**Apache Airflow Documentation**

**Introduction**

Apache Airflow is an open-source platform to programmatically author, schedule, and monitor workflows. It allows users to define workflows as Directed Acyclic Graphs (DAGs), where each task represents a unit of work. Airflow is highly extensible, scalable, and widely used for orchestrating data pipelines in real-world projects.

**Features of Apache Airflow**

* **Open Source** – Completely free and community-driven.
* **DAGs (Directed Acyclic Graphs)** – Workflows are represented as DAGs of tasks with dependencies.
* **Scalability** – Can handle thousands of tasks across distributed systems.
* **Extensibility** – Supports custom operators, hooks, and sensors.
* **Web UI** – Provides an intuitive UI for monitoring and managing workflows.
* **Scheduler** – Executes tasks on a defined schedule or in response to events.
* **Integration** – Supports integration with databases, cloud platforms, and messaging systems.

**Setting up Apache Airflow (Pipeline Setup)**

**Download docker-compose.yaml**

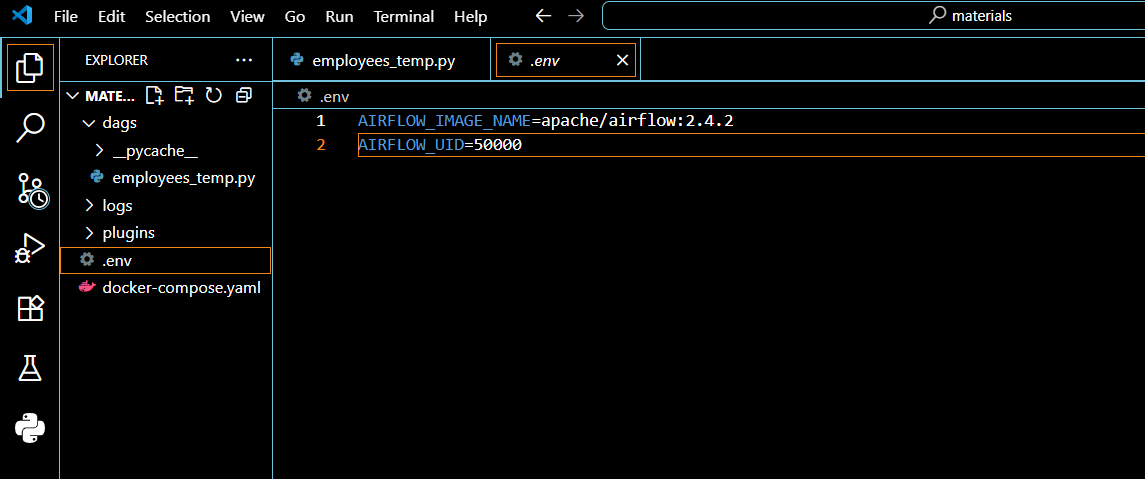
* In VS Code terminal, I ran the command to download the docker-compose.yaml file for Airflow.

**Save YAML file**

* I saved the docker-compose.yaml file inside a folder named **materials** and opened it in VS Code.

**Create .env file**

* In the same folder, I created a .env file and added the required environment configurations.



