as ML from "cljs/metabase.lib.js";
```

```
import { removeClause } from "./query";
```

```
import type { BreakoutClause, ColumnMetadata, Query } from "./types";
```

```
export function breakouttableColumns(  
  ...
```

### **frontend/src/metabase-lib/breakout.unit.spec.ts**

```
import { checkNotNull } from "metabase/lib/types";
```

```
import * as Lib from "metabase-lib";
```

```
import { columnFinder, createQuery, findTemporalBucket } from "./test-helpers";
```

```
describe("breakout", () => {  
  ...
```

### **frontend/src/metabase-lib/buckets.ts**

```
import { binning, isBinnable, withDefaultBinning } from "./binning";
```

```
import {
```

```
  isTemporalBucketable,
```

```
  temporalBucket,
```

```
  withDefaultTemporalBucket,
```

```
} from "./temporal_bucket";
```

```
import type {...
```

### **frontend/src/metabase-lib/column\_types.ts**

```
import * as ML from "cljs/metabase.lib.js";
```

```
import * as TYPES from "cljs/metabase.lib.types.isa";
```

```
import type Field from "metabase-lib/v1/metadata/Field";
```

```
import type { Field as ApiField,...
```

### **frontend/src/metabase-lib/comparison.ts**

```
import * as ML from "cljs/metabase.lib.js";
```

## Analyst Base

```
import type { DatasetQuery, DimensionReference } from "metabase-types/api";

import type { ColumnMetadata, Query } from "./types";

export function...
```

### frontend/src/metabase-lib/comparison.unit.spec.ts

```
import * as Lib from "metabase-lib";

import { createQueryWithClauses } from "./test-helpers";

describe("findColumnIndexesFromLegacyRefs", () => {
  const stageIndex = -1;

  it("should match...
```

### frontend/src/metabase-lib/database.ts

```
import * as ML from "cljs/metabase.lib.js";

import type { Query } from "./types";

/**
 * Get the Database ID (`:database`) associated with a query. If the query is using
 * the Saved Questions...
```

### frontend/src/metabase-lib/drills.ts

```
import * as ML from "cljs/metabase.lib.js";
import type { CardId, DatasetColumn, RowValue } from "metabase-types/api";

import type {
  ClickObjectDataRow,
  ClickObjectDimension,
  ColumnMetadata,
  ...
```

### frontend/src/metabase-lib/expression.ts

```
import * as ML from "cljs/metabase.lib.js";

import type {
  AggregationClause,
  ColumnMetadata,
  ExpressionArg,
  ExpressionClause,
  ExpressionOperator,
```

## Analyst Base

```
ExpressionOptions,  
ExpressionParts,  
...
```

### **frontend/src/metabase-lib/extractions.ts**

```
import * as ML from "cljs/metabase.lib.js";  
  
import { expressionParts } from "../expression";  
import type {  
  ColumnExtraction,  
  ColumnMetadata,  
  DrillThru,  
  ExpressionArg,  
  ExpressionClause,  
  ...
```

### **frontend/src/metabase-lib/fields.ts**

```
import * as ML from "cljs/metabase.lib.js";  
import type { FieldReference } from "metabase-types/api";  
  
import type {  
  Clause,  
  ColumnMetadata,  
  FieldValuesSearchInfo,  
  MetricMetadata,  
  Query,  
  ...
```

### **frontend/src/metabase-lib/filter.ts**

```
import dayjs from "dayjs";  
import type { Moment } from "moment-timezone"; // eslint-disable-line no-restricted-imports -- deprecated  
usage  
  
import * as ML from "cljs/metabase.lib.js";  
import type { ...
```

### **frontend/src/metabase-lib/index.ts**

Note: only metabase-lib v2 exports should be added here

### **frontend/src/metabase-lib/join.ts**

```
import * as ML from "cljs/metabase.lib.js";  
import type {  
  CardId,  
  ConcreteTableId,
```

## Analyst Base

```
Databaseld,  
VirtualTableId,  
} from "metabase-types/api";
```

```
import type {  
  Bucket,  
  CardMetadata,  
  Clause,  
  ...
```

### **frontend/src/metabase-lib/limit.ts**

```
import * as ML from "cljs/metabase.lib.limit";  
  
import type { Limit, Query } from "./types";  
  
export function currentLimit(query: Query, stageIndex: number): Limit {  
  return ML.current_limit(query,...
```

### **frontend/src/metabase-lib/metadata.ts**

```
import * as ML from "cljs/metabase.lib.js";  
import * as ML_MetadataCalculation from "cljs/metabase.lib.metadata.calculation";  
import type {  
  CardId,  
  CardType,  
  Databaseld,  
  DatasetColumn,  
  ...
```

### **frontend/src/metabase-lib/metadata.unit.spec.ts**

```
import * as Lib from "metabase-lib";  
  
import { SAMPLE_DATABASE, SAMPLE_METADATA } from "./test-helpers";  
  
describe("metadataProvider", () => {  
  // this is a very important optimization that the FE...
```

### **frontend/src/metabase-lib/metrics.ts**

```
import * as ML from "cljs/metabase.lib.js";  
  
import type { MetricMetadata, Query } from "./types";  
  
export function availableMetrics(  
  query: Query,  
  stageIndex: number,
```

## Analyst Base

```
): MetricMetadata[] {
```

```
...
```

### **frontend/src/metabase-lib/native.ts**

```
import * as ML from "cljs/metabase.lib.js";
```

```
import type { DatabaseId, TemplateTags } from "metabase-types/api";
```

```
import type { MetadataProvider, Query } from "../types";
```

```
export function nativeQuery(  
...
```

### **frontend/src/metabase-lib/native.unit.spec.ts**

```
import * as Lib from "metabase-lib";
```

```
import { SAMPLE_DATABASE, SAMPLE_METADATA } from "../test-helpers";
```

```
describe("native query template tags", () => {
```

```
  let metadataProvider:...
```

### **frontend/src/metabase-lib/order\_by.ts**

```
import * as ML from "cljs/metabase.lib.js";
```

```
import { removeClause } from "../query";
```

```
import type {
```

```
  ColumnMetadata,
```

```
  OrderByClause,
```

```
  OrderByDirection,
```

```
  Query,
```

```
} from "../types";
```

```
export function...
```

### **frontend/src/metabase-lib/order\_by.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
```

```
import * as Lib from "metabase-lib";
```

```
import { createMockCard } from "metabase-types/api/mocks";
```

```
import {
```

```
  SAMPLE_DB_ID,
```

```
...
```

### **frontend/src/metabase-lib/query.ts**

```
import * as ML from "cljs/metabase.lib.js";
```

```
import type {
```

## Analyst Base

```
CardId,  
CardType,  
DatabaseId,  
DatasetQuery,  
TableId,  
} from "metabase-types/api";
```

```
import type {  
  CardMetadata,  
  Clause,  
  ...
```

### **frontend/src/metabase-lib/query.unit.spec.ts**

```
import * as Lib from "metabase-lib";
```

```
import {  
  DEFAULT_QUERY,  
  SAMPLE_DATABASE,  
  SAMPLE_METADATA,  
  createQuery,  
} from "../test-helpers";
```

```
describe("fromLegacyQuery", () => {  
  // this is a very...
```

### **frontend/src/metabase-lib/segments.ts**

```
import * as ML from "cljs/metabase.lib.js";  
import type { SegmentId } from "metabase-types/api";
```

```
import type { Query, SegmentMetadata } from "../types";
```

```
export function availableSegments(  
  query:...
```

### **frontend/src/metabase-lib/temporal\_bucket.ts**

```
import * as ML from "cljs/metabase.lib.js";  
import type { TemporalUnit } from "metabase-types/api";
```

```
import { displayInfo } from "../metadata";  
import type { Bucket, Clause, ColumnMetadata, Query }...
```

### **frontend/src/metabase-lib/temporal\_bucket.unit.spec.ts**

```
import { describeTemporalInterval } from "../temporal_bucket";
```



## Analyst Base

```
describe("describeTemporalInterval", () => {  
  it("should return 'Previous 7 days' when include-current is false", () => {  
    const...
```

### frontend/src/metabase-lib/test-helpers.ts

istanbul ignore file

### frontend/src/metabase-lib/types.ts

```
import type { DefinedClauseName } from "metabase/querying/expressions";  
import type {  
  CardId,  
  DatabaseId,  
  DatasetColumn,  
  FieldId,  
  FieldValuesType,  
  RowValue,  
  SchemaId,  
  TableId,  
  ...
```

### frontend/src/metabase-lib/v1/Alert/constants.ts

```
export const ALERT_TYPE_ROWS = "alert-type-rows";  
export const ALERT_TYPE_TIMESERIES_GOAL = "alert-type-timeseries-goal";  
export const ALERT_TYPE_PROGRESS_BAR_GOAL =...
```

### frontend/src/metabase-lib/v1/Alert/index.ts

```
export {  
  ALERT_TYPE_ROWS,  
  ALERT_TYPE_TIMESERIES_GOAL,  
  ALERT_TYPE_PROGRESS_BAR_GOAL,  
} from "./constants";
```

### frontend/src/metabase-lib/v1/Dimension.ts

```
import { t } from "ttag";  
  
import ValidationError, {  
  VALIDATION_ERROR_TYPES,  
} from "metabase-lib/v1/ValidationError";  
import type Metadata from "metabase-lib/v1/metadata/Metadata";  
import type...
```

### frontend/src/metabase-lib/v1/DimensionOptions/DimensionOptions.ts

```
import type { TemplateTagDimension } from "metabase-lib/v1/Dimension";  
  
import type {
```

## Analyst Base

```
DimensionFK,  
DimensionOptionsProps,  
DimensionOptionsSection,  
} from "./types";
```

```
// eslint-disable-next-line...
```

### **frontend/src/metabase-lib/v1/DimensionOptions/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase-lib/v1/DimensionOptions/types.ts**

```
import type { TemplateTagDimension } from "metabase-lib/v1/Dimension";  
import type Field from "metabase-lib/v1/metadata/Field";
```

```
interface DimensionOptionsSectionItem {  
  dimension:...
```

### **frontend/src/metabase-lib/v1/Question.ts**

```
eslint-disable-next-line @typescript-eslint/ban-ts-comment  
@ts-nocheck
```

### **frontend/src/metabase-lib/v1/ValidationError/ValidationError.ts**

```
import type { ErrorType } from "./types";
```

```
// eslint-disable-next-line import/no-default-export -- deprecated usage  
export default class ValidationError extends Error {  
  type?: ErrorType;
```

```
...
```

### **frontend/src/metabase-lib/v1/ValidationError/constants.ts**

```
export const VALIDATION_ERROR_TYPES = {  
  MISSING_TAG_DIMENSION: "MISSING_TAG_DIMENSION",  
} as const;
```

### **frontend/src/metabase-lib/v1/ValidationError/index.ts**

```
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase-lib/v1/ValidationError/types.ts**

```
import type { VALIDATION_ERROR_TYPES } from "./constants";
```

```
export type ErrorType =  
  (typeof VALIDATION_ERROR_TYPES)[keyof typeof VALIDATION_ERROR_TYPES];
```

### **frontend/src/metabase-lib/v1/actions/utils.ts**

## Analyst Base

```
import type Question from "metabase-lib/v1/Question";
import type Database from "metabase-lib/v1/metadata/Database";
import type { WritebackAction } from "metabase-types/api";
```

```
export const...
```

### **frontend/src/metabase-lib/v1/actions/utils.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import type { Database, WritebackAction } from "metabase-types/api";
import {
  createMockDatabase,
  createMockQueryAction,
} from...
```

### **frontend/src/metabase-lib/v1/metadata/Base.ts**

```
eslint-disable-next-line @typescript-eslint/ban-ts-comment
@ts-nocheck
eslint-disable-next-line import/no-default-export -- deprecated usage
```

### **frontend/src/metabase-lib/v1/metadata/Base.unit.spec.ts**

```
eslint-disable-next-line @typescript-eslint/ban-ts-comment
@ts-nocheck
```

### **frontend/src/metabase-lib/v1/metadata/Database.ts**

```
import _ from "underscore";

import { generateSchemaId } from "metabase-lib/v1/metadata/utils/schema";
import type { NativeQuery, NormalizedDatabase } from "metabase-types/api";
```

```
import Question from...
```

### **frontend/src/metabase-lib/v1/metadata/Database.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import NativeQuery from "metabase-lib/v1/queries/NativeQuery";
import type { Database } from "metabase-types/api";
import {...
```

### **frontend/src/metabase-lib/v1/metadata/Field.ts**

```
eslint-disable-next-line @typescript-eslint/ban-ts-comment
@ts-nocheck
```

### **frontend/src/metabase-lib/v1/metadata/Field.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import { checkNotNull } from "metabase/lib/types";
import { TYPE } from "metabase-lib/v1/types/constants";
```

## Analyst Base

```
import type { Database, Field,...
```

### **frontend/src/metabase-lib/v1/metadata/ForeignKey.ts**

```
import type { NormalizedForeignKey } from "metabase-types/api";
```

```
import type Field from "../Field";
```

```
import type Metadata from "../Metadata";
```

```
interface ForeignKey
```

```
  extends Omit<NormalizedForeignKey,...
```

### **frontend/src/metabase-lib/v1/metadata/Metadata.ts**

```
import _ from "underscore";
```

```
import type {
```

```
  CardId,
```

```
  DatabaseId,
```

```
  FieldId,
```

```
  FieldReference,
```

```
  SchemaId,
```

```
  SegmentId,
```

```
  SettingKey,
```

```
  Settings,
```

```
  TableId,
```

```
} from "metabase-types/api";
```

```
import type...
```

### **frontend/src/metabase-lib/v1/metadata/Metadata.unit.spec.ts**

```
import { createMockMetadata } from "___support___/metadata";
```

```
import {
```

```
  createMockDatabase,
```

```
  createMockField,
```

```
  createMockSegment,
```

```
  createMockTable,
```

```
} from...
```

### **frontend/src/metabase-lib/v1/metadata/Schema.ts**

```
import { humanize, titleize } from "metabase/lib/formatting";
```

```
import type { NormalizedSchema } from "metabase-types/api";
```

```
import type Database from "../Database";
```

```
import type Metadata from...
```

### **frontend/src/metabase-lib/v1/metadata/Schema.unit.spec.ts**

## Analyst Base

```
import { createMockMetadata } from "__support__/metadata";
import type { Table } from "metabase-types/api";
import { createMockTable } from "metabase-types/api/mocks";
```

```
const TEST_TABLE =...
```

### **frontend/src/metabase-lib/v1/metadata/Segment.ts**

```
import type { Filter, NormalizedSegment } from "metabase-types/api";
```

```
import type Metadata from "../Metadata";
import type Table from "../Table";
```

```
interface Segment extends Omit<NormalizedSegment,...
```

### **frontend/src/metabase-lib/v1/metadata/Segment.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import type { Segment } from "metabase-types/api";
import { createMockSegment } from "metabase-types/api/mocks";
```

```
interface SetupOpts {
  ...
```

### **frontend/src/metabase-lib/v1/metadata/Table.ts**

```
import _ from "underscore";
```

```
// NOTE: this needs to be imported first due to some cyclical dependency nonsense
```

```
import { singularize } from "metabase/lib/formatting";
import type { NormalizedTable }...
```

### **frontend/src/metabase-lib/v1/metadata/Table.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import {
  createMockField,
  createMockForeignKey,
  createMockTable,
} from "metabase-types/api/mocks";
```

```
const TABLE_ORIGIN_ID = 1;
const...
```

### **frontend/src/metabase-lib/v1/metadata/utils/fields.ts**

```
import { isVirtualCardId } from "metabase-lib/v1/metadata/utils/saved-questions";
import {
  BOOLEAN,
```

## Analyst Base

COORDINATE,  
FOREIGN\_KEY,  
LOCATION,  
NUMBER,  
PRIMARY\_KEY,  
STRING,  
STRING\_LIKE,  
...

