



# Domino for Practitioners

Rev 3 Training Day

May 4, 2022

Dr. Melanie Veale, Field Data Scientist

►TR/01.ON ►TR/01►03  
►TR/01.ON ►TR/01►03

►RS/02.ON SEARCH ►RS/01  
►RS/02.ON SEARCH ►RS/01

►SEARCH►TR/01►03  
►SEARCH►TR/01►03

# AGENDA

Project Setup

Running Code

Publishing

Operationalizing

Settings and Collaboration

Git in Domino

Data in Domino

Workspaces and Jobs

Environments

On-demand Compute Clusters

Model APIs

Web Applications

Jobs and Scheduled Reports

Launchers

Use Domino programmatically

Integrated Model Monitoring





# End-User Course Objectives

**Upon Completion of this Course, Learners will be able to:**

Describe how to set up a Domino Project.

Explain the differences between running code in Workspaces and Jobs.

Articulate the benefits and use cases for On-Demand Distributed Computing.

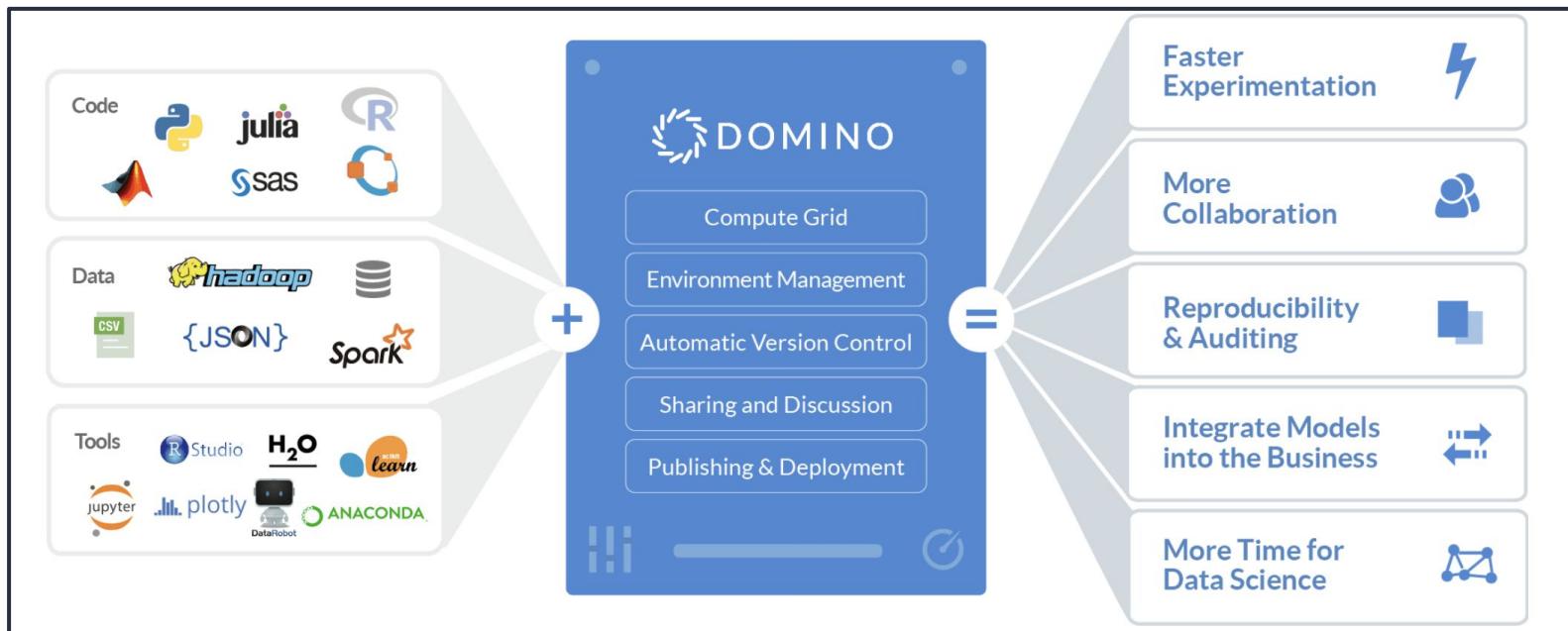
Describe the workflows of publishing and monitoring Model APIs.

Discuss Domino Web Apps, Scheduled Reports and Launchers.

Identify the different ways to interact with Domino programmatically.

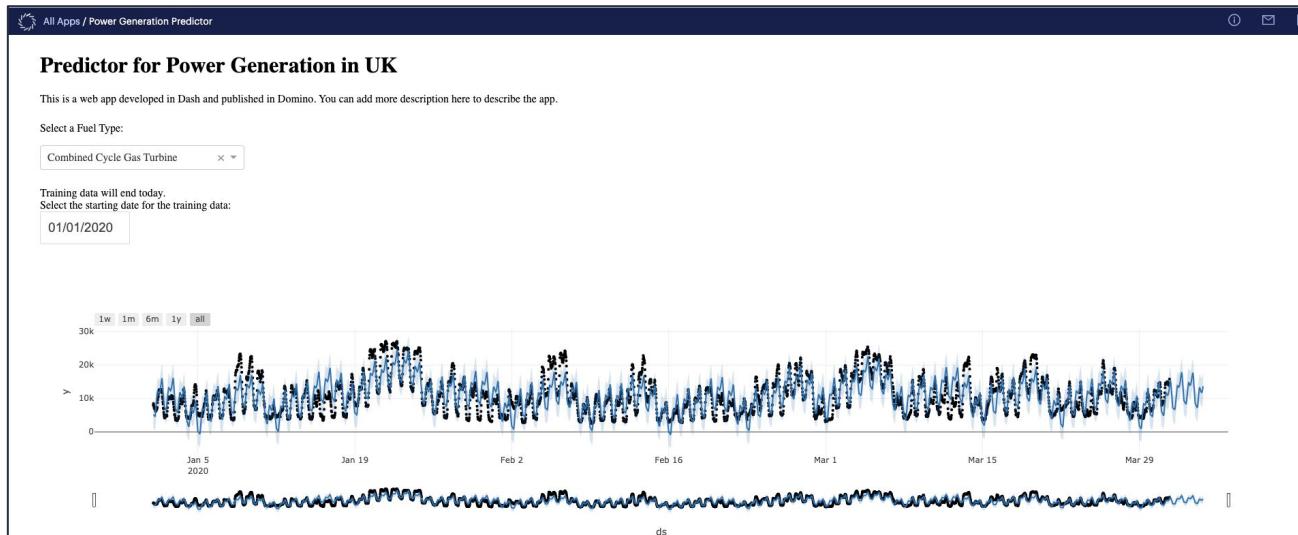
# What is Domino?

Domino makes it easy to access scalable compute, update and share environments, keep track of your experiments and file changes, and easily deploy models

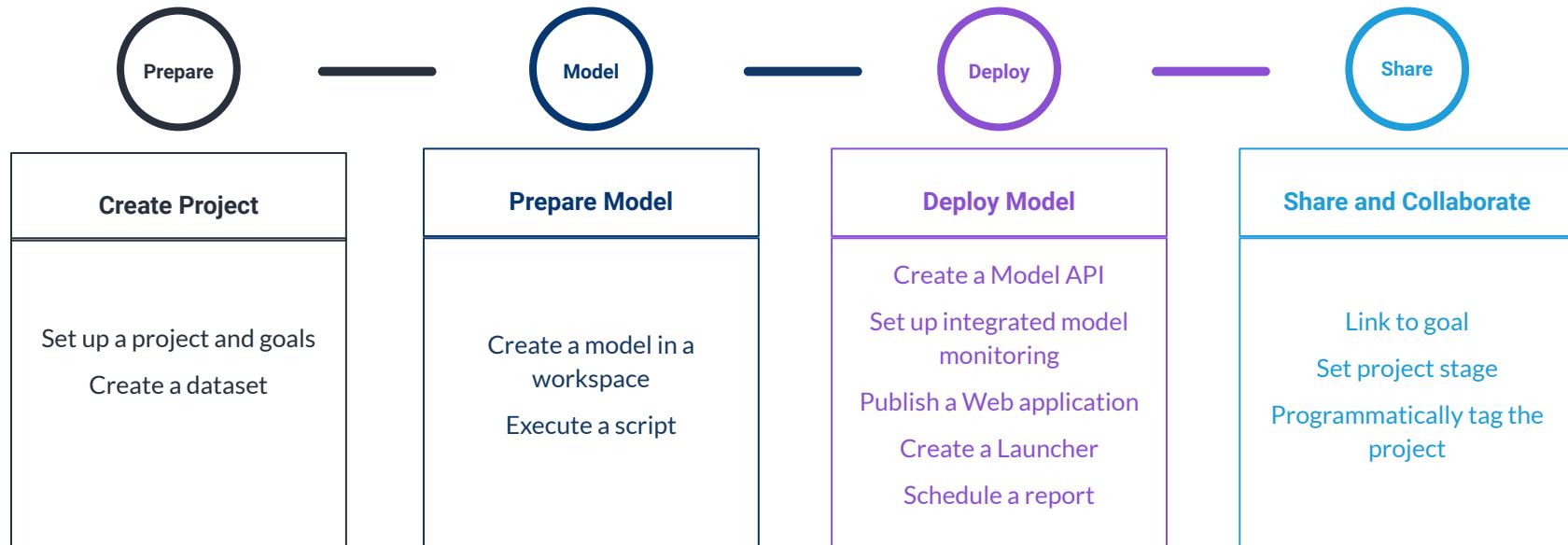


# Data Science Project (Python)

- The activities will use real-world data from the [Balancing Mechanism Reporting Service](#) to predict future electricity generation
- This [tutorial](#) is also covered in depth in our documentation



# Training Objectives (Including Online Courses)



# Self-Service Online Training

- Domino 101, 201, and Publishing are available as online self-service trainings
- Go to [learn.dominodatalab.com](https://learn.dominodatalab.com) and select 'Sign Up'
- Create an account using this access code: 2sypw-iehkb44g

The screenshot shows the 'Sign Up' page for learn.dominodatalab.com. At the top, there are 'Sign In' and 'Sign Up' buttons. The 'Sign Up' button is highlighted with a blue border. Below the buttons are several input fields:

- Access Code**: A text input field labeled 'Access code'.
- First Name** and **Last Name**: Two adjacent text input fields for personal information.
- Email**: A text input field labeled 'Email address'.
- Password** and **Password (Again)**: Two text input fields for password entry.
- What best describes your current role?**: A dropdown menu.
- What is your experience level with Domino?**: A dropdown menu.
- What kind of data science projects do you work on? (optional)**: A text input field for additional information.

At the bottom right is a large 'Sign Up' button.

# Course Goals



- Get comfortable clicking around in Domino
- Know where to find help and information

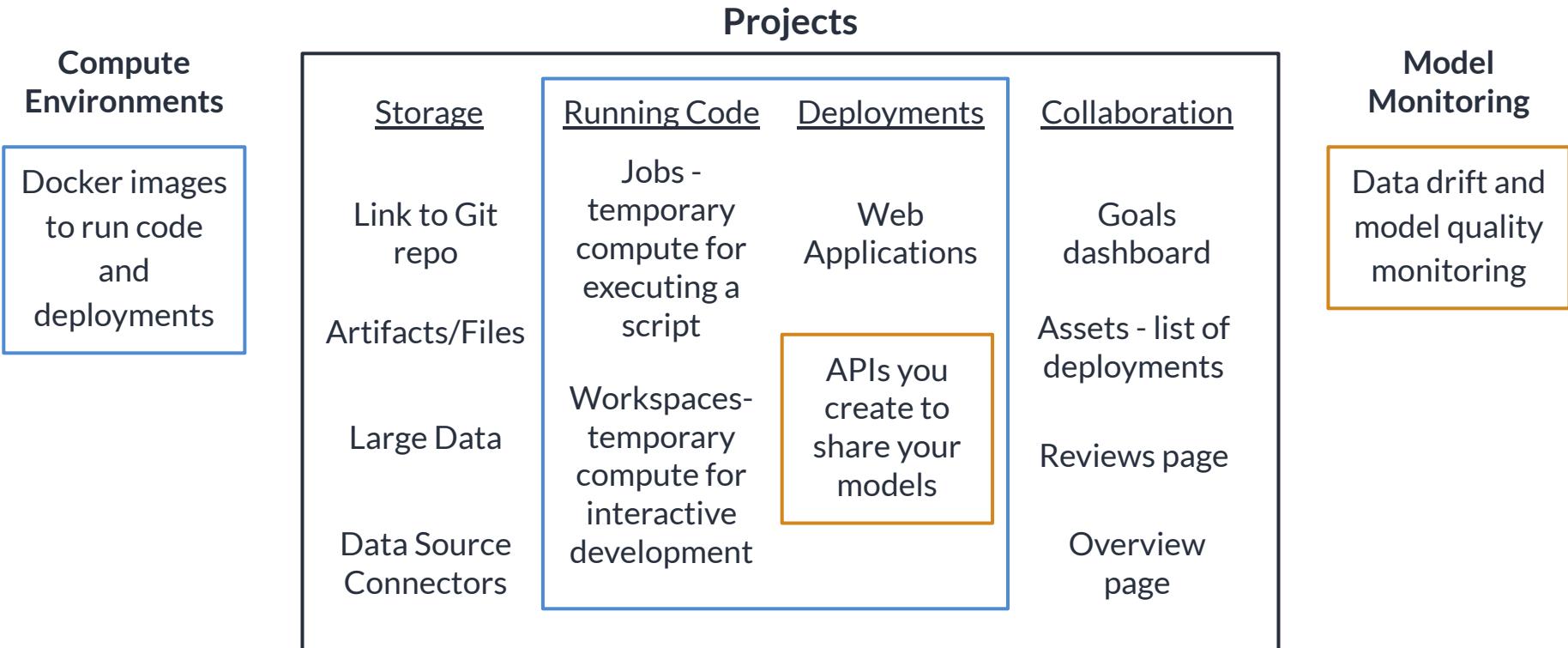


# Project Setup

# Projects: The Main Unit of Organization

- Projects hold all of your files and revision history
- You will run code inside of a project
- Project settings are basic options on how you run your code:
  - Compute environments (software your project needs to run)
  - Hardware tiers (compute resources your project runs on)
  - Access control - who can see and collaborate on your projects

# Domino Overview



# Project Types

## Domino File System (Classic)

- Store code files directly in the Domino project

## Git-Based

- Provides a full Git experience for your code

### New Project

Project Name

Project Visibility

Public (Anyone can view your project.)