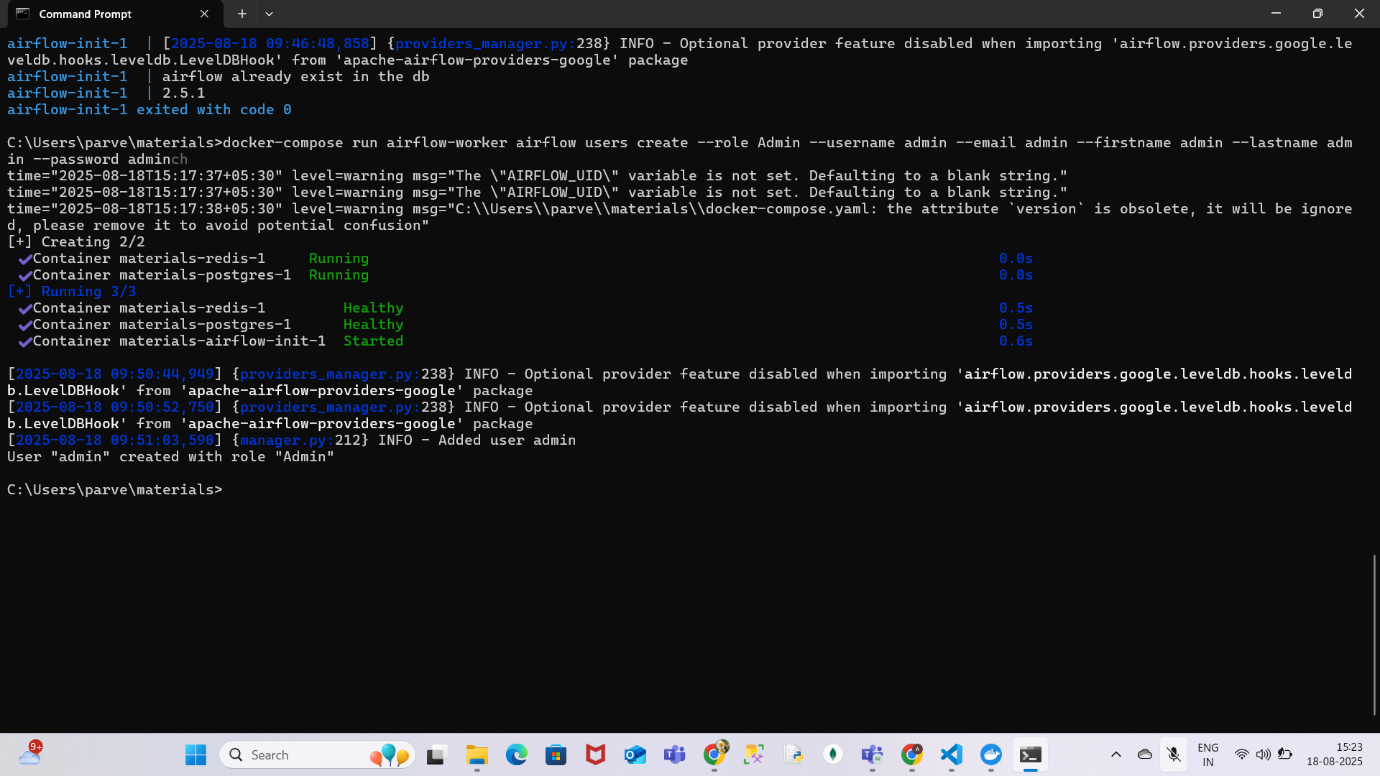
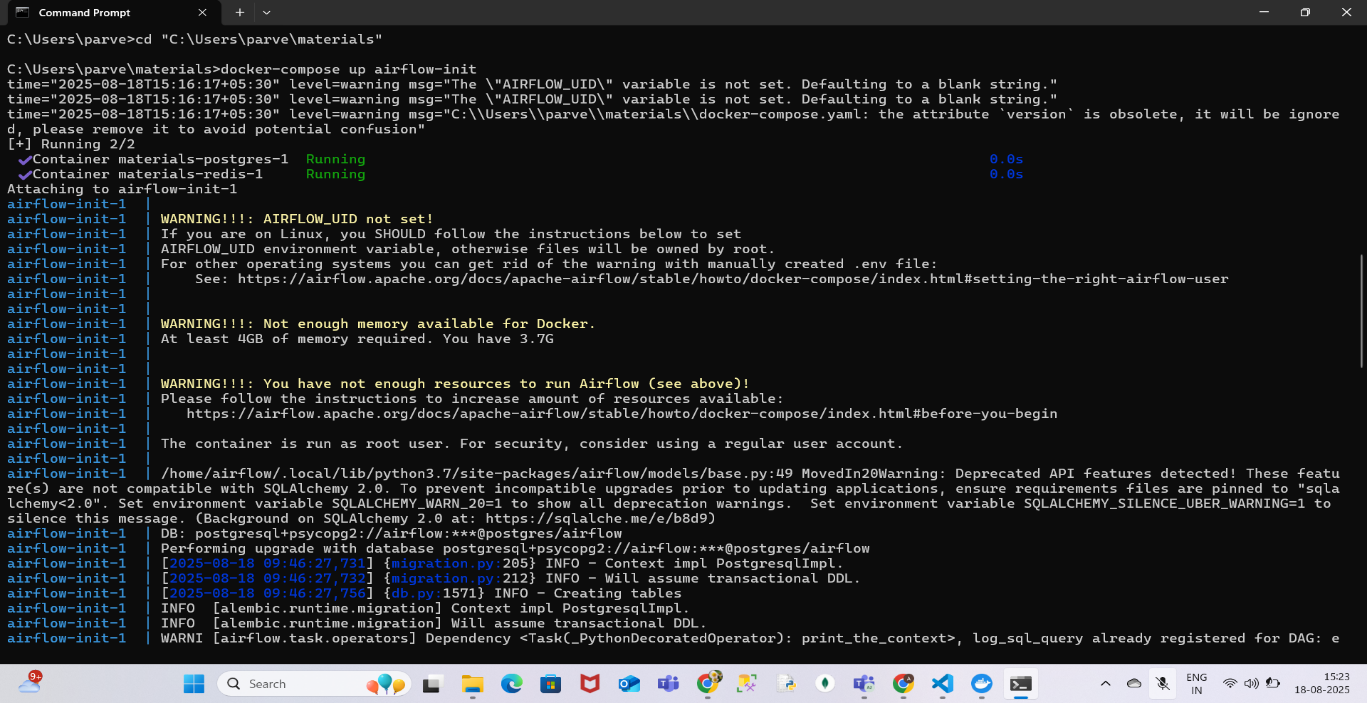
**Run Docker Compose**