### **frontend/src/metabase-lib/v1/metadata/utils/fields.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";  
import type { FieldReference } from "metabase-types/api";  
import { createMockField, createMockTable } from...
```

### **frontend/src/metabase-lib/v1/metadata/utils/models.ts**

```
import * as Lib from "metabase-lib";  
import type Question from "metabase-lib/v1/Question";  
import type Database from "metabase-lib/v1/metadata/Database";  
import type NativeQuery from...
```

### **frontend/src/metabase-lib/v1/metadata/utils/models.unit.spec.js**

```
import _ from "underscore";  
  
import { createMockMetadata } from "__support__/metadata";  
import Question from "metabase-lib/v1/Question";  
import {  
  checkCanBeModel,  
  checkCanRefreshModelCache,  
  ...  
}
```

### **frontend/src/metabase-lib/v1/metadata/utils/saved-questions.js**

```
import { generateSchemaId } from "metabase-lib/v1/metadata/utils/schema";  
  
export const SAVED_QUESTIONS_VIRTUAL_DB_ID = -1337;  
const ROOT_COLLECTION_VIRTUAL_SCHEMA_NAME = "Everything else";  
  
export...
```

### **frontend/src/metabase-lib/v1/metadata/utils/saved-questions.unit.spec.js**

```
import {  
  SAVED_QUESTIONS_VIRTUAL_DB_ID,  
  convertSavedQuestionToVirtualTable,  
  getCollectionVirtualSchemaId,  
  getCollectionVirtualSchemaName,  
}
```

## Analyst Base

```
getQuestionIdFromVirtualTableId,
```

```
...
```

### **frontend/src/metabase-lib/v1/metadata/utils/schema.ts**

```
import type { DatabaseId, SchemaId, SchemaName } from "metabase-types/api";
```

```
export const getSchemaName = (id: string | null | undefined): SchemaId => {  
  return parseSchemaId(id)[1];  
};
```

```
type...
```

### **frontend/src/metabase-lib/v1/metadata/utils/schema.unit.spec.js**

```
import { generateSchemaId, getSchemaName, parseSchemaId } from "./schema";
```

```
const SCHEMA_TEST_CASES = [  
  { dbId: 1, schemaName: 2, schema: "1:2" },  
  { dbId: 1, schemaName: "2", schema: "1:2" },  
  {...
```

### **frontend/src/metabase-lib/v1/operators/constants.js**

```
import { t } from "ttag";
```

```
import {  
  BOOLEAN,  
  COORDINATE,  
  FOREIGN_KEY,  
  LOCATION,  
  NUMBER,  
  PRIMARY_KEY,  
  STRING,  
  STRING_LIKE,  
  TEMPORAL,  
  TYPE,  
  UNKNOWN,  
} from...
```

### **frontend/src/metabase-lib/v1/operators/utils/index.js**

```
import _ from "underscore";
```

```
import {  
  FIELD_FILTER_OPERATORS,  
  FILTER_OPERATORS_BY_TYPE_ORDERED,  
} from "metabase-lib/v1/operators/constants";
```

## Analyst Base

```
export function doesOperatorExist(operatorName) {  
  ...
```

### **frontend/src/metabase-lib/v1/operators/utils/index.unit.spec.js**

```
import {  
  doesOperatorExist,  
  getOperatorByTypeAndName,  
  isEqualOperator,  
  isFuzzyOperator,  
} from "metabase-lib/v1/operators/utils/index";  
import {  
  COORDINATE,  
  FOREIGN_KEY,  
  NUMBER,  
  ...
```

### **frontend/src/metabase-lib/v1/parameters/constants.ts**

```
import { t } from "ttag";  
  
export const PARAMETER_OPERATOR_TYPES = {  
  number: [  
    {  
      type: "number/=",  
      operator: "=",  
      get name() {  
        return t`Number`;  
      },  
      get...
```

### **frontend/src/metabase-lib/v1/parameters/mock.ts**

```
import type { UiParameter } from "metabase-lib/v1/parameters/types";  
  
export const createMockUiParameter = (  
  opts?: Partial<UiParameter>,  
) : UiParameter => ({  
  id: "parameter-id",  
  slug: "slug",  
  ...
```

### **frontend/src/metabase-lib/v1/parameters/types.ts**

```
import type Field from "metabase-lib/v1/metadata/Field";  
import type { Parameter, ParameterTarget } from "metabase-types/api";
```



## Analyst Base

interface ValuePopulatedParameter extends...

### **frontend/src/metabase-lib/v1/parameters/utils/cards.ts**

```
import Question from "metabase-lib/v1/Question";
import type Metadata from "metabase-lib/v1/metadata/Metadata";
import type {
  ParameterWithTarget,
  UiParameter,
} from...
```

### **frontend/src/metabase-lib/v1/parameters/utils/click-behavior.ts**

```
import _ from "underscore";

import {
  formatDateTimeForParameter,
  formatDateToRangeForParameter,
} from "metabase/lib/formatting/date";
import type { ValueAndColumnForColumnNameDate } from...
```

### **frontend/src/metabase-lib/v1/parameters/utils/click-behavior.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import * as dateFormatUtils from "metabase/lib/formatting/date";
import { checkNotNull } from "metabase/lib/types";
import Question from...
```

### **frontend/src/metabase-lib/v1/parameters/utils/filters.ts**

```
import * as Lib from "metabase-lib";
import type { TemplateTagDimension } from "metabase-lib/v1/Dimension";
import type Field from "metabase-lib/v1/metadata/Field";
import { getParameterOperatorName...
```

### **frontend/src/metabase-lib/v1/parameters/utils/filters.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import { TemplateTagDimension } from "metabase-lib/v1/Dimension";
import Field from "metabase-lib/v1/metadata/Field";
import type...
```

### **frontend/src/metabase-lib/v1/parameters/utils/operators.ts**

```
import {
  doesOperatorExist,
  getOperatorByTypeAndName,
} from "metabase-lib/v1/operators/utils";
import { PARAMETER_OPERATOR_TYPES } from "metabase-lib/v1/parameters/constants";
import {
  ...
```

## Analyst Base

### **frontend/src/metabase-lib/v1/parameters/utils/operators.unit.spec.ts**

```
import { createMockUiParameter } from "metabase-lib/v1/parameters/mock";
```

```
import {  
  deriveFieldOperatorFromParameter,  
  getOperatorDisplayName,  
} from "./operators";
```

```
const option = {  
  type:...
```

### **frontend/src/metabase-lib/v1/parameters/utils/parameter-fields.ts**

```
import type Field from "metabase-lib/v1/metadata/Field";  
import type {  
  FieldFilterUiParameter,  
  UiParameter,  
} from "metabase-lib/v1/parameters/types";
```

```
export const isFieldFilterUiParameter = (  
  ...
```

### **frontend/src/metabase-lib/v1/parameters/utils/parameter-fields.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";  
import type Field from "metabase-lib/v1/metadata/Field";  
import { createMockUiParameter } from "metabase-lib/v1/parameters/mock";  
import {...
```

### **frontend/src/metabase-lib/v1/parameters/utils/parameter-parsing.ts**

```
import type { Query } from "history";
```

```
import {  
  normalizeBooleanParameterValue,  
  normalizeDateParameterValue,  
  normalizeNumberParameterValue,  
  normalizeStringParameterValue,  
  ...
```

### **frontend/src/metabase-lib/v1/parameters/utils/parameter-parsing.unit.spec.js**

```
import { createMockParameter } from "metabase-types/api/mocks";
```

```
import {  
  getParameterValueFromQueryParams,  
  getParameterValuesByIdFromQueryParams,
```

## Analyst Base

```
} from...
```

### **frontend/src/metabase-lib/v1/parameters/utils/parameter-source.ts**

```
import type Field from "metabase-lib/v1/metadata/Field";
import { isFuzzyOperator } from "metabase-lib/v1/operators/utils";
import type {
  Parameter,
  ValuesQueryType,
  ValuesSourceConfig,
  ...
```

### **frontend/src/metabase-lib/v1/parameters/utils/parameter-source.unit.spec.ts**

```
import { createMockMetadata } from "___support___/metadata";
import type Field from "metabase-lib/v1/metadata/Field";
import { createMockUiParameter } from "metabase-lib/v1/parameters/mock";
import {...
```

### **frontend/src/metabase-lib/v1/parameters/utils/parameter-type.ts**

```
import _ from "underscore";

import { FIELD_FILTER_PARAMETER_TYPES } from "metabase-lib/v1/parameters/constants";
import type { Parameter, ParameterType } from "metabase-types/api";

export function...
```

### **frontend/src/metabase-lib/v1/parameters/utils/parameter-type.unit.spec.js**

```
import {
  getParameterSubType,
  getParameterType,
} from "metabase-lib/v1/parameters/utils/parameter-type";

describe("parameters/utils/parameter-type", () => {
  describe("getParameterType", () =>...
```

### **frontend/src/metabase-lib/v1/parameters/utils/parameter-values.js**

```
import _ from "underscore";

import {
  getQueryType,
  getSourceConfig,
  getSourceType,
} from "../parameter-source";
import { getParameterType } from "../parameter-type";
```

## Analyst Base

export const...

### **frontend/src/metabase-lib/v1/parameters/utils/parameter-values.unit.spec.js**

```
import {
  getParameterValue,
  getParameterValuesBySlug,
  getValuePopulatedParameters,
  normalizeParameterValue,
} from "metabase-lib/v1/parameters/utils/parameter-values";
import {...
```

### **frontend/src/metabase-lib/v1/parameters/utils/targets.ts**

```
import _ from "underscore";

import { isNotNull } from "metabase/lib/types";
import * as Lib from "metabase-lib";
import type { TemplateTagDimension } from "metabase-lib/v1/Dimension";
import type...
```

### **frontend/src/metabase-lib/v1/parameters/utils/targets.unit.spec.ts**

```
import { createMockMetadata } from "___support___/metadata";
import { checkNotNull } from "metabase/lib/types";
import * as Lib from "metabase-lib";
import {
  columnFinder,
  createQuery,
  ...
```

### **frontend/src/metabase-lib/v1/parameters/utils/template-tag-options.ts**

```
import { t } from "ttag";

import type Field from "metabase-lib/v1/metadata/Field";
import type { ParameterOptions, TemplateTag } from "metabase-types/api";
import { createMockParameter } from...
```

### **frontend/src/metabase-lib/v1/parameters/utils/template-tag-options.unit.spec.js**

```
import _ from "underscore";

import { PARAMETER_OPERATOR_TYPES } from "metabase-lib/v1/parameters/constants";

import {
  getParameterOptions,
  getParameterOptionsForField,
} from...
```

## Analyst Base

### **frontend/src/metabase-lib/v1/parameters/utils/template-tags.ts**

```
import _ from "underscore";

import type { ParameterWithTarget } from "metabase-lib/v1/parameters/types";
import { getTemplateTagFromTarget } from "metabase-lib/v1/parameters/utils/targets";
import...
```

### **frontend/src/metabase-lib/v1/parameters/utils/template-tags.unit.spec.js**

```
import {
  getTemplateTagParameters,
  getTemplateTags,
  remapParameterValuesToTemplateTags,
} from "metabase-lib/v1/parameters/utils/template-tags";
import { createMockTemplateTag } from...
```

### **frontend/src/metabase-lib/v1/queries/InternalQuery.ts**

```
import type { DatasetQuery } from "metabase-types/api";

export class InternalQuery {
  static isDatasetQueryType(datasetQuery: DatasetQuery) {
    // eslint-disable-next-line...
```

### **frontend/src/metabase-lib/v1/queries/NativeQuery.ts**

```
eslint-disable-next-line @typescript-eslint/ban-ts-comment
@ts-nocheck
```

### **frontend/src/metabase-lib/v1/queries/StructuredQuery.ts**

```
export const STRUCTURED_QUERY_TEMPLATE = {
  database: null,
  type: "query",
  query: {
    "source-table": null,
  },
};
```

### **frontend/src/metabase-lib/v1/queries/drills/dashboard-click-drill.js**

```
import { getIn } from "icepick";
import querystring from "querystring";
import _ from "underscore";

import { renderLinkURLForClick } from "metabase/lib/formatting/link";
import * as Urls from...
```

### **frontend/src/metabase-lib/v1/queries/drills/native-drill-fallback.ts**

## Analyst Base

```
import * as Lib from "metabase-lib";
import type Question from "metabase-lib/v1/Question";
```

```
interface FallbackNativeDrillProps {
  question: Question;
}
```

```
export function nativeDrillFallback({...
```

### **frontend/src/metabase-lib/v1/queries/drills/types.ts**

```
import type { ClickObject } from "metabase-lib";
import type Question from "metabase-lib/v1/Question";
import type { VisualizationSettings } from "metabase-types/api";
```

```
export type {
  ClickObject,
  ...
```

### **frontend/src/metabase-lib/v1/queries/utils/card.js**

```
import { updateIn } from "icepick";
import _ from "underscore";
```

```
import { copy } from "metabase/lib/utils";
import * as Lib from "metabase-lib";
import { deriveFieldOperatorFromParameter } from...
```

### **frontend/src/metabase-lib/v1/queries/utils/column-key.ts**

```
import {
  createFieldReference,
  getBaseDimensionReference,
  getNormalizedDimensionReference,
  hasStringFieldName,
  isAggregationReference,
  isExpressionReference,
  isFieldReference,
  ...
```

### **frontend/src/metabase-lib/v1/queries/utils/column-key.unit.spec.ts**

```
import {
  getColumnKey,
  getColumnNameFromKey,
  getColumnSettings,
  getLegacyColumnKey,
  getObjectColumnSettings,
```

## Analyst Base

```
} from "metabase-lib/v1/queries/utils/column-key";
import type { DatasetColumn }...
```

### **frontend/src/metabase-lib/v1/queries/utils/dataset.ts**

```
import type { DatasetData, TableColumnOrderSetting } from "metabase-types/api";
```

```
import type { DatasetColumnReference } from "../column-key";
```

```
export const datasetContainsNoResults = (data:...
```

### **frontend/src/metabase-lib/v1/queries/utils/dataset.unit.spec.ts**

```
import {
  createMockColumn,
  createMockTableColumnOrderSetting,
} from "metabase-types/api/mocks";
import { ORDERS } from "metabase-types/api/mocks/presets";
```

```
import {
```

```
...
```

### **frontend/src/metabase-lib/v1/queries/utils/expression.js**

```
import { unique_expression_name } from "cljs/metabase.xrays.domain_entities.queries.util";
```

```
/**
```

```
 * Ensures expression's name uniqueness
```

```
 *
```

```
 * Example: if query has a "Double Total" expression,
```

```
 * and...
```

### **frontend/src/metabase-lib/v1/queries/utils/field.js**

```
import _ from "underscore";
```

```
// Metadata field "values" type is inconsistent
```

```
// https://github.com/metabase/metabase/issues/3417
```

```
export function getFieldValues(field) {
```

```
  const values = field &&...
```

### **frontend/src/metabase-lib/v1/queries/utils/index.ts**

The backend won't return more than 2,000 rows so in cases where we need to communicate or use that, use this constant

### **frontend/src/metabase-lib/v1/queries/utils/native-query-table.ts**

```
import _ from "underscore";
```

```
import type Table from "metabase-lib/v1/metadata/Table";
```

## Analyst Base

```
import { getQuestionVirtualTableId } from "metabase-lib/v1/metadata/utils/saved-questions";
```

```
import type...
```

### **frontend/src/metabase-lib/v1/queries/utils/native-query-table.unit.spec.ts**

```
import { createMockMetadata } from "___support___/metadata";
```

```
import type Question from "metabase-lib/v1/Question";
```

```
import type Table from "metabase-lib/v1/metadata/Table";
```

```
import {...
```

### **frontend/src/metabase-lib/v1/queries/utils/normalize.js**

```
export { normalize } from "cljs/metabase.lib.js";
```

### **frontend/src/metabase-lib/v1/queries/utils/pivot.ts**

```
import _ from "underscore";
```

```
import { isNotNull } from "metabase/lib/types";
```

```
import * as Lib from "metabase-lib";
```

```
import type Question from "metabase-lib/v1/Question";
```

```
import type {
```

```
...
```

### **frontend/src/metabase-lib/v1/queries/utils/range-for-value.ts**

```
import type { DatasetColumn } from "metabase-types/api";
```

```
/**
```

```
 * @returns min and max for a value in a column
```

```
*/
```

```
export const rangeForValue = (value: unknown, column: DatasetColumn) => {
```

```
  if...
```

### **frontend/src/metabase-lib/v1/references.ts**

```
import _ from "underscore";
```

```
import { normalize } from "metabase-lib/v1/queries/utils/normalize";
```

```
import type {
```

```
  AggregateFieldReference,
```

```
  DimensionReference,
```

```
  DimensionReferenceWithOptions,
```

```
...
```

### **frontend/src/metabase-lib/v1/references.unit.spec.ts**

```
import {
```

```
  isAggregationReference,
```

```
  isExpressionReference,
```



## Analyst Base

```
isFieldReference,  
isTemplateTagReference,  
normalizeReferenceOptions,  
} from "metabase-lib/v1/references";
```

```
describe("reference...
```

### **frontend/src/metabase-lib/v1/types/constants.ts**

```
import {  
  LEVEL_ONE_TYPES as cljs_LEVEL_ONE_TYPES,  
  TYPE as cljs_TYPE,  
} from "cljs/metabase.types.core";  
  
export const LEVEL_ONE_TYPES: string[] = cljs_LEVEL_ONE_TYPES;  
  
export const TYPE:...
```

### **frontend/src/metabase-lib/v1/types/utils/isa.js**

```
import { isa as cljs_isa } from "cljs/metabase.types.core";  
import { isVirtualCardId } from "metabase-lib/v1/metadata/utils/saved-questions";  
import {  
  BOOLEAN,  
  COORDINATE,  
  FOREIGN_KEY,  
  ...
```

### **frontend/src/metabase-lib/v1/types/utils/isa.unit.spec.js**

```
import {  
  BOOLEAN,  
  COORDINATE,  
  LOCATION,  
  NUMBER,  
  PRIMARY_KEY,  
  STRING,  
  STRING_LIKE,  
  TEMPORAL,  
  TYPE,  
} from "metabase-lib/v1/types/constants";  
import {  
  getFieldTypes,  
  isDimension,  
  ...
```

### **frontend/src/metabase-lib/v1/urls.ts**

## Analyst Base

```
import { utf8_to_b64url } from "metabase/lib/encoding";
import * as Urls from "metabase/lib/urls";
import * as Lib from "metabase-lib";
import type { ParameterWithTarget } from...
```

### frontend/src/metabase-lib/v1/utils/index.ts

```
export { memoizeClass } from "./memoize-class";
export { sortObject } from "./sort-object";
```

### frontend/src/metabase-lib/v1/utils/memoize-class.ts

```
type Constructor<T> = new (...args: any[]) => T;
```

```
function getWithFallback(
  map: Map<string, any>,
  key: string,
  fallback: () => void,
) {
  if (map.has(key)) {
    return map.get(key);
  } else...
```

### frontend/src/metabase-lib/v1/utils/sort-object.ts

`sortObject` copies objects for deterministic serialization.

Objects that have equal keys and values don't necessarily serialize to the same string. JSON.stringify prints properties in inserted...

