TURING

Penguin S3 Dashboard - How to Run the Local Environment

Summary

Summary

This guide provides a comprehensive step-by-step walkthrough for setting up and running the local environment for the Penguin S3 Dashboard. It includes instructions for repository access, environment configuration, running the backend and frontend services, and using the Celery task queue. By following this guide, you can seamlessly manage the development, testing, and deployment processes for the Penguin S3 Dashboard.

Why and How

- **Why?**
  This document is intended to ensure a smooth onboarding process for developers. It details how to set up the necessary environment and tools to enable efficient development and testing of the Penguin S3 Dashboard.
- **How?**
  - Access repositories and manage tokens.
  - Configure virtual machines if needed.
  - Set up NPM dependencies and ensure proper environment variable management.
  - Run multiple terminals to start the frontend, backend, and asynchronous task queue services.
- **Result:**
  A fully functioning local setup of the Penguin S3 Dashboard for testing and development purposes, ensuring all dependencies and configurations are met.

Introduction

The Penguin S3 Dashboard serves as a comprehensive tool for managing and visualizing data workflows. This document provides clear instructions for setting up a local development environment, with troubleshooting tips to handle common issues.

Support

Reach out to the following people if you have any questions or need help:

- David Truong Hong Phuoc and Aditya Patro to support GitHub, code and package repo access.
- Bhavna Kumrawat to support node.js, Web and Samuel Ezeala API service setup for Labeling Tool.

Repository Access

1. **Token Management:**
   - Generate a **personal access token (classic)** with the `write:packages` permission.
   - Use your Turing GitHub account for repository access.
2. **Repositories:**
   - **Read and write:**
     - Apple Validation Scripts

Running the environment

Terminal 1 : Start the frontend

Setting up a frontend project requires installing dependencies, ensuring your system has the correct software, and configuring your environment. Here's a step-by-step guide to setting up your project from scratch:

Prerequisites

Before starting, ensure you have the following:

1. **Operating System:** Windows, macOS, or Linux.
2. **Code Editor:** Install VS Code.
3. **Node.js and npm:**
   - Download Node.js version 20.15 from Node.js Official Website.
   - Install Node.js; npm (Node Package Manager) is bundled with it.

4. **Git:** Install Git from Git Official Website.

After installation, open Git Bash to verify:

```
Unset
  git --version
```

Step 1: Clone the Repository

Open Git Bash and navigate to your desired directory:
After installation, open Git Bash to verify:

```
Unset
git clone
https://<username?:<token>@github.com/TuringGpt/apple-validation-
scripts.git
```

Step 2: Navigate to the Frontend Directory & change the branch

```
Unset
 cd frontend
```

Switch to the Development Branch

Checkout the `dev` branch where the latest development work is available:

```
Unset
    git checkout dev
```

Create Your Working Branch

Create a new branch for your work to keep changes isolated:

```
Unset
      git checkout -b your-working-branch-name
```

Replace `your-working-branch-name` with a descriptive name for your task or feature.

Start Working

You can now make changes and commit them to your working branch.

Step 3: Install Dependencies

Run the following command in Git Bash:

```
Unset
npm install
```

This will install all the required dependencies listed in the `package.json` file.

Step 4: Install `nx` Globally (Optional)

If the project uses `nx` for monorepo management, you may need to install it globally:

```
Unset
npm install -g nx
```

If you prefer not to install it globally, you can use `npx nx` instead of `nx` in all commands.

Step 5: Set up the environment files

- **Create a `.env` File**

In the root of your project (or `frontend` directory), create a file named `.env`:

```
Unset
    touch .env
```

- **Updating files : Environment Variables**
- Download the necessary files Enviroment files
- Copy files to their respective locations:
    - `.env.local.frontend` → `frontend/.env`

Step 6: Start the Frontend

To start the frontend, use:

```
Unset
npm start
```

If `nx` is not globally installed, use:

```
Unset
npx nx serve
```

If everything is set up correctly, the frontend will start on:
`http://localhost:4200/`

---

7. Additional Tools to Install

- **Browser:** Install Google Chrome.
- **Extensions for VS Code:**
    - ESLint
    - Prettier
- Please, don't forget to update the path according to your actual path. If everything is fine, the frontend will start on the following URL: http://localhost:4200/

```
➜  Local:   http://localhost:4200/
```

Terminal 2 - Start the backend

To set up and start the backend with the instructions you provided, here's how you can proceed, step-by-step, assuming you're using a Python FastAPI application with a local PostgreSQL database:

---

1. Ensure Prerequisites

- **Python**: Installed on your machine (preferably Python 3.7 or higher).
- **PostgreSQL**: Installed and running locally.
- **Dependencies**: Install `pip` for dependency management.

Step 2: Clone the Repository

Open Git Bash and navigate to your desired directory:
After installation, open Git Bash to verify:

```
Unset
git clone
https://<username>:<token>@github.com/TuringGpt/apple-validation-scripts.git
```

- Navigate to the Frontend Directory & change the branch

Navigate to the Frontend Directory
Open your terminal and move to the backend directory:

```
Unset
    cd backend_fastapi
```

Switch to the Development Branch
Checkout the `dev` branch where the latest development work is available:

```
Unset
      git checkout dev
```

Create Your Working Branch
Create a new branch for your work to keep changes isolated:

```
Unset
      git checkout -b your-working-branch-name
```

Replace `your-working-branch-name` with a descriptive name for your task or feature.

Start Working
You can now make changes and commit them to your working branch.

---

4. Install Dependencies & Run Migrations

1. **First Method: Using the `start.sh` Script**

   Install all the Python dependencies listed in the `requirements.txt` file. This is handled via the `start.sh` script.

```
Unset
chmod +x ./start.sh  # Make the script executable if necessary
./start.she
```

2. **Second Method: Using Docker**

Follow the instructions provided here to set up the environment [Using Docker]

---

5. Configure PostgreSQL

- Ensure PostgreSQL is running locally.
- Create a database for the project:

```
Unset
 psql -U postgres

 CREATE DATABASE your_database_name;
```

Update the **environment variables** or the configuration file (`.env`) with your PostgreSQL database credentials:

```
Unset
DATABASE_URL=postgresql://<username>:<password>@localhost:5432/yo
ur_database_name
```

Step 6: Set up the environment files

- **Create a `.env` File**

  In the root of your project (or `backend_fastapi` directory), create a file named `.env`:

```
Unset
    touch .env
```

Step 6: Set up the environment files

- **Updating files : Environment Variables**
- Download the necessary files [Enviroment files]
- Copy files to their respective locations:
  - `.env.local.backend` → `backend_fastapi/.env`

---

7. Start the Server

To start the FastAPI server locally:

```
Unset
uvicorn app.main:app --reload --port 5000
```

- `app.main` refers to the module and the FastAPI instance.
- `--reload` enables live-reloading during development.
- `--port 5000` sets the server to run on port 5000.

Access the server at http://127.0.0.1:5000.

8. Verify Everything

Visit the FastAPI docs at http://127.0.0.1:5000/docs to ensure the API is working.