Searchable (Discoverable by other Domino users.)

Private (Only viewable or discoverable by your collaborators.)

Code Repository

Domino File System  Git (beta)

[Cancel](#) [Create Project](#)

# Project Settings: Hardware Tiers

- Compute resources for that job or workspace
- Your Domino admin defines these - can be custom for organizations
- Choose a hardware tier that will meet the needs of your workflow

Project settings

Hardware & Environment   Access & Sharing   Results   Exports

Hardware tier

Tier	Resources	Cost	Completion Time
<b>Small</b>	1 core · 4 GB RAM	\$0.002/min	< 1 MIN ⓘ
Small	1 core · 4 GB RAM	\$0.002/min	< 1 MIN ⓘ
<b>Large.2</b>	6 cores · 28 GB RAM	\$0.012/min	< 1 MIN ⓘ
Large.2-spark	6 cores · 24 GB RAM	\$0.012/min	< 1 MIN ⓘ
<b>Medium</b>	2 cores · 8 GB RAM	\$0.03/min	< 1 MIN ⓘ
<b>Large</b>	64 cores · 256 GB RAM	\$0.15/min	< 7 MIN ⓘ
<b>GPU (1 V100) - Global</b>	6 cores · 28 GB RAM	\$0.408/min	< 7 MIN ⓘ

# Project Settings: Compute Environments

Specification for the container where your job or workspace will run

Automatically versioned and easily shared

...More on this later!

The screenshot shows the 'Compute environment' settings page. At the top, there's a search bar containing 'Domino Analytics Distribution Py3.6 R3.6' and a 'Manage Environment' button. Below the search bar, under the 'Global' section, several compute environments are listed: 'Domino Analytics Distribution py3.6 R3.5 - shap & lime', 'Domino Analytics Distribution Py3.6 R3.6' (which is highlighted), 'HyperOpt Domino Analytics Distribution Py3.6 R3.6', 'Image\_Drift\_Keras\_TF', 'OpenAI Gym with Python 3.6', and 'Tensorflow 2.0 Py3.6 R3.5'. Under the 'test' section, there are 'testenv' and 'Private' entries. A note at the bottom right states: 'Name conflicts with value of a variable, just set it'. The interface has a clean, modern design with a light blue header and white background.

# Project Settings: Volume Size

- Controls the amount of storage available to that workspace
- Can be updated in project settings
- Default limits are shown - this is configurable by your Admin

Project settings

Hardware & Environment    Access & Sharing    Results    Exports    Integrations    Archive Project

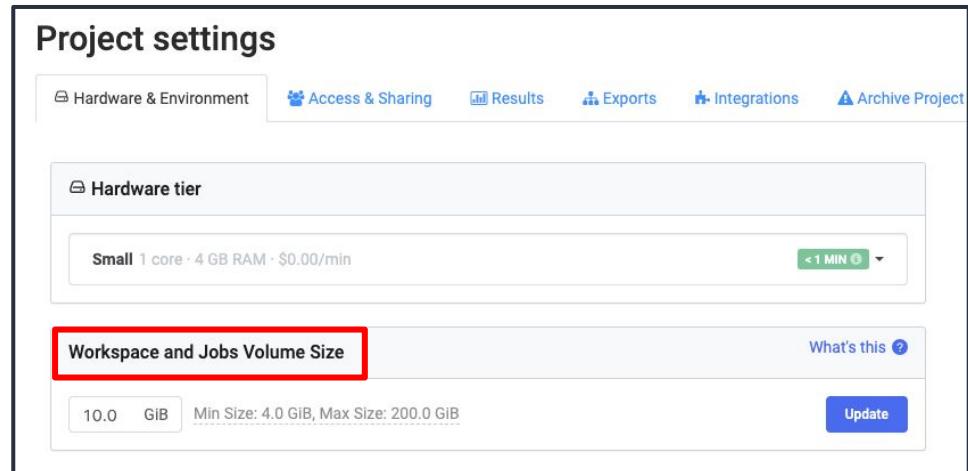
Hardware tier

Small 1 core · 4 GB RAM · \$0.00/min    <1 MIN ⏱

Workspace and Jobs Volume Size

10.0 GiB    Min Size: 4.0 GiB, Max Size: 200.0 GiB    Update

What's this ?



# Project Settings: Access and Sharing

Set the visibility of your project

## Project settings

Hardware & Environment    Access & Sharing    Results    Exports    Integrations    Archive Project or Transfer Ownership

**Visibility** [Learn more](#)

Public: Viewable by anyone  
 Searchable: Discoverable by other Domino users  
 Private: Only viewable or discoverable by your collaborators  
 Allow run execution by anonymous users

**Project Name & Description**

Name:

[Rename Project](#) [Cancel](#)

Description (optional):  
This project contains the code, data, and artifacts for the Getting Started in Domino Tutorial, found here for Python and here for R.

A description makes it easier for colleagues to find your project.

[Save](#) [Cancel](#)

# Project Settings: Access and Sharing

Add colleagues or groups (via organizations) to your project and set permission levels

 Collaborators and permissions Learn more 

**Invite Collaborators**

If your project has Git repositories connected, then collaborators will also need permissions with your Git host in order to start a workspace or make changes to Git files. [Learn more about working with Domino and Git.](#)

You can enter a welcome message here. If left blank - the default Domino welcome message will be sent.

**Invite**

Name	Role	Notifications preference	Remove
domino-andrealowe	★ Owner	Runs started by you (domino-andrealowe)	Cannot remove owner
domino-johnjoo	<input checked="" type="checkbox"/> Contributor Results Consumer Launcher User Project Importer	Runs started by domino-johnjoo	<b>Remove</b>



# Project Settings: Permissions

Collaborators on your projects can have different permissions levels

- Contributor - read/write files, start runs
- Results Consumer - read files and access published apps
- Launcher User - run Launchers and access those results
- Project Importer - import the project
- Owner - archive, change collaborator types, alter settings

# Organizations in Domino

Domino lets you group users into [Organizations](#)

Use organizations to:

- Permission projects to many users at once and easily add/remove collaborators from multiple projects
- Assign *owner* level permissions to a group of users
- Share compute environments
- Restrict hardware tiers to specific groups (controlled by Admins)

# Project Settings: Emails

Configure email settings about Job activity by you and your collaborators

The screenshot shows the Domino Project Settings interface. On the left, there's a sidebar with project navigation: Overview, Files, Workspaces, Jobs, Scheduled Jobs, Publish, Reviews, Activity, and Settings. The Settings option is currently selected. The main area has two tabs: 'Public: Viewable by anyone' (selected) and 'Private: Only viewable or discoverable by your collaborators'. Below these are checkboxes for 'Allow run execution by anonymous users' and 'Allow run execution by anonymous users' (disabled). To the right, there's a 'Rename Project' dialog with fields for 'Name' (set to 'Intro-to-Domino') and 'Description (optional)' (containing a note about the project being a tutorial). Below the dialog is a note: 'A description makes it easier for colleagues to find your project.' At the bottom are 'Save' and 'Cancel' buttons. The main content area below shows the 'Collaborators and permissions' section. It includes an 'Invite Collaborators' form with a placeholder 'Username, email address, or organization name' and a 'Welcome message' input field. An 'invite' button is present. A table lists collaborators: 'domino-andrealowe' (Owner) and 'domino-johnjoo' (Contributor). The 'Notifications preference' dropdown for domino-johnjoo is open, showing options: 'Runs started by you (domino-andrealowe)', 'Never', 'Runs started by domino-johnjoo', and 'All runs'. The 'Runs started by domino-johnjoo' option is selected. A 'Remove' button is also visible for the collaborator row.

# Projects: Fork/Merge

domino-andrea-admin

Intro-to-Domino Ideation

Quick Action

Overview

Activity

Workspaces

Jobs

Scheduled Jobs

Files

Datasets

Reviews

Publish

Settings

Small (Kubernetes)

Getting-Started...

< Back to Projects

## Intro-to-Domino

Updated May 26th 2020

About Manage

### Tags & Description

Tags: No tags [+](#)

Description: No description provided. [Edit](#)

### Readme

This project contains the code, data, and artifacts for the *Getting Started in Domino* Tutorial, found [here](#) for Python and [here](#) for R.

This tutorial will guide you through a common model lifecycle in Domino. You will start by working with data from the Balancing Mechanism Reporting Service in the UK. We will be exploring the Electricity Generation by Fuel Type and predicting the electricity generation in the future. You'll see examples of Jupyter, Dash, pandas, and Prophet used in Domino.

Table of Contents:

- Forecast\_Power\_Generation.ipynb
- Scheduled\_Forecast\_Power\_Generation.ipynb
- Forecast\_Power\_Generation\_for\_Launcher.ipynb
- forecast.ipynb
- forecast\_predictor.py
- data.csv
- app.sh

**Fork domino-andrea-admin/Intro-to-Domino**

Forking this project will create a copy of it under your account, so you can modify it as you wish. Your new project will contain the same files, but none of the Run history or project settings.

Name of new project

domino-andrea-admin

[Cancel](#) [Fork](#)

Not used for Git-Based Projects

Hardware Small (Kubernetes)

Environment Getting-Started-in-Domino

Total Runtime 9 days

Apps 1

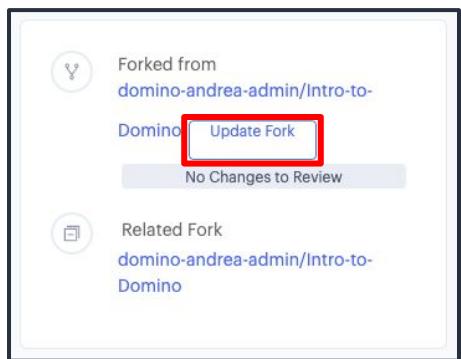
Jobs 1

Workspaces 16

Comments 0

# Projects: Fork/Merge

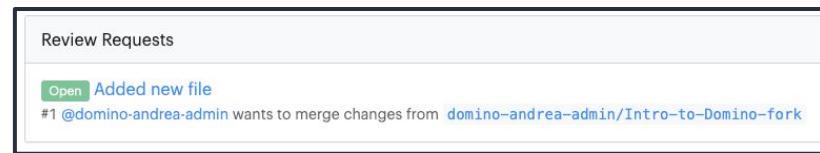
Bring changes from original project into your forked project:



Request to merge changes from forked project into original:



Review changes from forked versions of your project:



# Projects: Versioning and Reproducibility

You can revert your project or any individual file to any previous state

In Git-Based Projects this page is labeled Artifacts

The screenshot shows the Domino platform interface. On the left is a dark sidebar with various icons and links: a circular icon, a magnifying glass, a grid, a file folder, a workspace icon, a double arrow, a clock, a publish dropdown, reviews, activity, settings, small, and Domino Analytics. Below these are a pencil icon, a question mark, and a blue AL icon.

The main area has a header with "Local", "Other Projects", and "Git Repositories". It displays a project named "Intro-to-Domino /". A message indicates that "domino-andrealowe committed fd4440b "Reverted requirements.txt" (job #48) on April 2, 2020 @ 04:17 pm". A note says, "This is an old version of your files . The latest version was made at April 2nd 2020 @ 4:17 pm [View Latest](#)".

A red box highlights the "Revert Project" button in the top right corner of the header.

The table below lists the project's artifacts:

Name	Size	Modified
results	4.9 KB	
.ipynb_checkpoints	1.2 MB	
data.csv	113 KB	April 2nd 2020, 12:31:07 pm
README.md	837 B	February 28th 2020, 3:01:49 pm
model.pkl	0 B	April 2nd 2020, 12:31:48 pm
forecast_launcher.sh	138 B	February 12th 2020, 5:40:24 pm
requirements.txt	63 B	February 12th 2020, 5:29:17 pm
.dominoresults	313 B	February 6th 2020, 12:38:59 pm
Forecast_Power_Generation.ipynb	386 KB	April 1st 2020, 5:26:38 pm
forecast.ipynb	604 KB	March 30th 2020, 12:31:26 pm
app.py	9.3 KB	February 13th 2020, 9:15:14 am
Forecast_Power_Generation_for_Launcher.ipynb	389 KB	February 12th 2020, 5:39:39 pm
forecast_predictor.py	378 B	February 12th 2020, 5:45:05 pm
app.sh	13 B	February 13th 2020, 9:15:41 am

A blue circular icon with a white square is in the bottom right corner.

# Projects: Goals

Define goals and link code, files, or artifacts to them

Add comments and discussion around each goal

Track stages of goals separately from the project

The screenshot shows the Domino platform's interface for managing projects. On the left, a sidebar menu is open for the project "Akshay-Intro-To-Domino". The "Overview" tab is selected and highlighted with a red box. Other tabs in the sidebar include Ideation, Activity, Reviews, RUN, Workspaces, Jobs, MATERIALS (Data and Files), PUBLISH (Scheduled Jobs, App, Model APIs, Launchers), and Settings. To the right of the sidebar, the main content area displays the project details for "Akshay-Intro-To-Domino", updated on May 3rd, 2022. The "Manage" tab is also highlighted with a red box. Below this, the "Tags & Description" section shows "No tags" and "No description provided.". At the bottom of the screen, there are three icons with descriptions: a target icon for defining key milestones, a line graph icon for tracking high-level deliverables, and a pie chart icon for summarizing and communicating objectives. A blue "+ Add Goals" button is located at the bottom right.

# Notifications

Your Admins can send you notifications in Domino

These are different from the Project run notifications

The indicator color matches criticality:

- Default
- Critical

The screenshot shows the Domino application interface. On the left is a sidebar with the following items:

- Domino
- Search
- Lab
- Data
- Projects
- Environments
- Model APIs
- Tags

Below the sidebar is a main content area titled "Projects". It shows a single project named "monitoring-test" with the following details:

- Color: Green (indicates Default criticality)
- Name: monitoring-test
- Type: Ideation
- Status: No tags
- Description: No project description available.
- Last updated: Latest Results: January 31st 2022, 12:46 pm
- Status: 1 Stopped

At the bottom of the sidebar, there is a "Notifications" item, which is highlighted with a red box. Other items at the bottom include "Admin", "Help", and "integration-test".