### frontend/src/metabase-lib/v1/variables/TemplateTagVariable/TemplateTagVariable.ts

```
import NativeQuery from "metabase-lib/v1/queries/NativeQuery";
import Variable from "metabase-lib/v1/variables/Variable";
import type { TemplateTag, VariableTarget } from...
```

### frontend/src/metabase-lib/v1/variables/TemplateTagVariable/constants.ts

TODO: migrate icons away from metabase-lib  
eslint-disable-next-line no-restricted-imports

### frontend/src/metabase-lib/v1/variables/TemplateTagVariable/index.ts

eslint-disable-next-line import/no-default-export -- deprecated usage

### frontend/src/metabase-lib/v1/variables/Variable/Variable.ts

```
import type Metadata from "metabase-lib/v1/metadata/Metadata";
import type NativeQuery from "metabase-lib/v1/queries/NativeQuery";
```

```
// eslint-disable-next-line import/no-default-export -- deprecated...
```

## Analyst Base

### **frontend/src/metabase-lib/v1/variables/Variable/index.ts**

eslint-disable-next-line import/no-default-export -- deprecated usage

### **frontend/src/metabase-lib/viz/display.ts**

```
import * as Lib from "metabase-lib";
import type {
  CardDisplayType,
  VisualizationSettings,
} from "metabase-types/api";
```

```
type DefaultDisplay = {
  display: CardDisplayType;
  settings?:...
```

### **frontend/src/metabase-lib/viz/display.unit.spec.ts**

```
import { createMockMetadata } from "__support__/metadata";
import type { Field, Table } from "metabase-types/api";
import { createMockField, createMockTable } from "metabase-types/api/mocks";
import...
```

### **frontend/src/metabase-lib/viz/index.ts**

```
export * from "./display";
```

### **frontend/src/metabase-shared/color\_selector.js**

This runs in the Nashorn JavaScript engine and there are some limitations

1. This is not currently automatically built with the rest of the application, please run `yarn build-shared` after...

### **frontend/src/metabase-types/analytics/account.ts**

```
type AccountEventSchema = {
  event: string;
  version?: string | null;
  link?: string | null;
};
```

```
type ValidateEvent<
  T extends AccountEventSchema &
  Record<Exclude<keyof T, keyof...
```

### **frontend/src/metabase-types/analytics/action.ts**

```
type ActionEventSchema = {
  event: string;
  type: string;
```

## Analyst Base

```
action_id: number;
num_parameters?: number | null;
context?: string | null;
};
```

```
type ValidateEvent<
  T extends ActionEventSchema &
  ...
```

### **frontend/src/metabase-types/analytics/browse\_data.ts**

```
type BrowseDataEventSchema = {
  event: string;
  model_id?: number | null;
  table_id?: number | null;
};
```

```
type ValidateEvent<
  T extends BrowseDataEventSchema &
  Record<Exclude<keyof T, keyof...
```

### **frontend/src/metabase-types/analytics/cleanup.ts**

```
type CleanupEventSchema = {
  event: string;
  collection_id?: number | null;
  total_stale_items_found?: number | null;
  total_items_archived?: number | null;
  cutoff_date?: string |...
```

### **frontend/src/metabase-types/analytics/csv-upload.ts**

```
type CsvUploadEventSchema = {
  event: string;
  model_id?: number | null;
  upload_seconds?: number | null;
  size_mb: number | null;
  num_columns: number | null;
  num_rows: number | null;
  ...
```

### **frontend/src/metabase-types/analytics/dashboard.ts**

```
type DashboardEventSchema = {
  event: string;
  dashboard_id: number;
  question_id?: number | null;
  num_tabs?: number | null;
```

## Analyst Base

```
total_num_tabs?: number | null;  
duration_milliseconds?: number |...
```

### **frontend/src/metabase-types/analytics/database.ts**

```
type DatabaseEventSchema = {  
  event: string;  
  database?: string | null;  
  database_id?: number | null;  
  error_type?: string | null;  
  source?: string | null;  
  dbms_version?: string |...
```

### **frontend/src/metabase-types/analytics/downloads.ts**

```
type DownloadsEventSchema = {  
  event: string;  
  resource_type?: string | null;  
  accessed_via?: string | null;  
  export_type?: string | null;  
};
```

```
type ValidateEvent<  
  T extends DownloadsEventSchema...
```

### **frontend/src/metabase-types/analytics/embed-flow.ts**

```
import type { EmbedResourceDownloadOptions } from "metabase/public/lib/types";
```

```
type EmbedFlowParams = {  
  locked?: number;  
  enabled?: number;  
  disabled?: number;  
};
```

```
type EmbedFlowAppearance = {  
  ...
```

### **frontend/src/metabase-types/analytics/embed-share.ts**

```
type EmbedShareEventSchema = {  
  event: string;  
  authorized_origins_set?: boolean | null;  
  number_embedded_questions?: number | null;  
  number_embedded_dashboards?: number | null;  
};
```

```
type...
```

## Analyst Base

### frontend/src/metabase-types/analytics/embedding-homepage.ts

```
type EmbeddingHomepageEventSchema = {
  event: string;
  dismiss_reason?: string | null;
  initial_tab?: string | null;
};

type ValidateEvent<
  T extends EmbeddingHomepageEventSchema &
  ...
```

### frontend/src/metabase-types/analytics/event.ts

```
import type {
  ChecklistItemCTA,
  ChecklistItemValue,
} from "metabase/home/components/Onboarding/types";
import type { KeyboardShortcutId } from "metabase/palette/shortcuts";
import type { Engine,...
```

### frontend/src/metabase-types/analytics/index.ts

```
export * from "./account";
export * from "./action";
export * from "./browse_data";
export * from "./cleanup";
export * from "./csv-upload";
export * from "./dashboard";
export * from...
```

### frontend/src/metabase-types/analytics/invite.ts

```
type InviteEventSchema = {
  event: string;
  invited_user_id: number;
  source?: string | null;
};

type ValidateEvent<
  T extends InviteEventSchema &
  Record<Exclude<keyof T, keyof...
```

### frontend/src/metabase-types/analytics/model.ts

```
type ModelEventSchema = {
  event: string;
  model_id: number;
```

## Analyst Base

```
};
```

```
type ValidateEvent<  
  T extends ModelEventSchema &  
    Record<Exclude<keyof T, keyof ModelEventSchema>, never>,  
> = T;
```

```
export type...
```

### **frontend/src/metabase-types/analytics/question.ts**

```
type QuestionEventSchema = {  
  event: string;  
  question_id: number;  
  type?: string | null;  
  method?: string | null;  
  visualization_type?: string | null;  
  database_id?: number | null;  
  ...
```

### **frontend/src/metabase-types/analytics/schema.ts**

```
import type { AccountEvent } from "./account";  
import type { ActionEvent } from "./action";  
import type { BrowseDataEvent } from "./browse_data";  
import type { CleanupEvent } from "./cleanup";  
import...
```

### **frontend/src/metabase-types/analytics/search.ts**

```
type SearchEventSchema = {  
  event: string;  
  runtime_milliseconds?: number | null;  
  context?: string | null;  
  total_results?: number | null;  
  page_results?: number | null;  
  position?: number |...
```

### **frontend/src/metabase-types/analytics/serialization.ts**

```
type SerializationEventSchema = {  
  event: string;  
  source: string;  
  success: boolean;  
  direction?: string | null;  
  duration_ms?: number | null;  
  error_message?: string | null;  
  count?: number...
```

### **frontend/src/metabase-types/analytics/settings.ts**

```
type SettingsEventSchema = {  
  event: string;  
  source?: string | null;  
};
```

```
type ValidateEvent<  
  T extends SettingsEventSchema &  
    Record<Exclude<keyof T, keyof SettingsEventSchema>, never>,  
> =...
```

### **frontend/src/metabase-types/analytics/setup.ts**

```
type SetupEventSchema = {  
  event: string;  
  version: string;  
  step?: string | null;  
  step_number?: number | null;  
  usage_reason?: string | null;  
  database?: string | null;  
  ...
```

### **frontend/src/metabase-types/analytics/timeline.ts**

```
type TimelineEventSchema = {  
  event: string;  
  source?: string | null;  
  question_id?: number | null;  
  collection_id?: number | null;  
  time_matters?: boolean | null;  
};
```

```
type ValidateEvent<  
  T...
```

### **frontend/src/metabase-types/analytics/upsell.ts**

```
type UpsellEventSchema = {  
  event: string;  
  promoted_feature?: string | null;  
  upsell_location?: string | null;  
};
```

```
type ValidateEvent<  
  T extends UpsellEventSchema &  
    Record<Exclude<keyof T, ...
```



### **frontend/src/metabase-types/api/actions.ts**

```
import type { CardId } from "./card";
import type { DatabaseId } from "./database";
import type { BaseEntityId } from "./entity-id";
import type { Parameter, ParameterId, ParameterTarget } from...
```

### **frontend/src/metabase-types/api/activity.ts**

```
import type { CollectionId } from "./collection";
import type { DashboardId } from "./dashboard";
import type { DatabaseId, InitialSyncStatus } from "./database";
import type { ModerationReviewStatus...
```

### **frontend/src/metabase-types/api/admin.ts**

```
export type ApiKeyId = number;
```

```
export type ApiKey = {
  name: string;
  id: ApiKeyId;
  group: {
    id: number;
    name: string;
  };
  creator_id: number;
  masked_key: string;
  created_at:...
```

### **frontend/src/metabase-types/api/ai-entity-analysis.ts**

```
export interface AIEntityAnalysisResponse {
  summary: string;
}
```

```
export interface TimelineEventInfo {
  name: string;
  description?: string;
  timestamp: string;
}
```

```
export interface...
```

### **frontend/src/metabase-types/api/ai-sql-fixer.ts**

```
import type { DatasetQuery } from "metabase-types/api/query";
```

```
export type FixSqlQueryRequest = {
```

## Analyst Base

```
query: DatasetQuery;  
error_message: string;  
};
```

```
export type FixSqlQueryResponse = {  
  fixes:...
```

### **frontend/src/metabase-types/api/ai-sql-generation.ts**

```
import type { Databaseld } from "metabase-types/api/database";
```

```
export type GenerateSqlQueryRequest = {  
  prompt: string;  
  database_id: Databaseld;  
};
```

```
export type GenerateSqlQueryResponse = {  
  ...
```

### **frontend/src/metabase-types/api/alert.ts**

```
import type { CardId } from "./card";  
import type { CollectionId } from "./collection";  
import type { DashCardId, DashboardId } from "./dashboard";  
import type { BaseEntityId } from...
```

### **frontend/src/metabase-types/api/autocomplete.ts**

```
import type { CardType } from "./card";  
import type { Databaseld } from "./database";
```

```
export type AutocompleteRequest = {  
  databaseld: Databaseld;  
  prefix?: string;  
  substring?:...
```

### **frontend/src/metabase-types/api/automagic-dashboards.ts**

```
import type { Databaseld } from "metabase-types/api/database";
```

```
export type XrayEntityType =  
  | "table"  
  | "segment"  
  | "metric"  
  | "model"  
  | "question"  
  | "adhoc"  
  | "field"
```

|...

### **frontend/src/metabase-types/api/bookmark.ts**

```
import type { CardId, CardType } from "../card";
import type { CollectionId } from "../collection";
import type { DashboardId } from "../dashboard";
```

```
export const BOOKMARK_TYPES = [
  "card",
  ...
```

### **frontend/src/metabase-types/api/bug-report.ts**

```
import type {
  Card,
  Collection,
  Dashboard,
  DatasetData,
  Log,
  MetabaseInfo,
} from "metabase-types/api";
```

```
export type ReportableEntityName =
  | "question"
  | "model"
  | "dashboard"
  | ...
```

### **frontend/src/metabase-types/api/card.ts**

```
import type { EmbeddingParameters } from "metabase/public/lib/types";
import type { PieRow } from "metabase/visualizations/echarts/pie/model/types";
```

```
import type { Collection, CollectionId, ...
```

### **frontend/src/metabase-types/api/click-behavior.ts**

```
import type {
  CardId,
  DashboardId,
  DashboardTabId,
  DimensionTargetOptions,
  FieldReference,
  ParameterId,
  TemplateTagName,
} from "metabase-types/api";
```

## Analyst Base

// Used to set values for question...

### **frontend/src/metabase-types/api/cloud-migration.ts**

```
export type CloudMigrationState =
```

```
  | "init"
  | "setup"
  | "dump"
  | "upload"
  | "cancelled"
  | "error"
  | "done";
```

```
export type CloudMigration = {
```

```
  id: number;
  external_id: string;
  state:...
```

### **frontend/src/metabase-types/api/collection.ts**

```
import type { ColorName } from "metabase/lib/colors/types";
```

```
import type { IconName, IconProps } from "metabase/ui";
```

```
import type {
  BaseEntityId,
  CollectionEssentials,
  Dashboard,
  DashboardId,
  ...
```

### **frontend/src/metabase-types/api/content-translation.ts**

```
export type DictionaryArrayRow = {
```

```
  locale: string;
  msgid: string;
  msgstr: string;
};
```

```
export type DictionaryArray = DictionaryArrayRow[];
```

/\*\* Translations retrieved from the BE have ids...

### **frontend/src/metabase-types/api/dashboard.ts**

```
import type { EmbeddingParameters } from "metabase/public/lib/types";
```

```
import type {
  BaseEntityId,
  CardDisplayType,
  ClickBehavior,
```

## Analyst Base

Collection,  
CollectionAuthorityLevel,  
CollectionId,  
...

### **frontend/src/metabase-types/api/database.ts**

```
import type { ScheduleSettings } from "../settings";  
import type { Table } from "../table";  
  
import type { ISO8601Time, LongTaskStatus } from ".";  
  
export type DatabaseId = number;
```

export type...

### **frontend/src/metabase-types/api/dataset.ts**

```
import type {  
  CacheStrategy,  
  LocalFieldReference,  
  Parameter,  
  ParameterValueOrArray,  
  VisualizerColumnValueSource,  
} from "metabase-types/api";
```

```
import type { Card } from "../card";  
import...
```

### **frontend/src/metabase-types/api/email.ts**

```
export interface EmailSMTPSettings {  
  "email-smtp-host": string;  
  "email-smtp-password": string | null;  
  "email-smtp-port": number | null;  
  "email-smtp-security": "none" | "ssl" | "tls" |...
```

### **frontend/src/metabase-types/api/entity-id.ts**

/\*

BaseEntityId should NOT be created on the frontend - these IDs are generated by the backend when an entity is created. As FE developers we should not be touching or modifying these IDs without any...

### **frontend/src/metabase-types/api/field.ts**

```
import type { RowValue } from "../dataset";  
import type { FieldReference } from "../query";  
import type { Table, TableId } from "../table";
```

## Analyst Base

```
export type FieldId = number;
```

```
export interface...
```

### **frontend/src/metabase-types/api/geojson.ts**

```
import type { Feature, FeatureCollection } from "geojson";
```

```
export type GeoJSONData = Feature | FeatureCollection;
```

### **frontend/src/metabase-types/api/group.ts**

```
import type { User } from "../user";
```

```
export type GroupId = number;
```

```
export type Membership = {  
  user_id: number;  
  group_id: number;  
  membership_id: number;  
  is_group_manager?: boolean;  
};
```

```
export...
```

### **frontend/src/metabase-types/api/index.ts**

```
export * from "../actions";  
export * from "../activity";  
export * from "../admin";  
export * from "../ai-sql-fixer";  
export * from "../ai-sql-generation";  
export * from "../ai-entity-analysis";  
export *...
```

### **frontend/src/metabase-types/api/insight.ts**

```
import type { DateTimeAbsoluteUnit } from "../query";
```

```
export type InsightExpressionOperator =  
  | "+"  
  | "-"  
  | "*"  
  | "/"  
  | "log"  
  | "pow"  
  | "exp";
```

```
export type InsightExpressionOperand =...
```

### **frontend/src/metabase-types/api/logger.ts**

```
export type LogLevel =
```

```
| "off"  
| "fatal"  
| "error"  
| "warn"  
| "info"  
| "debug"  
| "trace";
```

```
export type LoggerDurationUnit =
```

```
| "days"  
| "hours"  
| "minutes"  
| "seconds"  
| ...
```

### **frontend/src/metabase-types/api/metabot.ts**

```
import type {
```

```
  CardDisplayType,  
  CardId,  
  CardType,  
  CollectionId,  
  DashboardId,  
  DatasetQuery,  
  PaginationResponse,  
  RowValue,  
  SearchModel,  
} from ".";
```

```
export type MetabotFeedbackType =
```

```
...
```

### **frontend/src/metabase-types/api/mocks/actions.ts**

```
import type {
```

```
  ActionFormSettings,  
  CardId,  
  FieldSettings,  
  PublicWritebackAction,  
  WritebackImplicitQueryAction,  
  WritebackParameter,
```

## Analyst Base

```
WritebackQueryAction,  
} from...
```

### **frontend/src/metabase-types/api/mocks/activity.ts**

```
import type {  
  PopularItem,  
  RecentCollectionItem,  
  RecentTableDatabaseInfo,  
  RecentTableItem,  
} from "metabase-types/api";
```

```
export const createMockRecentTableItem = (  
  opts?:....
```

### **frontend/src/metabase-types/api/mocks/ai-sql-fixer.ts**

```
import type {  
  FixSqlQueryRequest,  
  FixSqlQueryResponse,  
  SqlQueryFix,  
} from "metabase-types/api";
```

```
import { createMockNativeDatasetQuery } from "../query";
```

```
export function...
```

### **frontend/src/metabase-types/api/mocks/ai-sql-generation.ts**

```
import type {  
  GenerateSqlQueryRequest,  
  GenerateSqlQueryResponse,  
} from "metabase-types/api";
```

```
export const createMockGenerateSqlQueryRequest = (  
  opts?: Partial<GenerateSqlQueryRequest>,  
):...
```

