Hands-on Lab: Monitoring a DAG

Estimated time needed: **20** minutes

Introduction

In this lab, you will work with the Airflow Web UI and CLi to explore the DAGs further. You will be exposed to using the interactive tools to search for DAGs, introduces to various views of the DAGS and how you can use this to explore the DAG workflow, the individual tasks in the workflow and view the outcome of the tasks.

Objectives

After completing this lab you will be able to:

- Search for a DAG
- Pause/Unpause a DAG
- Get the Details of a DAG
- Explore grid view of a DAG
- Explore graph view of a DAG
- Explore Calendar view of a DAG
- Explore Task Duration view of a DAG
- Explore Details view of a DAG
- View the source code of a DAG
- Delete a DAG

About Skills Network Cloud IDE

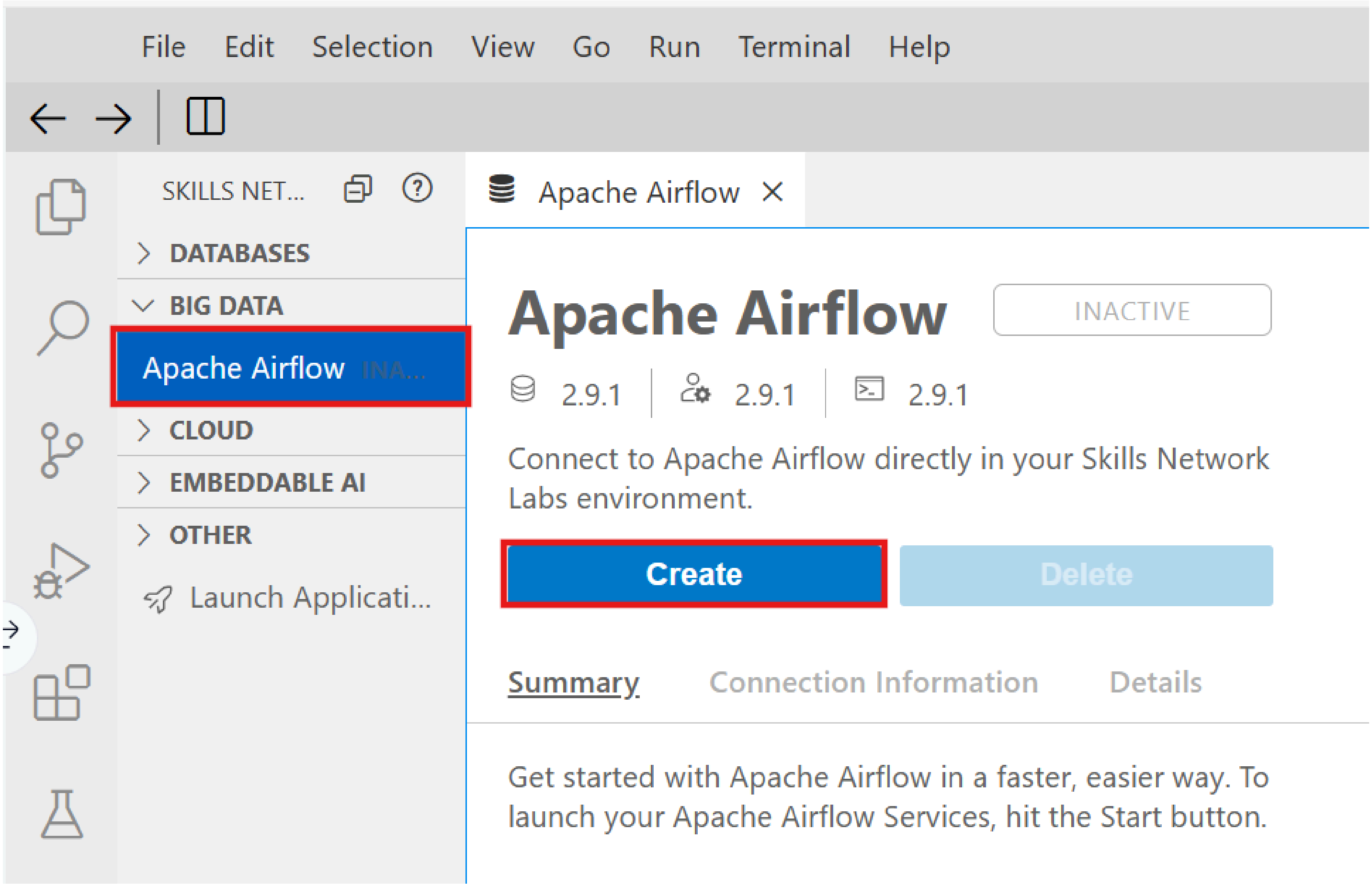
Skills Network Cloud IDE (based on Theia and Docker) provides an environment for hands on labs for course and project related labs. Theia is an open source IDE (Integrated Development Environment), that can be run on desktop or on the cloud. to complete this lab, we will be using the Cloud IDE based on Theia running in a Docker container.

Important notice about this lab environment

Please be aware that sessions for this lab environment are not persistent. A new environment is created for you every time you connect to this lab. Any data you may have saved in an earlier session will get lost. To avoid losing your data, please plan to complete these labs in a single session.

Exercise 1: Start Apache Airflow

1. Click on **Skills Network Toolbox**.
2. From the **BIG DATA** section, click **Apache Airflow**.
3. Click **Create** to start the Apache Airflow.