Terminal 3 - Start the Celery

Celery :Celery is an asynchronous task queue/job queue based on distributed message passing. It allows you to execute tasks asynchronously in Python, typically used in background jobs, periodic tasks, or other scenarios where you need to execute long-running or scheduled tasks outside of your main application flow.

1. Ensure Prerequisites

- **Celery Installed**: Ensure Celery is listed in your `requirements.txt` or install it manually.

```
Unset
pip install celery
```

- **Message Broker:** Celery requires a message broker (e.g., RabbitMQ or Redis). Start Celery and ensure that the message broker (Redis or RabbitMQ) is running in the standalone environment. For Redis, start the Redis server to enable Celery to work correctly.
-

```
Unset
celery -A app.jobs.celery_task.celery_app worker --loglevel=info
-E --concurrency=3
```

Using Docker: Set Up the Backend Locally

A step-by-step guide to set up and run the application using Docker:

Step 1: Ensure Prerequisites Are Installed

Ensure the following prerequisites are installed on your system:

- **Docker:** Install Docker from the official Docker website.
- **Source Files:** Your project directory should contain:
  - `Dockerfile` for building the image.
  - `requirements.txt` for Python dependencies.
  - `supervisord.conf` for process management.
  - Application code (e.g., `app.main` for FastAPI, Alembic for database migrations).

Step 2: Build the Docker Image

Run the following command in the directory containing your `Dockerfile`:

```
Unset
docker compose build
```

This command will:

- Install system dependencies (e.g., `redis-server`, `gcc`, `libpq-dev`, `supervisor`).
- Install Python dependencies from `requirements.txt`.
- Set up the working directory and copy application files into the container.

Step 3: Run the Container

Start the container with:

```
Unset
docker compose up
```

Step 4: Access the Application

Open a browser or use `curl` to access your FastAPI application at:

```
Unset
http://localhost:5000
```

Step 5: Confirm Services Inside the Container

Redis Server: Redis runs in the background with `redis-server --daemonize yes`. Confirm it is running by executing:

`docker exec -it <container-name> redis-cli ping`

- Replace `<container-name>` (e.g., `fastapi-container`) with the actual container name. It should respond with `PONG`.
- **Alembic Migrations:** Ensure migrations run before starting the application. This can typically be done via the `CMD` or `ENTRYPOINT` directives in the `Dockerfile`.
- **Supervisord:** `supervisord` manages multiple processes, such as the FastAPI application and Celery workers.

Step 6: Debugging and Logs

If something doesn't work:

1. Check container logs:
   `docker logs <container-name>`

2. Access the container shell for inspection:

```
docker exec -it <container-name> /bin/bash
```

Step 7: Stopping and Cleaning Up

1. Stop the container:
```
docker stop <container-name>
```

2.Remove the container:
```
docker rm <container-name>
```

3. Remove the Docker image:
```
docker rmi <image-name>
```

Step 8: Using Docker Compose for Multi-Service Management (Optional)

If you need to manage multiple services (e.g., PostgreSQL and Redis), simplify the process using a `docker-compose.yml` file. Start services with:

```
docker-compose up --build
```

Frontend Overview -

The Frontend of the Penguin S3 Dashboard is a critical component for user interaction. It provides features such as authentication, data visualization, file management, and log monitoring. This section outlines the core functionalities and their purpose, ensuring that users can efficiently manage and interact with S3 data.

1. Login Screen

Summary

The login screen ensures secure access to the Penguin S3 Dashboard using goggle auth which is done currently from frontend. It is designed to validate user credentials and restrict access to authorized personnel only.

What?

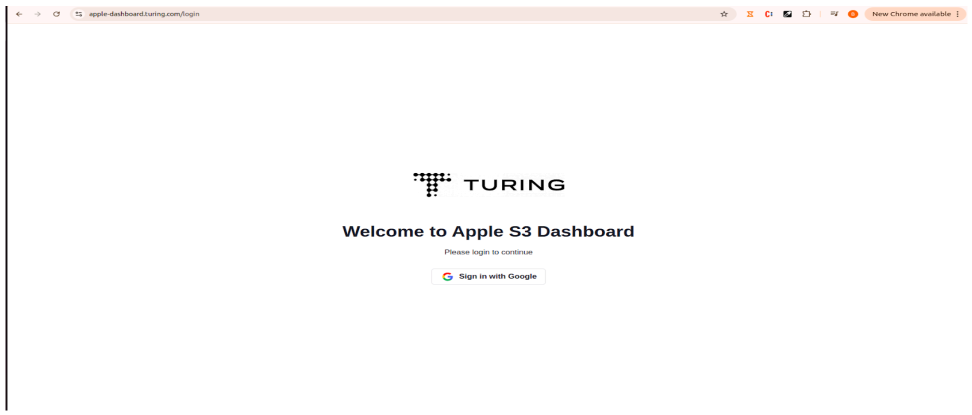
- A simple interface that requires a valid **Turing email ID** for authentication.
- Ensures secure access to the Penguin S3 Dashboard.

Why?

- To maintain the security and confidentiality of the system by restricting unauthorized access.
- To track and log user activity within the system.

Result

- **Enhanced Security:** Only authorized users can access the system.
- **Seamless Access Management:** Provides a straightforward way to authenticate and log in.



---

2. Dashboard

Summary

The dashboard is an upgraded version of the existing interface. It introduces a modern design with enhanced usability, performance, and user experience.

What?

- A redesigned UI for better usability.

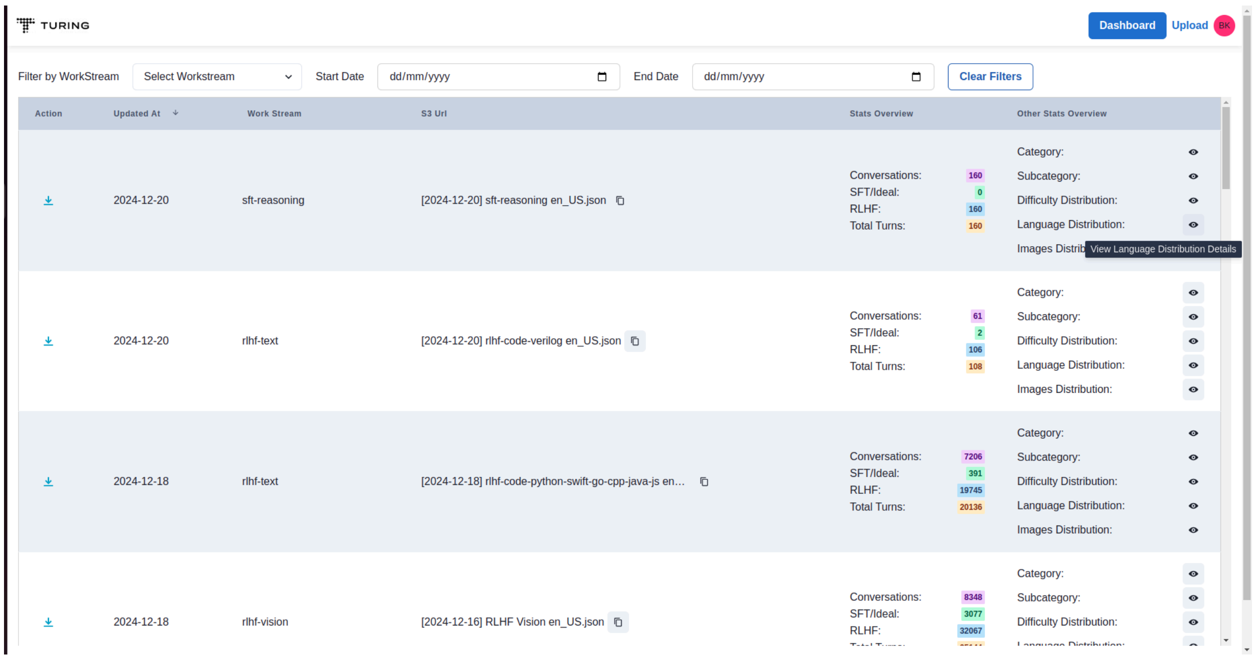
- Retains the core features of the original dashboard while improving data visualization and navigation.
- Uses updated APIs for faster and more efficient data retrieval.