# Activity: Fork a Project

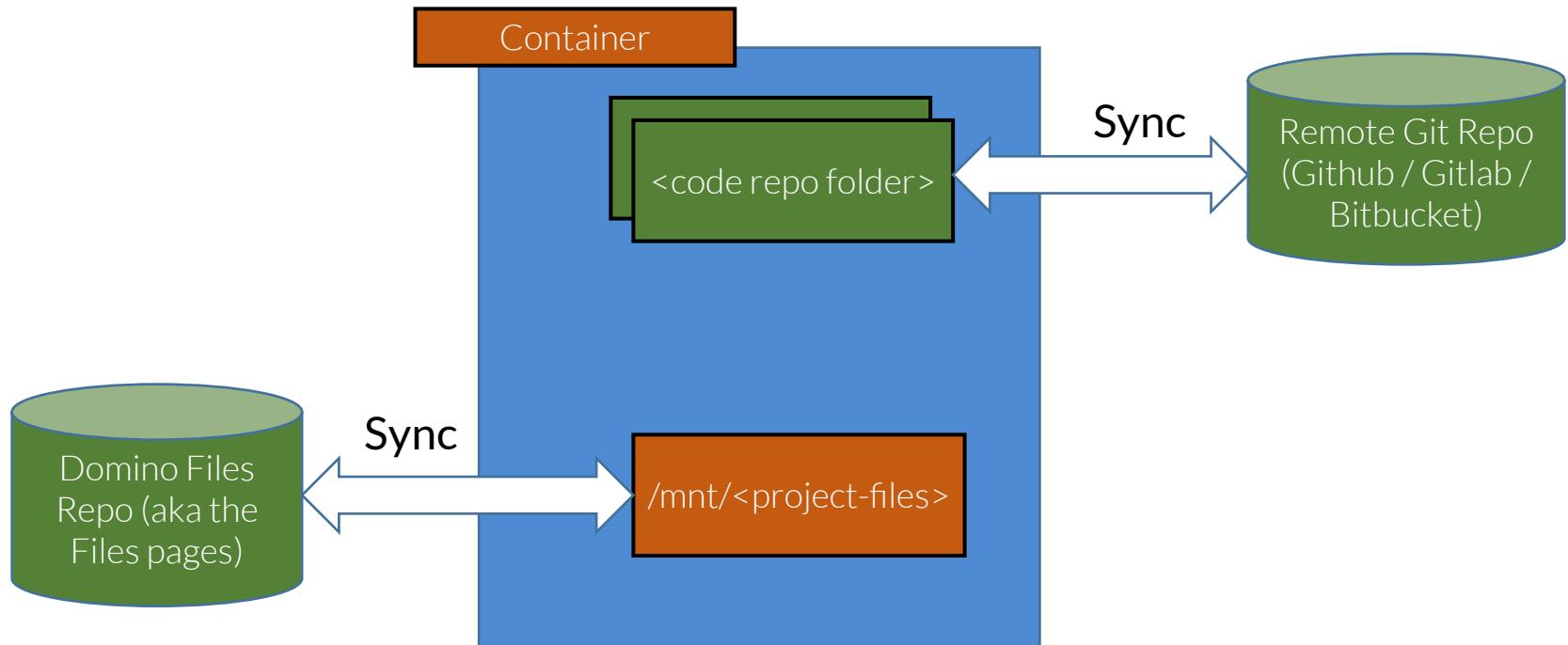
1. Sign into temporary training Domino instance and open the project:

[https://rev.workshop.domino.tech/u/melanie\\_veale/Intro-To-Domino/overview](https://rev.workshop.domino.tech/u/melanie_veale/Intro-To-Domino/overview)

2. Fork the project from the Overview page and name it '<your name>-Domino-Training'
3. Navigate to settings page:
  - a. Make sure project is set to private
  - b. Note Environment and Hardware Tier
4. Add a goal to the project called 'Train and serialize model'

# Git in Domino

# Integration with Git: Separation of Version Control Systems



# Git Based Projects

Users can build a project on an existing git repo

A new GitHub or GitLab repository can also be created directly

Create New Project

Hosted By

Project demo-git-based-project, Private

Code Github

Git Service Provider

Git Credentials

Repository

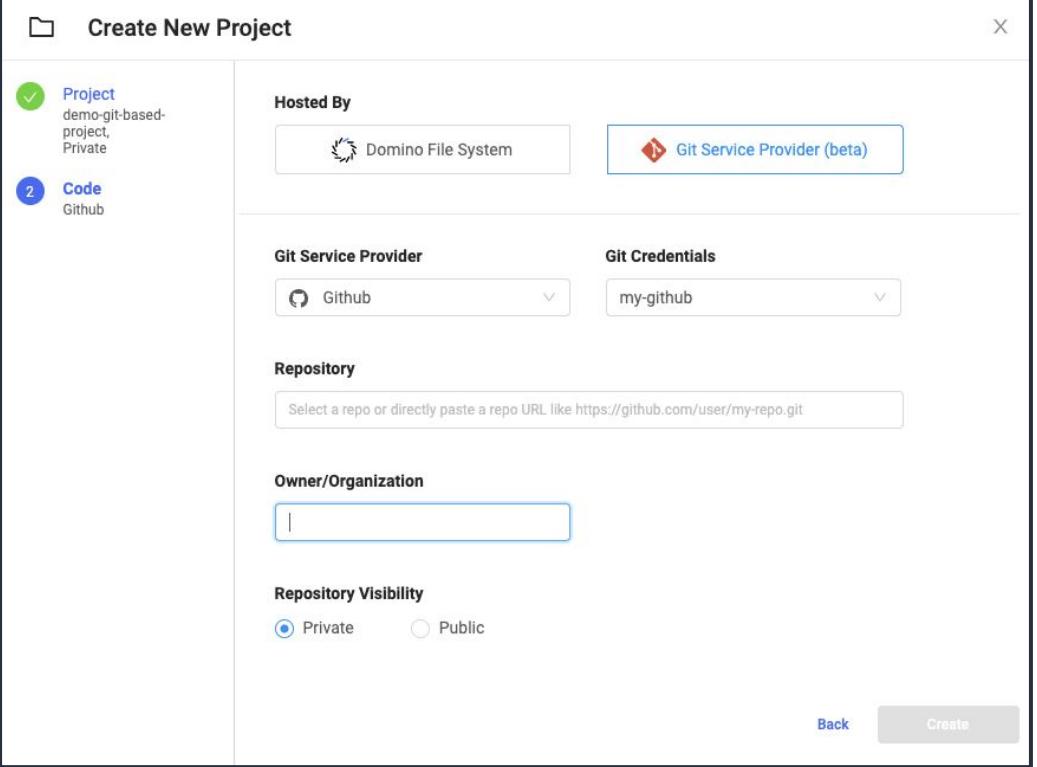
Select a repo or directly paste a repo URL like https://github.com/user/my-repo.git

Owner/Organization

Repository Visibility

Private Public

Back Create



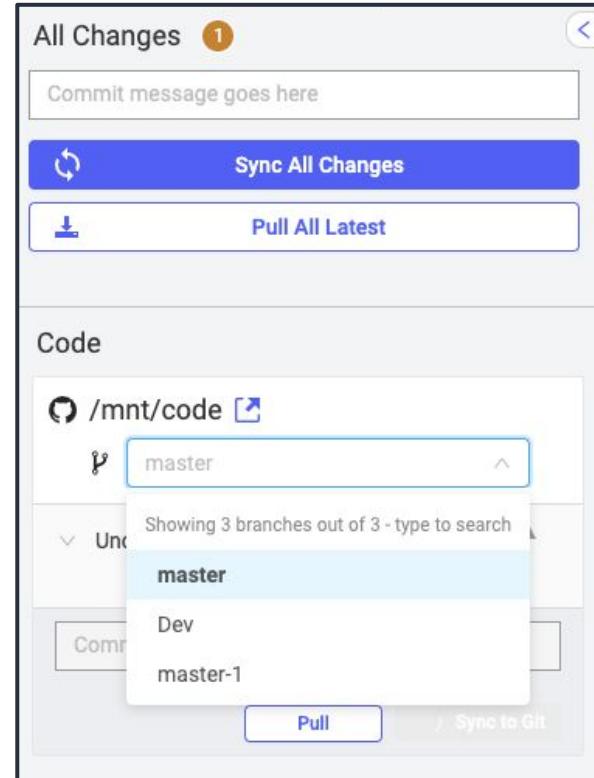
# Import Git Repos

Any number of git repos can be imported into a standard Domino project or a git-based project

The screenshot shows the Domino interface with a sidebar on the left containing navigation links: Overview, Activity, Reviews, RUN, Workspaces, Jobs, MATERIALS, Data, and Files. The 'Files' link is highlighted with a red box. The main content area has tabs at the top: Local, Other Projects, and GIT Repositories, with 'GIT Repositories' also highlighted with a red box. Below the tabs, there's a section titled 'Import from external Git repositories' with the sub-instruction 'This project can import files from external Git repositories hosted by services like GitHub.' A blue link 'Learn more about repositories.' is present, followed by a small question mark icon. To the right, a button labeled 'Add a New Repository' is enclosed in a red box. The status message 'Not importing any repositories.' is displayed at the bottom.

# Git Integration

- Make sure to add authentication before setting up git integration
- You can change branches and resolve conflicts directly in Domino
- Branch updates and status will be reflected in your Workspaces
- See details in [Domino 101](#)



# Data in Domino

# Domino Data Connectors

Create a data source in projects or workspaces

Data

Data Sources    Domino Datasets    External Volumes



You can connect to external data sources like Snowflake, Redshift, S3 etc.

Connect to External Data

model-monitoring-demo : andrea-demo's Jupyter (Python, R, Julia) session

Data

Data Sources



There are no Data Sources currently added to this Workspace

+ Add a Data Source

# Create a Data Source

Choose from:

- [Snowflake](#)
- AWS [S3](#) and [Redshift](#)
- MySQL
- Microsoft SQL Server  
(MSSQL)
- PostgreSQL
- Oracle Database
- Azure Data Lake Storage
- Google Cloud Storage

New Data Source

X

1 Configure      2 Authenticate      3 Permissions

Select Data Store

Select a data store

Snowflake

Amazon Redshift

Amazon S3

Account Name

Database (Optional)

Schema (Optional)

Warehouse (Optional)

Role (Optional)

Data Source Name

Description (Optional)

A clear description helps others understand the purpose of this data source

# Create a Data Source

Data Sources / snowflake-demo

## snowflake-demo

Settings Dependent Projects

### Details

DATA STORE Snowflake	OWNER andrea-demo	CREATED January 7th, 2022
DESCRIPTION —		LAST UPDATED January 7th, 2022

### Configuration

ACCOUNT NAME kma55258	DATABASE YELLOW_CAB	SCHEMA PUBLIC
WAREHOUSE LOAD_TESTDATA_WH		ROLE —

### Snowflake Authentication

USERNAME SYSTEM_TEST_USER	PASSWORD *****
CREDENTIAL TYPE Individual Credentials	

### Delete Data Source

Deleting will permanently remove the connection details and any associated user credentials provided for this data source. To use this data source in any executions, you will need to configure it again as a new data source.

[Delete Data Source](#)

### User Permissions

edit

Users with permissions can view and use this data source in projects. Data access still requires valid individual or service account credentials.

andrea-demo	Owner
-------------	-------

# Use Data Connectors

Workspaces will provide an option to copy code for connecting to the data source

Use similar code in your Job scripts, see [here](#) for more details

The screenshot shows a 'Data' workspace interface. At the top, there's a 'Data Sources' tab (which is selected) and a 'Datasets' tab. A blue button labeled '+ Add a Data Source' is visible. Below the tabs, there's a data source card for 'snowflake-demo'. The card includes a small icon of a snowflake, the name 'snowflake-demo', 'Added by: andrea-demo', and 'Last accessed: January 7th, 2022'. To the right of the card is a clipboard icon. At the bottom of the card is a black button with white text that says 'Click to copy to clipboard'.

```
from domino.data_sources import DataSourceClient  
  
# instantiate a client and fetch the datasource instance  
ds =  
DataSourceClient().get_datasource("snowflake-demo")  
  
# res is a simple wrapper of the query result  
res = ds.query("select * from {{TABLE NAME}} limit 100")  
  
# to_pandas() loads the result into a pandas dataframe  
df = res.to_pandas()
```

# Data Source Permissions

Admins can add service accounts

Admins can make data sources accessible to all, while users must add individuals or organizations

Permissions are checked as follows:

- **Workspaces and Jobs** - user who started the execution
- **Launchers** - user who started the launcher regardless of who created the launcher
- **Domino Apps** - user who published the app regardless of who is accessing the app
- **Model API** - no user identity, it is possible to inject an API key into an execution through an environment variable, and then use it explicitly when retrieving a data source

# Other Data Connections

To connect to other outside data sources you'll need:

- Network connectivity
- Package or driver
- Credentials

See detailed instructions [here](#), or check out the Domino 101 data section

 teradata.	✓	<ul style="list-style-type: none"><li>• Connecting to Teradata from Domino</li></ul>
 PostgreSQL		<ul style="list-style-type: none"><li>• Connecting to PostgreSQL from Domino</li></ul>
 Microsoft SQL Server		<ul style="list-style-type: none"><li>• Connecting to Microsoft SQL Server from Domino</li></ul>
 cloudera IMPALA		<ul style="list-style-type: none"><li>• Connecting to Impala from Domino</li></ul>
 ORACLE DATABASE		<ul style="list-style-type: none"><li>• Connecting to Oracle DB from Domino</li></ul>
 MySQL®		<ul style="list-style-type: none"><li>• Connecting to MySQL from Domino</li></ul>

# Domino Environment Variables

Use [environment variables](#) to store credentials securely.

## Project level environment variables

- Accessible to anyone with access to project
- Added via Project Settings page

## User level environment variables

- Accessible to only you when executing Runs
- Added via Account Settings page

The screenshot shows the 'Environment variables' section of the Domino Project Settings. It displays two project-level environment variables: 'S3\_KEY' and 'S3\_SECRET'. Each variable has a 'Remove' button to its right. Below this, there is a 'Set environment variable' form with fields for 'Name' and 'Value', and a '\$ Set variable' button. A note at the bottom states: 'These variables can be retrieved from the operating system's environment at runtime. To change the value of a variable, just set it again. Values are passed verbatim, without any escaping.'