Apache Airflow

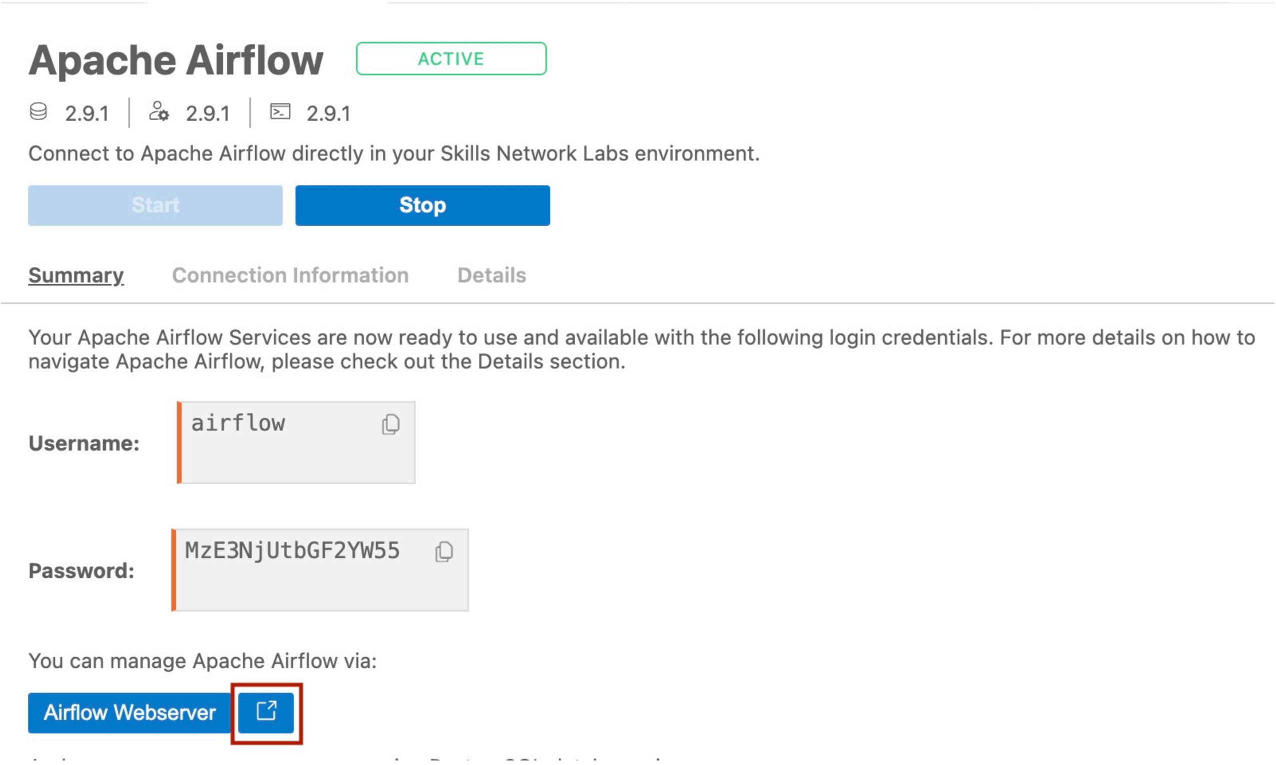
Connect to Apache Airflow directly in your Skills Network Labs environment.

Get started with Apache Airflow in a faster, easier way. To launch your Apache Airflow Services, hit the Start button.

Note: Please be patient, it will take a few minutes for Airflow to get started.

Exercise 2: Open the Airflow Web UI

When Airflow starts successfully, you should see an output similar to the one below. Once **Apache Airflow** has started, click on the highlighted icon to open **Apache Airflow Web UI** in the new window.



Connect to Apache Airflow directly in your Skills Network Labs environment.

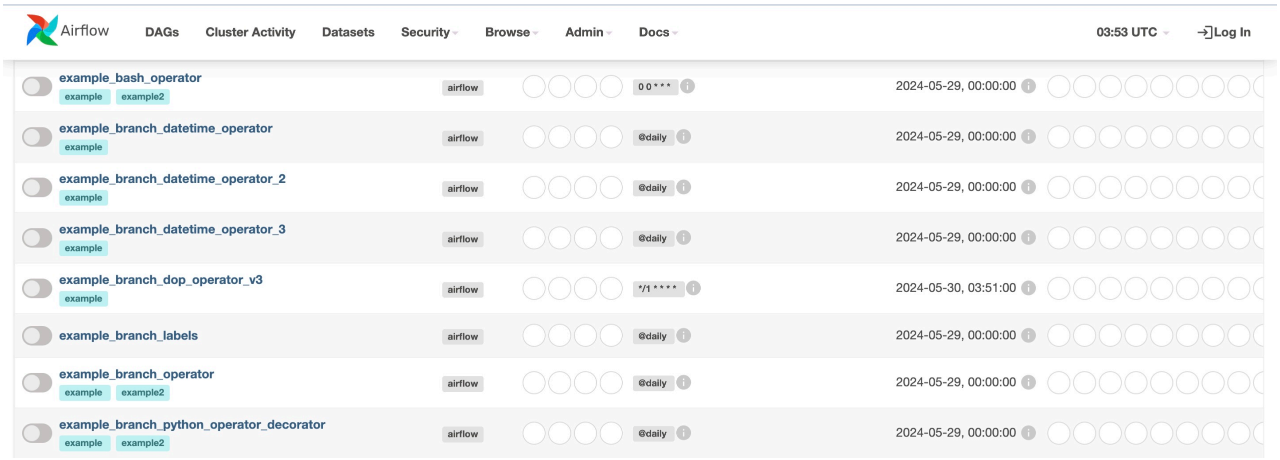
Your Apache Airflow Services are now ready to use and available with the following login credentials. For more details on how to navigate Apache Airflow, please check out the Details section.

Username: airflow

Password: MzE3NjUtbGF2YW55

You can manage Apache Airflow via:

You should land at a page that looks like this.



example_bash_operator	example example2	airflow	0 0 * * *	2024-05-29, 00:00:00
example_branch_datetime_operator	example	airflow	@daily	2024-05-29, 00:00:00
example_branch_datetime_operator_2	example	airflow	@daily	2024-05-29, 00:00:00
example_branch_datetime_operator_3	example	airflow	@daily	2024-05-29, 00:00:00
example_branch_dop_operator_v3	example	airflow	*/1 * * * *	2024-05-30, 03:51:00
example_branch_labels		airflow	@daily	2024-05-29, 00:00:00
example_branch_operator	example example2	airflow	@daily	2024-05-29, 00:00:00
example_branch_python_operator_decorator	example example2	airflow	@daily	2024-05-29, 00:00:00

Exercise 3: Submit a dummy DAG

For the purpose of monitoring, let's create a dummy DAG with three tasks.

- Task1 does nothing but sleep for 1 second.
- Task2 sleeps for 2 seconds.
- Task3 sleeps for 3 seconds.

This DAG is scheduled to run every 1 minute.

1. Using Menu->`File`->`New File` create a new file named `dummy_dag.py`.
2. Copy and paste the code below into it and save the file.

```
# import the libraries
from datetime import timedelta
# The DAG object; we'll need this to instantiate a DAG
from airflow import DAG
# Operators; we need this to write tasks!
from airflow.operators.bash_operator import BashOperator
# This makes scheduling easy
from airflow.utils.dates import days_ago
#defining DAG arguments
# You can override them on a per-task basis during operator initialization
default_args = {
    'owner': 'Your name',
    'start_date': days_ago(0),
    'email': ['your email'],
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
}
# defining the DAG
dag = DAG(
    'dummy_dag',
    default_args=default_args,
    description='My first DAG',
    schedule_interval=timedelta(minutes=1),
)
# define the tasks
# define the first task
task1 = BashOperator(
    task_id='task1',
    bash_command='sleep 1',
    dag=dag,
)
# define the second task
task2 = BashOperator(
    task_id='task2',
    bash_command='sleep 2',
    dag=dag,
)
# define the third task
task3 = BashOperator(
    task_id='task3',
    bash_command='sleep 3',
    dag=dag,
)
# task pipeline
task1 >> task2 >> task3
```

3. Set the `AIRFLOW_HOME` directory.

   ```
   export AIRFLOW_HOME=/home/project/airflow
   ```

4. Submitting a DAG is as simple as copying the DAG python file into `dags` folder in the `AIRFLOW_HOME` directory. Open a terminal and run the command below to submit the DAG.