* In the terminal (command prompt), I executed the command:

docker-compose up airflow-init

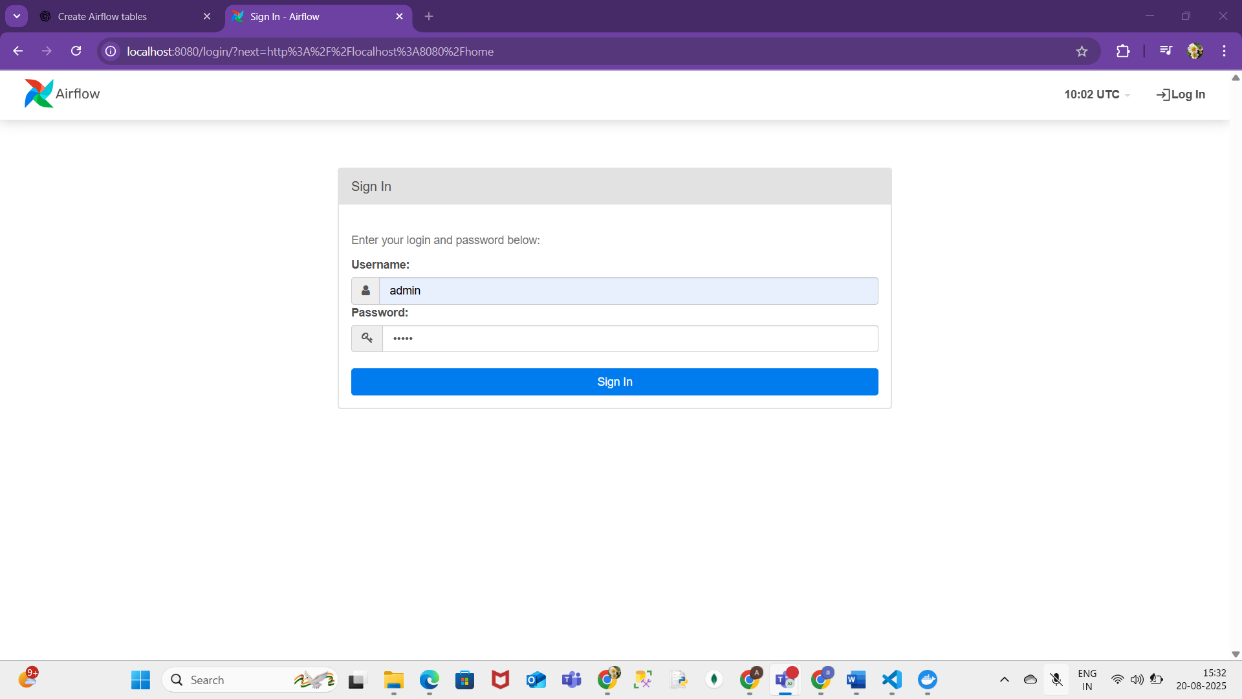
docker-compose up

This started the Airflow services (scheduler, webserver, postgres, etc.).



