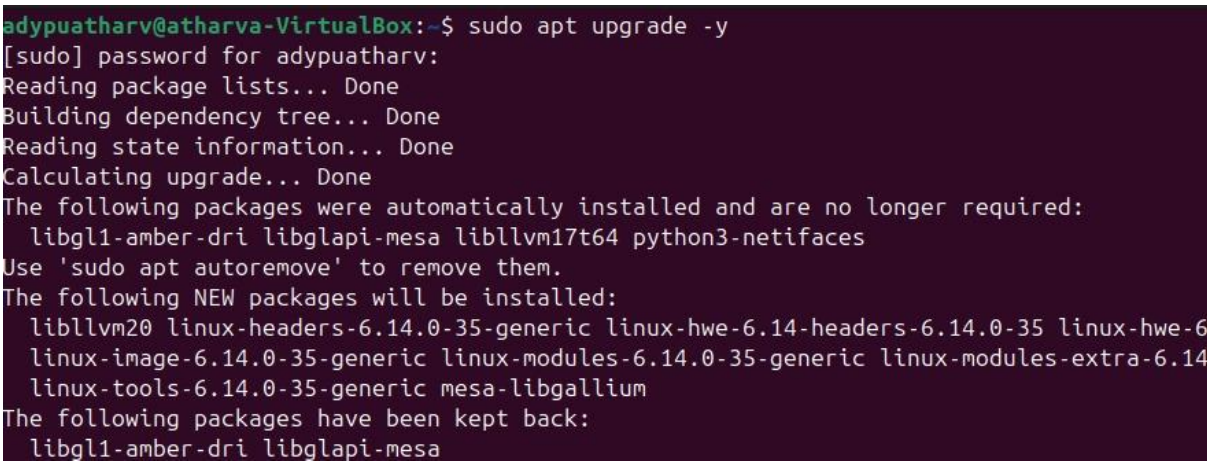
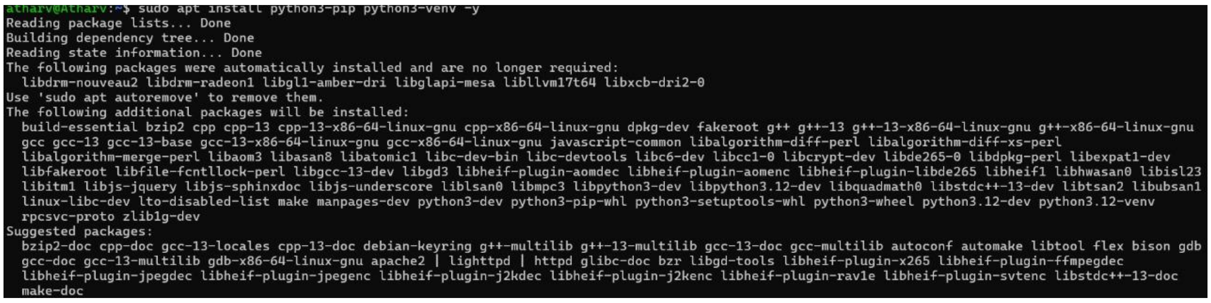
Step 1: Create and Activate Virtual Environment (if not already)

1.sudo apt update -y



```
adypuatharv@atharva-VirtualBox:~$ sudo apt upgrade -y
[sudo] password for adypuatharv:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  libgl1-amber-dri libglapi-mesa libllvm17t64 python3-netifaces
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  libllvm20 linux-headers-6.14.0-35-generic linux-hwe-6.14-headers-6.14.0-35 linux-hwe-6
  linux-image-6.14.0-35-generic linux-modules-6.14.0-35-generic linux-modules-extra-6.14
  linux-tools-6.14.0-35-generic mesa-libgallium
The following packages have been kept back:
  libgl1-amber-dri libglapi-mesa
```