   ```
   cp dummy_dag.py $AIRFLOW_HOME/dags
   ```

5. Verify that our DAG actually got submitted. Run the command below to list out all the existing DAGs.

   ```
   airflow dags list
   ```

6. Verify that `dummy_dag` is a part of the output.

   ```
   airflow dags list | grep dummy_dag
   ```

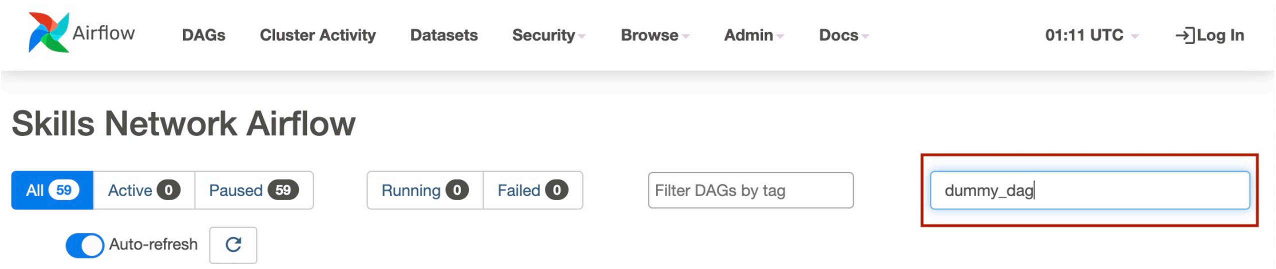
7. Run the command below to list out all the tasks in `dummy_dag`.

   ```
   airflow tasks list dummy_dag
   ```

   You should see 3 tasks in the output.

Exercise 4: Search for a DAG

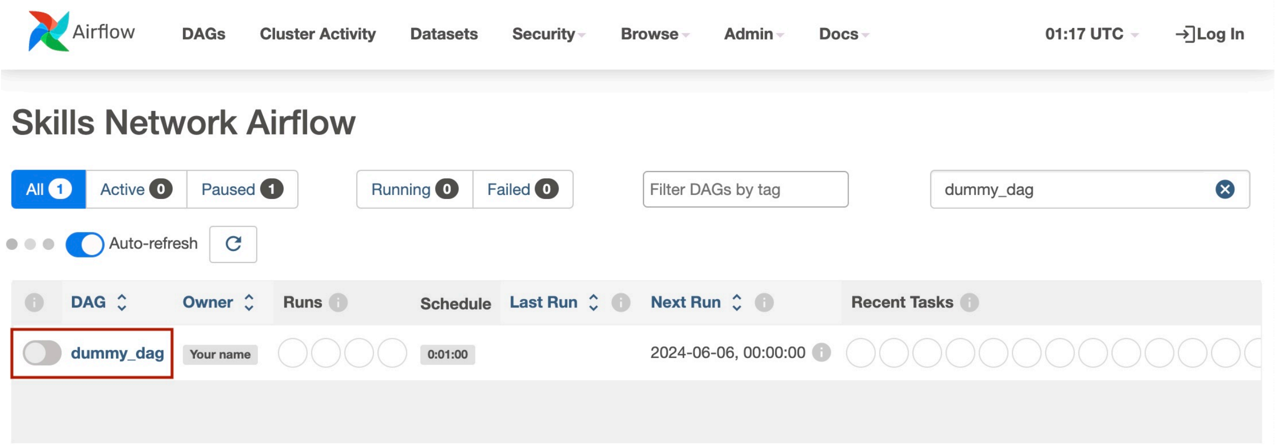
1. In the Web-UI, identify the `Search DAGs` text box as shown in the image below and type `dummy_dag` in the textbox and press enter.



dummy_dag

Note: It may take a couple of minutes for the dag to appear here. If you do not see your DAG, please give it a minute and try again.

2. You should see the `dummy_dag` listed as seen in the image below:

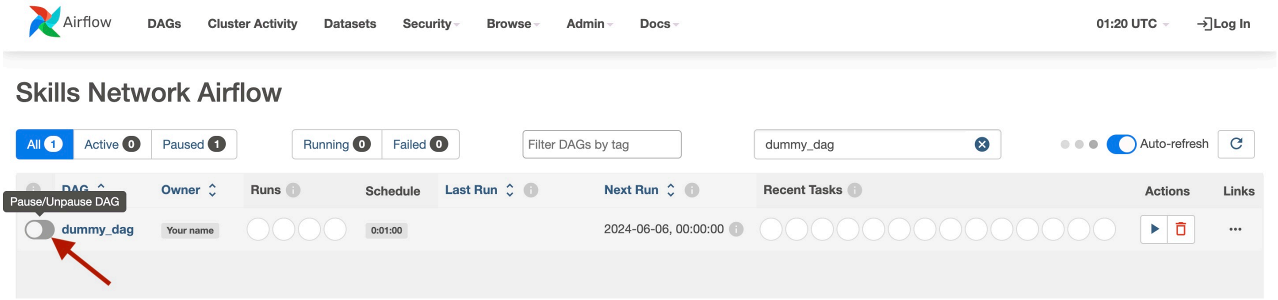


dummy_dag

DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks
dummy_dag	Your name		0:01:00		2024-06-06, 00:00:00	

Exercise 5: Pause/Unpause a DAG

1. Unpause the DAG using the Pause/Unpause button.

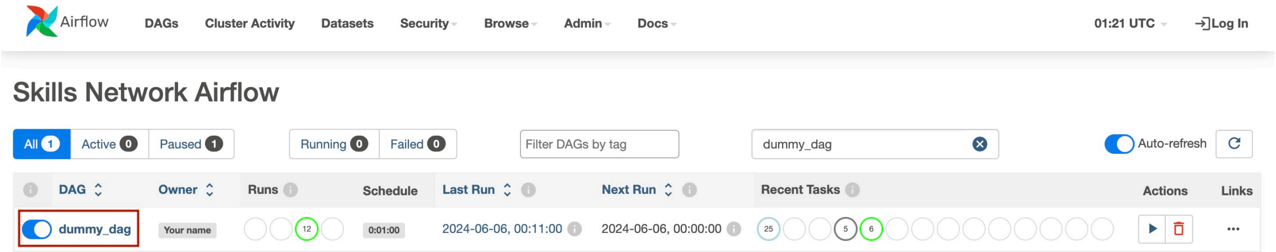


dummy_dag

DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks	Actions	Links
dummy_dag	Your name		0:01:00		2024-06-06, 00:00:00			...