Why?

- To improve the overall user experience with a modern interface.
- To address performance issues and enhance responsiveness.
- To adapt to evolving user needs with improved functionality.

Result

- **Streamlined Workflow:** Users can access and manage data more efficiently.
- **Enhanced User Experience:** A visually appealing and responsive interface improves engagement.
- **Faster Operations:** Optimized APIs and design reduce load times and improve performance.



Filter by WorkStream: Select Workstream | Start Date: dd/mm/yyyy | End Date: dd/mm/yyyy

Action	Updated At	Work Stream	S3 Url	Stats Overview	Other Stats Overview
	2024-12-20	sft-reasoning	[2024-12-20] sft-reasoning en_US.json	Conversations: 160 SFT/Ideal: 0 RLHF: 160 Total Turns: 160	Category: Subcategory: Difficulty Distribution: Language Distribution: Images Distrib
	2024-12-20	rlhf-text	[2024-12-20] rlhf-code-verilog en_US.json	Conversations: 61 SFT/Ideal: 2 RLHF: 106 Total Turns: 108	Category: Subcategory: Difficulty Distribution: Language Distribution: Images Distribution:
	2024-12-18	rlhf-text	[2024-12-18] rlhf-code-python-swift-go-cpp-java-js en…	Conversations: 7206 SFT/Ideal: 391 RLHF: 19745 Total Turns: 20136	Category: Subcategory: Difficulty Distribution: Language Distribution: Images Distribution:
	2024-12-18	rlhf-vision	[2024-12-16] RLHF Vision en_US.json	Conversations: 8348 SFT/Ideal: 3077 RLHF: 32067	Category: Subcategory: Difficulty Distribution:

API CURL

```
Curl
'https://penguin-dashboard.turing.com/api/s3files/?sort%5B0%5D=created_at%2CDESC&limit=25&page=1' \
 -H 'accept: */*' \
 -H 'accept-language: en-GB,en-US;q=0.9,en;q=0.8' \
 -H 'authorization: Bearer Bearer
ya29.a0AeDClZCuubtOqmnLfD4O254QK4vxD1ybTsfl-bZzq6FZXHq2YGKbNSKG0psAztOBkLoO_O74pev
WyVF-utLfPLXePFhtagazgvL-jsTLFI9lS4xszsDJW_f5Q61w5VhipdpJM9hblomMt0H9JNEcgUNSRJgNM
m9KNTbqaCgYKAbASARASFQHGX2Mi724bRlJGHRqrOKvAFlCmWQ0171' \
 -H 'priority: u=1, i' \
 -H 'referer: https://penguin-dashboard.turing.com/dashboard' \
 -H 'sec-ch-ua: "Not)A;Brand";v="99", "Google Chrome";v="127", "Chromium";v="127"'
\
 -H 'sec-ch-ua-mobile: ?0' \
 -H 'sec-ch-ua-platform: "Linux"' \
 -H 'sec-fetch-dest: empty' \
 -H 'sec-fetch-mode: cors' \
 -H 'sec-fetch-site: same-origin' \
 -H 'user-agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/127.0.0.0 Safari/537.36'
```

---

3. Upload Section

Summary

The upload section simplifies file management by allowing users to view uploaded files, convert Collab links to JSON, and perform new uploads through a three-tab interface.

What?

The upload section now has three tabs:

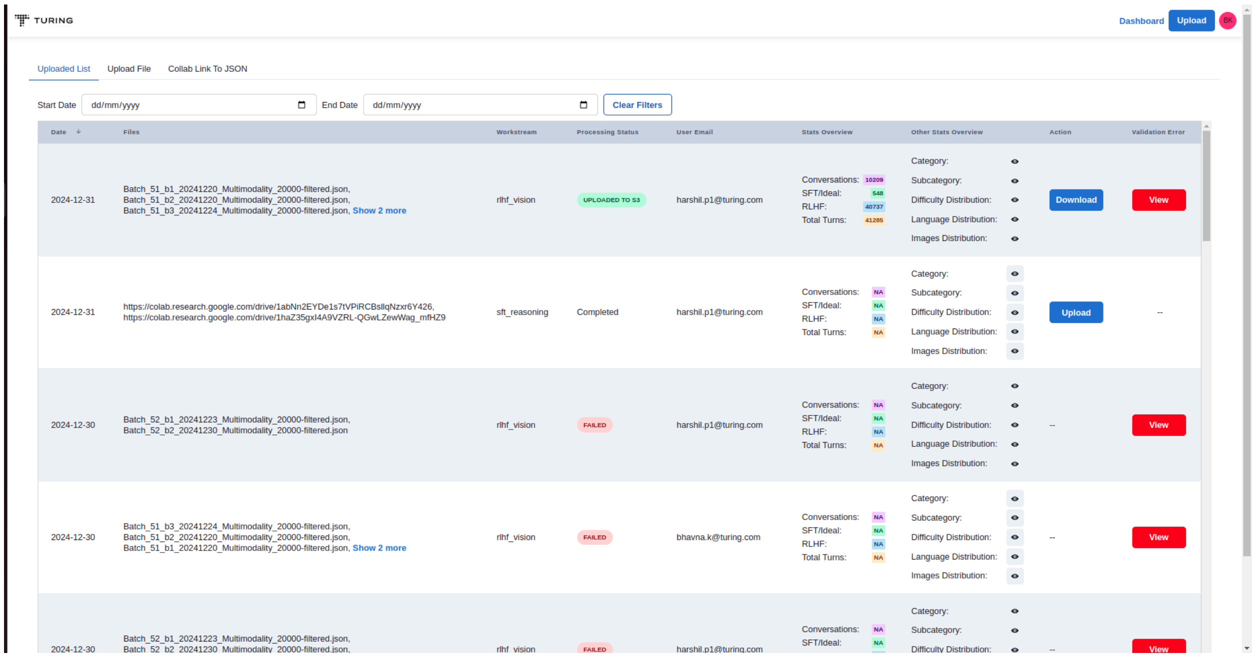
1. **Uploaded Files Tab**:
   - Displays a list of all previously uploaded files along with their statuses.
   - Provides details such as file name, upload timestamp, and current processing status.
2. **Upload Files Tab**:
   - Allows users to upload new files directly to S3.
   - Includes options to select files, validate entries, and confirm uploads.
3. **Collab Link to JSON Tab**:
   - Accepts Google Colab links (e.g., `https://colab.research.google.com/drive/...`).
   - Automatically converts the provided link to a structured JSON format.
   - Displays the converted JSON for preview and further use in integration processes.