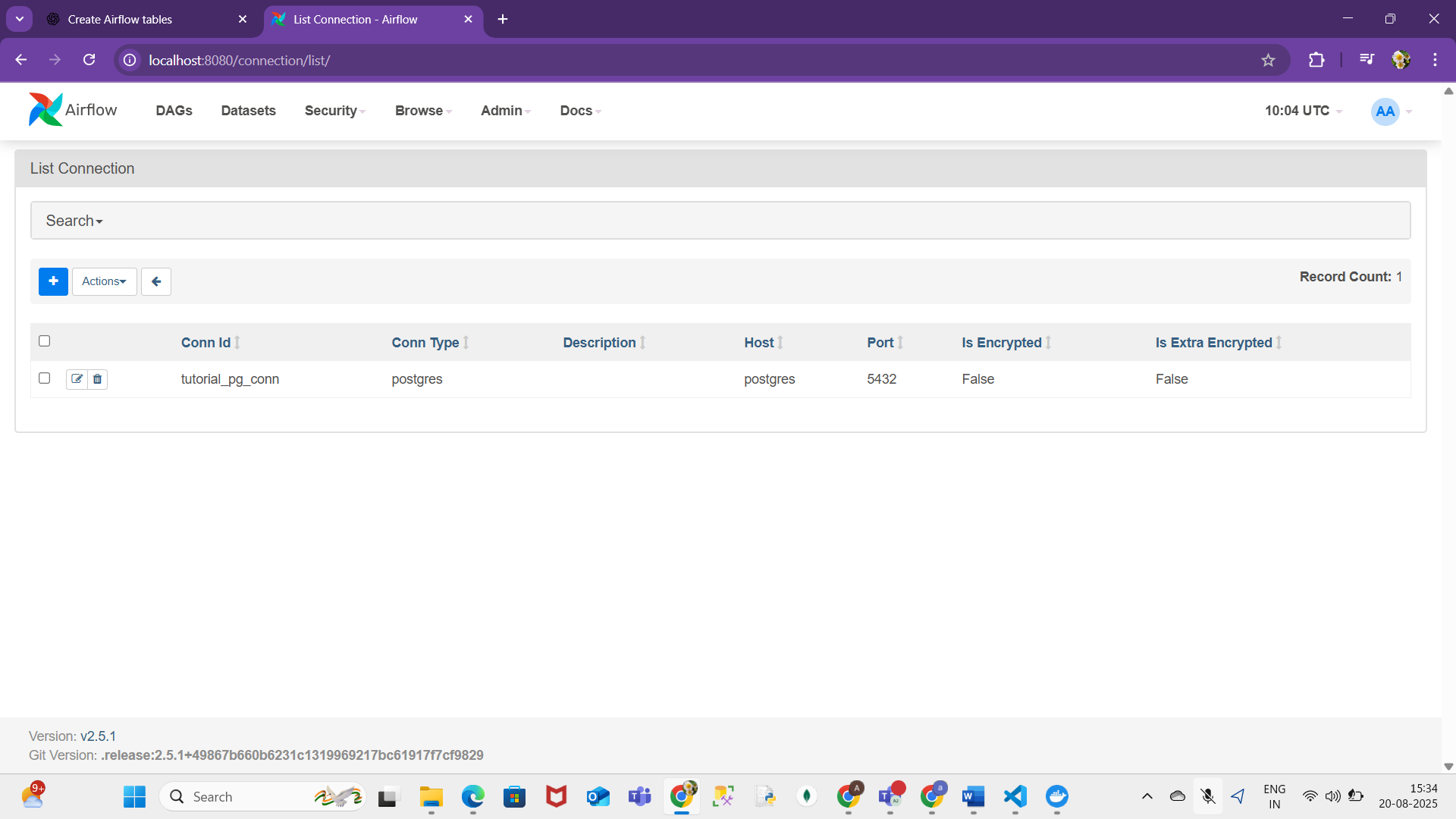
**Access Airflow Web UI**

* Opened the browser at **http://localhost:8080**.
* Logged in using:
  + - Username: **admin**
    - Password: **admin**