2. You can see the following details in this view.

- Owner of the DAG
- How many times this DAG has run
- Schedule of the DAG
- Last run time of the DAG
- Recent task status

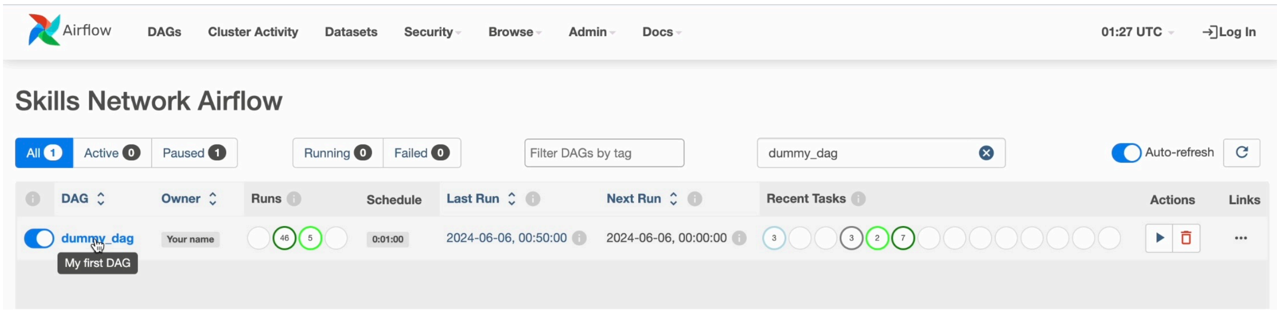


dummy_dag

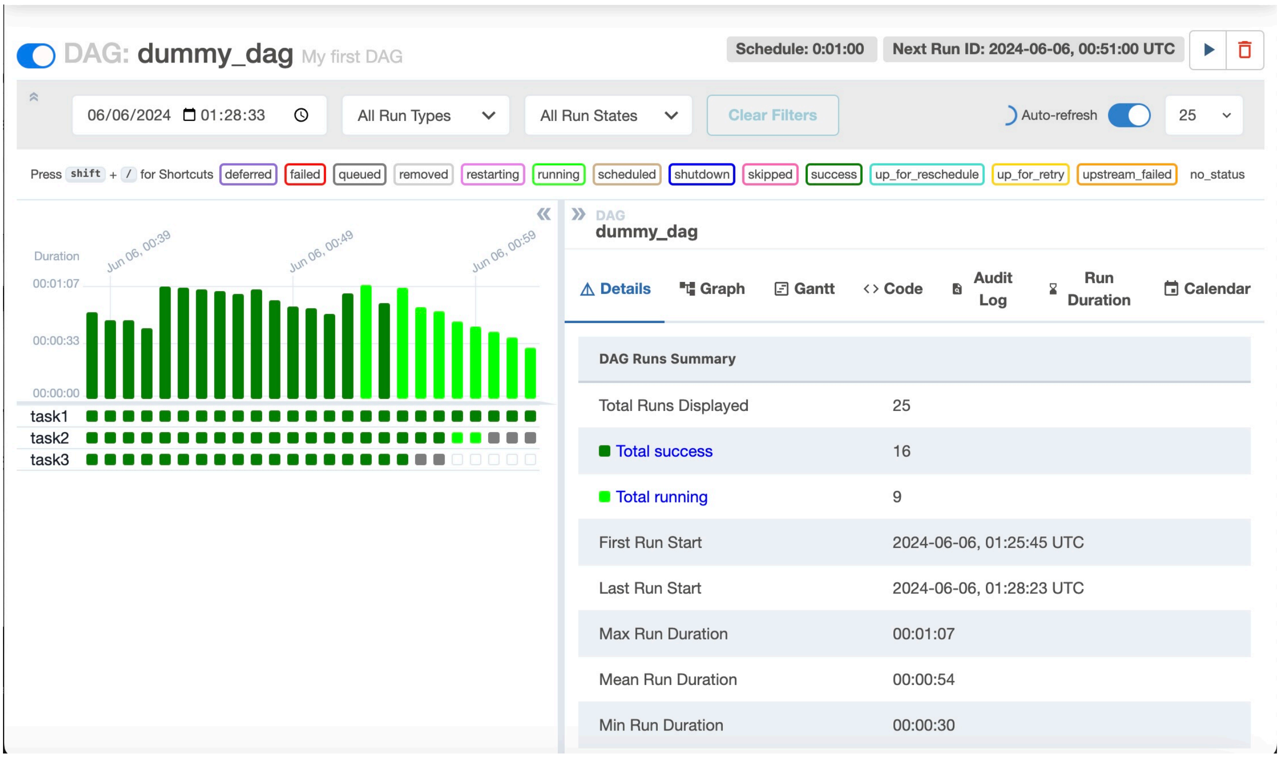
DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks	Actions	Links
dummy_dag	Your name	12	0:01:00	2024-06-06, 00:11:00	2024-06-06, 00:00:00	25 5 6		...