### **frontend/src/metabase-types/api/mocks/alert.ts**

```
import type { Alert, AlertCard } from "../alert";  
import type { Channel } from "../notification-channels";
```

```
import { createMockEntityId } from "../entity-id";  
import { createMockUserInfo } from...
```

### **frontend/src/metabase-types/api/mocks/automagic-dashboards.ts**

```
import type { DatabaseXray, TableXray } from "metabase-types/api";
```



## Analyst Base

```
export const createMockDatabaseCandidate = (  
  opts?: Partial<DatabaseXray>,  
) : DatabaseXray => ({  
  id: "1/public",  
  schema:...
```

### **frontend/src/metabase-types/api/mocks/bookmark.ts**

```
import type { Bookmark } from "metabase-types/api";
```

```
export const createMockBookmark = (opts?: Partial<Bookmark>): Bookmark => ({  
  id: "collection-1",  
  name: "My Collection",  
  item_id: 1,  
  type:...
```

### **frontend/src/metabase-types/api/mocks/card.ts**

```
import type {  
  Card,  
  CardQueryMetadata,  
  ColumnRangeFormattingSetting,  
  ColumnSingleFormattingSetting,  
  ModerationReview,  
  NativeDatasetQuery,  
  PublicCard,  
  SeriesOrderSetting,  
  ...
```

### **frontend/src/metabase-types/api/mocks/channel.ts**

```
import type {  
  ChannelDetails,  
  NotificationChannel,  
} from "metabase-types/api/notification-channels";
```

```
export const createMockChannelDetails = (  
  opts: Partial<ChannelDetails>,  
) : ChannelDetails...
```

### **frontend/src/metabase-types/api/mocks/collection.ts**

```
import type {  
  BaseEntityId,  
  Collection,  
  CollectionEssentials,  
  CollectionItem,
```

## Analyst Base

```
} from "metabase-types/api";
```

```
import { createMockEntityId } from "../entity-id";
```

```
export const...
```

### **frontend/src/metabase-types/api/mocks/dashboard.ts**

```
import type {  
  ActionDashboardCard,  
  Dashboard,  
  DashboardQueryMetadata,  
  DashboardTab,  
  QuestionDashboardCard,  
  VirtualCard,  
  VirtualDashboardCard,  
} from "metabase-types/api";
```

```
import {...
```

### **frontend/src/metabase-types/api/mocks/database.ts**

```
import type {  
  Database,  
  DatabaseData,  
  DatabaseFeature,  
  SavedQuestionDatabase,  
} from "metabase-types/api";
```

```
export const COMMON_DATABASE_FEATURES: DatabaseFeature[] = [  
  "actions",  
  ...
```

### **frontend/src/metabase-types/api/mocks/dataset.ts**

```
import type {  
  Dataset,  
  DatasetColumn,  
  DatasetData,  
  EmbedDataset,  
  EmbedDatasetData,  
  ErrorEmbedDataset,  
  ResultsMetadata,  
  TemplateTag,  
} from "metabase-types/api/dataset";
```

```
export const...
```

## Analyst Base

### frontend/src/metabase-types/api/mocks/entity-id.ts

```
import { nanoid } from "@reduxjs/toolkit";

import { type BaseEntityId, NANOID_LENGTH } from "../entity-id";

export const createMockEntityId = (value?: string): BaseEntityId =>
  (value || ...
```

### frontend/src/metabase-types/api/mocks/field.ts

```
import type {
  DateTimeFieldFingerprint,
  Field,
  FieldDimension,
  FieldFingerprint,
  FieldGlobalFingerprint,
  GetFieldValuesResponse,
  NumberFieldFingerprint,
  TextFieldFingerprint,
} from ...
```

### frontend/src/metabase-types/api/mocks/geojson.ts

```
import type { FeatureCollection } from "geojson";

export const createMockGeoJSONFeatureCollection = (
  opts?: Partial<FeatureCollection>,
): FeatureCollection => {
  return {
    type: ...
```

### frontend/src/metabase-types/api/mocks/group.ts

```
import type { Group } from "metabase-types/api";

export const createMockGroup = (
  opts?: Partial<Group>,
): Omit<Group, "members"> => ({
  id: 1,
  name: "All Users",
  member_count: 1,
  ...
```

### frontend/src/metabase-types/api/mocks/index.ts

```
export * from "../actions";
export * from "../activity";
```

## Analyst Base

```
export * from "./ai-sql-fixer";
export * from "./ai-sql-generation";
export * from "./alert";
export * from "./automagic-dashboards";
export *...
```

### frontend/src/metabase-types/api/mocks/logger.ts

```
import type { LoggerPreset } from "../logger";
```

```
export function createMockLoggerPreset(
  preset?: Partial<LoggerPreset>,
): LoggerPreset {
  return {
    display_name: "Logger preset",
    id: "1",
    ...
  }
}
```

### frontend/src/metabase-types/api/mocks/modelIndexes.ts

```
import type { ModelIndex } from "metabase-types/api";
```

```
export const createMockModelIndex = (
  opts?: Partial<ModelIndex>,
): ModelIndex => ({
  id: 1,
  model_id: 1,
  pk_ref: ["field", 1, null],
  ...
})
```

### frontend/src/metabase-types/api/mocks/models.ts

```
import type { ModelCacheRefreshStatus } from "metabase-types/api";
```

```
export const getMockModelCacheInfo = (
  opts?: Partial<ModelCacheRefreshStatus>,
): ModelCacheRefreshStatus => {
  const now = new...
```

### frontend/src/metabase-types/api/mocks/notification.ts

```
import type {
  Notification,
  NotificationCronSubscription,
  NotificationHandlerEmail,
  NotificationHandlerSlack,
  NotificationRecipientUser,
} from "metabase-types/api";
```

## Analyst Base

```
import {...
```

### **frontend/src/metabase-types/api/mocks/parameters.ts**

```
import type {
  Parameter,
  ParameterValues,
  SearchParameterValuesRequest,
} from "metabase-types/api";

export const createMockParameter = (opts?: Partial<Parameter>): Parameter => ({
  id: "1",
  ...
```

### **frontend/src/metabase-types/api/mocks/performance.ts**

```
import { type CacheConfig, CacheDurationUnit } from "metabase-types/api";

export const createMockCacheConfig = (
  opts?: Partial<CacheConfig>,
): CacheConfig => ({
  model: "database",
  model_id:...
```

### **frontend/src/metabase-types/api/mocks/permissions.ts**

```
import {
  DataPermission,
  DataPermissionValue,
} from "metabase/admin/permissions/types";
import type {
  Database,
  Group,
  GroupsPermissions,
  Impersonation,
  PermissionsGraph,
} from...
```

### **frontend/src/metabase-types/api/mocks/presets/index.ts**

```
export * from "../sample_cards";
export * from "../sample_database";
```

### **frontend/src/metabase-types/api/mocks/presets/sample\_cards.ts**

```
import type {
  Card,
  NativeDatasetQuery,
  StructuredDatasetQuery,
```

## Analyst Base

```
    UnsavedCard,  
  } from "metabase-types/api";  
import {  
  createMockNativeCard,  
  createMockStructuredCard,  
} from...
```

### **frontend/src/metabase-types/api/mocks/presets/sample\_database.ts**

```
import type {  
  Database,  
  DatasetColumn,  
  Field,  
  FieldDimensionOption,  
  GetFieldValuesResponse,  
  Table,  
} from "metabase-types/api";  
import {  
  createMockColumn,  
  createMockDatabase,  
  ...
```

### **frontend/src/metabase-types/api/mocks/pulse.ts**

```
import type { Pulse } from "metabase-types/api";  
  
export const createMockPulse = (opts?: Partial<Pulse>): Pulse => ({  
  name: "Pulse",  
  cards: [],  
  channels: [],  
  parameters: [],  
  ...opts,  
});
```

### **frontend/src/metabase-types/api/mocks/query.ts**

```
import type {  
  NativeDatasetQuery,  
  NativeQuery,  
  StructuredDatasetQuery,  
  StructuredQuery,  
} from "metabase-types/api";
```

```
export const createMockStructuredQuery = (  
  opts?:....
```

### **frontend/src/metabase-types/api/mocks/revision.ts**

## Analyst Base

```
import type { Revision } from "metabase-types/api";

export const createMockRevision = (opts?: Partial<Revision>): Revision => ({
  model_id: 1,
  id: 1,
  description: "created this",
  message:...
```

### **frontend/src/metabase-types/api/mocks/schema.ts**

```
import type {
  NormalizedCollection,
  NormalizedDatabase,
  NormalizedField,
  NormalizedFieldDimension,
  NormalizedSchema,
  NormalizedSegment,
  NormalizedTable,
  NormalizedTimeline,
} from...
```

### **frontend/src/metabase-types/api/mocks/search.ts**

```
import _ from "underscore";

import type { SearchResponse, SearchResult } from "metabase-types/api";

import { createMockCollection } from "../collection";

export const createMockSearchResult = (
  ...
```

### **frontend/src/metabase-types/api/mocks/segment.ts**

```
import type { Segment } from "metabase-types/api";

import { createMockStructuredQuery } from "../query";

export const createMockSegment = (opts?: Partial<Segment>): Segment => ({
  id: 1,
  name:...
```

### **frontend/src/metabase-types/api/mocks/series.ts**

```
import type { Card } from "../card";
import type { Series, SingleSeries } from "../dataset";

import { createMockCard } from "../card";
```

## Analyst Base

```
import type { MockDatasetOpts } from "../dataset";
import {...
```

### **frontend/src/metabase-types/api/mocks/session.ts**

```
import type { PasswordResetTokenStatus } from "metabase-types/api";
```

```
export const createMockPasswordResetTokenStatus = (
  opts?: Partial<PasswordResetTokenStatus>,
): PasswordResetTokenStatus => ({
  ...
```

### **frontend/src/metabase-types/api/mocks/settings.ts**

```
import dayjs from "dayjs";
```

```
import type {
  Engine,
  EngineField,
  EngineSource,
  EnterpriseSettingKey,
  EnterpriseSettings,
  FontFile,
  SettingDefinition,
  Settings,
  TokenFeatures,
  ...
```

### **frontend/src/metabase-types/api/mocks/slack.ts**

```
import type { SlackSettings } from "metabase-types/api";
```

```
export const createMockSlackSettings = (
  opts?: Partial<SlackSettings>,
): SlackSettings => ({
  "slack-app-token": null,
  ...
```

### **frontend/src/metabase-types/api/mocks/snippets.ts**

```
import type { NativeQuerySnippet } from "metabase-types/api";
```

```
import { createMockEntityId } from "../entity-id";
import { createMockUser } from "../user";
```

```
export const createMockNativeQuerySnippet =...
```

### **frontend/src/metabase-types/api/mocks/store.ts**



## Analyst Base

```
import type { StoreTokenStatus } from "metabase-types/api";
```

```
export const createMockStoreTokenStatus = (
  opts?: Partial<StoreTokenStatus>,
): StoreTokenStatus => ({
  ...opts,
  valid: false,
  ...

```

### frontend/src/metabase-types/api/mocks/table.ts

```
import type { ForeignKey, Schema, Table } from "metabase-types/api";
```

```
export const createMockTable = (opts?: Partial<Table>): Table => {
  return {
    id: 1,
    db_id: 1,
    display_name:...
```

### frontend/src/metabase-types/api/mocks/task.ts

```
import type { Task } from "metabase-types/api";
```

```
export const createMockTask = (task?: Partial<Task>): Task => ({
  id: 1,
  db_id: 1,
  duration: 100,
  started_at:...
```

### frontend/src/metabase-types/api/mocks/timeline.ts

```
import type {
  Timeline,
  TimelineData,
  TimelineEvent,
  TimelineEventData,
} from "../timeline";
```

```
import { createMockUserInfo } from "../user";
```

```
export const createMockTimeline = (opts?:...
```

### frontend/src/metabase-types/api/mocks/undo.ts

```
import type { Undo } from "metabase-types/store/undo";
```

```
export const createMockUndo = (opts?: Partial<Undo>): Undo => ({
  message: "The filter was auto-connected to all questions.",

```

## Analyst Base

actionLabel:...

### **frontend/src/metabase-types/api/mocks/user.ts**

```
import type { User, UserInfo, UserListResult } from "metabase-types/api";
```

```
export const createMockUser = (opts?: Partial<User>): User => ({
  id: 1,
  first_name: "Testy",
  last_name: "Tableton",
  ...
```

### **frontend/src/metabase-types/api/modelIndexes.ts**

```
import type { CardId } from "../card";
import type { FieldReference } from "../query";
```

```
export type NormalizedIndexedEntity = IndexedEntity;
```

```
export type ModelIndex = {
  id: number;
  model_id:...
```

### **frontend/src/metabase-types/api/models.ts**

```
import type {
  BaseUser,
  CardId,
  CollectionAuthorityLevel,
  CollectionId,
  DatabasId,
  UserId,
} from "metabase-types/api";
```

```
export type ModelCacheState =
  | "creating"
  | "refreshing"
  | ...
```

### **frontend/src/metabase-types/api/moderation.ts**

```
import type { BaseUser } from "../user";
```

```
export type VerifyItemRequest = {
  status: "verified" | null;
  moderated_item_id: number;
  moderated_item_type: "card" | "dashboard";
  text?:....
```

## **frontend/src/metabase-types/api/notification-channels.ts**

```
import type {
  ScheduleSettings,
  ScheduleType,
} from "metabase-types/api/settings";
import type { User } from "metabase-types/api/user";
import { isObject } from "metabase-types/guards";

type...
```

## **frontend/src/metabase-types/api/notification.ts**

```
import type { Card, CardId } from "metabase-types/api/card";
import type { Channel } from "metabase-types/api/notification-channels";
import type { PaginationRequest } from...
```

## **frontend/src/metabase-types/api/pagination.ts**

```
export interface PaginationRequest {
  limit?: number | null;
  offset?: number | null;
}
```

```
export interface PaginationResponse {
  limit: number | null;
  offset: number | null;
  total: number;
}
```

## **frontend/src/metabase-types/api/parameters.ts**

```
import type { CardId } from "../card";
import type { RowValue, TemporalUnit } from "../dataset";
import type { FieldReference } from "../query";
```

```
export type StringParameterType =
  | "string/="
  | ...
```

## **frontend/src/metabase-types/api/performance.ts**

\* 'Model' as in 'type of object'

## **frontend/src/metabase-types/api/permissions.ts**

```
import type {
  DataPermission,
  DataPermissionValue,
} from "metabase/admin/permissions/types";
```

## Analyst Base

```
import type {  
  CollectionId,  
  Databaseld,  
  SchemaName,  
  TableId,  
} from...
```

### **frontend/src/metabase-types/api/persist.ts**

```
import type { ModelCacheRefreshStatus } from "../models";  
import type { PaginationRequest, PaginationResponse } from "../pagination";
```

```
export type PersistedInfold = number;
```

```
export type...
```

### **frontend/src/metabase-types/api/pulse.ts**

```
import type { Card } from "../card";  
import type { Channel } from "../notification-channels";
```

```
export type Pulse = {  
  cards: Card[];  
  channels: Channel[];  
  name?: string;  
  parameters?:...
```

### **frontend/src/metabase-types/api/query.ts**

```
import type { UiParameter } from "metabase-lib/v1/parameters/types";  
import type {  
  CardId,  
  Databaseld,  
  FieldId,  
  SegmentId,  
  TableId,  
  TemplateTags,  
} from "metabase-types/api";
```

```
export...
```

### **frontend/src/metabase-types/api/revision.ts**

```
import type { CardId } from "metabase-types/api";
```

```
export interface Revision {  
  id: number;  
  description: string;
```

## Analyst Base

```
message: string | null;
timestamp: string;
is_creation: boolean;
...
```

### frontend/src/metabase-types/api/schema.ts

```
import type { WritebackAction } from "../actions";
import type { Card, CardId } from "../card";
import type { Collection, CollectionId, CollectionItemId } from "../collection";
import type { Dashboard }...
```

### frontend/src/metabase-types/api/search.ts

```
import type { UserId } from "metabase-types/api/user";

import type { CardId } from "../card";
import type { Collection, CollectionId, LastEditInfo } from "../collection";
import type { Dashboard,...
```

### frontend/src/metabase-types/api/segment.ts

```
import type { StructuredQuery } from "../query";
import type { Table, TableId } from "../table";
```

```
export type SegmentId = number;
```

```
export interface Segment {
  id: SegmentId;
  name: string;
  ...
}
```

### frontend/src/metabase-types/api/session.ts

```
export interface PasswordResetTokenStatus {
  valid: boolean;
}
```

### frontend/src/metabase-types/api/settings.ts

```
import type { ReactNode } from "react";
```

```
import type { SdkIframeEmbedSetupSettings } from
"metabase-enterprise/embedding_iframe_sdk_setup/types";
```

```
import type { InputSettingType } from...
```

### frontend/src/metabase-types/api/setup.ts

```
export type UsageReason =
  | "self-service-analytics"
```

```
| "embedding"  
| "both"  
| "not-sure";
```

### **frontend/src/metabase-types/api/slack.ts**

```
export interface SlackSettings {  
  "slack-app-token": string | null;  
  "slack-bug-report-channel": string | null;  
}
```

### **frontend/src/metabase-types/api/snippets.ts**

```
import type { RegularCollectionId } from "../collection";  
import type { BaseEntityId } from "../entity-id";  
import type { UserId, UserInfo } from "../user";
```

```
export type NativeQuerySnippetId = ...
```