Why?

- To provide users with an organized way to track and manage uploaded files.
- To streamline the file upload process and reduce the chances of errors during uploads.
- To introduce a convenient way for users to transform Collab links into JSON for easier integration.

Result

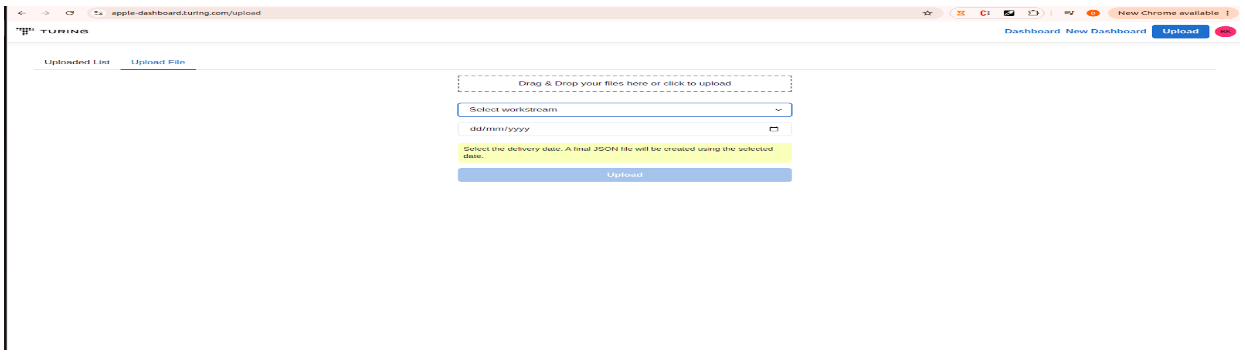
1. **Centralized File Management**:
   - Users can easily view, track, and upload files from one location.
2. **Efficient Workflow**:
   - The three-tab layout ensures intuitive navigation between file tracking, uploading, and link-to-JSON conversion.
3. **Seamless Integration**:
   - Simplifying the Collab-to-JSON conversion process supports downstream applications and integrations.
4. **Error Reduction**:
   - Validation during uploads minimizes processing issues, while automation ensures JSON output correctness.



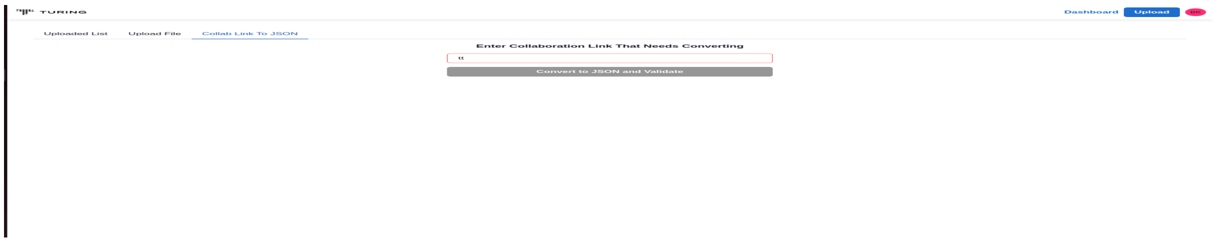
Start Date dd/mm/yyyy End Date dd/mm/yyyy

Date	Files	Workstream	Processing Status	User Email	Stats Overview	Other Stats Overview	Action	Validation Error
2024-12-31	Batch_51_b1_20241220_Multimodality_20000-filtered.json, Batch_51_b2_20241220_Multimodality_20000-filtered.json, Batch_51_b3_20241224_Multimodality_20000-filtered.json, Show 2 more	rlhf_vision	UPLOADED TO S3	harshil.p1@turing.com	Conversations: 10209 SFT/Ideal: 548 RLHF: 40737 Total Turns: 41285	Category: Subcategory: Difficulty Distribution: Language Distribution: Images Distribution:	Download	View
2024-12-31	https://colab.research.google.com/drive/1abNn2EYDe1s7tVPiRCBsllqNzxr6Y426, https://colab.research.google.com/drive/1haZ35gxI4A9VZRL-QGwLZewWag_mfHZ9	sft_reasoning	Completed	harshil.p1@turing.com	Conversations: NA SFT/Ideal: NA RLHF: NA Total Turns: NA	Category: Subcategory: Difficulty Distribution: Language Distribution: Images Distribution:	Upload	--
2024-12-30	Batch_52_b1_20241223_Multimodality_20000-filtered.json, Batch_52_b2_20241230_Multimodality_20000-filtered.json	rlhf_vision	FAILED	harshil.p1@turing.com	Conversations: NA SFT/Ideal: NA RLHF: NA Total Turns: NA	Category: Subcategory: Difficulty Distribution: Language Distribution: Images Distribution:	--	View
2024-12-30	Batch_51_b3_20241224_Multimodality_20000-filtered.json, Batch_51_b2_20241220_Multimodality_20000-filtered.json, Batch_51_b1_20241220_Multimodality_20000-filtered.json, Show 2 more	rlhf_vision	FAILED	bhavna.k@turing.com	Conversations: NA SFT/Ideal: NA RLHF: NA Total Turns: NA	Category: Subcategory: Difficulty Distribution: Language Distribution: Images Distribution:	--	View
2024-12-30	Batch_52_b1_20241223_Multimodality_20000-filtered.json, Batch_52_b2_20241230_Multimodality_20000-filtered.json,	rlhf_vision	FAILED	harshil.p1@turing.com	Conversations: NA SFT/Ideal: NA	Category: Subcategory: Difficulty Distribution:	--	View

Uploaded Files Tab


apple-dashboard.turing.com/upload
New Chrome available
TURING
Dashboard
New Dashboard
Upload
Uploaded List
Upload File
Drag & Drop your files here or click to upload
Select workstream
dd/mm/yyyy
Select the delivery date. A final JSON file will be created using the selected date.
Upload

Upload Files Tab


TURING
Dashboard
Upload
Uploaded List
Upload File
Collab Link To JSON
Enter Collaboration Link That Needs Converting
Convert to JSON and Validate

Collab Link to JSON Tab

---

5. Logs Section

Summary

The logs section provides a comprehensive record of server activities, aiding in monitoring and troubleshooting system operations.

What?