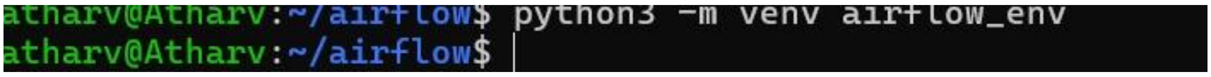
2. sudo apt install python3-pip python3.12-venv -y



```
atharv@Atharv:~$ sudo apt install python3-pip python3-venv -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libdrm-nouveau2 libdrm-radeon1 libgl1-amber-dri libglapi-mesa libllvm17t64 libxcb-dri2-0
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  build-essential bzip2 cpp cpp-13 cpp-13-x86-64-linux-gnu cpp-x86-64-linux-gnu dpkg-dev fakeroot g++ g++-13 g++-13-x86-64-linux-gnu g++-x86-64-linux-gnu
  gcc gcc-13 gcc-13-base gcc-13-x86-64-linux-gnu gcc-x86-64-linux-gnu javascript-common libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libaom3 libasan8 libatomic1 libc-dev-bin libc-devtools libc6-dev libcc1-0 libcrypt-dev libde265-0 libdpkg-perl libexpat1-dev
  libfakeroot libfile-fcntllock-perl libgcc-13-dev libgd3 libheif-plugin-aomdec libheif-plugin-aomenc libheif-plugin-libde265 libheif1 libhwasan0 libisl23
  libitm1 libjs-jquery libjs-sphinxdoc libjs-underscore liblsan0 libmpc3 libpython3-dev libpython3.12-dev libquadmath0 libstdc++-13-dev libtsan2 libubsan1
  linux-libc-dev lto-disabled-list make manpages-dev python3-dev python3-pip-whl python3-setuptools-whl python3-wheel python3.12-dev python3.12-venv
  rpcsvc-proto zlib1g-dev
Suggested packages:
  bzip2-doc cpp-doc gcc-13-locales cpp-13-doc debian-keyring g++-multilib g++-13-multilib gcc-13-doc gcc-multilib autoconf automake libtool flex bison gdb
  gcc-doc gcc-13-multilib gdb-x86-64-linux-gnu apache2 | lighttpd | httpd glibc-doc bzr libgd-tools libheif-plugin-x265 libheif-plugin-ffmpegdec
  libheif-plugin-jpegdec libheif-plugin-jpegenc libheif-plugin-j2kdec libheif-plugin-j2kenc libheif-plugin-rav1e libheif-plugin-svtenc libstdc++-13-doc
  make-doc
```

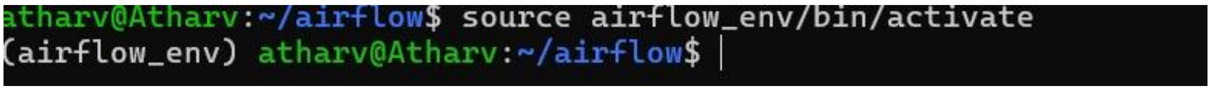
The command I ran earlier had already been executed, which is why it's showing only partially. Normally, it takes some time to complete.

3.   python3 -m venv airflow_env

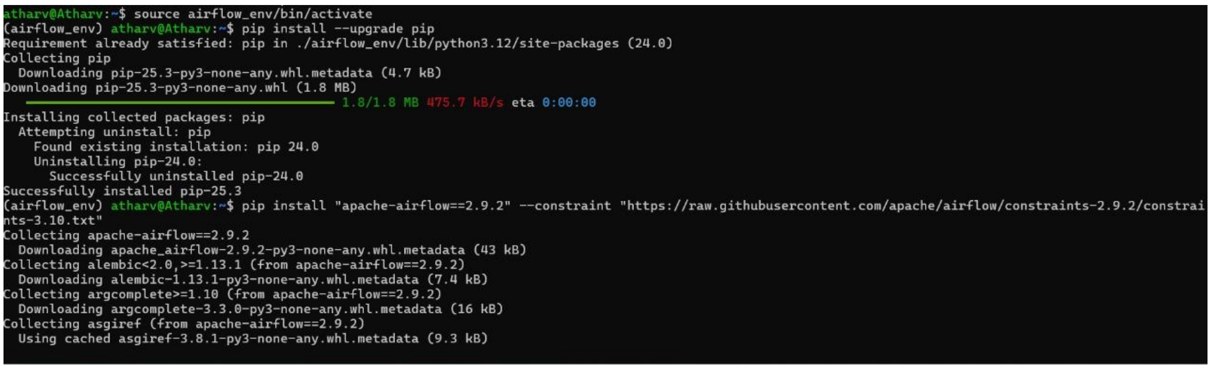


```
atharv@Atharv:~/airflow$ python3 -m venv airflow_env
atharv@Atharv:~/airflow$ |
```

source airflow_env/bin/activate



```
atharv@Atharv:~/airflow$ source airflow_env/bin/activate
(airflow_env) atharv@Atharv:~/airflow$ |
```



```
atharv@Atharv:~$ source airflow_env/bin/activate
(airflow_env) atharv@Atharv:~$ pip install --upgrade pip
Requirement already satisfied: pip in ./airflow_env/lib/python3.12/site-packages (24.0)
Collecting pip
  Downloading pip-25.3-py3-none-any.whl.metadata (4.7 kB)
Downloading pip-25.3-py3-none-any.whl (1.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 475.7 kB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 24.0
    Uninstalling pip-24.0:
      Successfully uninstalled pip-24.0
Successfully installed pip-25.3
(airflow_env) atharv@Atharv:~$ pip install "apache-airflow==2.9.2" --constraint "https://raw.githubusercontent.com/apache/airflow/constraints-2.9.2/constrai
nts-3.10.txt"
Collecting apache-airflow==2.9.2
  Downloading apache_airflow-2.9.2-py3-none-any.whl.metadata (43 kB)
Collecting alembic<2.0,>=1.13.1 (from apache-airflow==2.9.2)
  Downloading alembic-1.13.1-py3-none-any.whl.metadata (7.4 kB)
Collecting argcomplete>=1.10 (from apache-airflow==2.9.2)
  Downloading argcomplete-3.3.0-py3-none-any.whl.metadata (16 kB)
Collecting asgiref (from apache-airflow==2.9.2)
  Using cached asgiref-3.8.1-py3-none-any.whl.metadata (9.3 kB)
```