Example use in R:

```
sys.getenv("S3_KEY")
```

Example use in Python:

```
import os  
os.environ["S3_KEY"]
```

# Datasets vs Project Files

## Project Files:

- Files are synced whenever a code session is launched or stopped
- Large or many files can slow down these launch/sync times
- Limits based on project volume size
- Can only import data by importing entire project
- Automatically versioned

## Datasets:

- Stored in EFS
- Mounted to the machine for use during executions, no copying
- No copying means no additional wait for launching and syncing
- Large file size and large numbers of files allowed
- Manually versioned

# Datasets Overview

Each project will start with a default dataset

Local Datasets are always read/write - contents can be modified from the browser, Workpaces, Jobs, Apps, and Launchers

Shared Datasets are read only

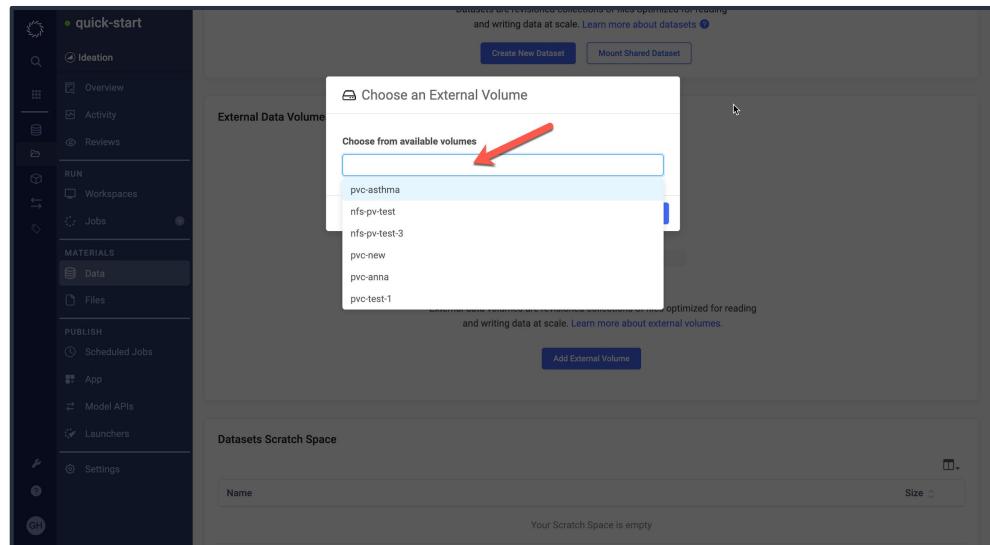
Your Datasets

Name	Description	Path	Snapshots
demo-project	This is the default dataset provided for your project and will be available in all your executions. You can add/remove data, rename or delete this dataset.	/domino/datasets/local/demo-project	0

[Create New Dataset](#)

# External Data Volumes Overview

- Domino projects can access data stored in external data volumes
  - Currently supports generic volumes as well as NFS, EFS, and SMB
- Setup must be done by an Admin
- Access is controlled separately from projects - can be limited to specific users or organizations



# Running Code in Domino *Workspaces*

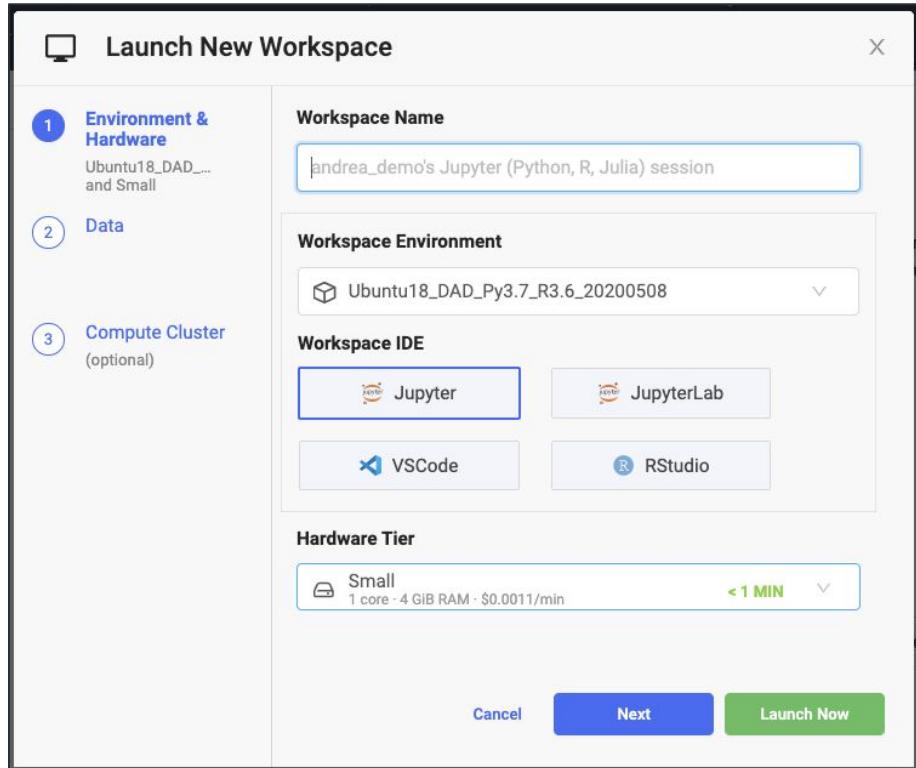
# Run Code in Domino

	<u>Workspaces</u>	<u>Jobs</u>
<b>For example</b>	RStudio Jupyter Notebook SAS Studio	my_script.py another_script.R wrapper_script.sh
<b>Saving to Domino</b>	Manual	Automatic
<b>Resource shutdown</b>	Manual	Automatic

# Configure a Workspace

Configure your workspace at launch:

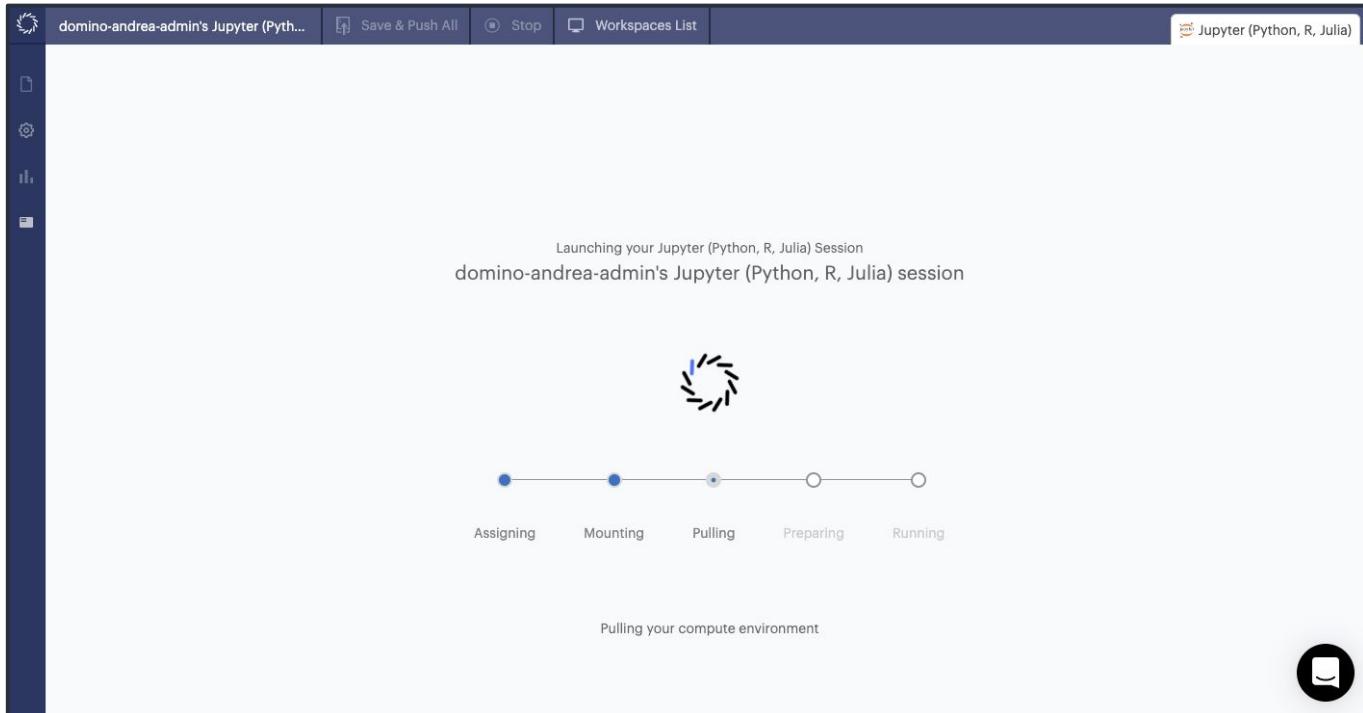
- Name (optional)
- Hardware Tier
- Environment
- Workspace IDE
- Data
- On-Demand Spark Cluster



# Using a Workspace

While the workspace is starting up, you can see:

- Status of your starting execution
- Logs for your starting execution



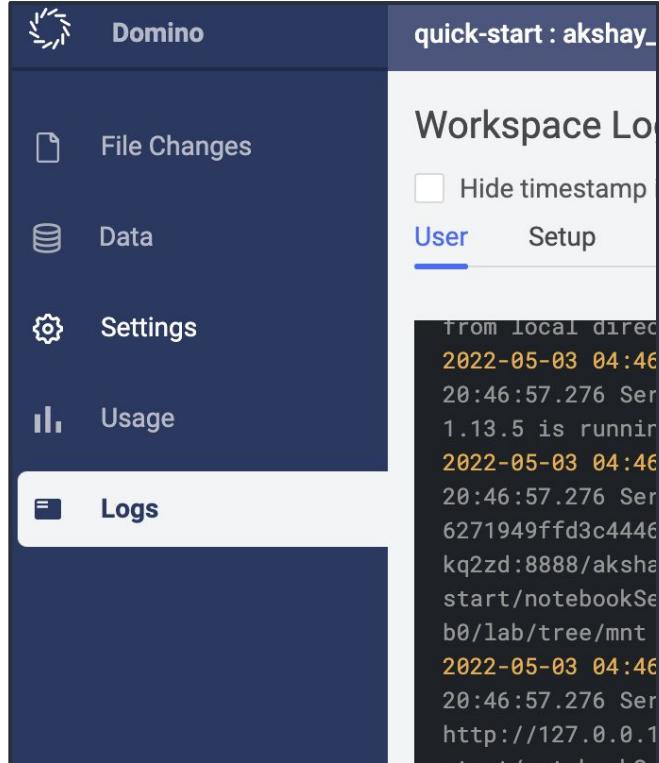
# Workspace Launch Process

<u>Stage</u>	<u>What's Happening</u>	<u>Wait Time Depends On...</u>
<b>Assigning</b>	Finding resources for your session	Availability of machines and launch time if needed
<b>Mounting</b>	Attaching persistent volume (files) to pod	Internal processes
<b>Pulling</b>	Copying compute environment (Docker image)	Level of customization to compute environment
<b>Preparing</b>	Launch Domino processes and move files into volume	Size/number of files

# Workspace Navigation Bar

On the left of the screen are the panes for

- **File Changes**, where changes to your files are tracked and synced
- **Data**, where your data sources, datasets and snapshots are tracked
- **Settings**, where you can see and edit your workspace configuration
- **Usage**, where you can view how your workspace is consuming resources
- **Logs**, where you can see the setup and user logs your workspace is generating



# Workspace Navigation Bar: Resource Usage

The screenshot shows the Domino workspace interface. On the left is a dark sidebar with icons for File Management, Settings, Usage (which is selected), and Logs. The main area has a title bar "domino-austin's JupyterLab session" and a "Save & Push All" button. Below this is a "Resource Usage" section with two graphs: "CPU Usage" (73.10% over 5 minutes) and "Memory Usage" (164 MB / 1024 MB). To the right is a file tree with the following structure:

- results
- vscode-set
- bignb.ipynb
- spark-test.ipynb
- Untitled.ipynb
- Untitled1.ipynb
- afile.thing
- Y: domino.yaml
- hello.py
- mynewfile.m
- README.m
- spark\_job\_1
- syncme.txt
- wait\_two\_h
- xss.html
- xss2.html

In the **Usage** pane, you'll find graphs of your workspace **CPU Usage** and **Memory Usage** over time.

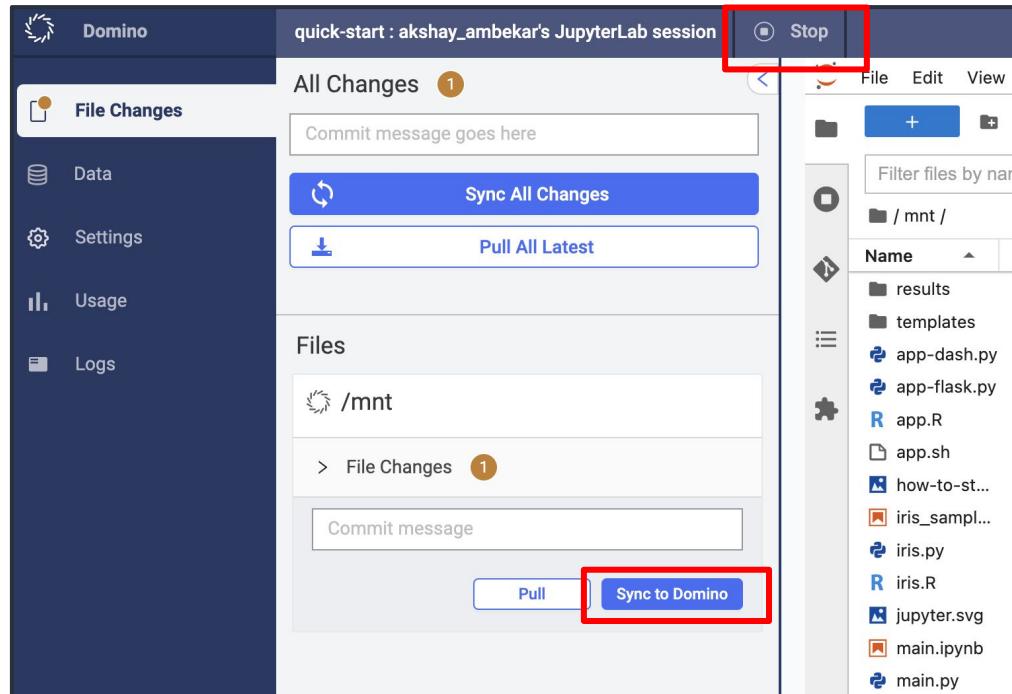
# Workspace Sync and Stop

Sync work in left navigation bar

Important - Stop session when finished to save on costs

Closing the browser tab does *not* stop a workspace

You can Stop workspaces without committing files



# What Restarts?

Persists between Sessions:

- Everything in the *Files* page (Classic Project) or *Code* and *Artifacts* folders (Git-Based Project)
- Workspace settings (Hardware tier, spark settings, volume size, etc)

Does not persist between Sessions:

- Any objects in memory
- Files outside of the */mnt* directory (*Files* page) including pip installed packages
- Compute Environment - it will load the latest version on restart

# Workspace Dashboard

The workspaces dashboard will show all stopped and running workspaces

See the state of the environment, files, and hardware for every workspace

The screenshot shows the Domino Workspace Dashboard interface. At the top, there are tabs: 'My Workspaces' (with a blue info icon), 'All Workspaces' (selected, indicated by a blue underline), 'Deleted', and 'Utility'. On the right, there are buttons for '+ Create New Workspace' and 'Open Last Workspace'.