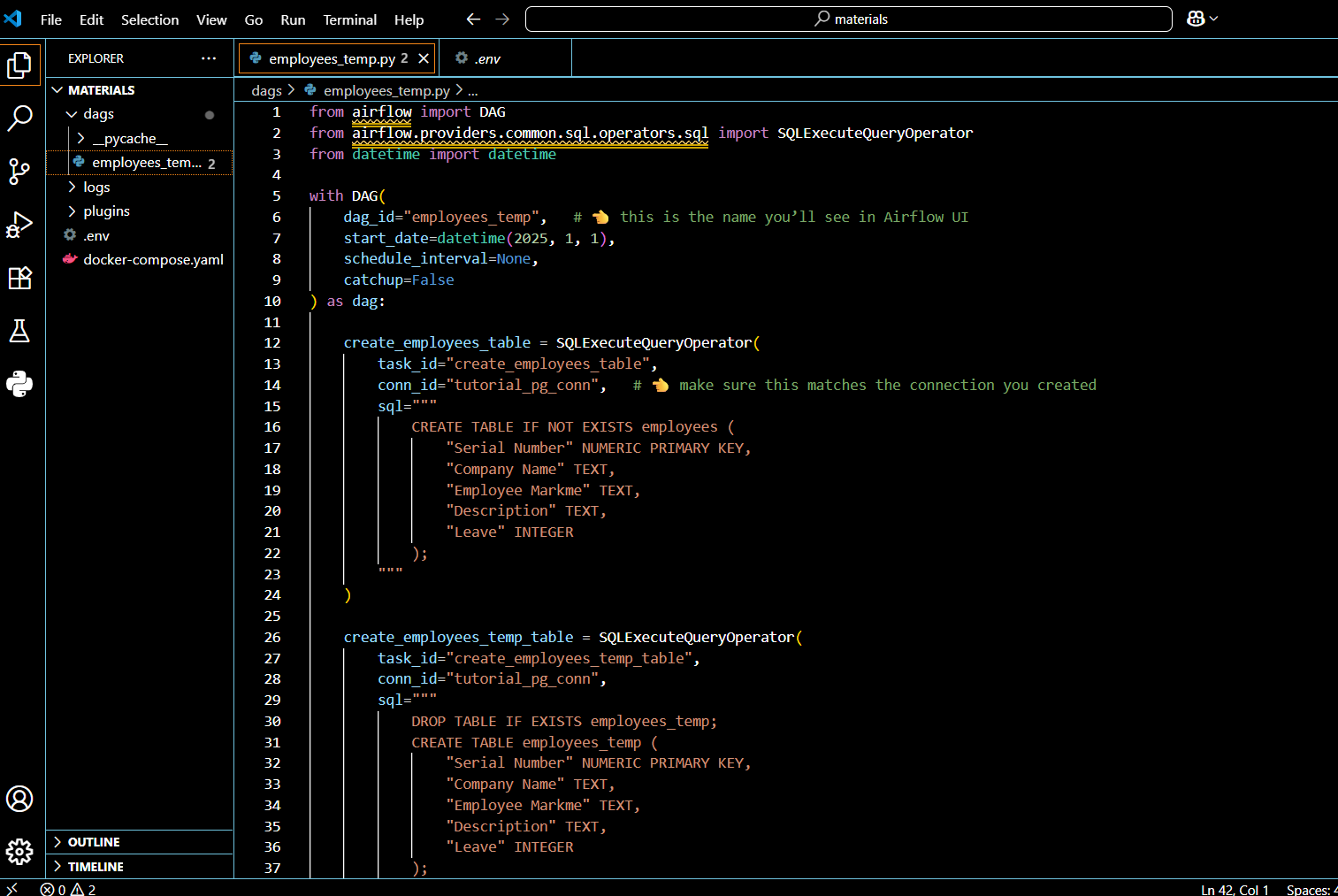
**Creating a Connection**

* Navigated to **Admin → Connections** in the Airflow UI.
* Clicked **+ Add Connection** and filled the details:
  + Connection ID: tutorial\_pg\_conn
  + Connection Type: Postgres
  + Host: postgres
  + Database: airflow
  + Login: airflow
  + Password: airflow
  + Port: 5432