Step 2: Install Apache Airflow (Stable Version with Dependencies)

You need to add the ~/airflow_env/bin to your PATH permanently.

echo 'export PATH="$HOME/airflow_env/bin:$PATH"' >> ~/.bashrc

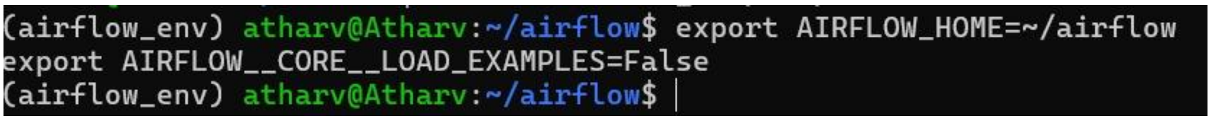
source ~/.bashrc

Step 4: Set Airflow Home Directory (Optional but Recommended)

By default, Airflow creates files in ~/airflow. You can choose a custom directory.

echo 'export AIRFLOW_HOME=~/airflow' >> ~/.bashrc

source ~/.bashrc

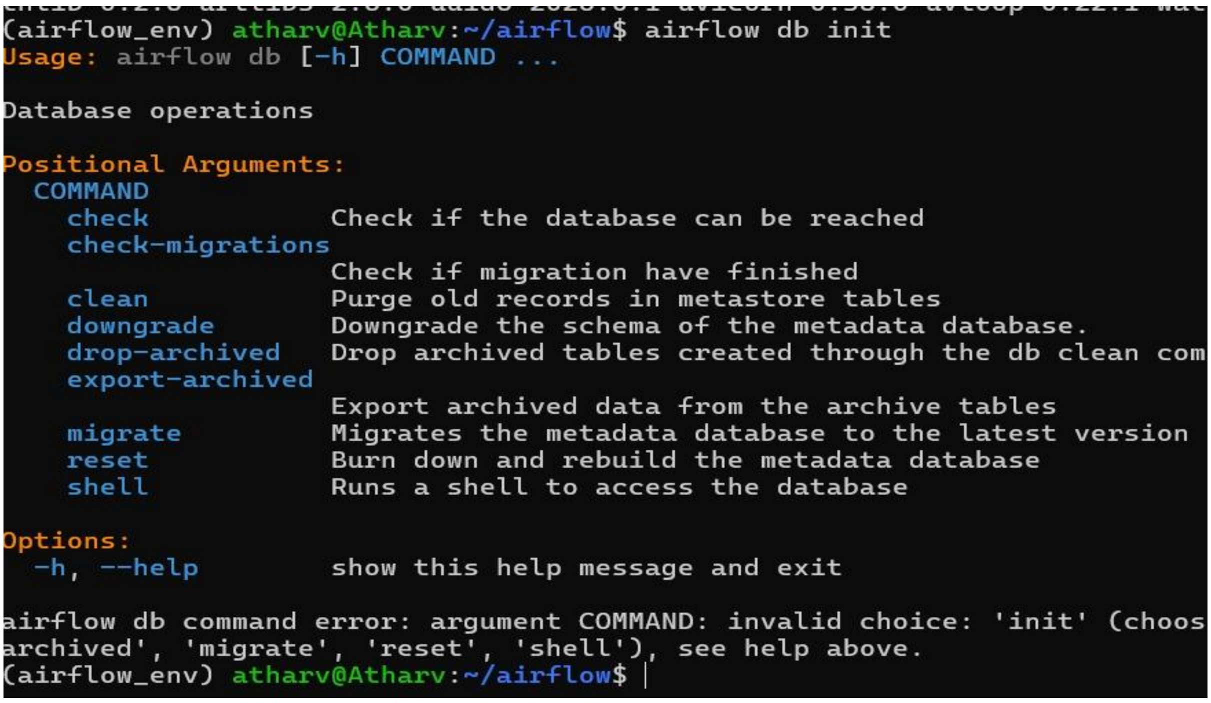


```
(airflow_env) atharv@Atharv:~/airflow$ export AIRFLOW_HOME=~/airflow
export AIRFLOW__CORE__LOAD_EXAMPLES=False
(airflow_env) atharv@Atharv:~/airflow$ 
```