The main area displays two workspace entries:

- Running:** andrea-demo (OWNER). The status is 'Running' with a green background. It features a laptop icon with a Jupyter logo on the screen. Below the icon are 'Open' (green button) and 'Stop' (blue button) buttons. To the right, there is a red box highlighting the 'Settings', 'Usage', 'Logs', and 'History' buttons. Below this section, it says 'andrea-demo's Jupyter (Python, R, Julia) session' with links to 'Edit' and 'Logs'. It also shows the start time 'Feb 16th, 2021 @ 04:01:09 PM' and 'LAST STARTED'.
- Stopped:** andrea-demo (OWNER). The status is 'Stopped' with a light blue background. It features a laptop icon with a Jupyter logo on the screen. Below the icon is a 'Start' (blue button). To the right, there is a red box highlighting the 'Settings', 'Usage', 'Logs', and 'History' buttons. Below this section, it says 'andrea-demo's Jupyter (Python, R, Julia) session' with links to 'Edit' and 'Logs'. It also shows the start time 'Feb 16th, 2021 @ 04:00:50 PM' and 'LAST UPTIME'.

# Workspace Limits

- Configurable by your Admin
- Defaults:
  - Four workspaces per *user per project*
  - Sixteen workspaces *total* per user

# Deleting a Workspace

**Deleting a workspace  
stops it permanently**

**Important:** Sync work  
before deleting!

Select *History* for more  
settings and logs from  
previous session

The screenshot shows the Domino platform's workspace management interface. At the top, there are tabs: 'My Workspaces' (with a blue notification badge), 'All Workspaces', 'Deleted' (which is highlighted in blue), and 'Utility'. To the right of these are buttons for '+ Create New Workspace' and 'Open Last Workspace'. Below the tabs, a list of workspaces is displayed. One workspace, 'andrea-demo', is shown in a card with a 'Deleted' status indicator. The card includes a thumbnail of a laptop displaying a Jupyter notebook interface, the owner 'andrea-demo OWNER', and a link to 'andrea-demo's Jupyter (Python, R, Julia) session'. A timestamp indicates it was 'LAST STARTED' on 'Feb 16th, 2021 @ 04:00:50 PM'. To the right of the card are three buttons: 'Settings', 'Logs', and 'History', with 'History' being the one highlighted with a red box. Below the card, there are three sections: 'Code' (managed by DFS), 'Environment & Hardware' (Domino Analytics Distribution Py3.8 R4.0, 2021Q1 - Revision #1, Small hardware tier, 10 GiB volume size), and 'Data' (no datasets configured). A close button 'X' is located at the top right of the workspace card.

# Set Automatic Time Limits for Workspaces

The screenshot shows the Domino web interface with a dark blue sidebar on the left and a light gray main content area.

**Left Sidebar:**

- Domino
- Search
- Lab
- Datasets
- Projects
- Environments
- Model APIs
- Tags

**User Profile:** domino-andrealowe

**Main Content Area:**

- Account Settings
- Login Profile
- Notifications
- API Key
- Workspace Settings** (highlighted in blue)
- Project Environment Default
- User Environment Variables
- Manage Compute Environments
- Organizations
- Kerberos Integration
- Git Credentials
- Jira Credentials

**Workspace Settings Sub-Panel:**

Shutdown my long-running workspaces after  ▾

**Bottom Navigation Bar:**

- Admin
- Help
- Download CLI
- Logout

**Bottom Right Corner:** DOMINO

# Activity: Develop a Model

1. Navigate to the Workspaces tab in your training project
2. Select the 'JupyterLab' button (optionally name the workspace)
3. Open the Forecast\_Power\_Generation.ipynb
4. Run all cells in order
  - a. Note how we install a new package directly into the notebook in cell
  - b. Observe how the serialized model is exported at the end
5. **Sync** your files, then **Stop** your session
6. Navigate to the Files tab and note the updated files
7. Link your model.pkl file to the goal we set up earlier

# Running Code in Domino *Jobs*

# Domino Jobs Overview

- Execute R, Python, or Bash scripts
- Domino keeps a snapshot of all files the code creates or modifies
  - Results between Jobs can be compared
  - Custom metrics can be tracked in the dashboard
- Multiple Jobs can be started at once and these are run in parallel
- Run Jobs from the Jobs dashboard or the code file itself

# Jobs Dashboard

Dominostats can be used to track custom metrics



# Jobs Dashboard

Tag Jobs to create separate dashboards for different experiments

The screenshot shows the Domino interface for managing jobs. On the left, a dark sidebar lists various sections: Akshay-Intro-To-Domino, Overview, Activity, Reviews, RUN (Workspaces, Jobs, +), MATERIALS (Data, Files), PUBLISH (Scheduled Jobs, App, Model APIs, Launchers). The 'Jobs' section is currently selected. In the main area, there's a chart titled 'Timeline shows your Diagnostic Stats across your jobs. Learn more about Diagnostic Stats in Domino'. Below it, a search bar has 'Q prod' typed into it, with a red box highlighting the search input field. To the right of the search bar is a table header with columns: NO., TITLE, COMMAND, STARTED, USER, CPU(%), MEMORY (GiB). A single row is visible: NO. 7, TITLE Test Powe, COMMAND launcher\_forec, STARTED 2 minutes ago, USER akshay\_ar, CPU(%) 0.36, MEMORY (GiB) 0, DURATION 14 s. At the bottom of the table, it says 'Showing 1 - 1 out of 1'. To the right of the table, there's a sidebar with a 'Test Powe' section containing 'Logs' and 'Results' tabs, both of which are highlighted with red boxes. Below this is a 'Search' bar and a link to 'results/launcher\_2003\_data.csv'. At the bottom right, there's a preview of a CSV file with 'HDR (0)' and 'HALF HOURLY OUTTURN GENERATION BY FUEL TYPE DATA (1)'.

NO.	TITLE	COMMAND	STARTED	USER	CPU(%)	MEMORY (GiB)
7	Test Powe	launcher_forec	2 minutes ago	akshay_ar	0.36	0

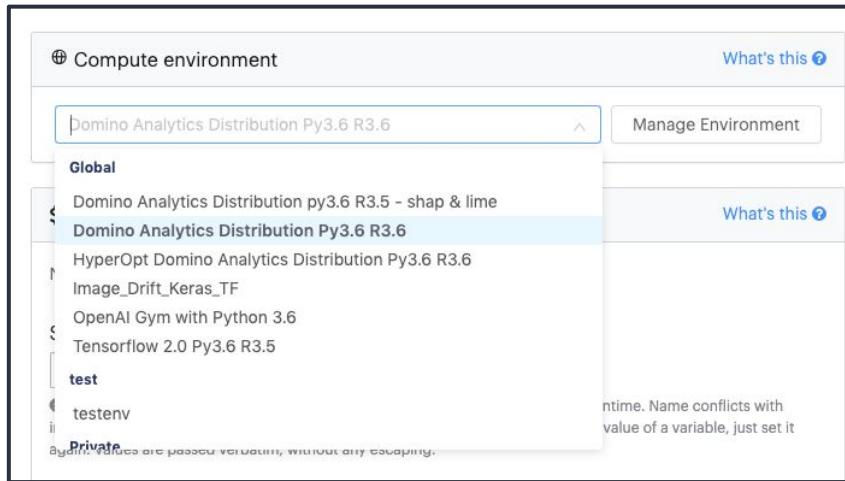
Showing 1 - 1 out of 1

DOMINO

# Environments

# Environments in Domino

- An environment is a Domino abstraction on top of a [Docker](#) image, that provides additional flexibility and versioning
- When Domino starts your run, it creates an isolated Docker container based on the environment associated with your project



# Environments in Domino

Domino comes with a set of Standard Environments, and it's easy to customize or create your own:

- Input Dockerfile instructions to add to an existing environment or
- Use a custom image, such as NVIDIA NGC containers, and have Domino make it automatically compatible with Workspaces and Jobs
- Learn more in [Domino 201](#)

New Environment

Name: NGC Tensorflow Image

Description: (Optional)

Base Environment / Image

Start from an existing Environment  
Inherit a base image, Dockerfile instructions & scripts from a Domino environment

Start from a custom base image  
Use a custom Docker or container image URI from a registry like NGC

FROM: nvcr.io/nvidia/tensorflow:21.11-tf2-py3

Automatically make compatible with Domino  
You can review and update this configuration later

Supported Clusters:  none,  Spark,  Ray,  Dask

Visibility:  Private integration-test

Globally Accessible

Cancel,  Customize before building,  Create Environment

# Activity: Change your Compute Environment

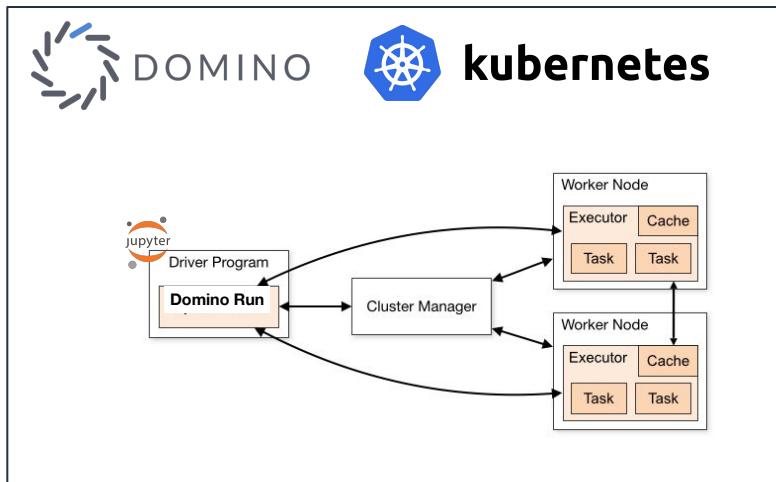
1. Navigate to the Settings for your training project
2. Update the Compute environment to 'Intro to Domino Power Forecasting'
3. (Optional) Navigate back to Workspaces and note your existing workspace still uses the default environment. You have two options to update:
  - a. Edit the Settings for that individual workspace
  - b. Create a new workspace, and note it picks up the new environment

# On-Demand Distributed Computing

# Spark in Domino

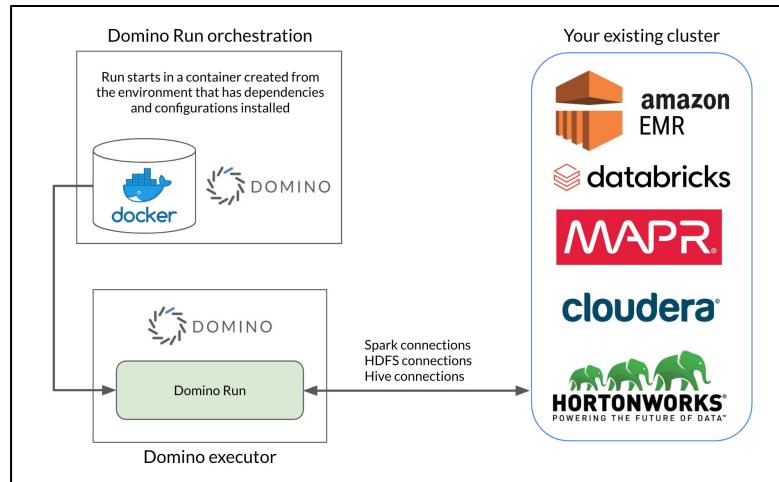
## On-Demand Clusters

Natively orchestrated directly on Domino controlled infrastructure



## External Clusters

Leverage existing long-lived Spark deployments in the enterprise



# Spark on Demand in Domino: Use Cases

## Distributed Machine Learning

Parallelize compute heavy workloads such as distributed training or hyper-parameter tuning

Take advantage of powerful machine learning algorithms from Spark MLlib

## Interactive Exploratory Analysis

Efficiently load large data sets in distributed manner

Explore and understand the data using a familiar interface with Spark SQL

For experienced Spark practitioners

## Featurization and Transformation

Sample, aggregate, and re-label large data sets

Optimal performance may require a practitioner who is skilled in tuning Spark

# Ray on Demand in Domino: Use Cases

## Distributed Multi-Node Training

Take existing PyTorch and Tensorflow models and scale them across multiple machines to dramatically reduce training times

Ray is suitable for both distributed CPU and GPU training

## Hyperparameter Optimization

Launch a distributed hyperparameter sweep with just a few lines of code.

Take advantage of a large number of advanced parameter search algorithms.

## Reinforcement Learning

Take advantage of a number of built-in reinforcement learning algorithms, along with a general framework for developing your own.

# Dask on Demand in Domino: Use Cases

## Working with Large Datasets

Great for scaling up Python data analysis or transformation where processing requires more resources than can be provided by a single machine.

Performed with minimal modification and without having to switch over to a different ecosystem like Spark.

## Distributed Training

Dask provides a simple way to parallelize existing scikit-learn models.