Exercise 6: Detailed view of a DAG

1. Click on the DAG name as shown in the image below to see the detailed view of the DAG.



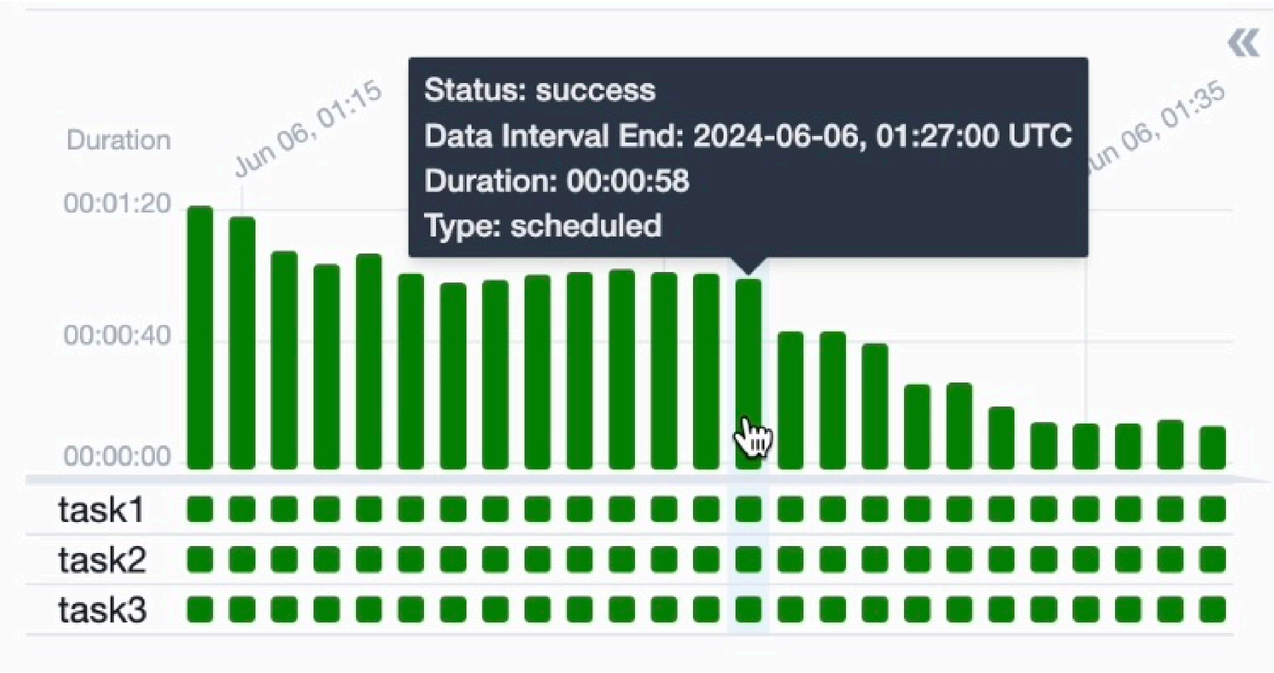
2. You will land on a DAG details page showing the default grid view with the three tasks listed.



DAG Runs Summary	
Total Runs Displayed	25
🟩 Total success	16
🟩 Total running	9
First Run Start	2024-06-06, 01:25:45 UTC
Last Run Start	2024-06-06, 01:28:23 UTC
Max Run Duration	00:01:07
Mean Run Duration	00:00:54
Min Run Duration	00:00:30

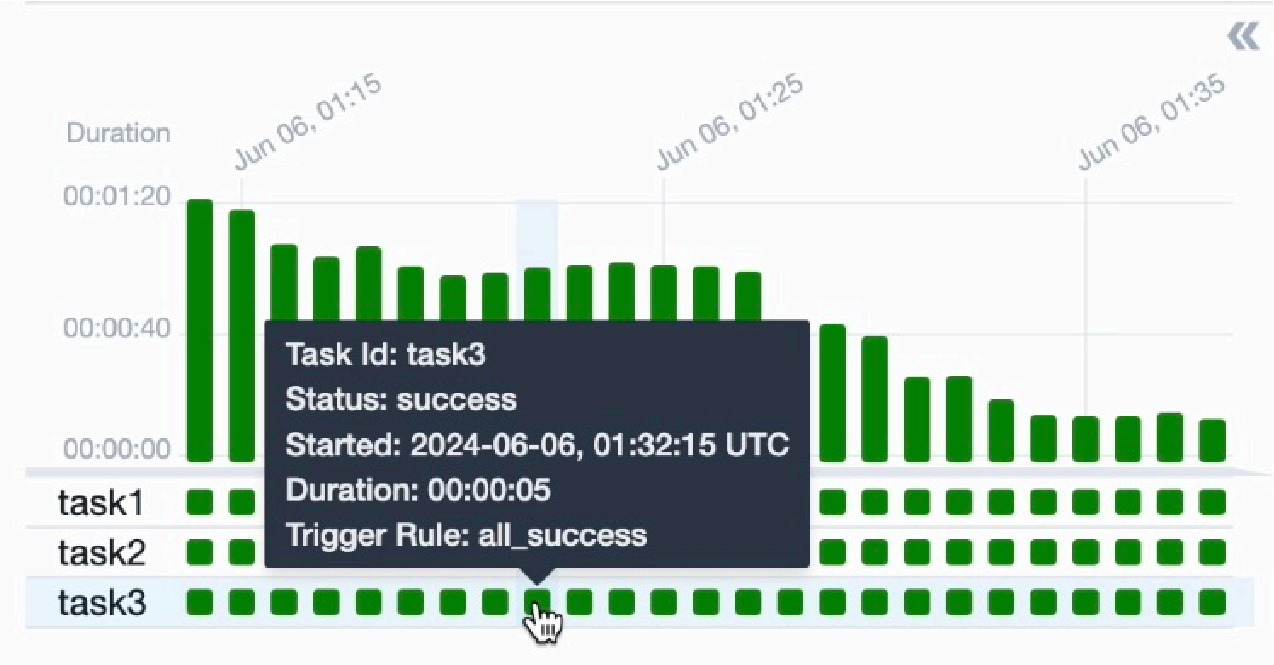
The Grid view shows your DAG tasks in the form of grids as seen in the image. You will observe the `Auto Refresh` button switched on by default on the right corner.

The grids in the image represent a single DAG run and the color indicates the status of the DAG run. Place your mouse on any grid to see the details.



Status: success
Data Interval End: 2024-06-06, 01:27:00 UTC
Duration: 00:00:58
Type: scheduled

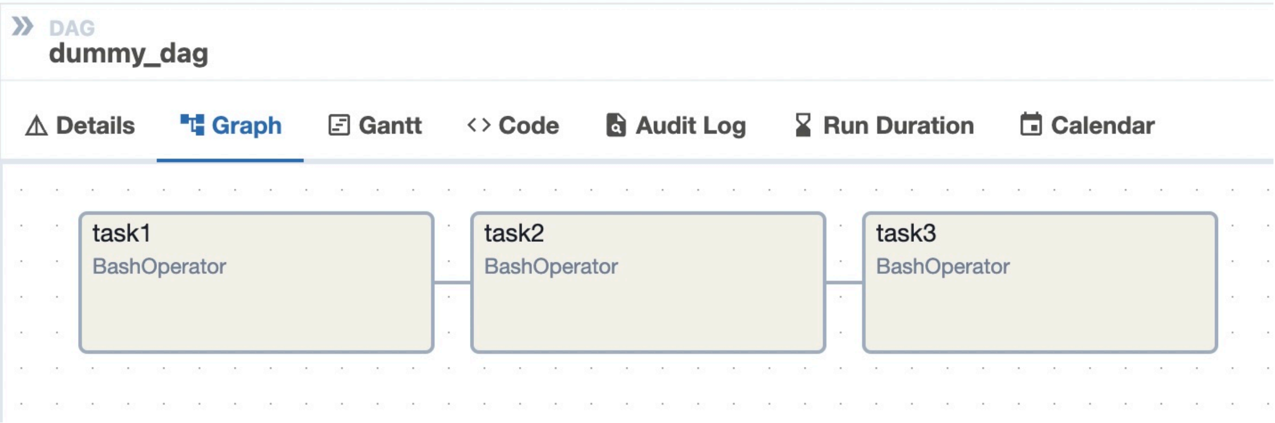
The squares in the image below represent a single task within a DAG run and the color indicates its status. Place your mouse on any square to see the task details.



