Coding challenge – Apache Airflow

Airflow features and building pipeline steps and view

What is Apache Airflow?

Apache Airflow is an open-source workflow orchestration platform for programmatically authoring, scheduling, and monitoring workflows. It is widely used for ETL pipelines, data processing, machine learning workflows, and automation.

Key Features of Airflow

1. DAGs (Directed Acyclic Graphs)
    - Pipelines are represented as DAGs.
    - A DAG is a collection of tasks with dependencies (execution order).
2. Flexible Scheduling
    - Cron-like expressions.
    - Timezone-aware.
    - Manual trigger option available.
3. Operators
    - BashOperator → runs bash commands.
    - PythonOperator → runs Python functions.
    - SQLExecuteQueryOperator → runs SQL queries.
    - Cloud integrations (AWS, GCP, Azure).
4. Extensible
    - You can build custom operators, hooks, and sensors.
5. Monitoring & UI
    - Web UI to monitor DAGs, view logs, retry failed tasks.
    - CLI and REST API support.
6. Scalability
    - Can run on a single machine or scale with Celery, Kubernetes, or other executors

Building pipeline steps and view

Step 1. Install Prerequisites

Make sure you have installed:

- Docker Desktop (latest version)
- docker-compose (comes with Docker Desktop)

Step 2. Create Airflow Project

Open PowerShell and create a working directory:

mkdir C:\Users\shree\materials

cd C:\Users\shree\materials

Inside this folder create the Airflow structure:

mkdir dags logs plugins

Step 3. Create docker-compose.yaml

Inside C:\Users\shree\materials, create a file docker-compose.yaml with this content:

```
version: '3'
x-airflow-common:
  &airflow-common
  image: apache/airflow:2.4.2
  environment:
    - AIRFLOW__CORE__EXECUTOR=CeleryExecutor
    - AIRFLOW__CORE__FERNET_KEY=YOUR_RANDOM_KEY
    - AIRFLOW__CORE__DAGS_ARE_PAUSED_AT_CREATION=true
    - AIRFLOW__CORE__LOAD_EXAMPLES=false
  volumes:
    - ./dags:/opt/airflow/dags
    - ./logs:/opt/airflow/logs
    - ./plugins:/opt/airflow/plugins
  user: "${AIRFLOW_UID:-50000}:0"
  depends_on:
```

```
    - postgres
    - redis

services:
  postgres:
    image: postgres:13
    environment:
      - POSTGRES_USER=airflow
      - POSTGRES_PASSWORD=airflow
      - POSTGRES_DB=airflow
    healthcheck:
      test: ["CMD", "pg_isready", "-U", "airflow"]
      interval: 10s
      retries: 5
    restart: always

  redis:
    image: redis:latest
    healthcheck:
      test: ["CMD", "redis-cli", "ping"]
      interval: 10s
      retries: 5
    restart: always

  airflow-webserver:
    <<: *airflow-common
    command: webserver
    ports:
      - "8080:8080"
    healthcheck:
```

```
    test: ["CMD", "curl", "--fail", "http://localhost:8080/health"]
    interval: 10s
    retries: 5
  restart: always

airflow-scheduler:
  <<: *airflow-common
  command: scheduler
  restart: always

airflow-worker:
  <<: *airflow-common
  command: celery worker
  restart: always
airflow-triggerer:
  <<: *airflow-common
  command: triggerer
  restart: always
```

Step 4. Initialize Airflow DB

Run this in PowerShell inside your project folder:

```
docker-compose up airflow-webserver airflow-scheduler airflow-worker airflow-triggerer -d

docker-compose run airflow-webserver airflow db init

docker-compose run airflow-webserver airflow users create \
  --username admin \
  --firstname Airflow \
  --lastname Admin \
  --role Admin \
  --email admin@example.com \
  --password admin
```

This will create the Admin user (admin/admin).

Step 5. Start Airflow

Start all services:

docker-compose up -d

Check containers:

docker ps

You should see airflow-webserver, airflow-scheduler, etc.