For larger memory-bound workloads, there are Dask specific ML libraries (e.g. Parallel Meta-estimators, Incremental Hyperparameter Optimizers)

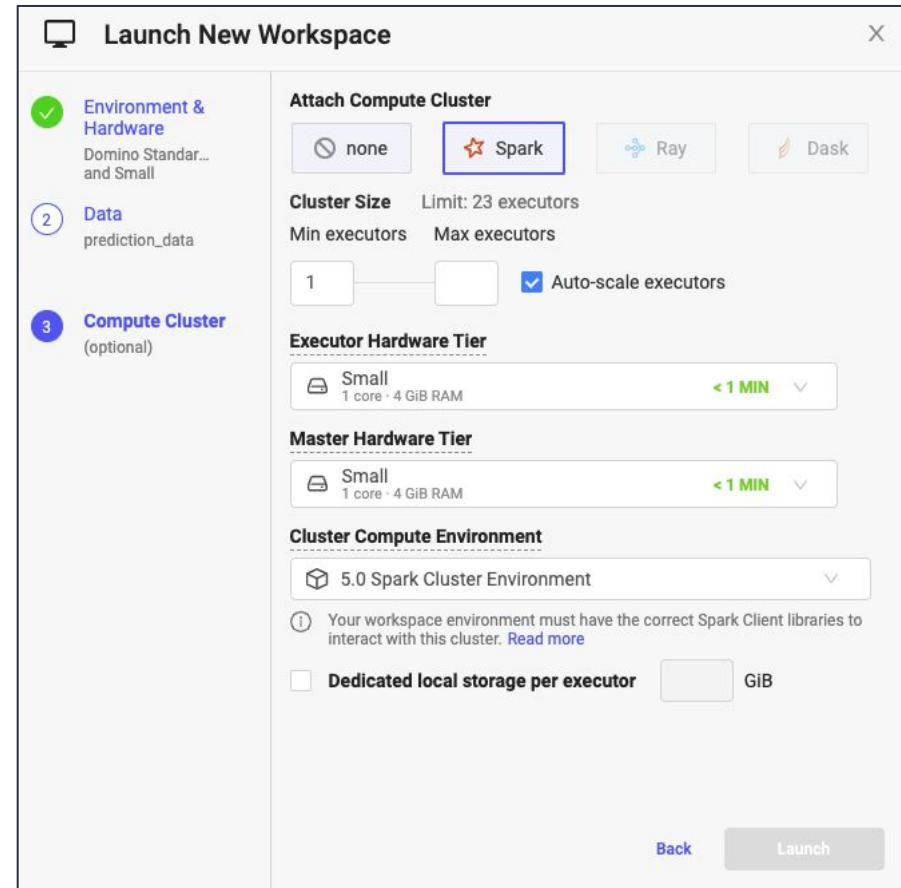
## Custom Distributed Computations

The low-level Dask scheduling APIs can be used to build custom algorithms that can benefit from parallelism.

You control the business logic while Dask handles task dependencies, network communication, workload resilience, diagnostics, etc.

# On-Demand Distributed Computing Overview

- Available only to the current execution
- Use with data residing outside the cluster - i.e. in Domino datasets or an S3 buckets
- Web UIs for the associated cluster are available in the same workspace
- Learn more in [Domino 201](#)
- Clusters can be auto-scaled



# Model APIs

# APIs: The Universal Translator

- A developer can create an application in the language of their choice AND still use your R or Python model
- Domino model APIs serve your code as a low-latency web service
- You don't need any API design experience to turn your Python or R code into a Model API
- Details in our [Publishing course](#)



# Build Model APIs in Domino

## Script:

```
load model

function (input):
    result = call model (input)
    return result
```



Publish New Version  
Step 2 of 2

The file containing the code to invoke (must be a Python or R file)

The function to invoke

Our API will handle marshalling input arguments to the function, and serializing the return values back as JSON.

[Choose files to exclude](#)

Calling your Model

Route:

Latest

Model URL: <https://try.dominodatalab.com:443/models/602eeb4f6c5f193a9019f4d2/latest/model>

Tester curl Java JavaScript PHP PowerShell Python R Ruby Other



# Host Domino Model APIs Externally

Model APIs can be exported to run in NVIDIA Fleet Command, Sagemaker, or any external container registry

1. Build Model API in Domino,
2. Use Domino's REST APIs to export
3. Stop model instances to remove from Domino cluster

# Activity: Create and Publish a Model API

1. Navigate to the ‘Publish/Model APIs’ page in your project
2. Select ‘New Model’
3. Name the model ‘<your name>-Power-Generation-Prediction’ (description optional) and select ‘Next’
4. Under file enter ‘model\_forecast.py’, and under the function to invoke enter ‘predict’
  - a. Select ‘Create Model’
  - b. Peruse the model code while waiting for the model to deploy
5. Test your model after it is running under the Model ‘Overview’ tab with the following sample

```
{  
    "data": {  
        "year": 2022,  
        "month": 6,  
        "day": 7  
    }  
}
```

6. (optional) Copy your model URL and token into the Model\_caller.ipynb notebook to call your model

# Web Applications

# Web Applications

- Each Domino project can host one web app
- Detailed instructions are available for [Dash](#), [Flask](#), [Shiny](#), [Streamlit](#), and [Django](#)
- See our [Publishing course](#) for more



# Web Apps in Domino

App Files



Start an app server



/mnt/...



app.sh



# Web Apps Access

Permission levels are separate from projects and include:

## **Anyone, including anonymous users**

Users do not need a Domino account, but they do need network access

## **Anyone with an account**

All users who have Domino accounts can view your App

## **Invited users (other users may request access)**

All users who have Domino accounts and are logged in to Domino can request access to your App if it appears in the Launchpad

## **Invited users only**

Only Domino users who are added by the App owner via the Invite People field can view the App.

Users cannot request access

# Take-home Activity: Create a Dash App

1. Peruse the files called 'app.sh' and 'app.py' in your training project
2. Navigate to Publish/App
3. Name your app '<your name> Power Generation in the UK' and add an optional description
4. (optional) Decide whether to show your App in the launchpad, and what permissions it should have
5. Click 'Publish' on the bottom right
6. Once app is running select 'View App' to use it
7. **Stop** the app when you are finished

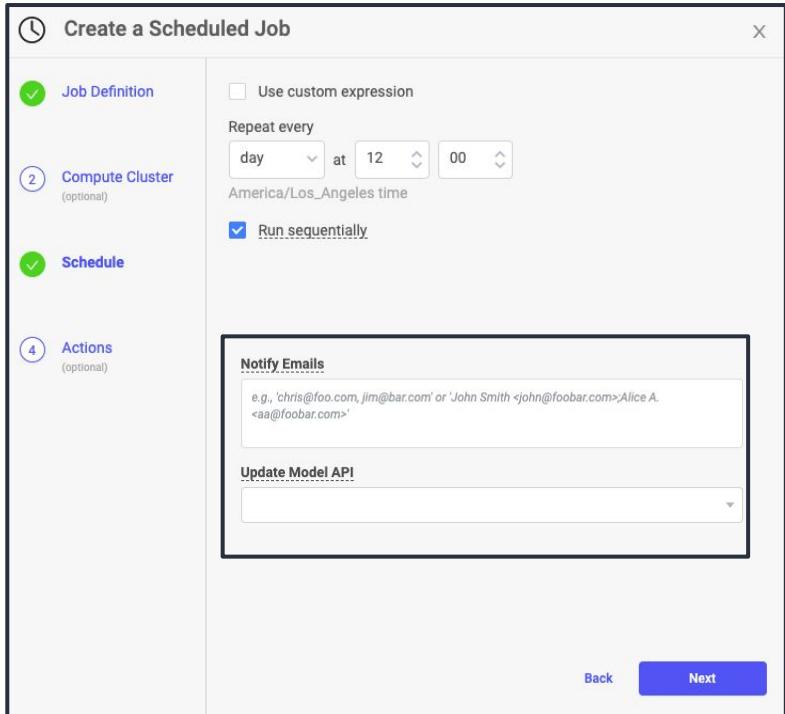
# Scheduled Jobs

# Scheduled jobs

- Schedule any Job in Domino to occur on a regular basis.
- Designate the recipients of the automatically generated email with the Job results attached

## Useful for

- Refreshing data, i.e. data pump
- Scheduling regular reports
- Retraining models on regular basis



# Take-home Activity: Create a Scheduled Report

1. Peruse the file called 'scheduled\_forecast.py' in your training project
2. Navigate to the 'Publish/Scheduled Jobs'
3. Under file name enter 'scheduled\_forecast.py'
4. Select a time in the next couple of minutes
5. Click 'Schedule'
6. Verify Job runs at scheduled time and look at the results
7. Click the : next to the scheduled job and the select **Pause**

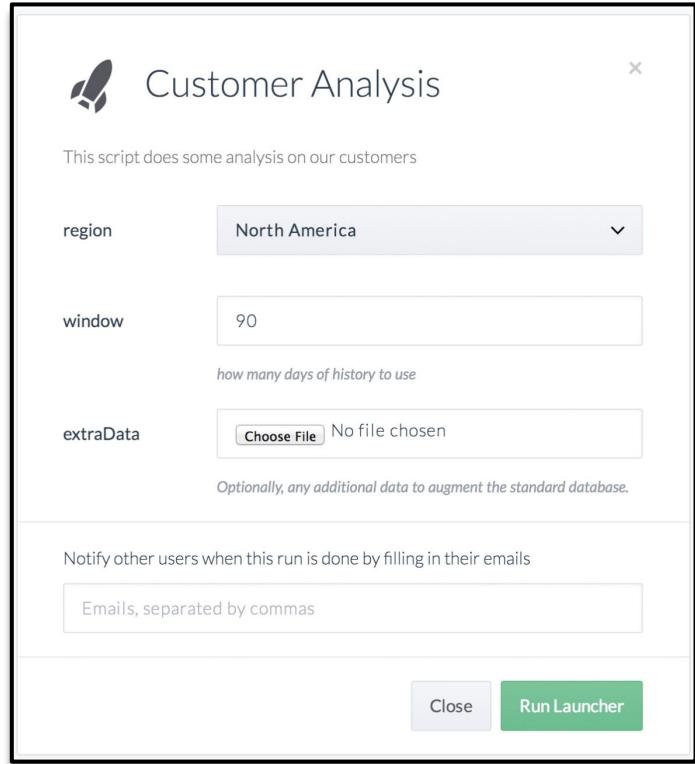
# Launchers

# Launchers Overview

Launchers are simple self service web forms on top of your script

Consumers of your project can upload data and specify parameters that are neatly consumed by your code

A separate permission level is available for Launcher Users, which limits them to only use Launchers and access Results.



The screenshot shows a modal window titled "Customer Analysis". The window contains the following fields:

- region:** A dropdown menu set to "North America".
- window:** An input field containing the value "90".  
how many days of history to use
- extraData:** A file input field with the placeholder "Choose File" and the message "No file chosen".  
Optionally, any additional data to augment the standard database.
- Emails:** A text input field with the placeholder "Emails, separated by commas".

At the bottom right are two buttons: "Close" and "Run Launcher".

# Take-home Activity: Create a Launcher

1. Peruse the file called 'launcher\_forecast.py' in your training project
  - a. Note the comments with the list of allowed values for fuel\_type
2. Navigate to the 'Publish/Launchers' tab and click New Launcher
3. Under command to run enter 'launcher\_forecast.py'
4. Click Add Parameter once to add the start\_date parameter
  - a. Change the name to date, type to "Date" and enter 01/01/2022 as default
5. Click Add Parameter again to add the fuel\_type parameter
  - a. Change the name to fuel\_type, type to "Select" and enter a comma separated list of accepted values - CCGT, OIL, COAL, NUCLEAR, WIND, PS, OCGT, BIOMASS
6. Click Save Launcher, then go back to the Launchers list
7. Run the Launcher with a different start\_date and fuel\_type and see the results

# Using Domino Programmatically

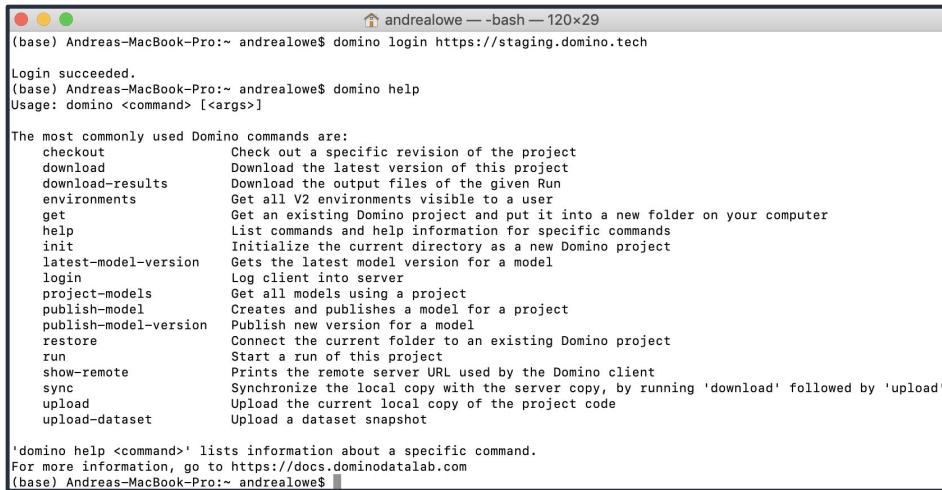
# Domino Command Line Interface (CLI)

- Work locally but run your code in Domino
- Sync a local folder with a Domino project
- Easily upload many/large files to Domino

To start, [download the CLI](#) by clicking on your user name in the bottom left

After the download completes, login using the terminal (you may be given a link for authentication)

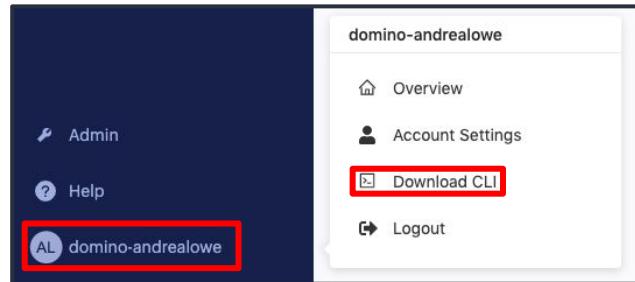
See list of possible commands [here](#) and in [Domino 201](#)



```
(base) Andreas-MacBook-Pro:~ andrealowe$ domino login https://staging.domino.tech
Login succeeded.
(base) Andreas-MacBook-Pro:~ andrealowe$ domino help
Usage: domino <command> [<args>]

The most commonly used Domino commands are:
checkout          Check out a specific revision of the project
download          Download the latest version of this project
download-results   Download the output files of the given Run
environments       Get all V2 environments visible to a user
get               Get an existing Domino project and put it into a new folder on your computer
help              List commands and help information for specific commands
init              Initialize the current directory as a new Domino project
latest-model-version Gets the latest model version for a model
login             Log client into server
project-models    Get all models using a project
publish-model     Creates and publishes a model for a project
publish-model-version Publish new version for a model
restore           Connect the current folder to an existing Domino project
run               Start a run of this project
show-remote        Prints the remote server URL used by the Domino client
sync              Synchronize the local copy with the server copy, by running 'download' followed by 'upload'
upload            Upload the current local copy of the project code
upload-dataset    Upload a dataset snapshot

'domino help <command>' lists information about a specific command.
For more information, go to https://docs.dominodatalab.com
(base) Andreas-MacBook-Pro:~ andrealowe$
```



# Domino API

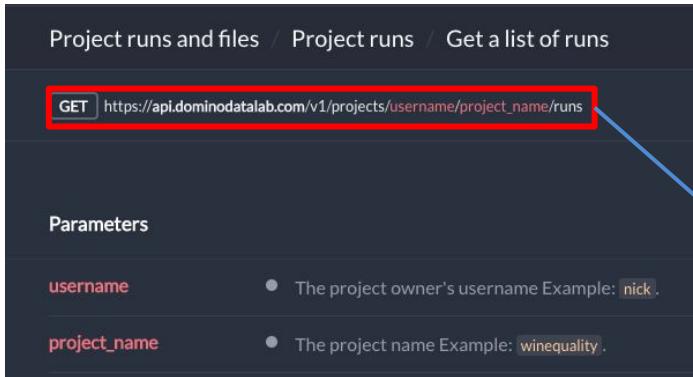
Much of Domino's functionality is available through a REST API

Available APIs can be found [here](#) (videos [here](#)) and can be used for workflows such as:

- Schedule extraction of new data and refresh models or datasets
- Parallelize your work with spread and collect flows
  - Start and inspect jobs programmatically
  - Automate meta-analysis across multiple experiments
- Get info about your work in Domino
  - Organizations, projects, models, and datasets
  - Admins can see compute time spend and other usage metrics

# Domino API - How to Use

## Domino's API:



The screenshot shows a screenshot of the Domino API documentation. At the top, it says "Project runs and files / Project runs / Get a list of runs". Below that is a red box around a "GET" button and the URL "https://api.dominodatalab.com/v1/projects/{username}/{project\_name}/runs". A blue arrow points from this URL to the Python script below.