Step 5: Initialize Airflow Database

airflow db init

All the commands are run in a in taken in a single screenshot

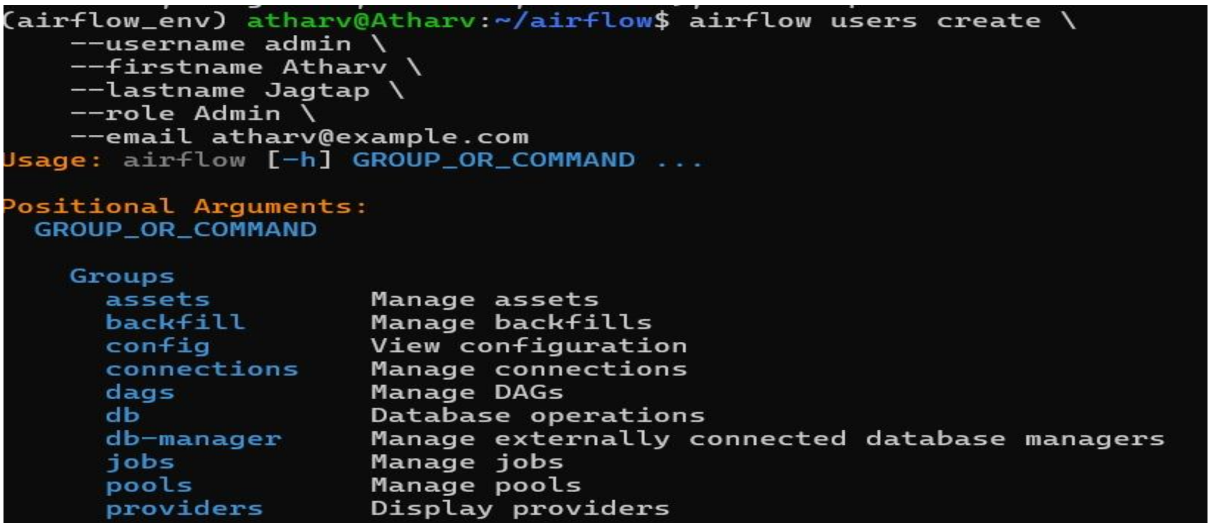


```
(airflow_env) atharv@Atharv:~/airflow$ airflow db init
Usage: airflow db [-h] COMMAND ...

Database operations

Positional Arguments:
  COMMAND
    check             Check if the database can be reached
    check-migrations
                      Check if migration have finished
    clean             Purge old records in metastore tables
    downgrade         Downgrade the schema of the metadata database.
    drop-archived     Drop archived tables created through the db clean com
    export-archived
                      Export archived data from the archive tables
    migrate           Migrates the metadata database to the latest version
    reset             Burn down and rebuild the metadata database
    shell             Runs a shell to access the database

Options:
  -h, --help          show this help message and exit

airflow db command error: argument COMMAND: invalid choice: 'init' (choos
archived', 'migrate', 'reset', 'shell'), see help above.
(airflow_env) atharv@Atharv:~/airflow$
```

Step 6: Create Airflow Admin User

```
airflow users create \
   --username admin \
   --firstname Admin \
   --lastname User \
   --role Admin \
   --email admin@example.com
```

It will ask for password → set a secure one.



```
(airflow_env) atharv@Atharv:~/airflow$ airflow users create \
    --username admin \
    --firstname Atharv \
    --lastname Jagtap \
    --role Admin \
    --email atharv@example.com
Usage: airflow [-h] GROUP_OR_COMMAND ...

Positional Arguments:
  GROUP_OR_COMMAND

    Groups
      assets          Manage assets
      backfill        Manage backfills
      config          View configuration
      connections     Manage connections
      dags            Manage DAGs
      db              Database operations
      db-manager      Manage externally connected database managers
      jobs            Manage jobs
      pools           Manage pools
      providers       Display providers
```

The above screenshot shows that the database exit because I have already runned this code first now when we run the second time it shows admin exist in the db

Step 7: Create folders for DAGs

```
mkdir -p ~/airflow/dags
```

Airflow will automatically look for DAGs in ~/airflow/dags.

Step 8 : Create a simple DAG

1.Open a text editor and create a file: nano ~/airflow/dags/etl_dag.py

2.Paste the following **example DAG code**:

```
from datetime import datetime
from airflow import DAG
from airflow.operators.python import PythonOperator
# Example ETL functions
def extract():
    print("Extracting data...")
def transform():
    print("Transforming data...")
def load():
    print("Loading data...")
# Define DAG
with DAG(
    'etl_dag',
    start_date=datetime(2025, 11, 7),
    schedule_interval='@daily',
    catchup=False,
    description='A simple ETL DAG with dependencies'
) as dag:
# Define tasks
    extract_task = PythonOperator(
        task_id='extract',
        python_callable=extract
    )
 transform_task = PythonOperator(
        task_id='transform',
```

        python_callable=transform

)

load_task = PythonOperator(

        task_id='load',

        python_callable=load

)