Task Id: task3
Status: success
Started: 2024-06-06, 01:32:15 UTC
Duration: 00:00:05
Trigger Rule: all_success

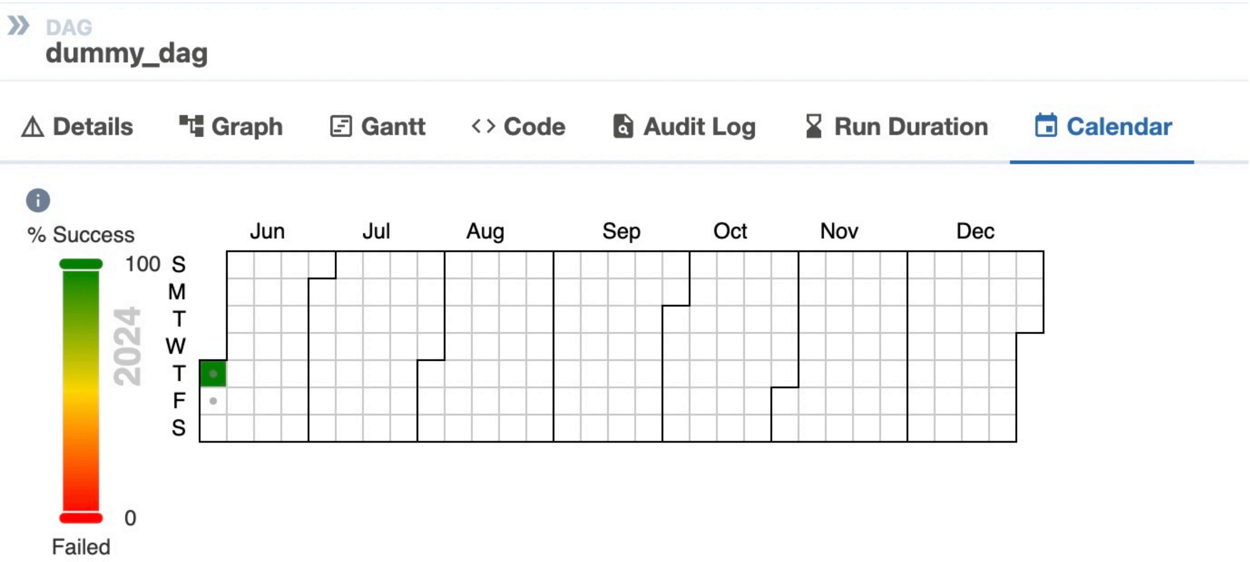
Exercise 7: Explore graph view of DAG

1. Click on the `Graph View` button to open the graph view. The graph view shows the tasks in a form of a graph. With the auto refresh on, each task status is also indicated with the color code.



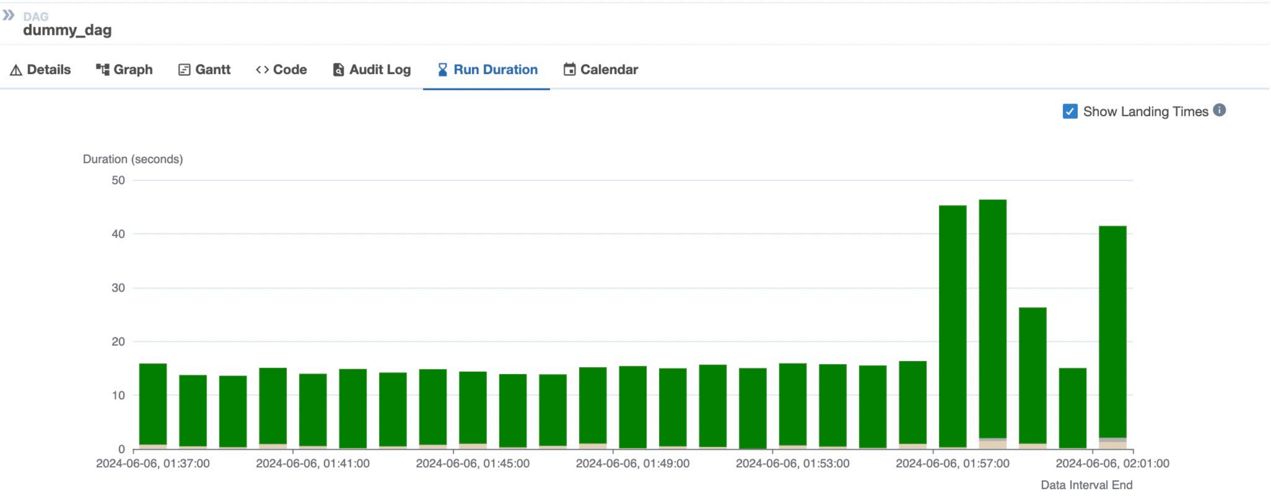
Exercise 8: Calender view

The calender view gives you an overview of all the dates when this DAG was run along with its status as a color code.