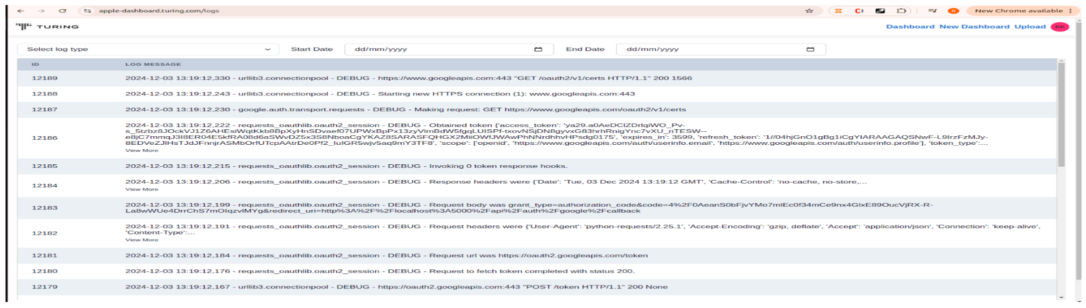
- Displays real-time logs of server-side operations.
- Includes information such as timestamps, actions performed, and error details (if any).

Why?

- To ensure transparency in system operations by keeping track of all activities.
- To help users and developers quickly identify and resolve issues.

Result

- **Improved Debugging:** Logs provide actionable insights into system errors and anomalies.
- **Operational Clarity:** Users gain a clear view of backend processes, ensuring smooth operations.
- **Proactive Issue Management:** Early detection and resolution of errors enhance system reliability.



ID	LOG MESSAGE
12189	2024-12-03 13:19:12,330 - urllib3.connectionpool - DEBUG - https://www.googleapis.com:443 "GET /oauth2/v1/certs HTTP/1.1" 200 1566
12188	2024-12-03 13:19:12,243 - urllib3.connectionpool - DEBUG - Starting new HTTPS connection (1): www.googleapis.com:443
12187	2024-12-03 13:19:12,230 - google.auth.transport.requests - DEBUG - Making request: GET https://www.googleapis.com/oauth2/v1/certs
12186	2024-12-03 13:19:12,222 - requests_oauthlib.oauth2_session - DEBUG - Obtained token {'access_token': 'ya29.a0AeDCIZDrlqiWO_Pv-s_5tzbz8JOckVJ1Z6AHEsiWqtKkb8BpXyHnSDvaef07UPWxBpPx13zyVlmBdW5fgqLUISPf-txovNSjDN8gyvxG83hrhRnigYnc7vXU_nTESW--e8jC7mmqJ3I8ER04E5kfRA08d6aSWvDZ5x3S8NboaCgYKAZ8SARASFQHGX2MiiOWfJWAwPhNNrdhhvHPsdg0175', 'expires_in': 3599, 'refresh_token': '1//04hjGnO1gBg1iCgYIARAAGAQSNwF-L9IrzFzMJy-8EDVeZJlHsTJdJFnnjrASMbOrfUTcpAAtrDe0Pf2_IuIGR5wjv5aq9mY3TF8', 'scope': ['openid', 'https://www.googleapis.com/auth/userinfo.email', 'https://www.googleapis.com/auth/userinfo.profile'], 'token_type':... View More
12185	2024-12-03 13:19:12,215 - requests_oauthlib.oauth2_session - DEBUG - Invoking 0 token response hooks.
12184	2024-12-03 13:19:12,206 - requests_oauthlib.oauth2_session - DEBUG - Response headers were {'Date': 'Tue, 03 Dec 2024 13:19:12 GMT', 'Cache-Control': 'no-cache, no-store,... View More
12183	2024-12-03 13:19:12,199 - requests_oauthlib.oauth2_session - DEBUG - Request body was grant_type=authorization_code&code=4%2F0AeanS0bFjvYMo7mlEc0f34mCe9nx4GlxE89OucVjRX-R-La8wWUe4DrrChS7mOIqzvlMYg&redirect_uri=http%3A%2F%2Flocalhost%3A5000%2Fapi%2Fauth%2Fgoogle%2Fcallback
12182	2024-12-03 13:19:12,191 - requests_oauthlib.oauth2_session - DEBUG - Request headers were {'User-Agent': 'python-requests/2.25.1', 'Accept-Encoding': 'gzip, deflate', 'Accept': 'application/json', 'Connection': 'keep-alive', 'Content-Type':... View More
12181	2024-12-03 13:19:12,184 - requests_oauthlib.oauth2_session - DEBUG - Request url was https://oauth2.googleapis.com/token
12180	2024-12-03 13:19:12,176 - requests_oauthlib.oauth2_session - DEBUG - Request to fetch token completed with status 200.
12179	2024-12-03 13:19:12,167 - urllib3.connectionpool - DEBUG - https://oauth2.googleapis.com:443 "POST /token HTTP/1.1" 200 None

---

6. Activity Logs Section

Summary

The Activity Logs Section provides users with a detailed record of actions performed within the system, making it easy to track changes, user activities, and file updates.

What Does This Screen Show?

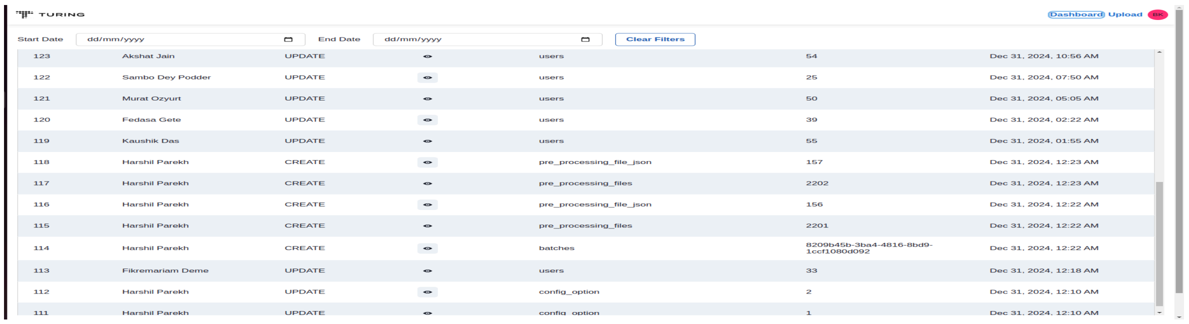
1. **Filters for Date Range**:
   - Users can filter the logs using a **Start Date** and **End Date** to view activities for a specific period.
2. **Columns Displayed**:
   - **ID**: A unique identifier for each activity log entry.
   - **User Name**: The name of the user who performed the activity.
   - **Activity Type**: Indicates the type of action (e.g., `CREATE`, `UPDATE`).
   - **Entity**: Represents the affected entity or section (e.g., `users`, `pre_processing_file_json`, `config_option`).
   - **Details/Records**: Displays additional details about the affected entity, such as count, specific ID, or changes.
   - **Timestamp**: Displays the exact date and time the action occurred.

Why Is This Useful?

- **Activity Tracking**: Allows administrators or team members to trace who made changes, when, and to which entities.
- **Error Diagnosis**: Facilitates root cause analysis by tracking changes and identifying unauthorized or mistaken actions.
- **Audit Compliance**: Ensures all actions are logged for accountability and audit readiness.

Result

- **Centralized Log Management**: Users can view all system activities in one place for efficient oversight.
- **Enhanced Productivity**: Filters and sorting options allow users to pinpoint information quickly.
- **Better System Control**: By tracking activities, users can ensure that actions align with organizational policies and procedures.