### **frontend/src/metabase-types/api/sorting.ts**

```
export enum SortDirection {  
  Asc = "asc",  
  Desc = "desc",  
}
```

```
export type SortingOptions<SortColumn extends string> = {  
  sort_column: SortColumn;  
  sort_direction: SortDirection;  
};
```

### **frontend/src/metabase-types/api/store.ts**

```
export interface StoreTokenStatus {  
  status?: string;  
  valid: boolean;  
  trial: boolean;  
}
```

```
export const supportedFormatTypes = [  
  "string",  
  "integer",  
  "float",  
  "datetime",  
  "currency",  
]...
```

### **frontend/src/metabase-types/api/subscription.ts**

## Analyst Base

```
import type { Card } from "./card";
import type { RegularCollectionId } from "./collection";
import type { DashboardId } from "./dashboard";
import type { BaseEntityId } from "./entity-id";
import...
```

### frontend/src/metabase-types/api/table.ts

```
import type { Card, CardType } from "./card";
import type { Database, DatabaseId, InitialSyncStatus } from "./database";
import type { Field, FieldDimensionOption, FieldId } from "./field";
import...
```

### frontend/src/metabase-types/api/task.ts

```
import type { DatabaseId } from "./database";
import type { PaginationRequest, PaginationResponse } from "./pagination";
import type { SortingOptions } from "./sorting";
```

```
// "unknown" status is only...
```

### frontend/src/metabase-types/api/timeline.ts

```
import type { CardId } from "./card";
import type {
  Collection,
  CollectionId,
  RegularCollectionId,
} from "./collection";
import type { UserInfo } from "./user";
```

```
export type TimelineId =...
```

### frontend/src/metabase-types/api/user.ts

```
import type { CollectionId } from "./collection";
import type { DashboardId } from "./dashboard";
import type { PaginationRequest, PaginationResponse } from "./pagination";
```

```
export type UserId =...
```

### frontend/src/metabase-types/api/util.ts

```
export interface Log {
  timestamp: string;
  process_uuid: string;
  fqns: string;
  msg: string;
  level: string;
```

## Analyst Base

```
exception: any;  
}
```

```
export interface MetabaseInfo {  
  "application-database":...
```

### **frontend/src/metabase-types/api/visualization-settings.ts**

SmartScalar (Trend Chart)

### **frontend/src/metabase-types/api/visualization.ts**

```
export const virtualCardDisplayTypes = [  
  "action",  
  "heading",  
  "link",  
  "placeholder",  
  "text",  
  "iframe",  
] as const;
```

```
export type VirtualCardDisplay = (typeof...
```

### **frontend/src/metabase-types/api/visualizer.ts**

```
import type {  
  VisualizationDisplay,  
  VisualizationSettings,  
} from "metabase-types/api";
```

```
export type VisualizerDataSourceType = "card";  
export type VisualizerDataSourceId =...
```

### **frontend/src/metabase-types/entities/common.ts**

```
import type { IconProps } from "metabase/ui";  
import type { Collection } from "metabase-types/api";
```

```
export type WrappedEntity<Entity> = {  
  getName: () => string;  
  getIcon: () => IconProps;  
  ...
```

### **frontend/src/metabase-types/entities/database.ts**

```
import type { Database } from "metabase-types/api";
```

```
export interface DatabaseEntity extends Database {  
  fetchIdFields: (query: any) => void;
```



```
}
```

### **frontend/src/metabase-types/entities/index.ts**

```
export * from "./table";
export * from "./database";
export * from "./common";
```

### **frontend/src/metabase-types/entities/table.ts**

```
import type { Table } from "metabase-types/api";

export interface TableEntity extends Table {
  updateProperty: (name: string, value: string | number | null) => void;
}
```

### **frontend/src/metabase-types/forms/index.ts**

```
*
```

### **frontend/src/metabase-types/guards/card.ts**

```
import type { Card, UnsavedCard } from "metabase-types/api";

export function isSavedCard(card: Card | UnsavedCard): card is Card {
  return "id" in card && card.id !== null;
}
```

### **frontend/src/metabase-types/guards/click-behavior.ts**

```
import type {
  BaseActionClickBehavior,
  DeleteActionClickBehavior,
  ImplicitActionClickBehavior,
  InsertActionClickBehavior,
  UpdateActionClickBehavior,
} from "metabase-types/api";
```

```
import {...
```

### **frontend/src/metabase-types/guards/common.ts**

```
export const isObject = (
  value: unknown,
): value is Record<string | number | symbol, unknown> => {
  return typeof value === "object" && value !== null;
};
```

### **frontend/src/metabase-types/guards/content-translation.ts**

```
import type { DictionaryArray, DictionaryArrayRow } from "metabase-types/api";
```

## Analyst Base

```
export const isDictionaryArrayRow = (  
  item: unknown,  
) : item is DictionaryArrayRow => {  
  return (  
    typeof item...
```

### **frontend/src/metabase-types/guards/dashboard.ts**

```
import type {  
  LinkEntity,  
  RestrictedLinkEntity,  
} from "metabase-types/api/dashboard";
```

```
export const isRestrictedLinkEntity = (  
  value: LinkEntity,  
) : value is RestrictedLinkEntity =>  
  !! (value...
```

### **frontend/src/metabase-types/guards/date-time.ts**

```
import type {  
  DateTimeAbsoluteUnit,  
  DateTimeRelativeUnit,  
} from "metabase-types/api";  
import {  
  dateTimeAbsoluteUnits,  
  dateTimeRelativeUnits,  
} from "metabase-types/api";
```

```
export const...
```

### **frontend/src/metabase-types/guards/dom.ts**

```
export function isElement(a: any): a is Element {  
  return a instanceof Element;  
}
```

### **frontend/src/metabase-types/guards/forms.ts**

```
import type {  
  CustomFormFieldDefinition,  
  FormFieldDefinition,  
  StandardFormFieldDefinition,  
} from "metabase-types/forms";
```

```
import { isReactComponent } from "../react";
```

```
export function...
```

### **frontend/src/metabase-types/guards/index.ts**

```
export * from "./card";
export * from "./click-behavior";
export * from "./common";
export * from "./dom";
export * from "./settings";
export * from "./forms";
export * from "./parameters";
export *...
```

### **frontend/src/metabase-types/guards/parameters.ts**

```
import type {
  ParameterDimensionTarget,
  ParameterTarget,
  StructuredParameterDimensionTarget,
} from "metabase-types/api";
```

```
export function isDimensionTarget(
  target: ParameterTarget |...
```

### **frontend/src/metabase-types/guards/react.ts**

```
import type * as React from "react";
import * as ReactIs from "react-is";
```

```
// checking to see if the `element` is in JSX.IntrinsicElements since they support refs
// tippy's `children` prop seems to...
```

### **frontend/src/metabase-types/guards/readme.md**

Type Guards

### **frontend/src/metabase-types/guards/settings.ts**

```
import { type EngineKey, engineKeys } from "metabase-types/api";
```

```
export function isEngineKey(value: string | undefined): value is EngineKey {
  return engineKeys.includes(value as EngineKey);
}
```

### **frontend/src/metabase-types/store/admin.ts**

```
import type {
  CollectionPermissions,
  Databaseld,
  GroupsPermissions,
  SettingDefinition,
} from "metabase-types/api";
```

## Analyst Base

```
export type AdminPathKey =  
  | "data-model"  
  | "settings"  
  | "metabot"  
  ...
```

### **frontend/src/metabase-types/store/app.ts**

```
import type { ChecklistItemValue } from "metabase/home/components/Onboarding/types";  
import type { CollectionId } from "metabase-types/api/collection";
```

```
export interface AppErrorDescriptor {  
  ...
```

### **frontend/src/metabase-types/store/auth.ts**

```
export interface AuthState {  
  loginPending: boolean;  
  redirect: boolean;  
}
```

### **frontend/src/metabase-types/store/collection.ts**

```
import type { IconName } from "metabase/ui";  
import type { Collection, CollectionId } from "metabase-types/api";
```

```
// see entities/collections/getExpandedCollectionsById.js  
export type...
```

### **frontend/src/metabase-types/store/dashboard.ts**

```
import type { DisplayTheme } from "metabase/public/lib/types";  
import type {  
  DashCardDataMap,  
  DashCardId,  
  Dashboard,  
  DashboardCard,  
  DashboardId,  
  DashboardTab,  
  DashboardTabId,  
  ...
```

### **frontend/src/metabase-types/store/downloads.ts**

```
export interface Download {  
  id: number;  
  title: string;  
  status: "complete" | "in-progress" | "error";
```

## Analyst Base

```
error?: string;  
}
```

```
export type DownloadsState = Download[];
```

### **frontend/src/metabase-types/store/embed.ts**

```
import type {  
  EmbeddingDataPicker,  
  EmbeddingEntityType,  
} from "../embedding-data-picker";
```

```
export interface InteractiveEmbeddingOptions {  
  font: string | undefined;  
  top_nav: boolean;  
  search:...
```

### **frontend/src/metabase-types/store/embedding-data-picker.ts**

```
export interface EmbeddingDataPickerState {  
  entityTypes: EmbeddingEntityType[];  
}
```

```
/**
```

```
 * `question` only works on multi-stage data picker, not the simple data picker.  
 * The reason being that we...
```

### **frontend/src/metabase-types/store/entities.ts**

```
import type {  
  NormalizedCard,  
  NormalizedCollection,  
  NormalizedDashboard,  
  NormalizedDatabase,  
  NormalizedField,  
  NormalizedIndexedEntity,  
  NormalizedNativeQuerySnippet,  
  NormalizedSchema,  
  ...
```

### **frontend/src/metabase-types/store/index.ts**

```
export * from "../admin";  
export * from "../app";  
export * from "../auth";  
export * from "../collection";  
export * from "../dashboard";  
export * from "../embed";
```

## Analyst Base

```
export * from "../entities";  
export * from...
```

### **frontend/src/metabase-types/store/mocks/admin.ts**

```
import type { AdminAppState, AdminState } from "metabase-types/store";
```

```
export const createMockAdminState = (  
  opts?: Partial<AdminState>,  
) : AdminState => ({  
  app: createMockAdminAppState(),  
  ...  

```

### **frontend/src/metabase-types/store/mocks/app.ts**

```
import type { AppState } from "metabase-types/store";
```

```
export const createMockAppState = (opts?: Partial<AppState>): AppState => ({  
  isNavbarOpen: true,  
  errorPage: null,  
  isDndAvailable: false,  
  ...  

```

### **frontend/src/metabase-types/store/mocks/auth.ts**

```
import type { AuthState } from "metabase-types/store";
```

```
export const createMockAuthState = (opts?: Partial<AuthState>): AuthState => ({  
  loginPending: false,  
  redirect: true,  
  ...opts,  
});
```

### **frontend/src/metabase-types/store/mocks/dashboard.ts**

```
import { createMockDashboard } from "metabase-types/api/mocks";  
import type { DashboardState, StoreDashboard } from "metabase-types/store";
```

```
export const createMockDashboardState = (  
  opts:...
```

### **frontend/src/metabase-types/store/mocks/downloads.ts**

```
import type { Download } from "../downloads";
```

```
export const createMockDownload = (props: Partial<Download> = {}): Download => {  
  return {  
    id: Date.now(),  
    title: "file.csv",  

```

## Analyst Base

status:...

### **frontend/src/metabase-types/store/mocks/embed.ts**

```
import { DEFAULT_INTERACTIVE_EMBEDDING_OPTIONS } from "metabase/redux/embed";
import type {
  EmbedState,
  InteractiveEmbeddingOptions,
} from "metabase-types/store";
```

export const...

### **frontend/src/metabase-types/store/mocks/embedding-data-picker.ts**

```
import { DEFAULT_EMBEDDING_ENTITY_TYPES } from "metabase/redux/embedding-data-picker";
```

```
import type { EmbeddingDataPickerState } from "../embedding-data-picker";
```

export const...

### **frontend/src/metabase-types/store/mocks/entities.ts**

```
import type { EntitiesState } from "metabase-types/store";
```

// This is a helper for cases when entities state doesn't matter

// Most likely, createMockEntitiesState from \_\_support\_\_/store would be a...

### **frontend/src/metabase-types/store/mocks/index.ts**

```
export * from "./admin";
export * from "./app";
export * from "./auth";
export * from "./dashboard";
export * from "./entities";
export * from "./embed";
export * from "./parameters";
export * from...
```

### **frontend/src/metabase-types/store/mocks/parameters.ts**

```
import type { ParametersState } from "metabase-types/store/parameters";
```

```
export const createMockParametersState = (
  opts?: Partial<ParametersState>,
): ParametersState => ({
  parameterValuesCache:...
```

### **frontend/src/metabase-types/store/mocks/qb.ts**

```
import type {
  QueryBuilderDashboardState,
```

## Analyst Base

```
QueryBuilderState,  
QueryBuilderUIControls,  
} from "metabase-types/store";
```

```
export const createMockQueryBuilderUIControlsState = (  
  opts?:....
```

### **frontend/src/metabase-types/store/mocks/requests.ts**

```
import type { RequestsState } from "../requests";
```

```
export const createMockRequestsState = (): RequestsState => {  
  return {  
    entities: {},  
    plugins: {},  
  };  
};
```

### **frontend/src/metabase-types/store/mocks/routing.ts**

```
import type { Location } from "history";  
import type { RouterState } from "react-router-redux";
```

```
export const createMockRoutingState = (  
  opts?: Partial<RouterState>,  
) : RouterState => {  
  return {  
    ...
```

### **frontend/src/metabase-types/store/mocks/settings.ts**

```
import type { EnterpriseSettings, Settings } from "metabase-types/api";  
import { createMockSettings } from "metabase-types/api/mocks";  
import type { SettingsState } from...
```

### **frontend/src/metabase-types/store/mocks/setup.ts**

```
import type {  
  InviteInfo,  
  Locale,  
  SetupState,  
  SubscribeInfo,  
  UserInfo,  
} from "metabase-types/store";
```

```
export const createMockLocale = (opts?: Partial<Locale>): Locale => ({  
  name:...
```

### **frontend/src/metabase-types/store/mocks/state.ts**



## Analyst Base

```
import type { SdkStoreState } from "embedding-sdk/store/types";
import type { EnterpriseState } from "metabase-enterprise/settings/types";
import { createMockUser } from...
```

### frontend/src/metabase-types/store/mocks/upload.ts

```
import type { FileUpload } from "../upload";

export const createMockUploadState = (uploads = {}) => {
  return { ...uploads };
};
```

```
export const createMockUpload = (props?: Partial<FileUpload>):...
```

### frontend/src/metabase-types/store/mocks/visualizer.ts

```
import type { VisualizerState } from "../visualizer";

export const createMockVisualizerState = (
  opts?: Partial<VisualizerState>,
): VisualizerState => ({
  initialState: {
    display: null,
    ...
  }
});
```

### frontend/src/metabase-types/store/parameters.ts

```
import type { ParameterValues } from "metabase-types/api";

export type ParameterValuesCache = Record<string, ParameterValues>;

export interface ParametersState {
  parameterValuesCache:...
```

### frontend/src/metabase-types/store/qb.ts

```
import type { Deferred } from "metabase/lib/promise";
import type { QueryModalType } from "metabase/query_builder/constants";
import type { Widget } from...
```

### frontend/src/metabase-types/store/requests.ts

```
type EntityKey = string;

type QueryKey = string;

type RequestType = string; // only "fetch"?

// See initialRequestState in metabase/redux/requests.js
```

## Analyst Base

```
export interface RequestState {  
  loading:...
```

### **frontend/src/metabase-types/store/settings.ts**

```
import type { EnterpriseSettings } from "metabase-types/api";
```

```
export interface SettingsState {  
  values: EnterpriseSettings;  
  loading: boolean;  
}
```

### **frontend/src/metabase-types/store/setup.ts**

```
import type { SetupStep } from "metabase/setup/types";  
import type { DatabaseData, UsageReason } from "metabase-types/api";
```

```
export interface Locale {  
  name: string;  
  code: string;  
}
```

```
export...
```

### **frontend/src/metabase-types/store/state.ts**

```
import type { RouterState } from "react-router-redux";
```

```
import type { User } from "metabase-types/api";
```

```
import type { AdminState } from "../admin";  
import type { AppState } from "../app";  
import type...
```

### **frontend/src/metabase-types/store/undo.ts**

```
import type { ReactNode, RefObject } from "react";
```

```
import type { IconName } from "metabase/ui";  
import type { DashCardId, DashboardTabId } from "metabase-types/api";
```

```
export interface Undo {  
  id:...
```

### **frontend/src/metabase-types/store/upload.ts**

```
import type { CollectionId, TableId } from "metabase-types/api";
```

```
export enum UploadMode {
```

## Analyst Base

```
append = "append",
create = "create",
replace = "replace",
}
```

```
export type FileUpload = {
  status:...
```

### **frontend/src/metabase-types/store/visualizer.ts**

```
import type {
  Card,
  Dataset,
  DatasetColumn,
  VisualizerDataSource,
  VisualizerDataSourceId,
  VisualizerVizDefinition,
} from "metabase-types/api";
```

```
type BaseDraggedItem<T> = {
  id: string;
  ...
```

### **frontend/src/types/dayjs.d.ts**

```
import type { ManipulateType } from "dayjs";
```

```
type SupportedUnit = QUnitType | "isoWeek" | ManipulateType;
```

```
declare module "dayjs" {
  interface Dayjs {
    add(value: number, unit: SupportedUnit):...
```

### **frontend/src/types/emotion.d.ts**

```
import "@emotion/react";
```

```
import type { MantineTheme } from "@mantine/core";
interface _EmotionCompatibilityTheme {
  fn: { themeColor: (colorName: string) => string };
}
```

```
declare module...
```