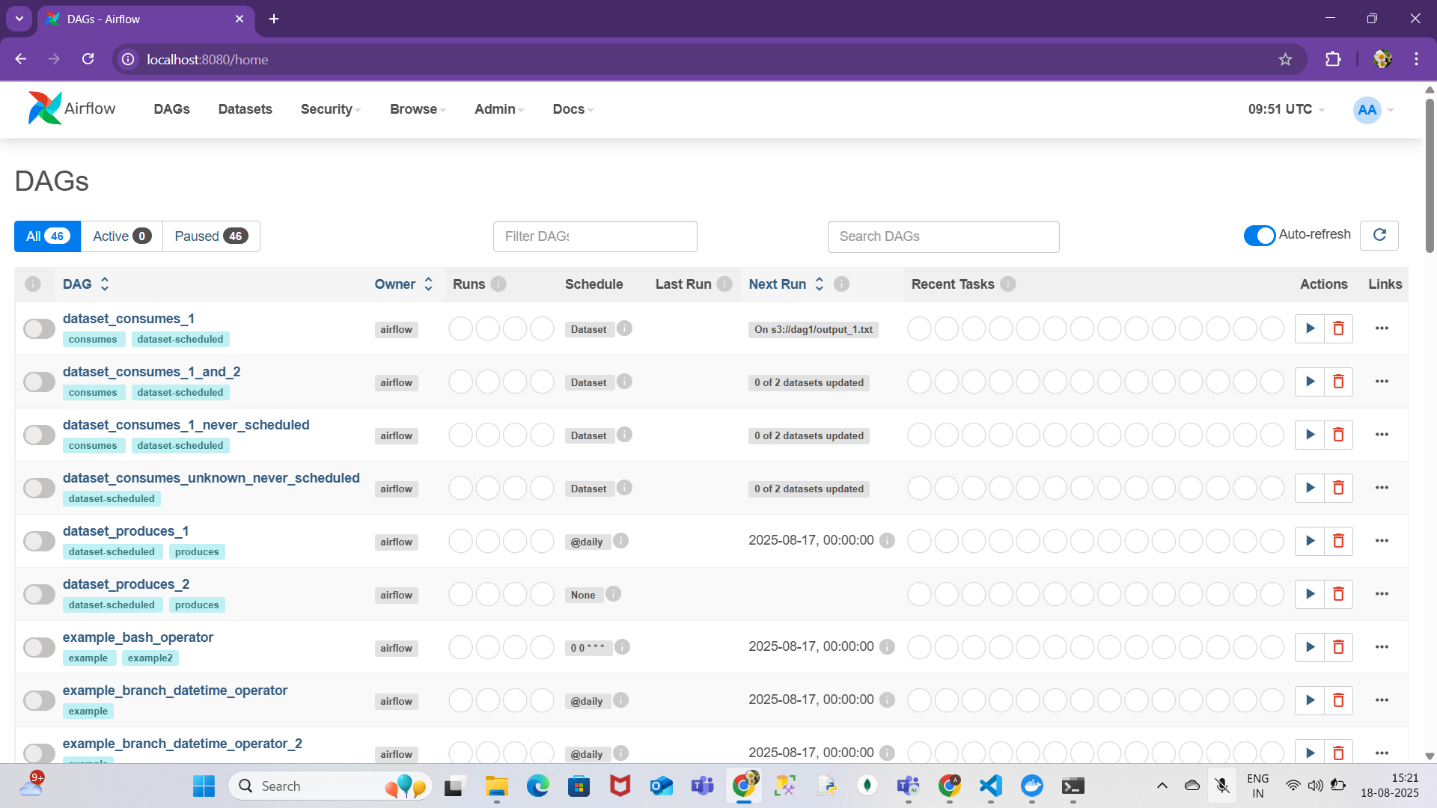


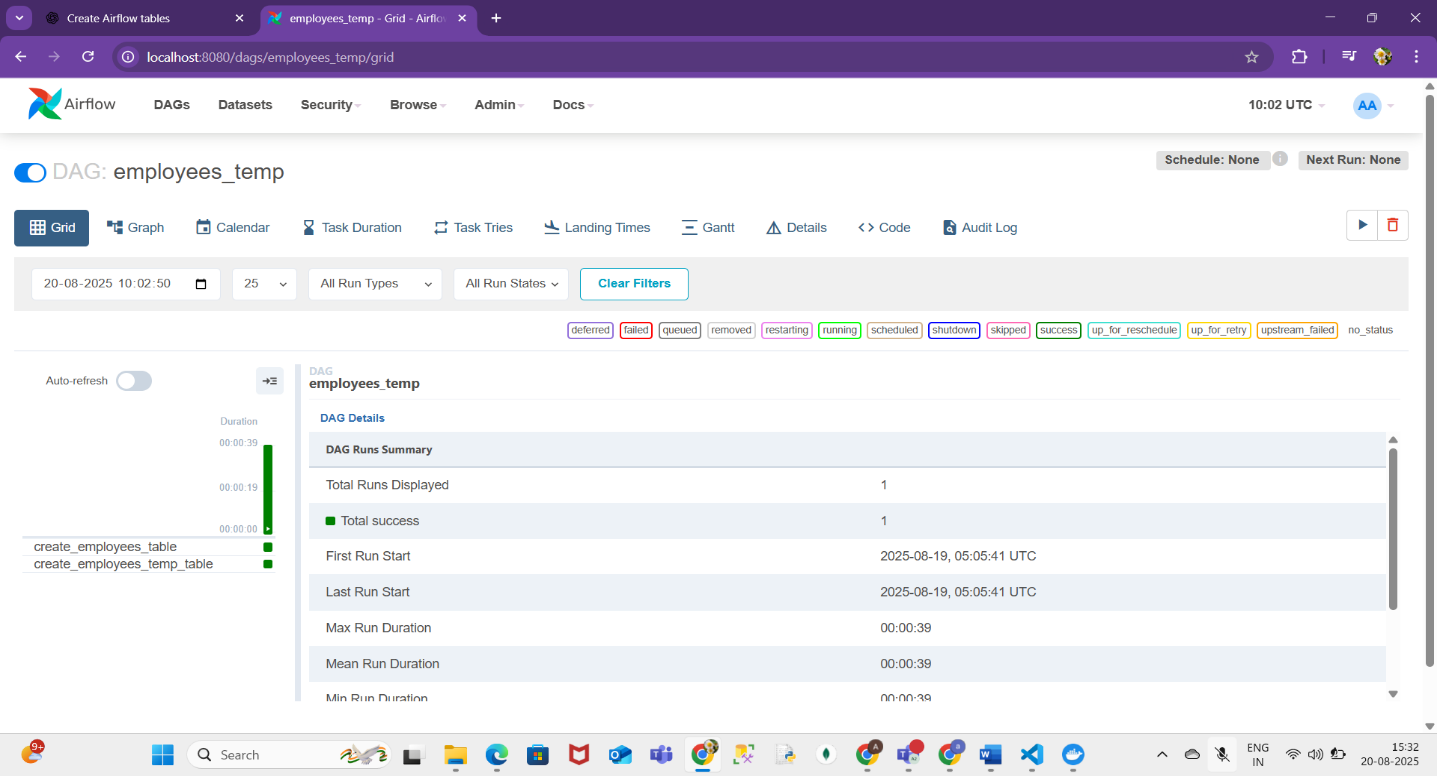
**DAGs in Airflow**

* **Definition:**  
  A DAG (Directed Acyclic Graph) is a collection of tasks with defined dependencies. In Airflow, DAGs are written in Python and describe the workflow structure.
* **Creating a DAG File:**
  + Inside the dags/ folder in VS Code, I created a Python file, for example employees\_temp.py.
  + In this file, I defined tasks for creating staging and final tables in Postgres.