123	Akshat Jain	UPDATE	👁	users	54	Dec 31, 2024, 10:56 AM
122	Sambo Dey Podder	UPDATE	👁	users	25	Dec 31, 2024, 07:50 AM
121	Murat Ozyurt	UPDATE	👁	users	50	Dec 31, 2024, 05:05 AM
120	Fedasa Gete	UPDATE	👁	users	39	Dec 31, 2024, 02:22 AM
119	Kaushik Das	UPDATE	👁	users	55	Dec 31, 2024, 01:55 AM
118	Harshil Parekh	CREATE	👁	pre_processing_file_json	157	Dec 31, 2024, 12:23 AM
117	Harshil Parekh	CREATE	👁	pre_processing_files	2202	Dec 31, 2024, 12:23 AM
116	Harshil Parekh	CREATE	👁	pre_processing_file_json	156	Dec 31, 2024, 12:22 AM
115	Harshil Parekh	CREATE	👁	pre_processing_files	2201	Dec 31, 2024, 12:22 AM
114	Harshil Parekh	CREATE	👁	batches	8209b45b-3ba4-4816-8bd9-1ccf1080d092	Dec 31, 2024, 12:22 AM
113	Fikremariam Deme	UPDATE	👁	users	33	Dec 31, 2024, 12:18 AM
112	Harshil Parekh	UPDATE	👁	config_option	2	Dec 31, 2024, 12:10 AM
111	Harshil Parekh	UPDATE	👁	config_option	1	Dec 31, 2024, 12:10 AM

---

7. **Configuration Settings Modal**

Summary

The Configuration Settings modal allows users to toggle and manage specific application settings, including file upload and validation options.

What Does This Screen Show?

Settings Options:

- **Enable Penguin S3 Upload**:
  Toggles the functionality for uploading files to the Penguin S3 bucket.
  - **Toggle ON**: Activates Penguin S3 uploads.
  - **Toggle OFF**: Deactivates Penguin S3 uploads.
- **Enable Turing S3 Upload**:
  Toggles the functionality for uploading files to the Turing S3 bucket.
  - **Toggle ON**: Activates Turing S3 uploads.

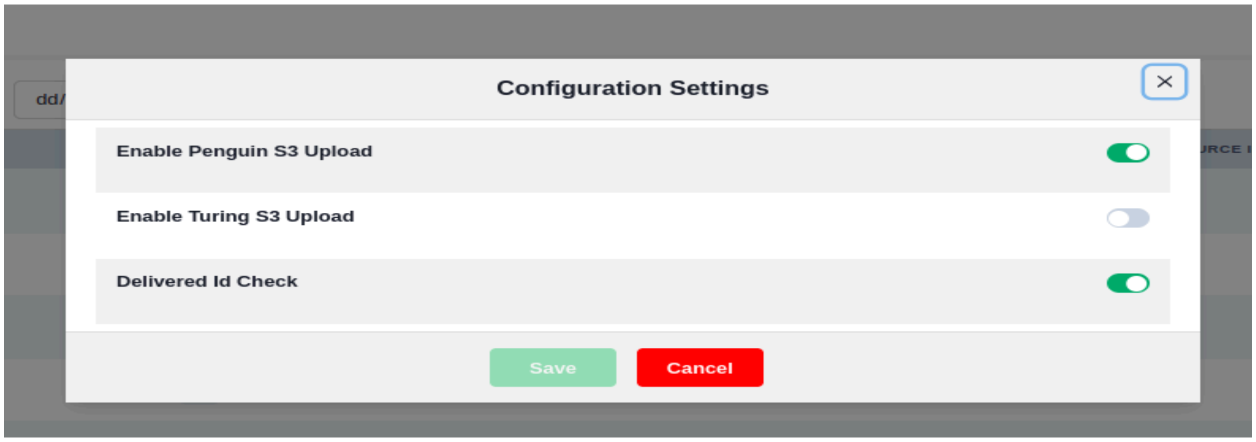
    - **Toggle OFF**: Deactivates Turing S3 uploads.
- **Delivered ID Check**:
  Enables or disables the ID validation process for delivered files.
    - **Toggle ON**: Ensures that IDs are checked before proceeding.
    - **Toggle OFF**: Skips the ID validation process.

Why Is This Useful?

- Provides administrators with granular control over file upload and validation processes.
- Enables quick and intuitive configuration changes without requiring direct database modifications.
- Helps ensure that the correct settings are applied based on the operational needs.

Result

- **Customizable Functionality**: Users can tailor upload and validation processes to suit different workflows.
- **Error Reduction**: Prevents unwanted actions by ensuring settings are explicitly toggled before activation.
- **Efficient Management**: Allows instant configuration updates without downtime.


Configuration Settings

Enable Penguin S3 Upload ●

Enable Turing S3 Upload ○

Delivered Id Check ●

8. Generate Token Modal

What Does This Modal Show?

The **Generate Token Modal** provides users with a **Bearer Token**, which is valid for a specified time period (24 hours in this case). This token is primarily used for API authentication.

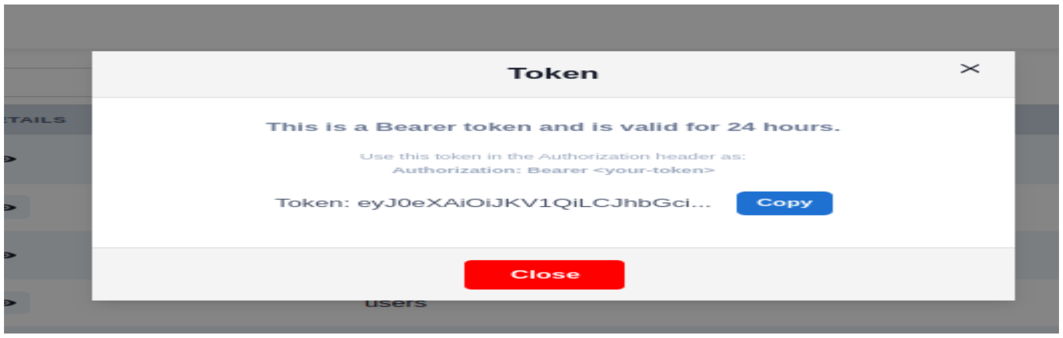
1. **Token Details**:
   - **Token Validity**:
     States that the token is valid for a specific period, ensuring security by requiring token regeneration after expiration.
     - Example: "This is a Bearer token and is valid for 24 hours."
   - **Usage Instruction**:
     Guides the user on how to include the token in API requests:
     - `Authorization: Bearer <your-token>`
   - **Token String**:
     Displays the generated token (e.g., `eyJ0eXAiOiJKV1QiLCJh...`). The token is truncated in the display for better readability.

Why Is This Useful?

- **Simplified API Access**: Provides users with a ready-to-use Bearer Token for authentication.
- **Enhanced Security**: Ensures tokens expire within a limited timeframe, reducing unauthorized access risks.
- **User Convenience**: Includes an easy-to-use **Copy** button for quick token sharing or implementation.