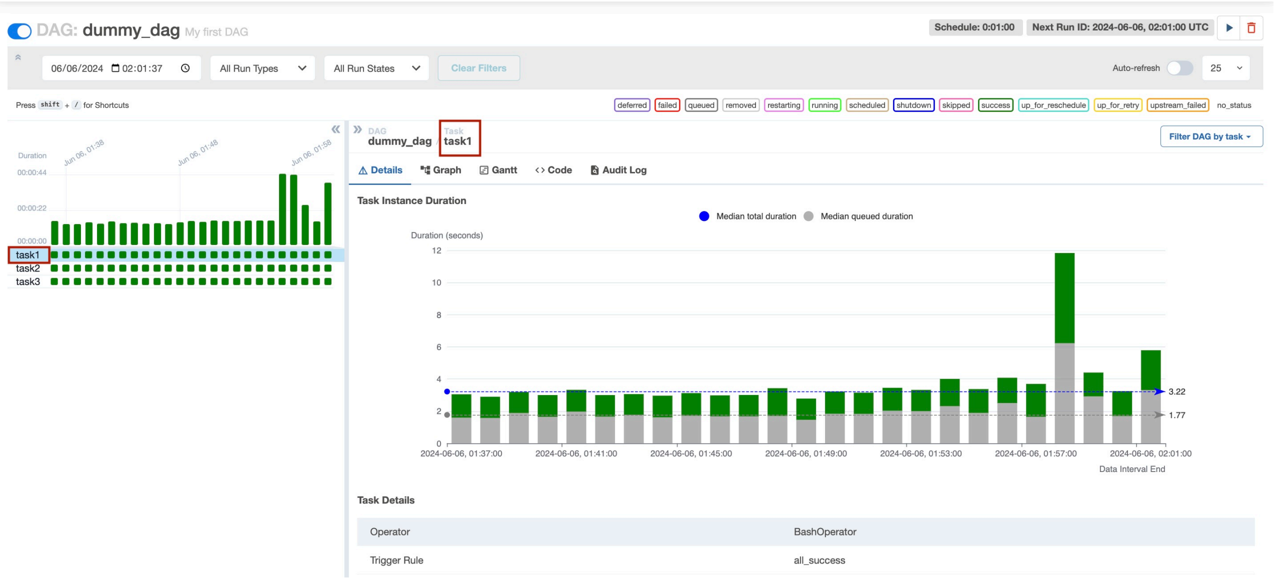


Exercise 9: DAG and Task Duration view

The DAG duration gives you an overview of how much time the entire workflow took.



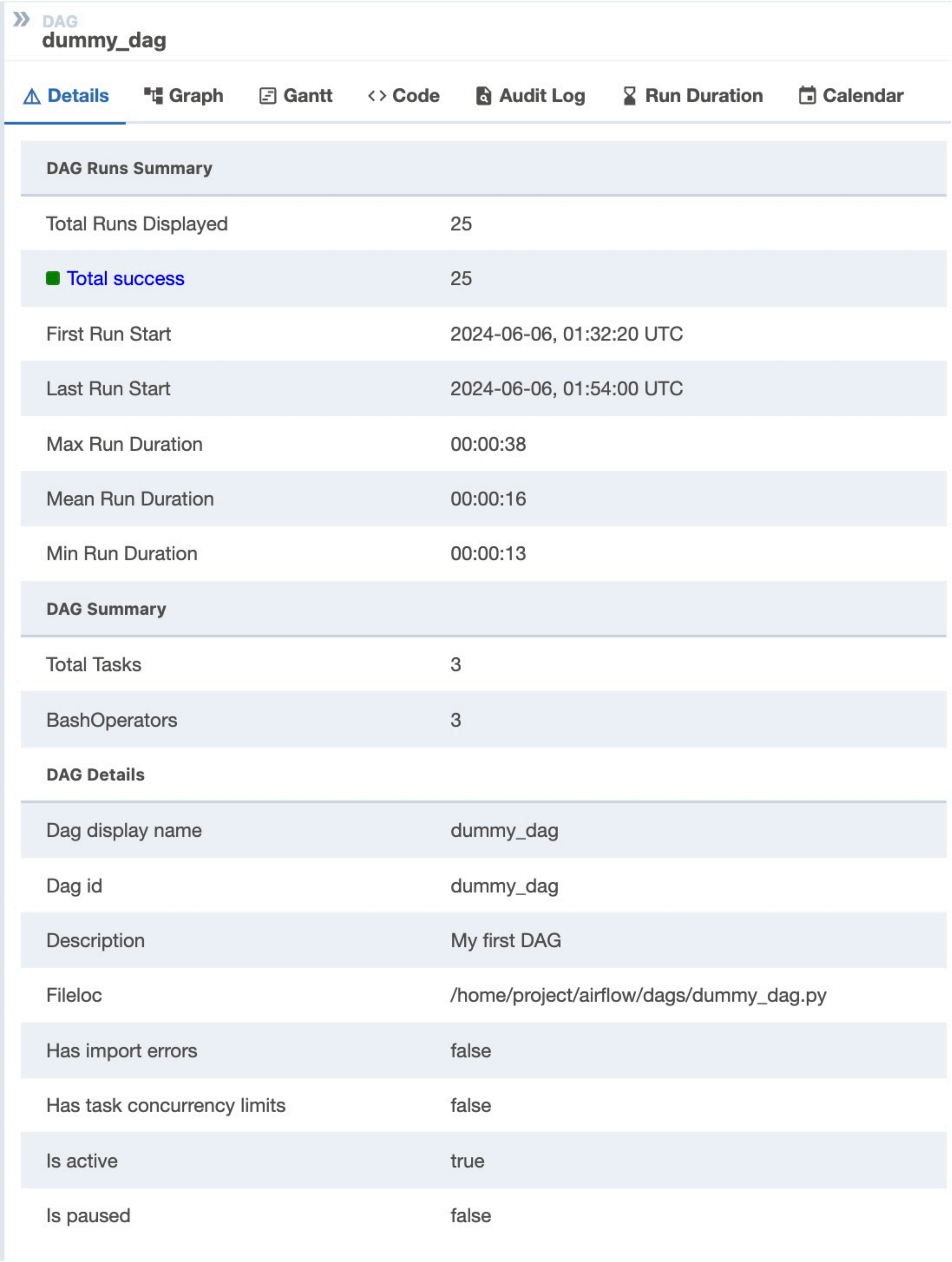
The Task Duration view gives you an overview of how much time each task took to execute, over a period of time.



Operator	BashOperator
Trigger Rule	all_success

Exercise 10: Details view

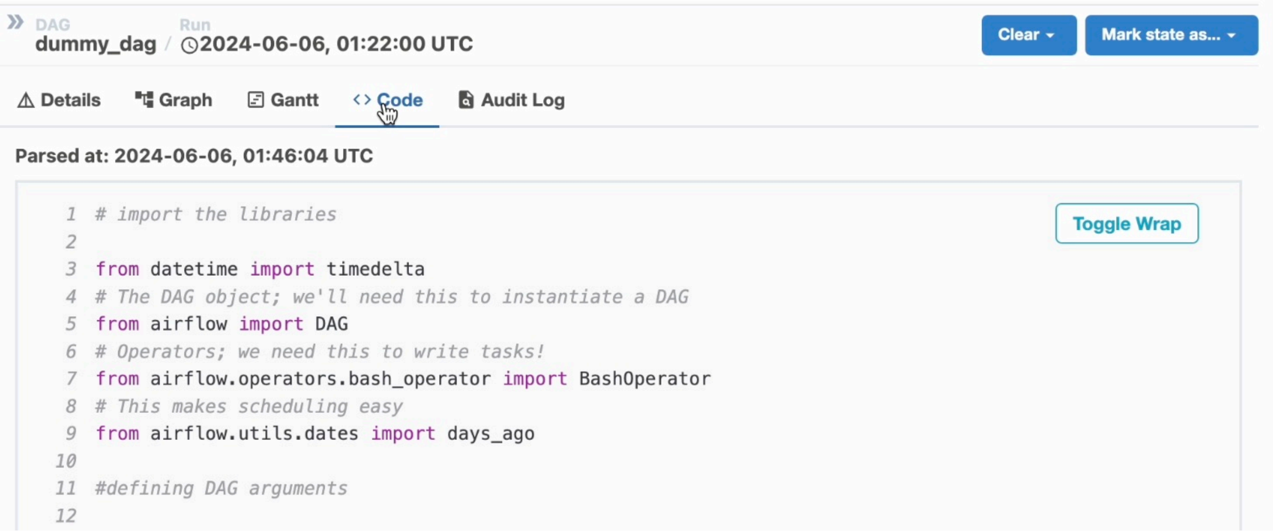
The Details view give you all the details of the DAG as specified in the code of the DAG.