**Parameters**

<b>username</b>	● The project owner's username Example: <code>nick</code> .
<b>project_name</b>	● The project name Example: <code>winequality</code> .

## Python script:

```
import requests
import json
import os

api_key = os.environ['DOMINO_USER_API_KEY']
username = os.environ['DOMINO_PROJECT_OWNER']
project_name = os.environ['DOMINO_PROJECT_NAME']

url = 'https://mycompany.domino.tech/v1/projects/{username}/{project_name}/runs'.format(username=username,
                                         project_name=project_name)

headers = {'X-Domino-Api-Key': api_key}

r = requests.get(url, headers=headers)
```

- Replace 'api.dominodatalab.com' with your company's Domino URL
- Find your API key in account settings or reference it using the [built-in environment variable](#) if within a Domino workspace/job

# Activity: Use the Domino API to Tag a Project

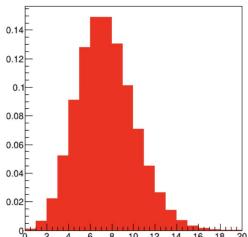
1. Peruse the file called 'tag-project-api.py'
2. Use the Domino API to create a script to tag the project with 'time-series'
  - a. Refer to the API docs found [here](#) for the specification
  - b. Keep the tag 'time-series' or change it to a different tag
3. Select 'Run' from the top right corner
  - a. Note you can also run the same code from a workspace, either by pasting into a notebook or by running the script from a Jupyter(Lab) terminal
4. After the job has completed, navigate to the project overview page and look at the tags

# Integrated Model Monitoring

# Model Monitoring Approach

## Training Data

Used as reference distributions

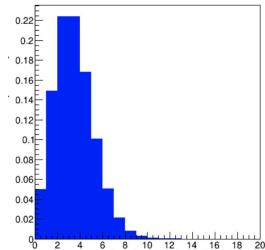


*Statistical Tests*

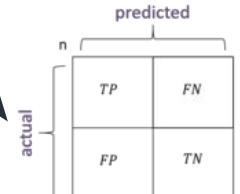
Divergence Metrics

## Prediction Data

Production data with features & predictions



Pred. Class
1
0
1
1



*Confusion Matrix*

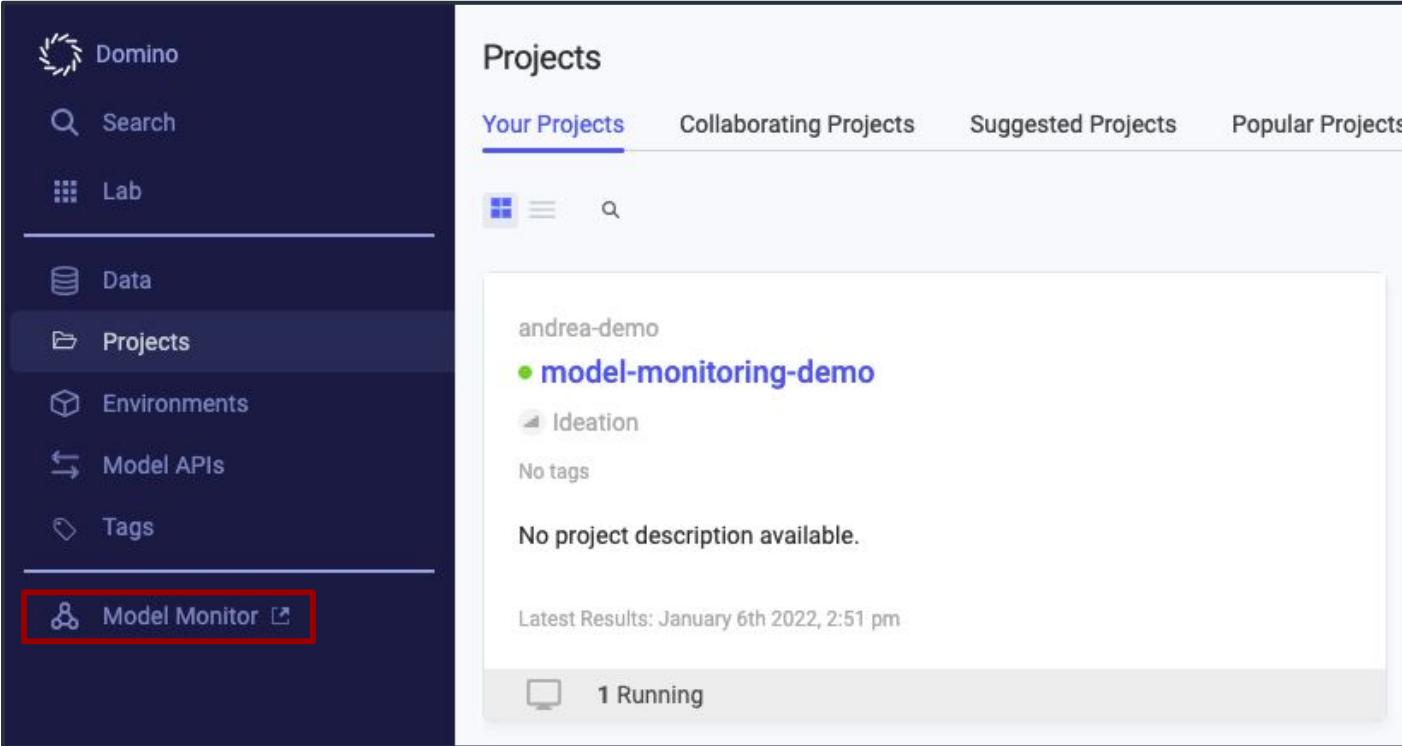
Model Quality Metrics

## Ground Truth Data

Ground Truth labels created by experts or actual outcomes

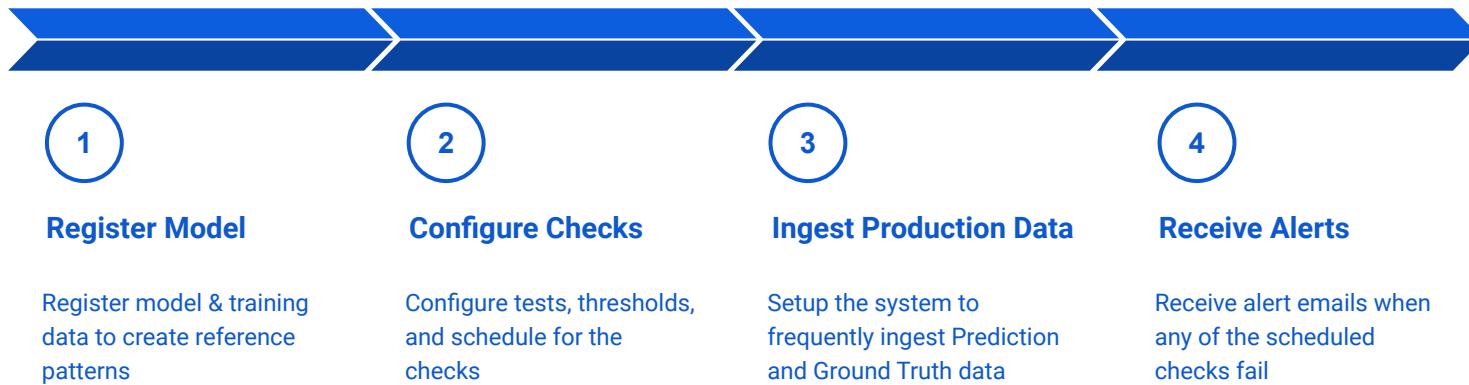
Act. Class
1
1
0
1

# Access in Domino Navigation Bar



The screenshot shows the Domino navigation bar interface. On the left, there's a sidebar with various icons and labels: Domino (with a sunburst icon), Search (magnifying glass icon), Lab (grid icon), Data (database icon), Projects (folder icon), Environments (cube icon), Model APIs (left arrow icon), Tags (tag icon), and Model Monitor (gauge icon). The 'Projects' item is currently selected, as indicated by a blue underline. A red box highlights the 'Model Monitor' button at the bottom of the sidebar. The main content area is titled 'Projects' and includes tabs for 'Your Projects' (which is active), 'Collaborating Projects', 'Suggested Projects', and 'Popular Projects'. Below the tabs are filter icons for 'Grid', 'List', and 'Search'. The main content pane displays a project named 'andrea-demo' with a green dot next to it, labeled 'model-monitoring-demo'. It also shows 'Ideation' and 'No tags' status. A message 'No project description available.' is present. At the bottom of this pane, it says 'Latest Results: January 6th 2022, 2:51 pm'. The footer of the main content area shows a monitor icon and the text '1 Running'.

# Model Monitor Workflow



# Integrated Model Monitoring

- Domino Model APIs can automatically send prediction data for monitoring
- Training sets can be created using the [Domino Data API](#)
- These are a versioned set of data, column info, and other metadata

```
from domino.training_sets import TrainingSetClient
tsv = TrainingSetClient.create_training_set_version(
    training_set_name="",
    df=training_df,
    key_columns=[],
    target_columns=[""],
    exclude_columns=[],
    meta={"experiment_id": "123456"},
    monitoring_meta=model.MonitoringMeta(**{
        "categorical_columns": [],
        "timestamp_columns": [],
        "ordinal_columns": []
    }),
    project_name=""
)
```

# Set up Model APIs for Integrated Monitoring

Use Domino's client libraries to ingest prediction data for monitoring:

```
from domino_data_capture.data_capture_client import DataCaptureClient  
data_capture_client = DataCaptureClient(feature_names, predict_names, metadata_names)
```

Then add to model function:

```
data_capture_client.capturePrediction(feature_values, predict_values,  
metadata_values, event_id, timestamp, prediction_probability, sample_weight)
```

See R version [here](#)

# Data Storage for Integrated Model Monitoring

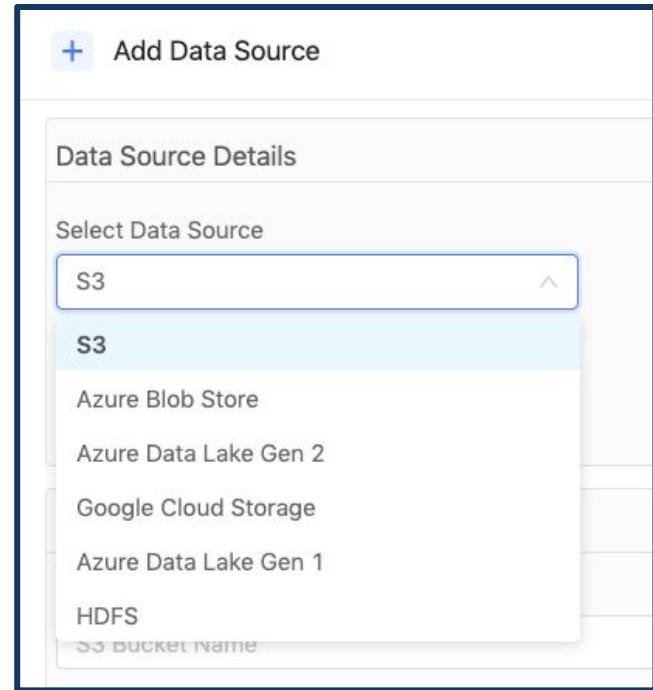
- Each instance of your model API will store one Parquet file per hour
  - Do not alter file names, these are stored as:  
`/domino/datasets/local/prediction_data/<model_version_id>/$$date$$=<date_in_utc>/  
$$hour$$=<hour_in_utc>/predictions_<uuid>.parquet`
- This prediction data is automatically consumed by the model monitor after setup
- By default, a daily cleanup job deletes data older than 30 days, or the oldest data beyond 300GB

# Upload Data for Any Model

DMM can be used with the following data sources:

- Amazon S3
- Azure Blob
- Azure Data Lake Gen 1
- Azure Data Lake Gen 2
- Google Cloud Storage
- HDFS

See [docs](#) for specifics on using each data type



# Data Drift Monitoring

Set up preferred tests, thresholds, and disable alerts

Disable alerts for specific features here

**Data Drift** ⓘ

Analyze Checks History Schedule Checks Register Prediction

Date Filter Select Other Actions Recalculate Search 🔍

Calculated On: 17 Feb, 2021 | 6:39 AM (UTC)  ⓘ Data used for Check

Feature <span>⬇️</span>	Training data	Prediction data	Test Type	Test Condition	Threshold	Drift <span>⬇️</span>	Drift Trend
<span>🔔</span> BIOMASS Numerical			Kullback-Leibler Diver...	Less than <span>⬇️</span>	0.3	0.8303	No date filter applied
<span>🔔</span> CCGT Numerical			Chi-Square Kolmogorov-Smirnov Wasserstein Distance Energy Distance Population Stability Index	Less than <span>⬇️</span>	0.3	0.5946	No date filter applied
<span>🔔</span> COAL Numerical				Less than <span>⬇️</span>	0.3	1.153	No date filter applied
<span>🔔</span> INTEM Numerical			Kullback-Leibler Diver...	Less than <span>⬇️</span>	0.3	0.7228	No date filter applied
<span>🔔</span> INTEW Numerical			Kullback-Leibler Diver...	Less than <span>⬇️</span>	0.3	0.0664	No date filter applied