Open browser → http://localhost:8080
Login → admin / admin

Step 6. Create Your First DAG

Inside C:\Users\shree\materials\dags, save this file as tuto.py:

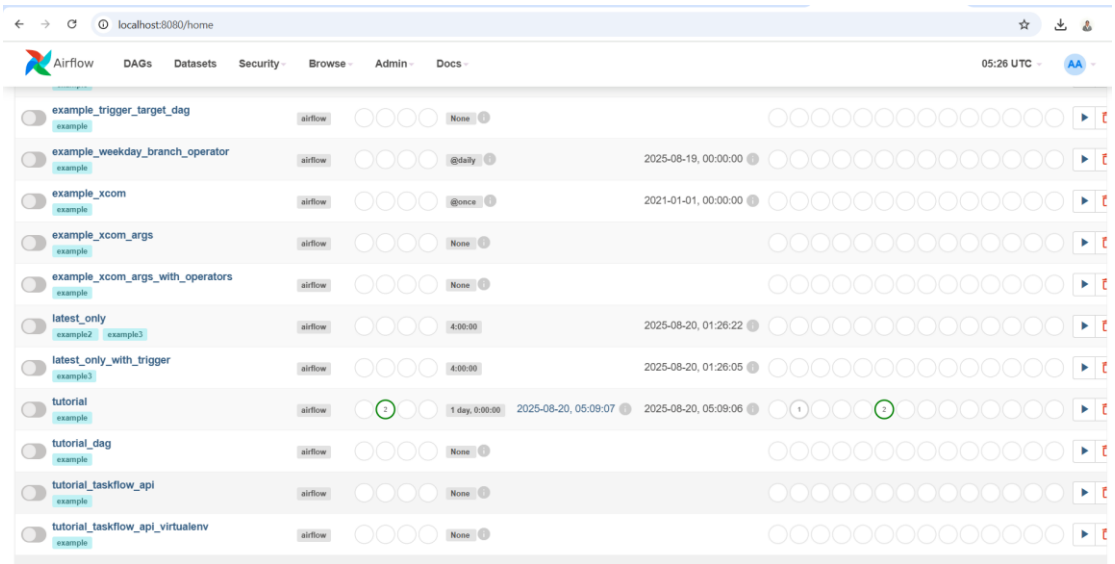
from datetime import datetime, timedelta

from airflow import DAG

from airflow.operators.bash import BashOperator

```
default_args = {
    "owner": "airflow",
    "depends_on_past": False,
    "start_date": datetime(2024, 1, 1),
    "email": ["airflow@airflow.com"],
    "email_on_failure": False,
    "email_on_retry": False,
    "retries": 1,
    "retry_delay": timedelta(minutes=5),
}

dag = DAG(
    "tutorial",
    default_args=default_args,
    description="My first DAG tutorial",
    schedule_interval=timedelta(days=1),
    catchup=False,
)

t1 = BashOperator(
    task_id="print_date",
    bash_command="date",
    dag=dag,
)

t2 = BashOperator(
    task_id="sleep",
    bash_command="sleep 5",
    retries=3,
```