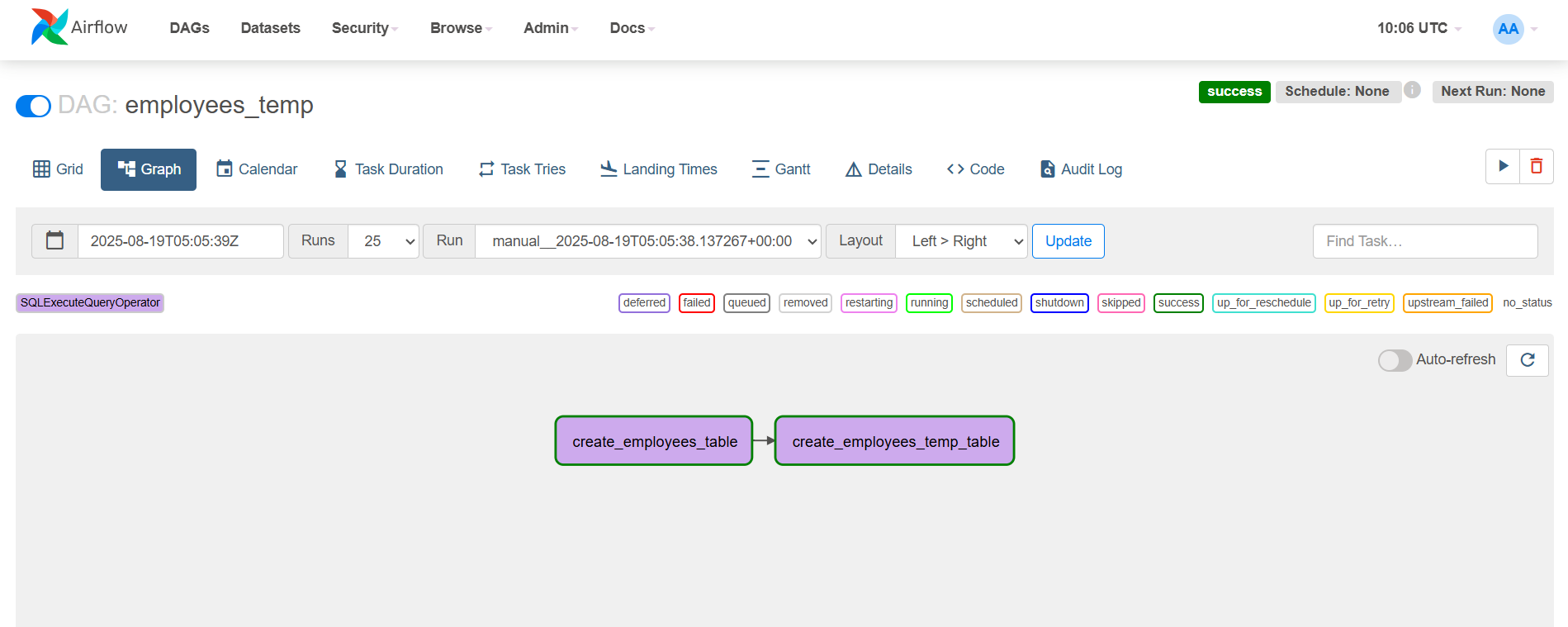
* **Running the DAG:**
  + After saving the DAG file, I refreshed Airflow UI.
  + The DAG appeared in the DAGs list.
  + I unpaused it (enabled it) and clicked **Trigger DAG**.





**DAG Views**

* **Graph View** → Shows tasks in a flow (arrows indicate dependencies).
* **Tree View** → Shows runs of the DAG in a timeline.



**Conclusion**

Apache Airflow makes it easy to manage data pipelines by combining code-based workflows with powerful scheduling and monitoring features. Using Docker Compose for setup, creating connections, and writing DAGs in Python, we can orchestrate complex workflows reliably and efficiently.