 # Define the task dependencies (graph)

    extract_task >> transform_task >> load_task

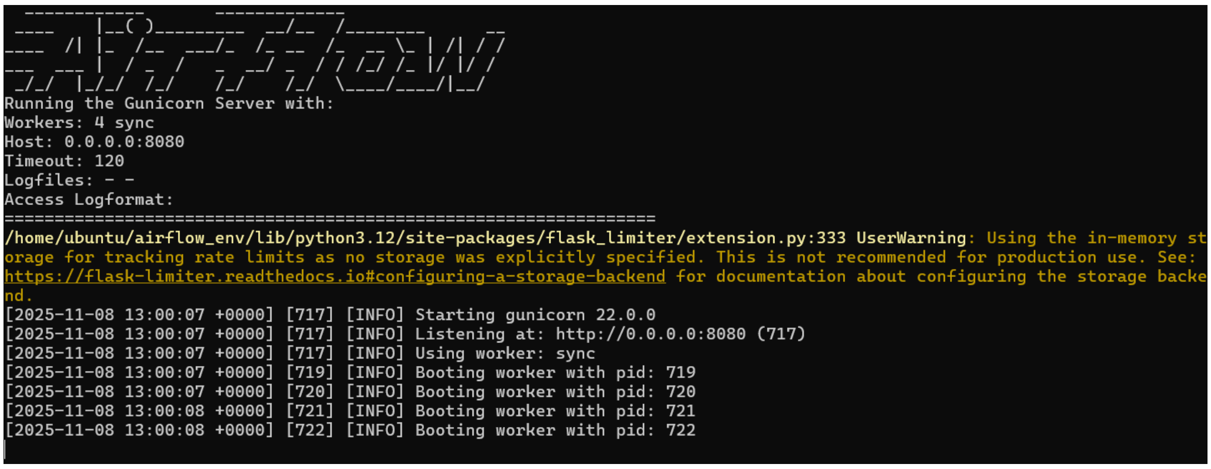
Save and exit (Ctrl+O, Enter, Ctrl+X).

Step : Start Airflow services

1. **Start the webserver**:

airflow webserver --port 8080

By default, Airflow UI will be at http://localhost:8080.

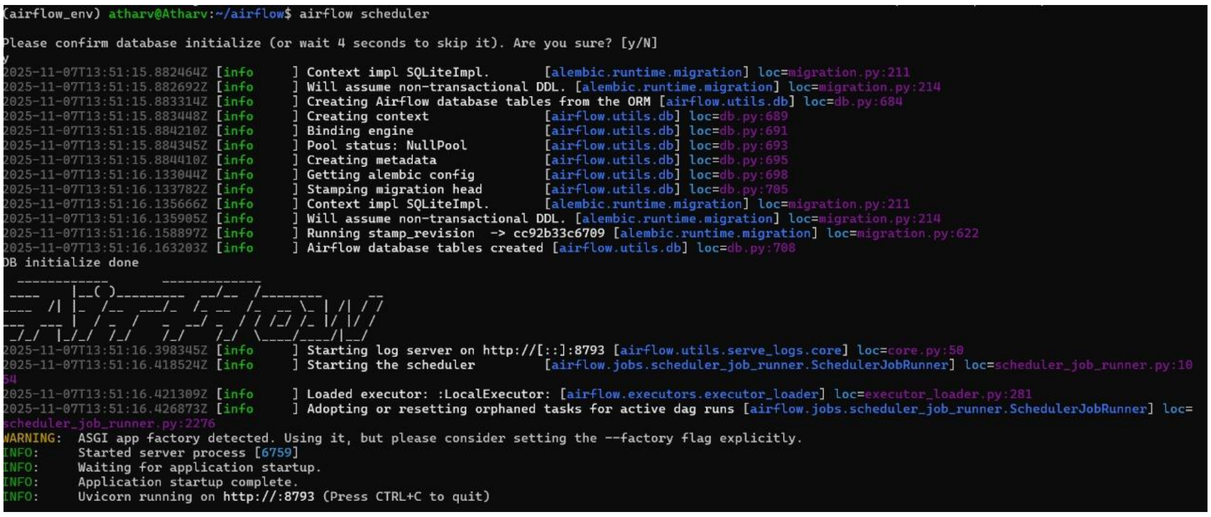


```
  ____________       _____________
 ____    |__( )_________  __/__  /________      __
____  /| |_  /__  ___/_  /_ __  /_  __ \_ | /| / /
___  ___ |  / _  /   _  __/ _  / / /_/ /_ |/ |/ /
 _/_/  |_/_/  /_/    /_/    /_/  \____/____/|__/
Running the Gunicorn Server with:
Workers: 4 sync
Host: 0.0.0.0:8080
Timeout: 120
Logfiles: - -
Access Logformat:
=================================================================
/home/ubuntu/airflow_env/lib/python3.12/site-packages/flask_limiter/extension.py:333 UserWarning: Using the in-memory st
orage for tracking rate limits as no storage was explicitly specified. This is not recommended for production use. See:
https://flask-limiter.readthedocs.io#configuring-a-storage-backend for documentation about configuring the storage backe
nd.
[2025-11-08 13:00:07 +0000] [717] [INFO] Starting gunicorn 22.0.0
[2025-11-08 13:00:07 +0000] [717] [INFO] Listening at: http://0.0.0.0:8080 (717)
[2025-11-08 13:00:07 +0000] [717] [INFO] Using worker: sync
[2025-11-08 13:00:07 +0000] [719] [INFO] Booting worker with pid: 719
[2025-11-08 13:00:07 +0000] [720] [INFO] Booting worker with pid: 720
[2025-11-08 13:00:08 +0000] [721] [INFO] Booting worker with pid: 721
[2025-11-08 13:00:08 +0000] [722] [INFO] Booting worker with pid: 722
```