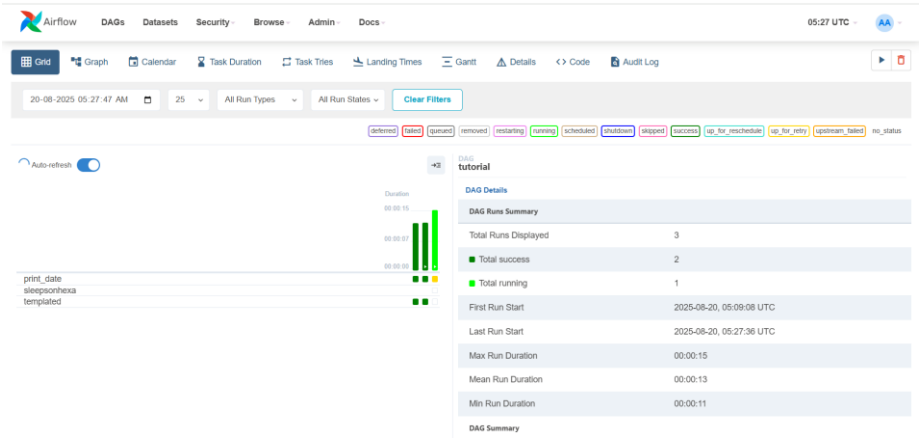
```
    dag=dag,
)

templated_command = """
{% for i in range(5) %}
  echo "{{ ds }}"
  echo "{{ macros.ds_add(ds, 7) }}"
  echo "{{ params.my_param }}"
{% endfor %}
"""

t3 = BashOperator(
    task_id="templated",
    bash_command=templated_command,
    params={"my_param": "Hello from Airflow"},
    dag=dag,
)

t1 >> [t2, t3]
```



Step 7. Run the DAG

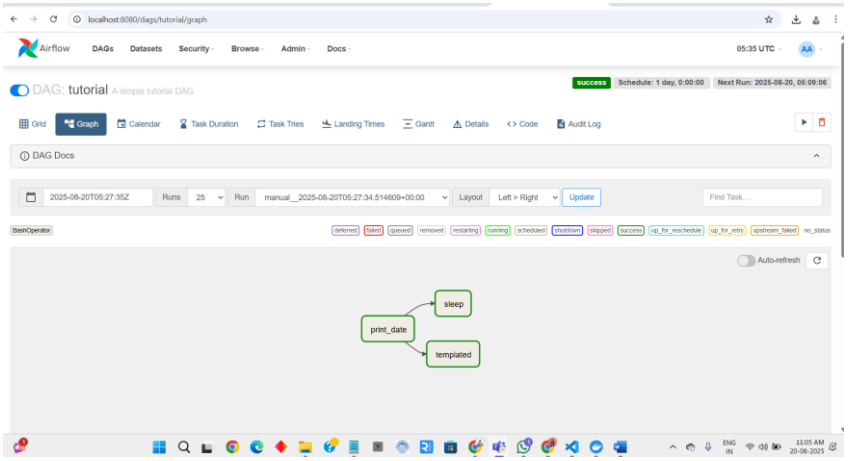
From UI

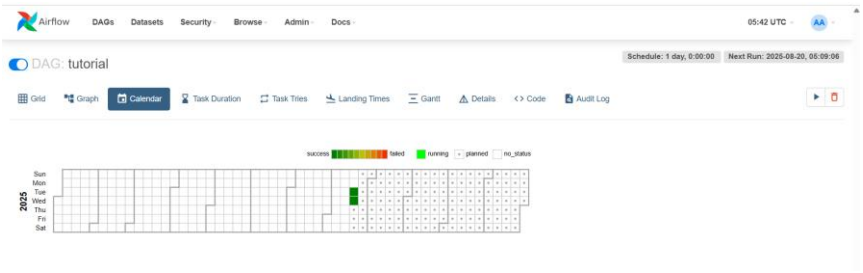
1. Go to Airflow UI → http://localhost:8080
2. Find DAG tutorial
3. Switch it ON
4. Click Trigger DAG
5. View Graph → Logs to see execution