# Schedule Data Drift and Model Quality Checks

Automate notifications if data drift or model quality metrics degrade beyond a set threshold

*Select Data to Check* ensures that only predictions with timestamps in selected range are used

Data Drift  ⓘ

Analyze Checks History Schedule Checks

Enabled

Check Name

Repeat Every  at  :00 Hrs

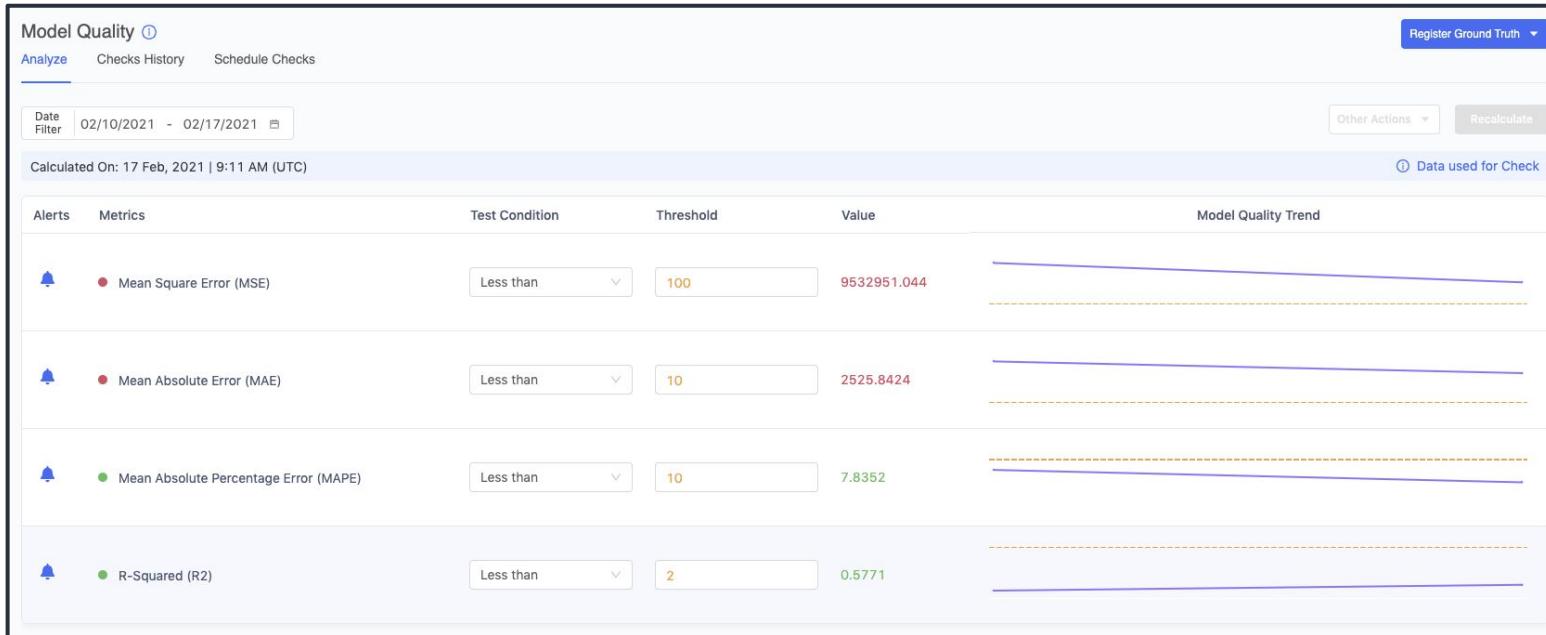
Select Data to Check

Use new data since last check time

Data since last

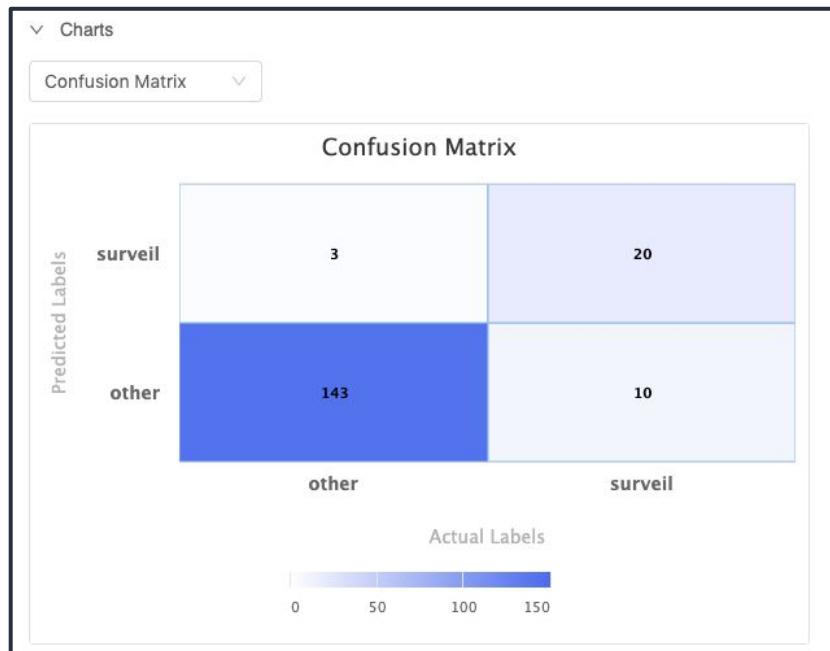
# Model Quality Monitoring

You can set up preferred tests, thresholds, disable alerts, and filter on date



# Model Quality Monitoring

Classification models will also show the confusion matrix and report



Charts

Classification Report

Class	Precision	Recall	F1	Support
other	0.9346	0.9795	0.9565	146
surveil	0.8696	0.6667	0.7547	30
Accuracy	-	-	0.9261	176
Weighted Avg	0.9235	0.9261	0.9221	176
Macro Avg	0.9021	0.8231	0.8556	176

# Register Model

Register model with a config file

The config file is a JSON object with info about the model

This can be customized to override binning method or provide pre-computed distributions

The screenshot shows a 'Register Model' dialog box. At the top, there is a dashed box labeled 'Drag & drop your JSON files or Click to browse'. Below this, a green checkmark icon indicates 'JSON is valid'. The main area contains a JSON code block:

```
1  [
2      "variables": [
3          {
4              "valueType": "numerical",
5              "variableType": "feature",
6              "name": "petal.length"
7          },
8          {
9              "valueType": "numerical",
10             "variableType": "feature",
11             "name": "sepal.length"
12         },
13         {
14             "valueType": "numerical",
15             "variableType": "feature",
16             "name": "petal.width"
17         }
18     ]
19 
```

At the bottom right of the dialog are two buttons: 'Cancel' and 'Register Model'.

# Register Predictions and Ground Truth

Choose data from any data source and configure any new columns

Map ground truth column to prediction column

+ Register New Prediction X

1 Add Prediction Dataset  
2 Configure New Columns  
3 Preview Config File

Choose Data Source  
DMM-power-generation

Upload new .csv file Search by File Name

Select	File Name	Upload Date
<input type="radio"/>	ground_truth_2021-02-16.csv	Feb 16, 2021
<input type="radio"/>	predictions_2021-02-16.csv	Feb 16, 2021
<input type="radio"/>	ground_truth_2021-02-15.csv	Feb 15, 2021
<input type="radio"/>	predictions_2021-02-15.csv	Feb 15, 2021
<input type="radio"/>	ground_truth_partition2.csv	Feb 15, 2021
<input type="radio"/>	ground_truth_partition1.csv	Feb 15, 2021
<input type="radio"/>	predictions_partition1.csv	Feb 15, 2021
<input type="radio"/>	predictions_partition2.csv	Feb 15, 2021
<input type="radio"/>	ground_truth_2021-02-14.csv	Feb 14, 2021

Back Next

# Take-home Activity: Explore Data Drift

1. Navigate to the Model Monitor page
2. Open any model
3. Open the data drift page
4. Change the test type and recalculate

# Activate Monitoring

house-price-predictions  Running

[Archive](#) [Open in Workspace](#) [New Version](#)

[Overview](#) [Versions](#) [Monitoring](#) [Audit Log](#) [Settings](#)

**Configure monitoring ▾**

- Data
- Target Ranges
- Alert Recipients
- Schedule



**Monitoring Is Not Enabled**

Monitoring alerts you when your model shows signs of degraded performance.  
To learn how to set up monitoring, click the button below

[Learn More](#)

# Activate Monitoring

**Configure Data** X

**Prediction Data**  
[house-price-predictions/version\\_1/](#)  
Model data is required for all monitoring. It is generated based on this model's prediction capture function. [Learn more ↗](#)

**Training Data** **Version**

Allows drift monitoring

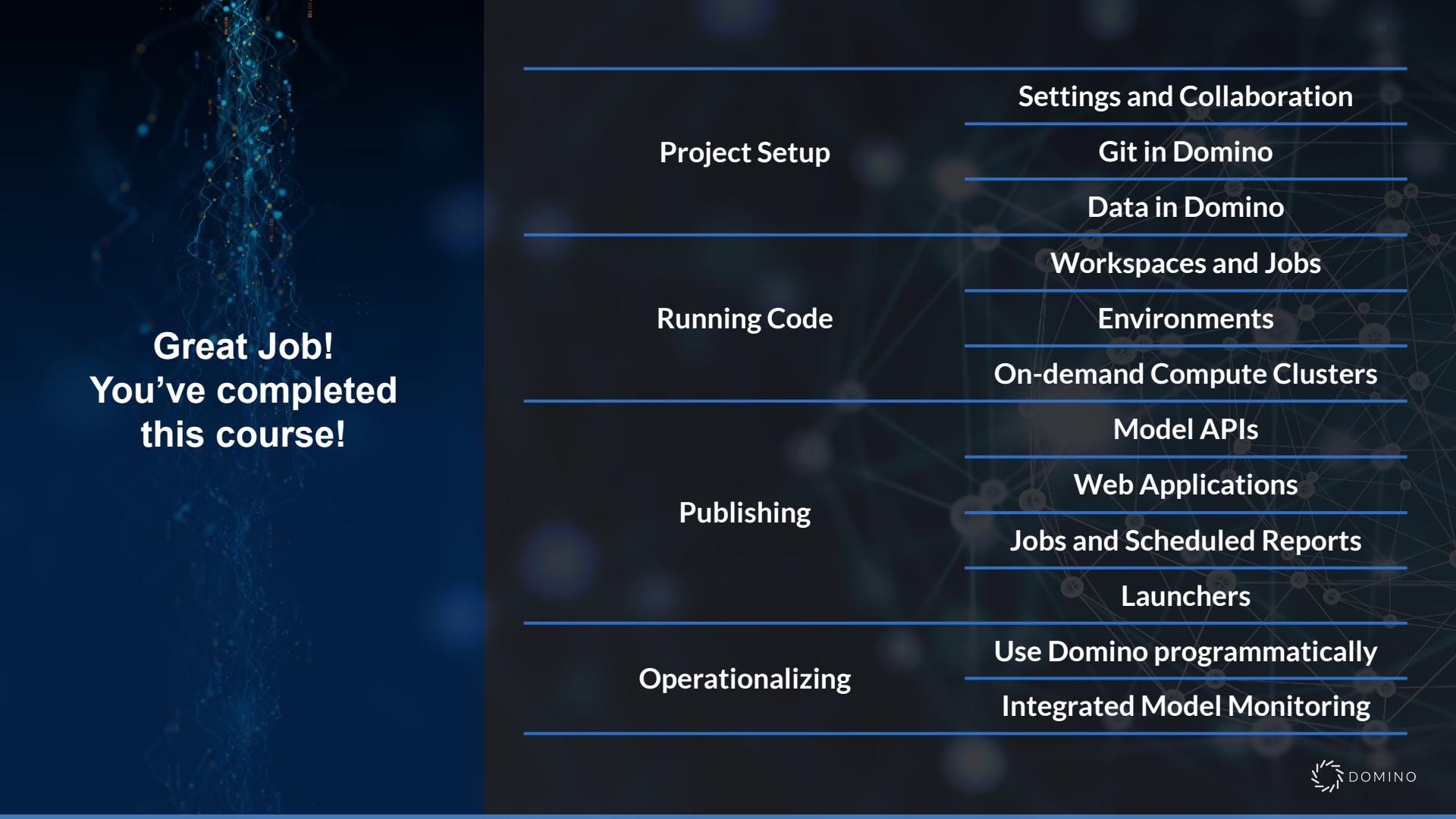
**Model type**

[Cancel](#) [Save](#)

# Activity: Configure Monitoring

For the Model API, enable monitoring by selecting the training set to track drift against

1. Go to your newly published Model API overview page and navigate to the ‘Monitoring’ tab
2. Open ‘Configure Monitoring’ > ‘Data’ to select the right training set and version and set the model type to ‘Regression’
3. Call the model multiple times by pasting your model URL and token into the `Model_caller.ipynb` notebook and running
4. Refresh and view drift, set thresholds, etc

A dark blue background featuring a faint, abstract network graph composed of numerous small, glowing nodes connected by thin lines.

**Great Job!**  
**You've completed**  
**this course!**

---

## Project Setup

---

## Running Code

---

## Publishing

---

## Operationalizing

---

---

**Settings and Collaboration**

---

**Git in Domino**

---

**Data in Domino**

---

**Workspaces and Jobs**

---

**Environments**

---

**On-demand Compute Clusters**

---

**Model APIs**

---

**Web Applications**

---

**Jobs and Scheduled Reports**

---

**Launchers**

---

**Use Domino programmatically**

---

**Integrated Model Monitoring**

---



# Need Help!?

The Docs:

[docs.dominodatalab.com](https://docs.dominodatalab.com)

Community:

[community.dominodatalab.com](https://community.dominodatalab.com)

Submit a Ticket:

[tickets.dominodatalab.com](https://tickets.dominodatalab.com)

Self-service Training:

[learn.dominodatalab.com](https://learn.dominodatalab.com)



 Support