Result

- **Secure API Authentication**: Reduces the chance of long-term token misuse.
- **Efficient Workflows**: Enables users to generate and use authentication tokens without complex configurations.
- **Clear Instructions**: Helps users understand token usage with concise instructions.


Token

This is a Bearer token and is valid for 24 hours.

Use this token in the Authorization header as:
Authorization: Bearer <your-token>

Token: eyJ0eXAiOiJKV1QiLCJhbGci...

8. Admin Role: User Management & Role Management

Overview

Admins have access to manage users and roles within the system. Role Management provides admins the flexibility to define custom roles with granular permissions tailored to organizational requirements.

User Management

1. **Add Users**:
    - Admins can add new users to the system with basic details like name, email, and role assignment.
    - Each user must be assigned to a specific role to define their permissions and actions within the platform.

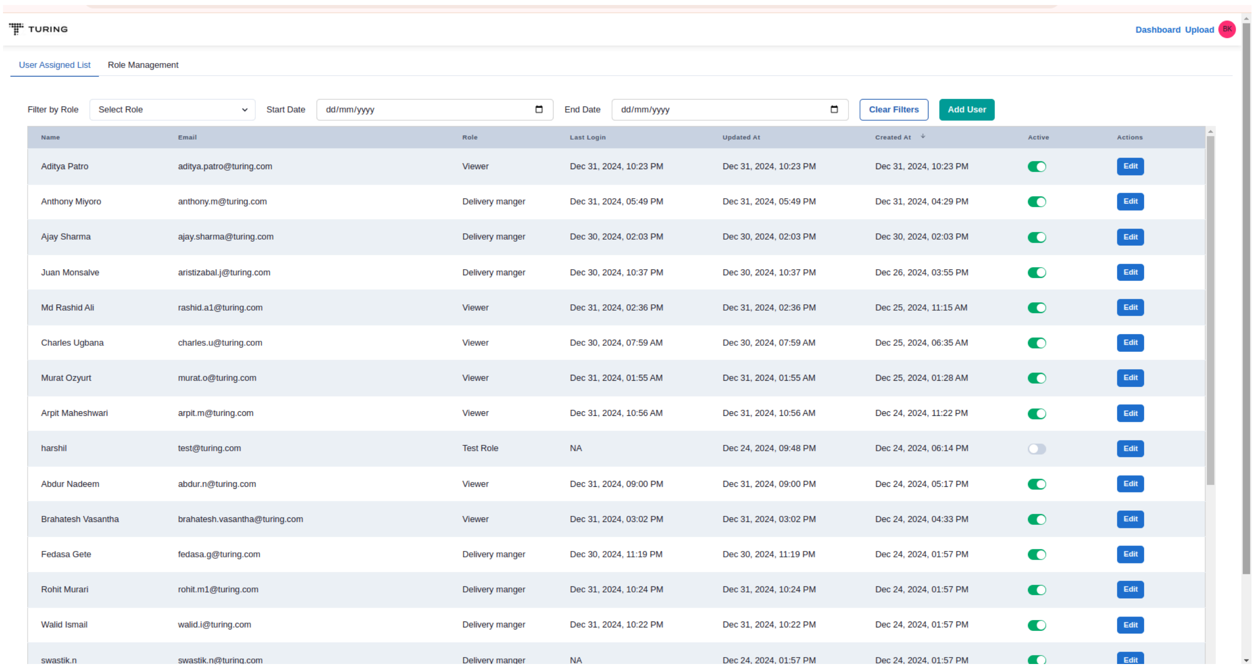
Role Management

1. **Default Admin Permissions**:
    - Admins have **full access to all permissions** by default.
    - Admins can create roles and assign these roles to other users.
2. **Create Role Workflow**:
    - **Role Name**:
        - Field to specify the name of the role (e.g., "Manager," "Viewer").
    - **Description**:
        - Field to add a description for the role's purpose (e.g., "Manager role for managing logs and configurations").
    - **Permissions**:
        - Admins can define specific permissions for the role, including:
            - **User Management**: Access to manage user accounts.
            - **Logs**: Access to view or manage system logs.
            - **Upload to S3**: Permissions to upload files to S3.
            - **Download from S3**: Permissions to download files from S3.
            - **Configuration**: Permissions to access and manage system configuration settings.
    - **Select All**:
        - A checkbox option to grant **all available permissions** to the role.
3. **Actions**:
    - **Save**:

        - Saves the new role with its name, description, and assigned permissions.
    - **Cancel**:
        - Discards the operation and closes the modal without saving.

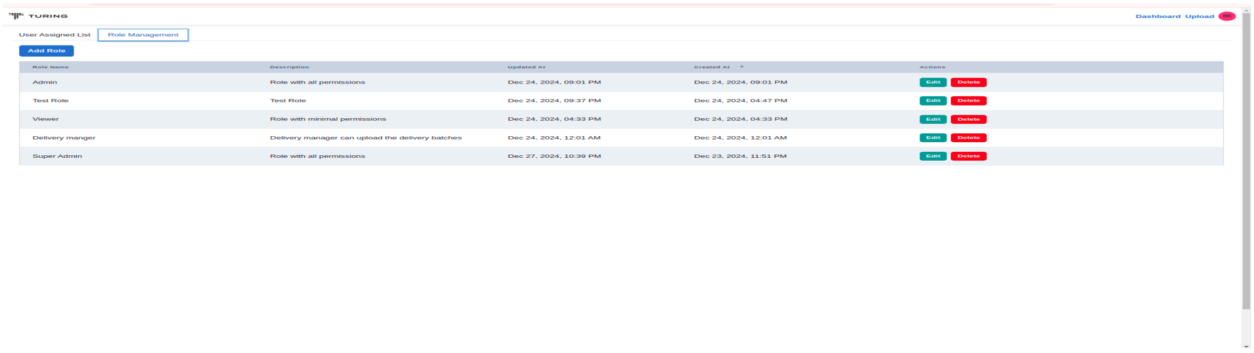
Use Case Scenarios

1. **Assigning Roles to Users**:
   Admins can create roles (e.g., “S3 Access Only”) with specific permissions, ensuring users have access to only what is needed.
2. **Flexible Permissions Management**:
   Enables the addition or removal of permissions from roles as organizational needs evolve.