2. **In another terminal**, activate the virtual environment and start the scheduler:

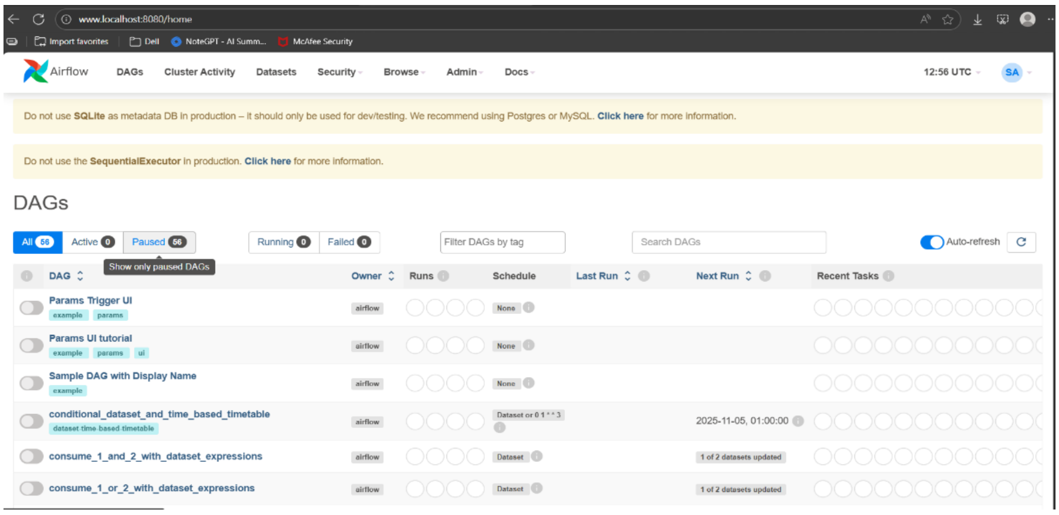
source airflow_env/bin/activate

airflow scheduler



```
(airflow_env) atharv@Atharv:~/airflow$ airflow scheduler

Please confirm database initialize (or wait 4 seconds to skip it). Are you sure? [y/N]
y
2025-11-07T13:51:15.882464Z [info     ] Context impl SQLiteImpl.       [alembic.runtime.migration] loc=migration.py:211
2025-11-07T13:51:15.882692Z [info     ] Will assume non-transactional DDL. [alembic.runtime.migration] loc=migration.py:214
2025-11-07T13:51:15.883314Z [info     ] Creating Airflow database tables from the ORM [airflow.utils.db] loc=db.py:684
2025-11-07T13:51:15.883448Z [info     ] Creating context               [airflow.utils.db] loc=db.py:689
2025-11-07T13:51:15.884210Z [info     ] Binding engine                 [airflow.utils.db] loc=db.py:691
2025-11-07T13:51:15.884345Z [info     ] Pool status: NullPool          [airflow.utils.db] loc=db.py:693
2025-11-07T13:51:15.884410Z [info     ] Creating metadata              [airflow.utils.db] loc=db.py:695
2025-11-07T13:51:16.133044Z [info     ] Getting alembic config         [airflow.utils.db] loc=db.py:698
2025-11-07T13:51:16.133782Z [info     ] Stamping migration head        [airflow.utils.db] loc=db.py:705
2025-11-07T13:51:16.135666Z [info     ] Context impl SQLiteImpl.       [alembic.runtime.migration] loc=migration.py:211
2025-11-07T13:51:16.135905Z [info     ] Will assume non-transactional DDL. [alembic.runtime.migration] loc=migration.py:214
2025-11-07T13:51:16.158897Z [info     ] Running stamp_revision  -> cc92b33c6709 [alembic.runtime.migration] loc=migration.py:622
2025-11-07T13:51:16.163203Z [info     ] Airflow database tables created [airflow.utils.db] loc=db.py:708
DB initialize done
  ____________       _____________
 ____    |__( )_________  __/__  /________      __
____  /| |_  /__  ___/_  /_ __  /_  __ \_ | /| / /
___  ___ |  / _  /   _  __/ _  / / /_/ /_ |/ |/ /
 _/_/  |_/_/  /_/    /_/    /_/  \____/____/|__/
2025-11-07T13:51:16.398345Z [info     ] Starting log server on http://[::]:8793 [airflow.utils.serve_logs.core] loc=core.py:50
2025-11-07T13:51:16.418524Z [info     ] Starting the scheduler         [airflow.jobs.scheduler_job_runner.SchedulerJobRunner] loc=scheduler_job_runner.py:10
54
2025-11-07T13:51:16.421309Z [info     ] Loaded executor: :LocalExecutor: [airflow.executors.executor_loader] loc=executor_loader.py:281
2025-11-07T13:51:16.426873Z [info     ] Adopting or resetting orphaned tasks for active dag runs [airflow.jobs.scheduler_job_runner.SchedulerJobRunner] loc=
scheduler_job_runner.py:2276
WARNING:  ASGI app factory detected. Using it, but please consider setting the --factory flag explicitly.
INFO:     Started server process [6759]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://:8793 (Press CTRL+C to quit)
```