DAG Runs Summary	
Total Runs Displayed	25
🟩 Total success	25
First Run Start	2024-06-06, 01:32:20 UTC
Last Run Start	2024-06-06, 01:54:00 UTC
Max Run Duration	00:00:38
Mean Run Duration	00:00:16
Min Run Duration	00:00:13
DAG Summary	
Total Tasks	3
BashOperators	3
DAG Details	
Dag display name	dummy_dag
Dag id	dummy_dag
Description	My first DAG
Fileloc	/home/project/airflow/dags/dummy_dag.py
Has import errors	false
Has task concurrency limits	false
Is active	true
Is paused	false

Exercise 11: Code view

The Code view lets you view the code of the DAG.

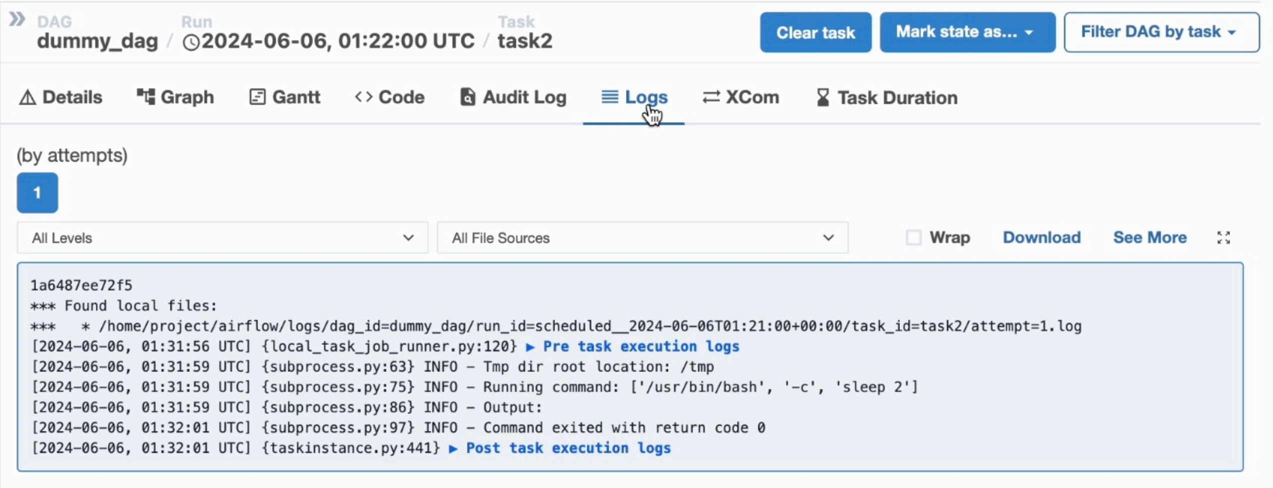


Parsed at: 2024-06-06, 01:46:04 UTC

```
# import the libraries

from datetime import timedelta
# The DAG object; we'll need this to instantiate a DAG
from airflow import DAG
# Operators; we need this to write tasks!
from airflow.operators.bash_operator import BashOperator
# This makes scheduling easy
from airflow.utils.dates import days_ago

#defining DAG arguments

```

Exercise 12: Task logs

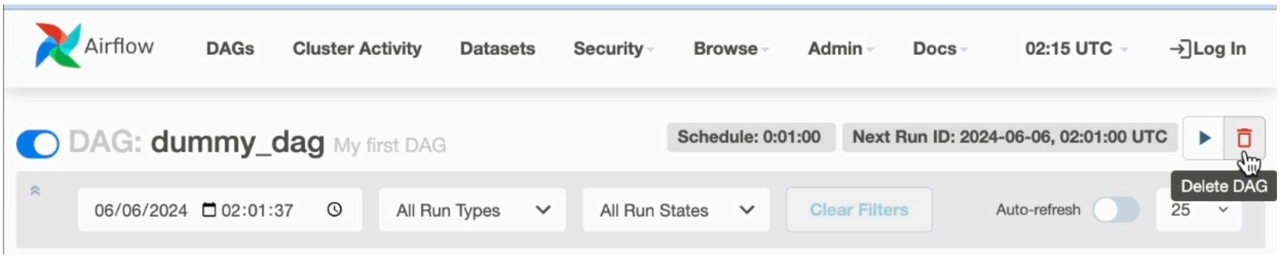
You can view the logs of an individual task with task logs.



```
1a6487ee72f5
*** Found local files:
***   * /home/project/airflow/logs/dag_id=dummy_dag/run_id=scheduled__2024-06-06T01:21:00+00:00/task_id=task2/attempt=1.log
[2024-06-06, 01:31:56 UTC] {local_task_job_runner.py:120} ▶ Pre task execution logs
[2024-06-06, 01:31:59 UTC] {subprocess.py:63} INFO - Tmp dir root location: /tmp
[2024-06-06, 01:31:59 UTC] {subprocess.py:75} INFO - Running command: ['/usr/bin/bash', '-c', 'sleep 2']
[2024-06-06, 01:31:59 UTC] {subprocess.py:86} INFO - Output:
[2024-06-06, 01:32:01 UTC] {subprocess.py:97} INFO - Command exited with return code 0
[2024-06-06, 01:32:01 UTC] {taskinstance.py:441} ▶ Post task execution logs
```

Exercise 13: Delete a DAG

To delete a DAG click on the delete button.



You will get a confirmation pop up as shown in the image below. Click OK to delete the DAG.

Practice exercises

1. Unpause any existing DAG and monitor it.
2. View the details on any existing DAG. View the code of the DAG. Delve into the task details and view the logs of each task.

Authors

Lavanya T S
Ramesh Sannareddy

Other Contributors

Rav Ahuja