### **frontend/src/types/global.d.ts**

```
interface Window {
  MetabaseBootstrap: any;
```

## Analyst Base

```
MetabaseRoot?: string;  
MetabaseNonce?: string;  
}
```

// This allows importing static SVGs from TypeScript files

```
declare module "*.svg" {  
  const content:...
```

### **frontend/src/types/import-svg.d.ts**

```
declare module "*.svg" {  
  const value: string;
```

```
  // eslint-disable-next-line import/no-default-export  
  export default value;  
}
```

```
declare module "*.svg?source" {  
  const value: string;
```

```
//...
```

### **frontend/src/types/mantine.d.ts**

```
import type { EmbeddingThemeOptions } from "metabase/embedding-sdk/theme/private";
```

```
interface _EmotionCompatibilityTheme {  
  fn: {  
    themeColor: (colorName: string) => string;  
  };  
}
```

```
declare...
```

### **frontend/src/types/react-ansi-style.d.ts**

```
declare module "react-ansi-style";
```

### **frontend/src/types/react.d.ts**

```
import type * as React from "react";
```

```
declare module "react" {  
  // to make forwardRef work with generic function components  
  // eslint-disable-next-line @typescript-eslint/ban-types  
  function...
```

### **frontend/src/types/slugg.d.ts**

## Analyst Base

```
declare module "slug" {  
  type Separator = string;  
  type ToStrip = string | RegExp;
```

```
  type Options = {  
    separator?: Separator;  
    toStrip?: ToStrip;  
    toLowerCase?: boolean;  
  };  
  
  //...
```

### **frontend/src/types/underscore.d.ts**

\*

### **frontend/test/.eslintrc**

```
{  
  "rules": {  
    "import/no-commonjs": 0,  
    "no-color-literals": 0  
  },  
  "env": {  
    "node": true  
  }  
}
```

### **frontend/test/\_\_mocks\_\_/aceSearchBoxExtMock.js**

```
module.exports = {};
```

### **frontend/test/\_\_mocks\_\_/fileMock.js**

```
module.exports = "test-file-stub";
```

### **frontend/test/\_\_mocks\_\_/styleMock.js**

```
module.exports = {};
```

### **frontend/test/\_\_runner\_\_/run\_timezone\_tests**

```
!/usr/bin/env bash
```

### **frontend/test/\_\_support\_\_/components/mantineSelect.ts**

```
import userEvent from "@testing-library/user-event";
```

```
import { screen, within } from "__support__/ui";
```

```
export type ViewMantineSelectOptionsParams = {  
  /** This function will identify the root...
```

## Analyst Base

### frontend/test/\_\_support\_\_/content-translation.ts

```
import * as EnterpriseContentTranslationUtilsModule from "metabase-enterprise/content_translation/utils";

/**
 * One of the utility functions that makes the content translation feature tick
 * is...
```

### frontend/test/\_\_support\_\_/echarts.ts

```
import type {
  BreakoutSeriesModel,
  SeriesModel,
} from "metabase/visualizations/echarts/cartesian/model/types";
import { createMockColumn } from "metabase-types/api/mocks";

export const...
```

### frontend/test/\_\_support\_\_/enterprise.js

```
*
```

### frontend/test/\_\_support\_\_/entities-store.js

```
import { combineReducers, configureStore } from "@reduxjs/toolkit";
import promise from "redux-promise";

import * as entities from "metabase/redux/entities";
import requestsReducer from...
```

### frontend/test/\_\_support\_\_/events.ts

```
type MockEvent = {
  preventDefault: jest.Mock;
  returnValue?: string;
};

type CallMockEventType = (
  mockEventListener: jest.SpyInstance,
  eventName: string,
) => MockEvent;
```

```
// calls event...
```

### frontend/test/\_\_support\_\_/metadata.ts

```
import { getMetadata } from "metabase/selectors/metadata";
import type { Settings } from "metabase-types/api";
import {
  createMockSettingsState,
```

## Analyst Base

```
createMockState,  
} from...
```

### **frontend/test/\_\_support\_\_/mocks.js**

```
global.ga = () => {};  
global.snowplow = () => {};  
  
global.window.matchMedia = () => ({  
  addEventListener: () => {},  
  removeEventListener: () => {},  
});
```

```
/**  
 * jsdom doesn't have scrollBy or...
```

### **frontend/test/\_\_support\_\_/server-mocks/action.ts**

```
import fetchMock from "fetch-mock";  
  
import type {  
  CardId,  
  GetPublicAction,  
  WritebackAction,  
} from "metabase-types/api";  
import {  
  createMockImplicitQueryAction,  
  createMockQueryAction,  
}...
```

### **frontend/test/\_\_support\_\_/server-mocks/activity.ts**

```
import fetchMock, { type UserRouteConfig } from "fetch-mock";  
  
import type {  
  Dashboard,  
  PopularItem,  
  RecentContexts,  
  RecentItem,  
} from "metabase-types/api";  
  
export function...
```

### **frontend/test/\_\_support\_\_/server-mocks/ai-entity-analysis.ts**

```
import fetchMock from "fetch-mock";  
  
import type { AIEntityAnalysisResponse } from "metabase-types/api";
```

## Analyst Base

```
export function setupAnalyzeChartEndpoint(response: AIEntityAnalysisResponse) {  
  const name...
```

### **frontend/test/\_\_support\_\_/server-mocks/ai-sql-fixer.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { FixSqlQueryResponse } from "metabase-types/api";
```

```
export function setupFixNativeQueryEndpoint(response: FixSqlQueryResponse) {  
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/ai-sql-generation.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { GenerateSqlQueryResponse } from "metabase-types/api";
```

```
export function setupGenerateSqlQueryEndpoint(  
  response: GenerateSqlQueryResponse,  
) {  
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/alert.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { Alert, Card } from "metabase-types/api";
```

```
export function setupAlertsEndpoints(card: Card, alerts: Alert[]) {  
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/api-key.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { ApiKey } from "metabase-types/api";
```

```
export function setupApiKeyEndpoints(apiKeys: ApiKey[]) {  
  fetchMock.get("path:/api/api-key", apiKeys);  
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/audit.ts**

```
import fetchMock, { type UserRouteConfig } from "fetch-mock";
```

```
import type {  
  CardId,
```



## Analyst Base

```
CollectionId,  
DashboardId,  
User,  
} from "metabase-types/api";
```

```
interface AuditInfo {  
  dashboard_overview:...
```

### **frontend/test/\_\_support\_\_/server-mocks/automagic-dashboards.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { DatabaseId, DatabaseXray } from "metabase-types/api";
```

```
export function setupDatabaseCandidatesEndpoint(  
  id: DatabaseId,  
  candidates:...
```

### **frontend/test/\_\_support\_\_/server-mocks/bookmark.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { Bookmark } from "metabase-types/api";
```

```
export function setupBookmarksEndpoints(bookmarks: Bookmark[]) {  
  fetchMock.get("path:/api/bookmark",...
```

### **frontend/test/\_\_support\_\_/server-mocks/bug-report.ts**

```
import fetchMock from "fetch-mock";
```

```
export const setupBugReportEndpoints = (responses = [{ success: true }]) => {  
  fetchMock.post(  
    "path:/api/slack/bug-report",  
    { status: 200, body:...
```

### **frontend/test/\_\_support\_\_/server-mocks/card.ts**

```
import fetchMock from "fetch-mock";
```

```
import { getQuestionVirtualTableId } from "metabase-lib/v1/metadata/utils/saved-questions";  
import type {  
  Card,  
  CardId,  
  CardQueryMetadata,  
  Dataset,  
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/channel.ts**

## Analyst Base

```
import fetchMock from "fetch-mock";
```

```
import type { NotificationChannel } from "metabase-types/api";
```

```
export const setupWebhookChannelsEndpoint = (  
  channels: NotificationChannel[] = [],  
) => {  
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/collection.ts**

```
import fetchMock from "fetch-mock";  
import _ from "underscore";
```

```
import { ROOT_COLLECTION } from "metabase/entities/collections/constants";  
import {  
  SAVED QUESTIONS_VIRTUAL_DB_ID,  
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/constants.ts**

```
export const PERMISSION_ERROR = "You don't have permissions to do that.";
```

### **frontend/test/\_\_support\_\_/server-mocks/content-translation.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { DictionaryArray, DictionaryResponse } from "metabase-types/api";
```

```
export function setupContentTranslationEndpoints({  
  dictionary = [],  
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/dashboard.ts**

```
import fetchMock from "fetch-mock";
```

```
import { getNextId } from "__support__/utils";  
import type {  
  Dashboard,  
  DashboardId,  
  DashboardQueryMetadata,  
  FieldId,  
  GetPublicDashboard,  
} from...
```

### **frontend/test/\_\_support\_\_/server-mocks/dashcard.ts**

```
import fetchMock from "fetch-mock";
```

## Analyst Base

```
import type { DashboardId, Dataset } from "metabase-types/api";
import type { StoreDashcard } from "metabase-types/store";
```

```
export function...
```

### **frontend/test/\_\_support\_\_/server-mocks/database.ts**

```
import fetchMock from "fetch-mock";
import _ from "underscore";
```

```
import { SAVED_QUESTIONS_DATABASE } from "metabase/databases/constants";
import { isTypePK } from...
```

### **frontend/test/\_\_support\_\_/server-mocks/dataset.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { CardQueryMetadata, ParameterValues } from "metabase-types/api";
import type { MockDatasetOpts } from "metabase-types/api/mocks";
import {...
```

### **frontend/test/\_\_support\_\_/server-mocks/email.ts**

```
email settings, why aren't they in the settings endpoint? who knows? \_()\_/
```

### **frontend/test/\_\_support\_\_/server-mocks/embed.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { Card, Dashboard, DashboardCard } from "metabase-types/api";
import { createMockDataset } from "metabase-types/api/mocks";
```

```
export function...
```

### **frontend/test/\_\_support\_\_/server-mocks/entity-id.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { TranslateEntityIdRequest } from "metabase/api";
import type {
  BaseEntityId,
  CardId,
  CollectionId,
  DashboardId,
} from...
```

### **frontend/test/\_\_support\_\_/server-mocks/field.ts**

```
import fetchMock from "fetch-mock";
```

## Analyst Base

```
import type {  
  Field,  
  FieldId,  
  FieldValue,  
  GetFieldValuesResponse,  
} from "metabase-types/api";  
import { createMockFieldValues } from...
```

### **frontend/test/\_\_support\_\_/server-mocks/gdrive.ts**

```
import fetchMock from "fetch-mock";  
  
import type { GdrivePayload } from "metabase-types/api";  
  
export function setupGdriveGetFolderEndpoint({  
  errorCode,  
  ...gdrivePayload  
}: Partial<GdrivePayload>...
```

### **frontend/test/\_\_support\_\_/server-mocks/geojson.ts**

```
import fetchMock from "fetch-mock";  
import type { FeatureCollection } from "geojson";  
  
export function setupGeoJSONEndpoint({  
  featureCollection,  
  url,  
}: {  
  featureCollection: FeatureCollection;  
  ...  
}
```

### **frontend/test/\_\_support\_\_/server-mocks/google.ts**

```
import fetchMock from "fetch-mock";  
  
export function setupUpdateGoogleAuthEndpoint(  
  { status }: { status?: number } = { status: 204 },  
) {  
  fetchMock.put(  
    new RegExp("/api/google/settings"),  
    ...  
  )
```

### **frontend/test/\_\_support\_\_/server-mocks/group.ts**

```
import fetchMock from "fetch-mock";  
  
import type { Group } from "metabase-types/api";
```

## Analyst Base

```
export const setupGroupsEndpoint = (groups: Omit<Group, "members">[]) => {  
  ...
```

### **frontend/test/\_\_\_support\_\_\_/server-mocks/impersonation.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { Databaseld, GroupId, Impersonation } from "metabase-types/api";
```

```
export const setupExistingImpersonationEndpoint = (  
  impersonation:...
```

### **frontend/test/\_\_\_support\_\_\_/server-mocks/index.ts**

```
export * from "./action";
```

```
export * from "./activity";
```

```
export * from "./ai-entity-analysis";
```

```
export * from "./ai-sql-fixer";
```

```
export * from "./ai-sql-generation";
```

```
export * from "./alert";
```

```
export * from...
```

### **frontend/test/\_\_\_support\_\_\_/server-mocks/logger.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { LoggerPreset } from "metabase-types/api";
```

```
export function setupLoggerPresetsEndpoint(response: LoggerPreset[]) {  
  ...
```

### **frontend/test/\_\_\_support\_\_\_/server-mocks/metabot.ts**

```
import fetchMock, { type UserRouteConfig } from "fetch-mock";
```

```
import type {
```

```
  MetabotApiEntity,
```

```
  MetabotId,
```

```
  MetabotInfo,
```

```
  SuggestedMetabotPrompt,
```

```
  SuggestedMetabotPromptsResponse,
```

```
} from...
```

### **frontend/test/\_\_\_support\_\_\_/server-mocks/model-indexes.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { CardId, ModelIndex } from "metabase-types/api";
```

```
import { createMockModelIndex } from "metabase-types/api/mocks";
```

## Analyst Base

export function...

### **frontend/test/\_\_support\_\_/server-mocks/native-query-snippet.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { NativeQuerySnippet } from "metabase-types/api";
```

```
const PATH = "path:/api/native-query-snippet";
```

```
export function setupNativeQuerySnippetEndpoints(  
  ...
```

```
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/notification.ts**

```
import fetchMock from "fetch-mock";
```

```
import type {  
  ListNotificationsRequest,  
  Notification,  
} from "metabase-types/api";
```

```
export const setupListNotificationEndpoints = (  
  { card_id }:...  
);
```

### **frontend/test/\_\_support\_\_/server-mocks/performance.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { CacheConfig } from "metabase-types/api";  
export function setupPerformanceEndpoints(cacheConfigs: CacheConfig[]) {  
  fetchMock.get("path:/api/cache",...  
);
```

### **frontend/test/\_\_support\_\_/server-mocks/permissions.ts**

```
import fetchMock from "fetch-mock";
```

```
import type {  
  CollectionPermissionsGraph,  
  Database,  
  Group,  
} from "metabase-types/api";  
import { createMockPermissionsGraph } from...
```

### **frontend/test/\_\_support\_\_/server-mocks/persist.ts**

```
import fetchMock from "fetch-mock";
```

## Analyst Base

```
import type { ModelCacheRefreshStatus } from "metabase-types/api";
```

```
export function setupModelPersistenceEndpoints(  
  persistedModels:...
```

### **frontend/test/\_\_support\_\_/server-mocks/premium-features.ts**

```
import fetchMock from "fetch-mock";
```

```
export const setupTokenStatusEndpoint = ({  
  valid,  
  features,  
}): {  
  valid: boolean;  
  features?: string[];  
}) => {  
  const name = "premium-token-status";  
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/public.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { EmbedDataset, PublicCard } from "metabase-types/api";
```

```
export function setupPublicQuestionEndpoints(  
  uuid: string,  
  publicCard: PublicCard,  
) {  
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/pulse.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { ChannelApiResponse } from "metabase-types/api";
```

```
export const setupNotificationChannelsEndpoints = (  
  channelData:...
```

### **frontend/test/\_\_support\_\_/server-mocks/revision.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { Revision } from "metabase-types/api";
```

```
export function setupRevisionsEndpoints(revisions: Revision[]) {  
  fetchMock.get("path:/api/revision",...
```

## Analyst Base

### **frontend/test/\_\_\_support\_\_\_/server-mocks/search.ts**

```
import fetchMock from "fetch-mock";
import { match } from "ts-pattern";

import type { CollectionItem, SearchResult } from "metabase-types/api";
import type { EmbeddingDataPicker } from...
```

### **frontend/test/\_\_\_support\_\_\_/server-mocks/segment.ts**

```
import fetchMock from "fetch-mock";

import type { Segment } from "metabase-types/api";
import { createMockSegment } from "metabase-types/api/mocks";

export function setupSegmentEndpoint(segment:...
```

### **frontend/test/\_\_\_support\_\_\_/server-mocks/session.ts**

```
import fetchMock from "fetch-mock";

import type {
  EnterpriseSettings,
  PasswordResetTokenStatus,
  Settings,
} from "metabase-types/api";

export function setupPropertiesEndpoints(
  settings:...
```

### **frontend/test/\_\_\_support\_\_\_/server-mocks/settings.ts**

```
import fetchMock from "fetch-mock";

import type {
  MaskedScimApiKey,
  UnmaskedScimApiKey,
} from "metabase-enterprise/user_provisioning/types";
import type {
  EnterpriseSettingKey,
  ...
```

### **frontend/test/\_\_\_support\_\_\_/server-mocks/setup.ts**

```
import fetchMock from "fetch-mock";

export function setupErrorSetupEndpoints() {
  fetchMock.post("path:/api/setup", 400);
```



## Analyst Base

```
}

export function setupForTokenCheckEndpoint(response: { valid: boolean })...