Login in using username and password



Do not use **SQLite** as metadata DB in production – it should only be used for dev/testing. We recommend using Postgres or MySQL. **Click here** for more information.

Do not use the **SequentialExecutor** in production. **Click here** for more information.

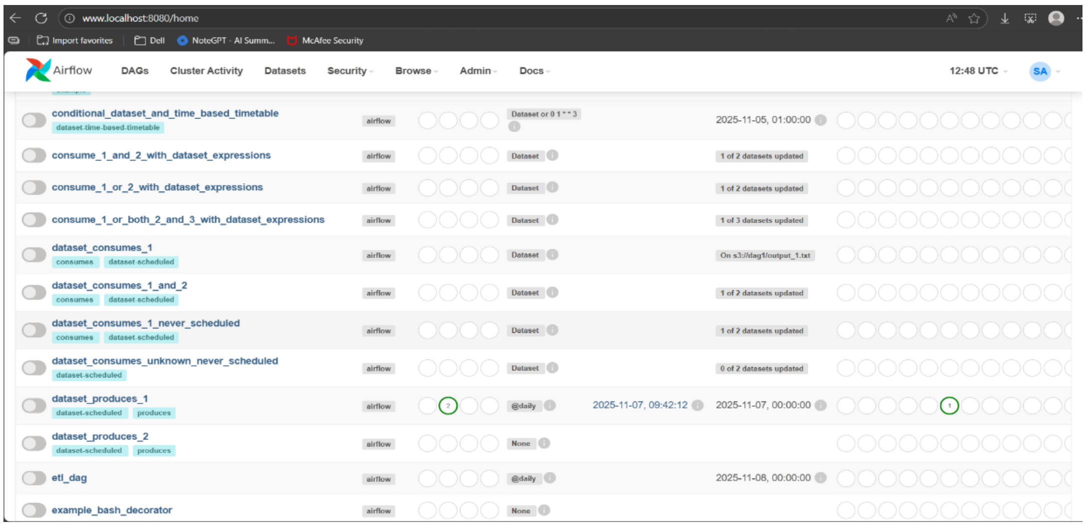
DAGs

All 56 | Active 0 | Paused 56 | Running 0 | Failed 0 | Filter DAGs by tag | Search DAGs | Auto-refresh

Show only paused DAGs

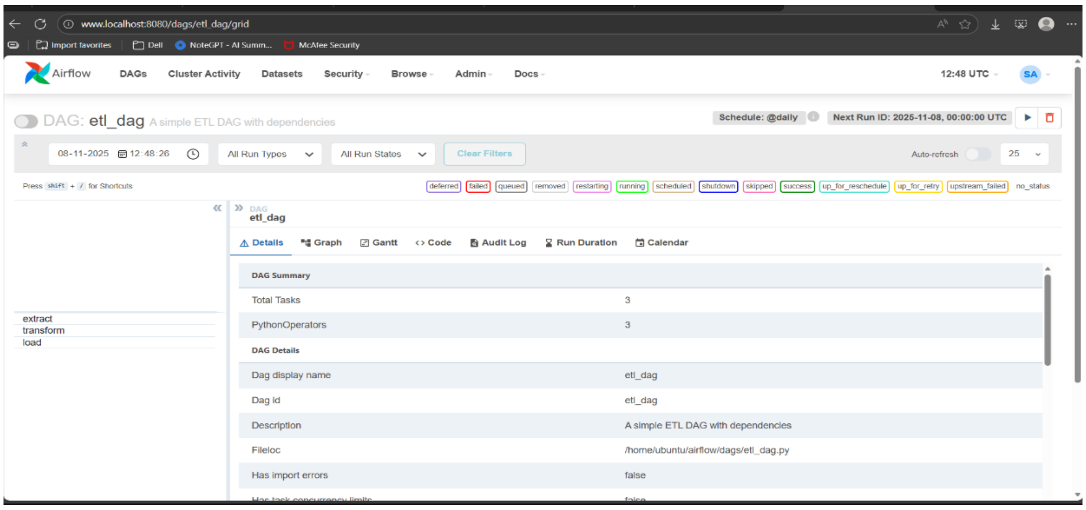
DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks
Params Trigger UI example params	airflow		None			
Params UI tutorial example params ui	airflow		None			
Sample DAG with Display Name example	airflow		None			
conditional_dataset_and_time_based_timetable dataset time based timetable	airflow		Dataset or 0 1 * * 3		2025-11-05, 01:00:00	
consume_1_and_2_with_dataset_expressions	airflow		Dataset		1 of 2 datasets updated	
consume_1_or_2_with_dataset_expressions	airflow		Dataset		1 of 2 datasets updated	