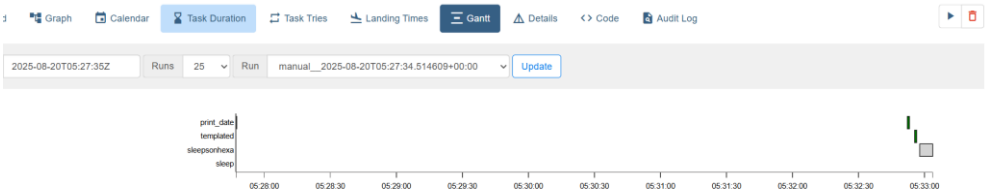


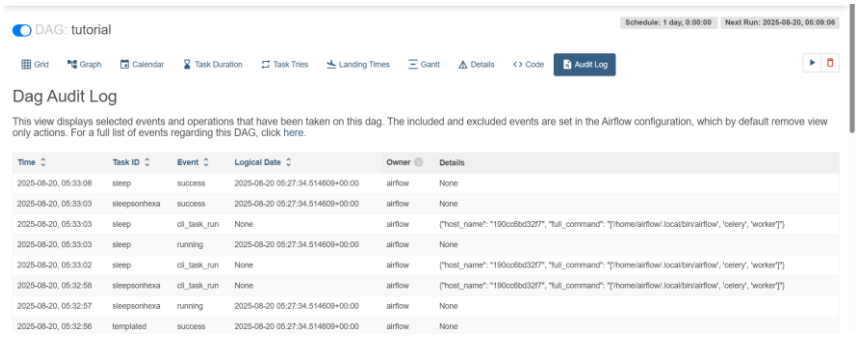
Airflow Views in UI

1. **Tree View** → shows past runs and task statuses in a tree layout.
2. **Graph View** → DAG structure (tasks as nodes).
3. **Grid View** → modern replacement of Tree View with better navigation.
4. **Gantt View** → task execution timelines.
5. **Code View** → preview DAG Python code in the UI.
6. **Task Logs** → per-task execution logs (super useful for debugging).




Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
05:42 UTC
AA
DAG: tutorial
Schedule: 1 day, 0:00:00
Next Run: 2025-08-20, 05:09:06
Grid
Graph
Calendar
Task Duration
Task Tries
Landing Times
Gantt
Details
Code
Audit Log
success
failed
running
planned
no_status
2025
Sun
Mon
Tue
Wed
Thu
Fri
Sat


Graph
Calendar
Task Duration
Task Tries
Landing Times
Gantt
Details
Code
Audit Log
2025-08-20T05:27:35Z
Runs
25
Run
manual__2025-08-20T05:27:34.514609+00:00
Update
print_date
templated
sleepsonhexa
sleep
05:28:00
05:28:30
05:29:00
05:29:30
05:30:00
05:30:30
05:31:00
05:31:30
05:32:00
05:32:30
05:33:00


DAG: tutorial
Schedule: 1 day, 0:00:00
Next Run: 2025-08-20, 05:09:06
Grid
Graph
Calendar
Task Duration
Task Tries
Landing Times
Gantt
Details
Code
Audit Log
Dag Audit Log
This view displays selected events and operations that have been taken on this dag. The included and excluded events are set in the Airflow configuration, which by default remove view only actions. For a full list of events regarding this DAG, click here.
Time
Task ID
Event
Logical Date
Owner
Details
2025-08-20, 05:33:08 sleep success 2025-08-20 05:27:34.514609+00:00 airflow None
2025-08-20, 05:33:03 sleepsonhexa success 2025-08-20 05:27:34.514609+00:00 airflow None
2025-08-20, 05:33:03 sleep cli_task_run None airflow {"host_name": "190cc6bd32f7", "full_command": "[/home/airflow/.local/bin/airflow', 'celery', 'worker']"}
2025-08-20, 05:33:03 sleep running 2025-08-20 05:27:34.514609+00:00 airflow None
2025-08-20, 05:33:02 sleep cli_task_run None airflow {"host_name": "190cc6bd32f7", "full_command": "[/home/airflow/.local/bin/airflow', 'celery', 'worker']"}
2025-08-20, 05:32:58 sleepsonhexa cli_task_run None airflow {"host_name": "190cc6bd32f7", "full_command": "[/home/airflow/.local/bin/airflow', 'celery', 'worker']"}
2025-08-20, 05:32:57 sleepsonhexa running 2025-08-20 05:27:34.514609+00:00 airflow None
2025-08-20, 05:32:56 templated success 2025-08-20 05:27:34.514609+00:00 airflow None