frontend/test/__support__/server-mocks/slack.ts

import fetchMock from "fetch-mock";

export function setupSlackManifestEndpoint() {
  fetchMock.get("path:/api/slack/manifest", "manifest-content");
}

export function setupSlackSettingsEndpoint() {
  ...
frontend/test/__support__/server-mocks/store.ts

import fetchMock from "fetch-mock";

import type { StoreTokenStatus } from "metabase-types/api";

export function setupStoreTokenEndpoints(tokenStatus: StoreTokenStatus) {
  ...
frontend/test/__support__/server-mocks/table.ts

import fetchMock from "fetch-mock";

import type { ForeignKey, Table } from "metabase-types/api";

import { setupFieldEndpoints } from "../field";

export function setupTableEndpoints(
  table:...
frontend/test/__support__/server-mocks/task.ts

import fetchMock, { type UserRouteConfig } from "fetch-mock";

import type { ListTasksResponse, Task } from "metabase-types/api";

export function setupTasksEndpoints(
  response: ListTasksResponse,
  ...
frontend/test/__support__/server-mocks/timeline.ts

import fetchMock from "fetch-mock";
```

## Analyst Base

```
import type { Timeline } from "metabase-types/api";
```

```
export function setupTimelinesEndpoints(timelines: Timeline[]) {  
  fetchMock.get("path:/api/timeline",...
```

### **frontend/test/\_\_support\_\_/server-mocks/user-key-value.ts**

```
import fetchMock from "fetch-mock";
```

```
import type { UserKeyValue, UserKeyValueKey } from "metabase-types/api";
```

```
export const setupUserKeyValueEndpoints = ({  
  namespace,  
  key,  
  value,  
}:...)
```

### **frontend/test/\_\_support\_\_/server-mocks/user.ts**

```
import fetchMock from "fetch-mock";
```

```
import type {  
  User,  
  UserAttributeKey,  
  UserListResult,  
} from "metabase-types/api";
```

```
export function setupUserEndpoints(user: UserListResult) {  
  ...
```

### **frontend/test/\_\_support\_\_/server-mocks/util.ts**

```
import fetchMock from "fetch-mock";
```

```
export function setupPasswordCheckEndpoint() {  
  fetchMock.post("path:/api/session/password-check", 204);  
}
```

```
type ResponseInfo = {  
  url: string;  
  body:...
```

### **frontend/test/\_\_support\_\_/settings.ts**

```
import MetabaseSettings from "metabase/lib/settings";  
import type { EnterpriseSettings, Settings } from "metabase-types/api";  
import { createMockSettings } from "metabase-types/api/mocks";  
import {...
```

### **frontend/test/\_\_support\_\_/store.ts**

```
import type { Schema as NormalizrSchema } from "normalizr";
import { normalize } from "normalizr";
```

```
import {
  ActionSchema,
  CollectionSchema,
  DashboardSchema,
  DatabaseSchema,
  FieldSchema,
  ...
```

### **frontend/test/\_\_support\_\_/testDataset.ts**

```
import type { DatasetColumn, DatasetData } from "metabase-types/api";
import { createMockDatasetData } from "metabase-types/api/mocks";
import { PRODUCTS, PRODUCTS_ID } from...
```

### **frontend/test/\_\_support\_\_/timezones.js**

```
export default function testAcrossTimezones(runTests) {
  // run_timezone_tests sets "TZ" environment variable to change the timezone
  const clientTz = process.env["TZ"] || "[default]";
  //...
```

### **frontend/test/\_\_support\_\_/ui-mocks.js**

```
jest.mock("metabase/common/components/Popover");
```

```
// Replace addEventListener with a test implementation which collects all event listeners to `eventListeners`
map
export const eventListeners =...
```

### **frontend/test/\_\_support\_\_/utils.ts**

```
import type { CallLog } from "fetch-mock";
```

```
import { act, waitFor } from "./ui";
```

```
export const getNextId = (() => {
  let id = 0;
  return (startingId?: number) => {
    if (startingId) {
      id =...
```

### **frontend/test/\_\_support\_\_/visualizations.js**

```
import { getComputedSettingsForSeries } from "metabase/visualizations/lib/settings/visualization";
import { formatValueForTooltip } from "metabase/visualizations/lib/tooltip";
```

export function...

### **frontend/test/generate-loki-report-json.js**

```
const { readdir, writeFile } = require("fs");
const { join: joinPath, relative } = require("path");
```

```
const { promisify } = require("util");
```

```
const asyncReaddir = promisify(readdir);
const...
```

### **frontend/test/jest-setup-env.js**

```
import "@testing-library/jest-dom";
import fetchMock from "fetch-mock";
```

```
beforeEach(() => {
  fetchMock.mockGlobal();
});
```

```
afterEach(() => {
  fetchMock.removeRoutes();
  ...
```

### **frontend/test/jest-setup.js**

```
eslint-disable no-undef
```

### **frontend/test/metabase/lib/api.unit.spec.js**

```
import fetchMock from "fetch-mock";
```

```
import api, { GET, POST, PUT } from "metabase/lib/api";
api.basename = "";
```

```
describe("api", () => {
  const successResponse = { body: { status: "ok" } };
  ...
```

### **frontend/test/metabase/lib/browser.unit.spec.js**

```
import { parseHashOptions, stringifyHashOptions } from "metabase/lib/browser";
```

```
describe("browser", () => {
  describe("parseHashOptions", () => {
    it("should parse with prepended '#'", () => {
```

...

### **frontend/test/metabase/lib/card.unit.spec.js**

```
import {
  createCard,
  deserializeCardFromUrl,
  serializeCardForUrl,
} from "metabase/lib/card";
import {
  b64_to_utf8,
  b64url_to_utf8,
  utf8_to_b64,
  utf8_to_b64url,
} from...
```

### **frontend/test/metabase/lib/dashboard\_grid.unit.spec.js**

```
import { getPositionForNewDashCard } from "metabase/lib/dashboard_grid";

const getPos = (cards) => getPositionForNewDashCard(cards, 2, 2, 6);

describe("dashboard_grid", () => {
  ...
}
```

### **frontend/test/metabase/lib/data\_grid.unit.spec.js**

```
import _ from "underscore";

import {
  COLLAPSED_ROWS_SETTING,
  COLUMN_SHOW_TOTALS,
  COLUMN_SORT_ORDER,
  COLUMN_SPLIT_SETTING,
  multiLevelPivot,
  pivot,
} from "metabase/lib/data_grid";
import...
```

### **frontend/test/metabase/lib/dom.unit.spec.js**

```
import {
  getSelectionPosition,
  parseDataUri,
  setSelectionPosition,
} from "metabase/lib/dom";

describe("getSelectionPosition/setSelectionPosition", () => {
```

## Analyst Base

```
let container;
```

```
beforeEach(() =>...
```

### **frontend/test/metabase/lib/formatting.unit.spec.js**

```
import moment from "moment-timezone"; // eslint-disable-line no-restricted-imports -- deprecated usage
import { isElementType } from "react-dom/test-utils";
```

```
import { mockSettings } from...
```

### **frontend/test/metabase/lib/pulse.unit.spec.js**

```
import {
  getActivePulseParameters,
  getPulseParameters,
  recipientIsValid,
} from "metabase/lib/pulse";
import MetabaseSettings from "metabase/lib/settings";
```

```
describe("recipientIsValid", () =>...
```

### **frontend/test/metabase/lib/schema\_metadata.unit.spec.js**

```
import {
  foreignKeyCountsByOriginTable,
  getSemanticTypeIcon,
  getSemanticTypeName,
} from "metabase/lib/schema_metadata";
import { TYPE } from...
```

### **frontend/test/metabase/lib/string.unit.spec.js**

```
import { regexpEscape } from "metabase/lib/string";
```

```
describe("regexpEscape", () => {
  const testCases = [
    ["nothing special here", "nothing special here"],
    ["somewhat .\\ special", ...
```

### **frontend/test/metabase/lib/time.unit.spec.js**

```
import moment from "moment-timezone"; // eslint-disable-line no-restricted-imports -- deprecated usage
```

```
import { parseTime, parseTimestamp } from "metabase/lib/time";
```

```
describe("time", () => {
```

```
...
```

### **frontend/test/metabase/lib/urls.unit.spec.js**

## Analyst Base

```
import {  
  bookmark,  
  browseDatabase,  
  collection,  
  dashboard,  
  extractCollectionId,  
  extractEntityId,  
  extractQueryParams,  
  isCollectionPath,  
  model,  
  modelDetail,  
  modelEditor,  
  ...  
}
```

### **frontend/test/metabase/lib/user.unit.spec.ts**

```
import { getFullName } from "metabase/lib/user";  
import { createMockUser } from "metabase-types/api/mocks";  
  
describe("lib/user", () => {  
  describe("getFullName", () => {  
    test("has both...
```

### **frontend/test/metabase/lib/validate.unit.spec.js**

```
import validate, { validators } from "metabase/lib/validate";  
  
describe("validators", () => {  
  describe("required", () => {  
    it("should return an error if the value is null", () => {  
      ...  
    }  
  }  
}
```

### **frontend/test/metabase-bootstrap.js**

```
import "regenerator-runtime/runtime";  
import "metabase/css/vendor.css";  
import "metabase/css/index.module.css";  
  
window.MetabaseBootstrap = {  
  "enable-xrays": true,  
  "available-timezones": [  
    ...  
  ]  
}
```

### **frontend/test/metabase-lib/lib/Question.unit.spec.js**

```
import { assoc, assocIn, dissoc } from "icepick";  
import { parse } from "url";
```

## Analyst Base

```
import { createMockMetadata } from "__support__/metadata";
import { deserializeCardFromUrl } from...
```

### frontend/test/metabase-lib/lib/metadata/Table.unit.spec.js

```
import { createMockMetadata } from "__support__/metadata";
import Database from "metabase-lib/v1/metadata/Database";
import Table from "metabase-lib/v1/metadata/Table";
import {
  ORDERS_ID,
  ...
```

### frontend/test/metabase-lib/lib/normalize.unit.spec.js

```
import { normalize } from "cljs/metabase.legacy_mbql.js";
```

```
// This test is here mostly to make sure the shared CLJS lib works correctly.
```

```
describe("normalize", () => {
  it("should normalize an MBQL...
```

### frontend/test/metabase-lib/lib/queries/NativeQuery.unit.spec.js

```
import { assocIn } from "icepick";
```

```
import { createMockMetadata } from "__support__/metadata";
import Question from "metabase-lib/v1/Question";
import NativeQuery, {
  updateCardTemplateTagNames,
}...
```

### frontend/test/metabase-lib/lib/utils.unit.spec.js

```
import { memoizeClass, sortObject } from "metabase-lib/v1/utils";
```

```
describe("sortObject", () => {
  it("should serialize identically regardless of property creation order", () => {
    const o1 =...
```

### frontend/test/register-visualizations.js

We need to mock this *\*before\** registering the visualizations. Otherwise ``ChartWithLegend`` will already load the real one.

### jest.config.js

```
@ts-check
```

### loki.config.js

```
module.exports = {
  diffingEngine: "looks-same",
```



## Analyst Base

```
storiesFilter: [  
  "DataGrid",  
  "static-viz",  
  "viz",  
  "^visualizations/shared",  
  "^app/embed",  
  "^design system",  
  ...  
]
```

### mage/resources/splash.txt

### modules/drivers/clickhouse/.docker/clickhouse/single\_node\_tls/Dockerfile

```
FROM clickhouse/clickhouse-server:25.2-alpine  
COPY .docker/clickhouse/single_node_tls/certificates /etc/clickhouse-server/certs  
RUN chown clickhouse:clickhouse -R /etc/clickhouse-server/certs \  
...
```

### modules/drivers/clickhouse/LICENSE.txt

This directory (modules/drivers/clickhouse/) includes code from the Clickhouse Metabase Driver (<https://github.com/ClickHouse/metabase-clickhouse-driver>) that is licensed under the Apache License,...

### modules/drivers/clickhouse/README.md

ClickHouse Driver for Metabae  
## Attribution

### modules/drivers/starburst/LICENSE.txt

This directory (modules/drivers/starburst/) includes code from the Starburst Metabase Driver (<https://github.com/starburstdata/metabase-driver>) that is licensed under the Apache License, Version...

### postcss.config.js

```
eslint-env node  
eslint-disable import/no-commonjs
```

### release/.env-template

To release locally, copy this file to a .env and fill in all values

### release/.eslintrc

```
{  
  "overrides": [  
    {  
      "files": ["*.ts", "*.js"],  
      "rules": {
```

## Analyst Base

```
    "no-console": "off"
  }
}
]
```

### **release/.gitignore**

```
node_modules/
.env
cache.json
*.log
dist/
version.properties
version-info.json
v*.html
channels.html
```

### **release/README.md**

Metabase Release

### **release/generate-changelog.ts**

changelog preview only, doesn't publish anything

### **release/release-offline.ts**

\*

### **release/src/backports.ts**

```
import "dotenv/config";
import type { Octokit } from "@octokit/rest";
import dayjs from 'dayjs';
```

```
import { sendBackportReminder } from "./slack";
import type { Issue } from "./types";
```

const...

### **release/src/constants.ts**

```
export const nonUserFacingLabels = [
  'CI & Tests',
  'Building & Releasing',
];
```

```
export const hiddenLabels = [
  'Type:Documentation',
];
```

## Analyst Base

### release/src/github.ts

```
import type { GithubCheck, GithubProps, Issue, ReleaseProps } from "./types";
import {
  getLastReleaseTag,
  getMilestoneName,
  getNextVersions,
  getOSSVersion,
} from "./version-helpers";
```

export...

### release/src/index.ts

```
export * from "./backports";
export * from "./github";
export * from "./linked-issues";
export * from "./milestones";
export * from "./release-notes";
export * from "./release-status";
export * from...
```

### release/src/linked-issues.ts

```
export function getLinkedIssues(body: string) {
  const matches = body.match(
    /(close(s|d)?|fixe?(s|d)?|resolve(s|d)?)(:?) (#|https?:Vgithub\.comV.+metabaseVissuesV)(\d+)/gi,
  );

  if...
```

### release/src/linked-issues.unit.spec.ts

```
import { getBackportSourcePRNumber, getLinkedIssues, getPRsFromCommitMessage } from
"./linked-issues";

const closingKeywords = [
  "Close",
  "Closes",
  "Closed",
  "Fix",
  "Fixes",
  "Fixed",
  ...
```

### release/src/milestones.ts

```
import fs from "fs";
```

## Analyst Base

```
import { graphql } from "@octokit/graphql";
import _ from "underscore";
```

```
import { hiddenLabels, nonUserFacingLabels } from "./constants";
import {
  findMilestone,
  ...
```

### **release/src/milestones.unit.spec.ts**

```
import { getNextMilestone } from "./milestones";
import type { Milestone } from "./types";
```

```
describe('milestones', () => {
  const testMilestones = [
    { number: 577, title: '0.57.7' },
    {...
```

### **release/src/release-branch-commits.ts**

```
import {$} from 'zx';
```

```
export async function getReleaseBranchCommits(
  { versionNumber }: { versionNumber: number }
) {
  const lastMajorVersion = versionNumber - 1;
```

```
  // find where the last branch...
```

### **release/src/release-channel-log.ts**

```
import fs from 'fs';
```

```
import { $ } from 'zx';
```

```
const releaseChannels = [
  "nightly",
  "beta",
  "latest",
] as const;
```

```
const editions = ["oss", "ee"] as const;
```

```
type CommitInfo = {
  version:...
```

### **release/src/release-log-run.ts**

```
import { generateReleaseLog } from './release-log';
```

```
generateReleaseLog();
```

### **release/src/release-log.ts**

```
import fs from 'fs';
```

```
import { $ } from 'zx';
```

```
import { issueNumberRegex } from './linked-issues';
```

```
type CommitInfo = {  
  versions: string[],  
  message: string,  
  hash: string,  
  date:...
```

### **release/src/release-log.unit.spec.ts**

```
import { processCommit } from './release-log';
```

```
describe('Release Log', () => {  
  describe('processCommit', () => {  
    it('should return an array of commit info objects with versions', () => {  
      ...
```

### **release/src/release-notes-templates.ts**

```
export const githubReleaseTemplate = `## Upgrading
```

```
> Before you upgrade, back up your Metabase application database!
```

```
Check out our [upgrading...
```

### **release/src/release-notes.ts**

```
import { match } from "ts-pattern";
```

```
import { hiddenLabels, nonUserFacingLabels } from './constants';  
import { getMilestoneIssues, hasBeenReleased } from './github';  
import { issueNumberRegex } from...
```

### **release/src/release-notes.unit.spec.ts**

```
import {  
  categorizeIssues,  
  generateReleaseNotes,
```

## Analyst Base

```
getChangelogUrl,  
getReleaseTitle,  
getWebsiteChangelog,  
markdownIssueLinks,  
} from "./release-notes";  
import {  
  githubReleaseTemplate,  
  ...
```

### **release/src/release-status.ts**

```
eslint-disable no-console
```

### **release/src/slack.ts**

```
import 'dotenv/config';  
import fs from 'fs';  
  
import { WebClient } from '@slack/web-api';  
import dayjs from 'dayjs';  
import relativeTime from 'dayjs/plugin/relativeTime';  
import fetch from...
```

### **release/src/types.ts**

```
import type { Octokit } from "@octokit/rest";  
  
export interface CacheType {  
  version?: string;  
  majorVersion?: string;  
  releaseBranch?: string;  
  commitHash?: string;  
  versionType?: "major" |...
```

### **release/src/version-helpers.ts**

```
import type { GithubProps, Tag } from "./types";  
  
// https://regexpr.com/711ip  
export const isValidVersionString = (versionString: string) => {  
  return...
```

### **release/src/version-helpers.unit.spec.ts**

```
import type { Tag } from "./types";  
import {  
  filterOutNonSupportedPrereleaseIdentifier,  
  findNextPatchVersion,  
  getBuildRequirements,
```

## Analyst Base

```
getDotXs,  
getEnterpriseVersion,  
...
```

### release/src/version-info.ts

```
import fetch from "node-fetch";  
  
import { getMilestoneIssues } from "../github";  
import type {  
  Issue,  
  ReleaseChannel,  
  ReleaseProps,  
  VersionInfo,  
  VersionInfoFile,  
} from "../types";  
import {  
  ...
```

### release/src/version-info.unit.spec.ts