The above dag with name etl_dag is the dag which we created now using the code



DAG	Owner	Schedule	Last Run	Next Run
conditional_dataset_and_time_based_timetable (dataset-time-based-timetable)	airflow	Dataset or 0 1 * * 3		2025-11-05, 01:00:00
consume_1_and_2_with_dataset_expressions	airflow	Dataset		1 of 2 datasets updated
consume_1_or_2_with_dataset_expressions	airflow	Dataset		1 of 2 datasets updated
consume_1_or_both_2_and_3_with_dataset_expressions	airflow	Dataset		1 of 3 datasets updated
dataset_consumes_1 (consumes, dataset-scheduled)	airflow	Dataset		On s3://dag1/output_1.txt
dataset_consumes_1_and_2 (consumes, dataset-scheduled)	airflow	Dataset		1 of 2 datasets updated
dataset_consumes_1_never_scheduled (consumes, dataset-scheduled)	airflow	Dataset		1 of 2 datasets updated
dataset_consumes_unknown_never_scheduled (dataset-scheduled)	airflow	Dataset		0 of 2 datasets updated
dataset_produces_1 (dataset-scheduled, produces)	airflow	@daily	2025-11-07, 09:42:12	2025-11-07, 00:00:00
dataset_produces_2 (dataset-scheduled, produces)	airflow	None		
etl_dag	airflow	@daily		2025-11-08, 00:00:00
example_bash_decorator	airflow	None		

The below Sreenshot represents the detail of the dag

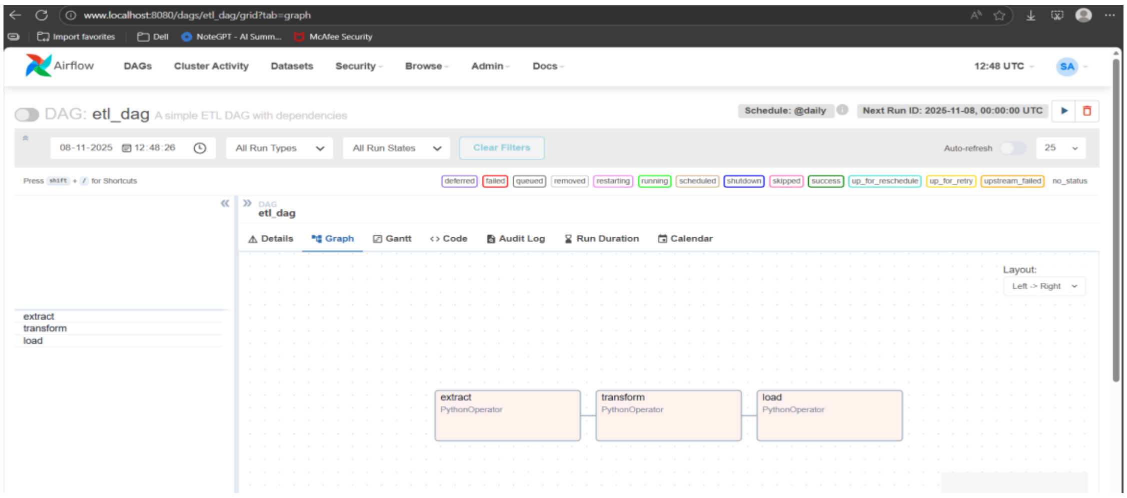


DAG: etl_dag A simple ETL DAG with dependencies

Schedule: @daily Next Run ID: 2025-11-08, 00:00:00 UTC

DAG Summary	
Total Tasks	3
PythonOperators	3
DAG Details	
Dag display name	etl_dag
Dag id	etl_dag
Description	A simple ETL DAG with dependencies
Fileloc	/home/ubuntu/airflow/dags/etl_dag.py
Has import errors	false

The below screenshot represents the graph of the dag



DAG: etl_dag A simple ETL DAG with dependencies

Schedule: @daily Next Run ID: 2025-11-08, 00:00:00 UTC