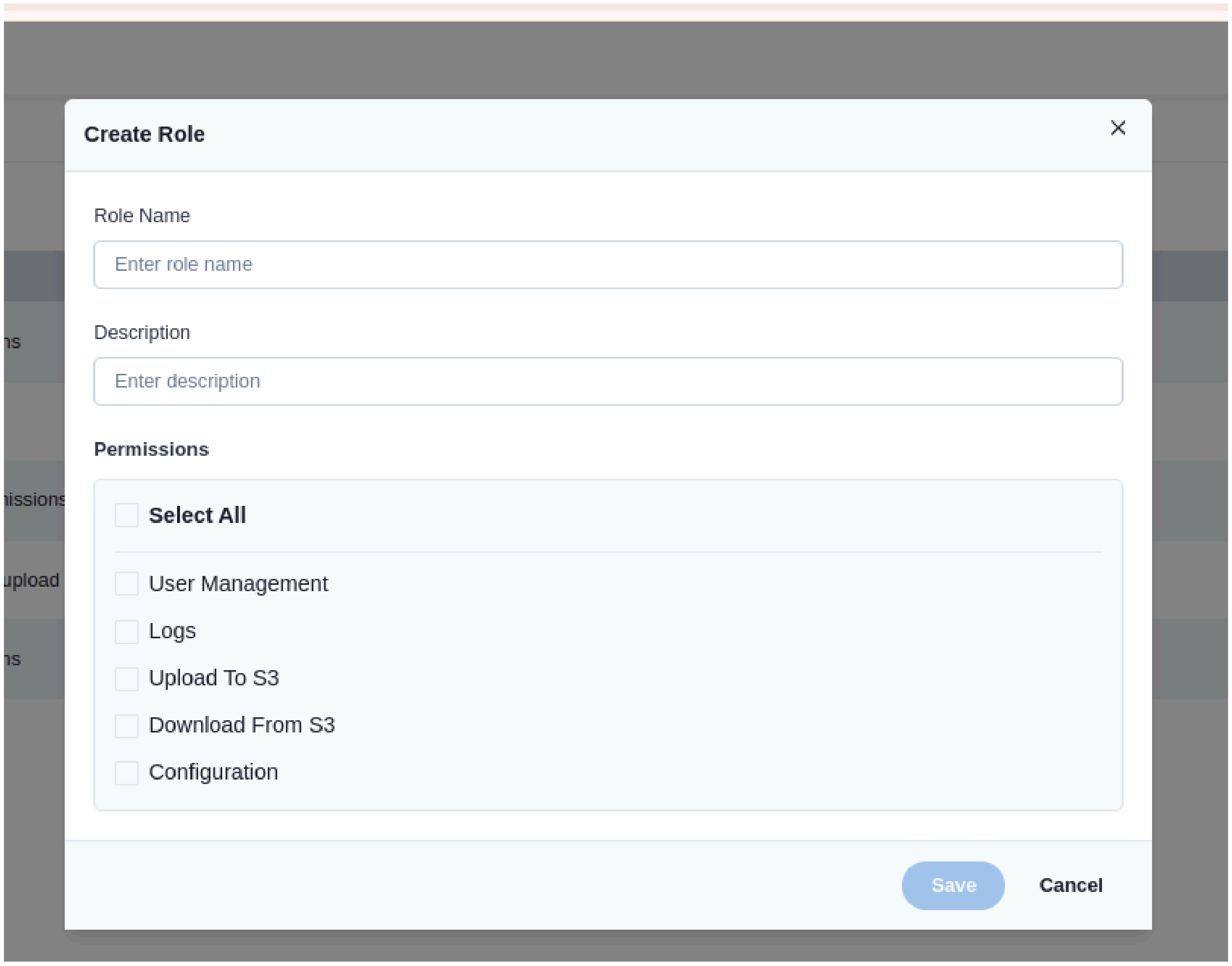
Name	Email	Role	Last Login	Updated At	Created At	Active	Actions
Aditya Patro	aditya.patro@turing.com	Viewer	Dec 31, 2024, 10:23 PM	Dec 31, 2024, 10:23 PM	Dec 31, 2024, 10:23 PM	●	Edit
Anthony Miyoro	anthony.m@turing.com	Delivery manger	Dec 31, 2024, 05:49 PM	Dec 31, 2024, 05:49 PM	Dec 31, 2024, 04:29 PM	●	Edit
Ajay Sharma	ajay.sharma@turing.com	Delivery manger	Dec 30, 2024, 02:03 PM	Dec 30, 2024, 02:03 PM	Dec 30, 2024, 02:03 PM	●	Edit
Juan Monsalve	aristizabal.j@turing.com	Delivery manger	Dec 30, 2024, 10:37 PM	Dec 30, 2024, 10:37 PM	Dec 26, 2024, 03:55 PM	●	Edit
Md Rashid Ali	rashid.a1@turing.com	Viewer	Dec 31, 2024, 02:36 PM	Dec 31, 2024, 02:36 PM	Dec 25, 2024, 11:15 AM	●	Edit
Charles Ugbana	charles.u@turing.com	Viewer	Dec 30, 2024, 07:59 AM	Dec 30, 2024, 07:59 AM	Dec 25, 2024, 06:35 AM	●	Edit
Murat Ozyurt	murat.o@turing.com	Viewer	Dec 31, 2024, 01:55 AM	Dec 31, 2024, 01:55 AM	Dec 25, 2024, 01:28 AM	●	Edit
Arpit Maheshwari	arpit.m@turing.com	Viewer	Dec 31, 2024, 10:56 AM	Dec 31, 2024, 10:56 AM	Dec 24, 2024, 11:22 PM	●	Edit
harshil	test@turing.com	Test Role	NA	Dec 24, 2024, 09:48 PM	Dec 24, 2024, 06:14 PM	○	Edit
Abdur Nadeem	abdur.n@turing.com	Viewer	Dec 31, 2024, 09:00 PM	Dec 31, 2024, 09:00 PM	Dec 24, 2024, 05:17 PM	●	Edit
Brahatesh Vasantha	brahatesh.vasantha@turing.com	Viewer	Dec 31, 2024, 03:02 PM	Dec 31, 2024, 03:02 PM	Dec 24, 2024, 04:33 PM	●	Edit
Fedasa Gete	fedasa.g@turing.com	Delivery manger	Dec 30, 2024, 11:19 PM	Dec 30, 2024, 11:19 PM	Dec 24, 2024, 01:57 PM	●	Edit
Rohit Murari	rohit.m1@turing.com	Delivery manger	Dec 31, 2024, 10:24 PM	Dec 31, 2024, 10:24 PM	Dec 24, 2024, 01:57 PM	●	Edit
Walid Ismail	walid.i@turing.com	Delivery manger	Dec 31, 2024, 10:22 PM	Dec 31, 2024, 10:22 PM	Dec 24, 2024, 01:57 PM	●	Edit
swastik.n	swastik.n@turing.com	Delivery manger	NA	Dec 24, 2024, 01:57 PM	Dec 24, 2024, 01:57 PM	●	Edit

User Managment



Role Name	Description	Updated At	Created At	Actions
Admin	Role with all permissions	Dec 24, 2024, 09:01 PM	Dec 24, 2024, 09:01 PM	Edit Delete
Test Role	Test Role	Dec 24, 2024, 09:37 PM	Dec 24, 2024, 04:47 PM	Edit Delete
Viewer	Role with minimal permissions	Dec 24, 2024, 04:33 PM	Dec 24, 2024, 04:33 PM	Edit Delete
Delivery manger	Delivery manager can upload the delivery batches	Dec 24, 2024, 12:01 AM	Dec 24, 2024, 12:01 AM	Edit Delete
Super Admin	Role with all permissions	Dec 27, 2024, 10:39 PM	Dec 23, 2024, 11:51 PM	Edit Delete

Role Managment



Create Role

Role Name

Enter role name

Description

Enter description

Permissions

- [ ] Select All
- [ ] User Management
- [ ] Logs
- [ ] Upload To S3
- [ ] Download From S3
- [ ] Configuration

Add Role