```
import type { Issue, VersionInfoFile } from "../types";  
import {  
  generateVersionInfoJson, getVersionInfoUrl, updateVersionInfoChannelJson,  
  updateVersionInfoLatestJson } from...
```

### resources/.keep-me

**\*\* DO NOT DELETE \*\***

This file works as a sentinel to allow us to check if we are running in a jar file on JVMs >= 24

**\*\* DO NOT DELETE \*\***

### resources/certificates/README.md

Directory Contents  
# `rds-combined-ca-bundle.pem`

### resources/common\_passwords.txt

```
123456  
password  
12345678  
qwerty  
123456789  
12345  
1234  
111111  
1234567
```

## Analyst Base

dragon  
123123  
baseball  
abc123  
football  
monkey  
letmein  
shadow  
master  
696969  
michael  
mustang  
666666  
qwertyuiop  
123321  
1234567890  
pussy...

### **resources/frontend\_client/app/fonts/Merriweather/OFL.txt**

Copyright 2016 The Merriweather Project Authors (<https://github.com/EbenSorkin/Merriweather>), with Reserved Font Name "Merriweather".

This Font Software is licensed under the SIL Open Font License,...

### **resources/frontend\_client/app/iframeResizer.js**

! iFrame Resizer (iframeSizer.min.js) - v4.3.2 - 2021-04-26

### **resources/frontend\_client/inline\_js/asset\_loading\_error.js**

```
(function () {  
  window.Metabase = window.Metabase || {};  
  window.Metabase.AssetErrorLoad = function (tag) {  
    console.error(  
      `Could not download asset ${tag.src} from your metabase...`  
    );  
  };  
})
```

### **resources/frontend\_client/inline\_js/index\_bootstrap.js**

```
(function() {  
  window.MetabaseBootstrap =  
    JSON.parse(document.getElementById("_metabaseBootstrap").textContent);  
  window.MetabaseUserLocalization = ...  
})
```

### **resources/frontend\_client/inline\_js/init.js**

```
const content = [  
  ["no-sql-required", "Make decisions with data - no SQL required"],  
  ["bring-data-to-slack", "Bring your charts and data into Slack"],  
]
```



## Analyst Base

["click-charts-dive-deeper", "Click on...

### **resources/frontend\_shared/color\_selector.js**

```
var shared;!function(){var t={168:function(t,e,r){let n=r(874),a={};for(let t of Object.keys(n))a[n[t]]=t;let...
```

### **resources/frontend\_shared/static\_viz\_interface.js**

```
const toJSArray = (a) => {  
  const jsArray = [];  
  for (let i = 0; i < a.length; i++) {  
    jsArray[i] = a[i];  
  }  
  return jsArray;  
};
```

```
function toJSMap(m) {  
  const o = {};  
  for (let i = 0; i <...
```

### **resources/openapi/api-intro.md**

- **\*\*The API is subject to change.\*\*** We rarely change API endpoints, and almost never remove them, but if you write code that relies on the API, there's a chance you might have to update your code in...

### **rspack.config.js**

```
@ts-check  
eslint-env node  
eslint-disable import/no-commonjs
```

### **rspack.embedding-sdk-bundle.config.js**

```
eslint-env node  
eslint-disable import/no-commonjs  
eslint-disable import/order
```

### **rspack.embedding-sdk-cli.config.js**

```
@ts-check  
eslint-disable no-undef
```

### **rspack.embedding-sdk-package.config.js**

```
eslint-env node  
eslint-disable import/no-commonjs  
eslint-disable import/order
```

### **rspack.iframe-sdk-embed.config.js**

```
eslint-disable import/no-commonjs  
eslint-disable no-undef
```

### **rspack.main.config.js**

## Analyst Base

@ts-check

eslint-env node

eslint-disable import/no-commonjs

### **rspack.shared.config.js**

eslint-env node

eslint-disable import/no-commonjs

### **rspack.static-viz.config.js**

```
const rspack = require("@rspack/core");
```

```
const YAML = require("json-to-pretty-yaml");
```

```
const { StatsWriterPlugin } = require("webpack-stats-plugin");
```

```
const { WEBPACK_BUNDLE } =...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/account/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Account creation",
  "self": {
    "vendor": "com.metabase",
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/account/jsonschema/1-0-1**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Account creation",
  "self": {
    "vendor": "com.metabase",
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/account/jsonschema/1-0-2**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Account creation",
  "self": {
    "vendor": "com.metabase",
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/action/jsonschema/1-0-0**

## Analyst Base

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Action events",
  "self": {
    "vendor": "com.metabase",
    "name": ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/browse\_data/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Schema for tracking clicks on the Browse Data page",
  "self": {
    ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/cleanup/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Viewing or archiving stale collection items",
  "self": {
    "vendor": ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/cleanup/jsonschema/1-0-1**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Viewing or archiving stale collection items",
  "self": {
    "vendor": ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/csvupload/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "CSV upload events",
  "self": {
    "vendor": "com.metabase",
    "name": ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/csvupload/jsonschema/1-0-1**

```
{
```

## Analyst Base

```
"$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
"self": {  
  "vendor": "com.metabase",  
  "name": "csvupload",  
  "format":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/csvupload/jsonschema/1-0-2**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "self": {  
    "vendor": "com.metabase",  
    "name": "csvupload",  
    "format":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/csvupload/jsonschema/1-0-3**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "self": {  
    "vendor": "com.metabase",  
    "name": "csvupload",  
    "format":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/dashboard/jsonschema/1-0-0**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Dashboard events",  
  "self": {  
    "vendor": "com.metabase",  
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/dashboard/jsonschema/1-0-1**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Dashboard events",  
  "self": {  
    "vendor": "com.metabase",  
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/dashboard/jsonschema/1-1-0**

## Analyst Base

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Dashboard events",
  "self": {
    "vendor": "com.metabase",
    "name": ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/dashboard/jsonschema/1-1-1**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Dashboard events",
  "self": {
    "vendor": "com.metabase",
    "name": ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/dashboard/jsonschema/1-1-2**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Dashboard events",
  "self": {
    "vendor": "com.metabase",
    "name": ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/dashboard/jsonschema/1-1-3**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Dashboard events",
  "self": {
    "vendor": "com.metabase",
    "name": ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/dashboard/jsonschema/1-1-4**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Dashboard events",
  "self": {
    "vendor": "com.metabase",
    "name": ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/dashboard/jsonschema**

## Analyst Base

### a/1-1-5

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Dashboard events",
  "self": {
    "vendor": "com.metabase",
    "name": ...
```

### snowplow/iglu-client-embedded/schemas/com.metabase/database/jsonschema/1-0-0

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Database connection",
  "self": {
    "vendor": "com.metabase",
    "name": ...
```

### snowplow/iglu-client-embedded/schemas/com.metabase/database/jsonschema/1-0-1

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Database connection",
  "self": {
    "vendor": "com.metabase",
    "name": ...
```

### snowplow/iglu-client-embedded/schemas/com.metabase/downloads/jsonschema/1-0-0

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Downloads results from questions and dashcards",
  "self": {
    "vendor": ...
```

### snowplow/iglu-client-embedded/schemas/com.metabase/embed\_flow/jsonschema/1-0-0

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "User interactions with public links, static embedding, and public...
```

### snowplow/iglu-client-embedded/schemas/com.metabase/embed\_flow/jsonschema/1-0-1

```
{
```

## Analyst Base

```
"$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
"description": "User interactions with public links, static embedding, and public..."
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/embed\_flow/jsonschema/1-0-2**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "User interactions with public links, static embedding, and public..."
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/embed\_flow/jsonschema/1-0-3**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "User interactions with public links, static embedding, and public..."
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/embed\_share/jsonschema/1-0-0**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Schema for tracking embedding enabled and disabled events",  
  "self": {  
    ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/embed\_share/jsonschema/1-0-1**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Schema for tracking embedding enabled and disabled events",  
  "self": {  
    ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/embed\_share/jsonschema/1-0-2**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Schema for tracking embedding enabled and disabled events",  
  "self": {  
    ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/embedding\_homepage/jsonschema/1-0-0**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "User interactions with the embedding-homepage",
```

## Analyst Base

```
"self": {  
  "vendor":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/instance/jsonschema/1-0-0**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Metabase instance",  
  "self": {  
    "vendor": "com.metabase",  
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/instance/jsonschema/1-1-0**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Metabase instance",  
  "self": {  
    "vendor": "com.metabase",  
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/instance/jsonschema/1-1-1**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Metabase instance",  
  "self": {  
    "vendor": "com.metabase",  
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/instance/jsonschema/1-1-2**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Metabase instance",  
  "self": {  
    "vendor": "com.metabase",  
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/instance/jsonschema/1-1-3**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
```



## Analyst Base

```
"description": "Metabase instance",
"self": {
  "vendor": "com.metabase",
  "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/instance\_stats/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Schema for daily stats ping, tracking instance metrics and settings",
  ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/instance\_stats/jsonschema/2-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Schema for daily stats ping, tracking instance metrics and settings",
  ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/invite/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "New user invite",
  "self": {
    "vendor": "com.metabase",
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/invite/jsonschema/1-0-1**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "New user invite",
  "self": {
    "vendor": "com.metabase",
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/llm\_usage/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Metabot events",
  "self": {
```

## Analyst Base

```
"vendor": "com.metabase",  
"name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/metabot/jsonschema/1-0-0**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Metabot events",  
  "self": {  
    "vendor": "com.metabase",  
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/metabot/jsonschema/1-0-1**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Metabot events",  
  "self": {  
    "vendor": "com.metabase",  
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/model/jsonschema/1-0-0**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Model events",  
  "self": {  
    "vendor": "com.metabase",  
    "name":...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/question/jsonschema/1-0-0**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "New question creation",  
  "self": {  
    "vendor": "com.metabase",  
    ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/question/jsonschema/1-0-1**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
```

## Analyst Base

```
"description": "New question creation",  
"self": {  
  "vendor": "com.metabase",  
  ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/question/jsonschema/1-0-2**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "New question creation",  
  "self": {  
    "vendor": "com.metabase",  
    ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/question/jsonschema/1-0-3**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Events related to questions",  
  "self": {  
    "vendor": "com.metabase",  
    ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/question/jsonschema/1-0-4**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Events related to questions",  
  "self": {  
    "vendor": "com.metabase",  
    ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/question/jsonschema/1-0-5**

```
{  
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",  
  "description": "Events related to questions",  
  "self": {  
    "vendor": "com.metabase",  
    ...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/question/jsonschema/1-0-6**

## Analyst Base

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Events related to questions",
  "self": {
    "vendor": "com.metabase",
    ...
  }
}
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/search/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Search events",
  "self": {
    "vendor": "com.metabase",
    "name": ...
  }
}
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/search/jsonschema/1-0-1**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Search events",
  "self": {
    "vendor": "com.metabase",
    "name": ...
  }
}
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/search/jsonschema/1-1-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Search events",
  "self": {
    "vendor": "com.metabase",
    "name": ...
  }
}
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/serialization/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Serialization operation",
  "self": {
    "vendor": "com.metabase",
    ...
  }
}
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/serialization/jsonsche**

## Analyst Base

### ma/1-0-1

```
{
  "$schema":
"http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Serialization operation",
  "self": {
    "vendor":...
```

### snowplow/iglu-client-embedded/schemas/com.metabase/settings/jsonschema/1-0-0

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Settings events",
  "self": {
    "vendor": "com.metabase",
    "name":...
```

### snowplow/iglu-client-embedded/schemas/com.metabase/settings/jsonschema/1-0-1

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Settings events",
  "self": {
    "vendor": "com.metabase",
    "name":...
```

### snowplow/iglu-client-embedded/schemas/com.metabase/settings/jsonschema/1-0-2

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Settings events",
  "self": {
    "vendor": "com.metabase",
    "name":...
```

### snowplow/iglu-client-embedded/schemas/com.metabase/setup/jsonschema/1-0-0

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Setup flow",
  "self": {
    "vendor": "com.metabase",
    "name":...
```

## **snowplow/iglu-client-embedded/schemas/com.metabase/setup/jsonschema/1-0-1**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Setup flow",
  "self": {
    "vendor": "com.metabase",
    "name": "...
```

## **snowplow/iglu-client-embedded/schemas/com.metabase/setup/jsonschema/1-0-2**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Setup flow",
  "self": {
    "vendor": "com.metabase",
    "name": "...
```

## **snowplow/iglu-client-embedded/schemas/com.metabase/setup/jsonschema/1-0-3**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Setup flow",
  "self": {
    "vendor": "com.metabase",
    "name": "...
```

## **snowplow/iglu-client-embedded/schemas/com.metabase/simple\_event/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Generic event to track interactions and actions that happen within...
```

## **snowplow/iglu-client-embedded/schemas/com.metabase/task/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Task history",
  "self": {
    "vendor": "com.metabase",
    "name": "...
```

## **snowplow/iglu-client-embedded/schemas/com.metabase/timeline/jsonschema/1-0-0**

## Analyst Base

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Event timelines",
  "self": {
    "vendor": "com.metabase",
    "name": "...
```

### **snowplow/iglu-client-embedded/schemas/com.metabase/upsell/jsonschema/1-0-0**

```
{
  "$schema": "http://iglucentral.com/schemas/com.snowplowanalytics.self-desc/schema/jsonschema/1-0-0#",
  "description": "Upsell events",
  "self": {
    "vendor": "com.metabase",
    "name": "...
```

### **src/metabase/DO\_NOT\_ADD\_NEW\_FILES\_HERE.txt**

DO NOT ADD ANY NEW `metabase.x` NAMESPACEs HERE. ADD THEM TO THE APPROPRIATE MODULES.

NEW FOLDERS (MODULES) ARE OK!

PLEASE RE-READ...

### **src/metabase/api/DO\_NOT\_ADD\_NEW\_FILES\_HERE.txt**

DO NOT ADD ANY NEW NAMESPACEs WITH API ENDPOINTS HERE. ADD THEM TO THE APPROPRIATE MODULES.

PLEASE RE-READ...

### **src/metabase/channel/README.md**

`channel` module

### **src/metabase/cmd/README\_BEFORE\_ADDING\_FILES.txt**

DO NOT ADD ANY COMMAND NAMESPACEs HERE. ADD THEM TO THE APPROPRIATE MODULES.

PLEASE

RE-READ

<https://www.notion.so/metabase/Guide-to-Backend-Module-Organization-19169354c9018046ab46e4234aac905...>

### **src/metabase/cmd/resources/config-file-intro.md**

---

title: "Metabase config file template"

---

## Analyst Base

# Metabase config file template

You can generate this doc page by changing into the top-level Metabase directory and running:

...

clojure -M:doc:ee...

**src/metabase/cmd/resources/config-file-outro.md**

...

**src/metabase/cmd/resources/env-var-intro.md**

---

title: Environment variables

redirect\_from:

- /docs/latest/operations-guide/environment-variables

---

# Environment variables

\_This documentation was generated from source by...

**src/metabase/cmd/resources/other-env-vars.md**

# Other environment variables

**src/metabase/events/DO\_NOT\_ADD\_NEW\_FILES\_HERE.txt**

DO NOT ADD ANY NEW EVENT HANDLER NAMESPACES HERE. ADD THEM TO THE APPROPRIATE MODULES.

PLEASE RE-READ...

**src/metabase/models/DO\_NOT\_ADD\_NEW\_FILES\_HERE.txt**

DO NOT ADD ANY NEW MODEL NAMESPACES HERE. ADD THEM TO THE APPROPRIATE MODULES.

PLEASE

RE-READ

<https://www.notion.so/metabase/Guide-to-Backend-Module-Organization-19169354c9018046ab46e4234aac>  
e905...

**src/metabase/notification/README.md**

Introduction to Notification

**src/metabase/server/README.md**

`metabase.server.\*`

**src/metabase/setup/README.md**



## Analyst Base

`setup` Module

### **src/metabase/sync/README.md**

`sync` Module

### **src/metabase/task/DO\_NOT\_ADD\_NEW\_FILES\_HERE.txt**

DO NOT ADD ANY NEW TASK NAMESPACES HERE. ADD THEM TO THE APPROPRIATE MODULES.

PLEASE

RE-READ

<https://www.notion.so/metabase/Guide-to-Backend-Module-Organization-19169354c9018046ab46e4234aac905...>

### **src/metabase/task/QUARTZ.md**

Quartz Scheduler in Metabase

### **test\_resources/certificates/README.md**

Driver Testing Certificates

### **test\_resources/fake\_ag\_token.txt**

airgap\_eyJhbGciOiJSU0EtT0FFUCIsImVuYyI6IkExMjhDQkMtSFMyNTYifQ.rHUDSPzcF-Kth1pIXbnLIJiEgwx2rABSyj9PU98ubYGa7vUFwxrR9f4-UhF6EYrjvbtOx0QG9rFOSIfDmgG8jhCplyuXpfliXdeeFpYt8Pludf-bmB85gU6hQEpyaTq\_hihPwCjp...

### **test\_resources/ssh/README.md**

ssh test keys

### **test\_resources/ssh/ssh\_test**

-----BEGIN OPENSSH PRIVATE...

### **test\_resources/ssh/ssh\_test\_invalid**

-----BEGIN RSA PRIVATE...

### **test\_resources/ssh/ssh\_test\_passphrase**

-----BEGIN OPENSSH PRIVATE...

### **test\_resources/ssl/README.md**

test certificates

### **test\_resources/ssl/mongo/README.md**

Mongo server with SSL support

# Running the server

### **test\_resources/ssl/oracle/README.md**

Oracle certificates