

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

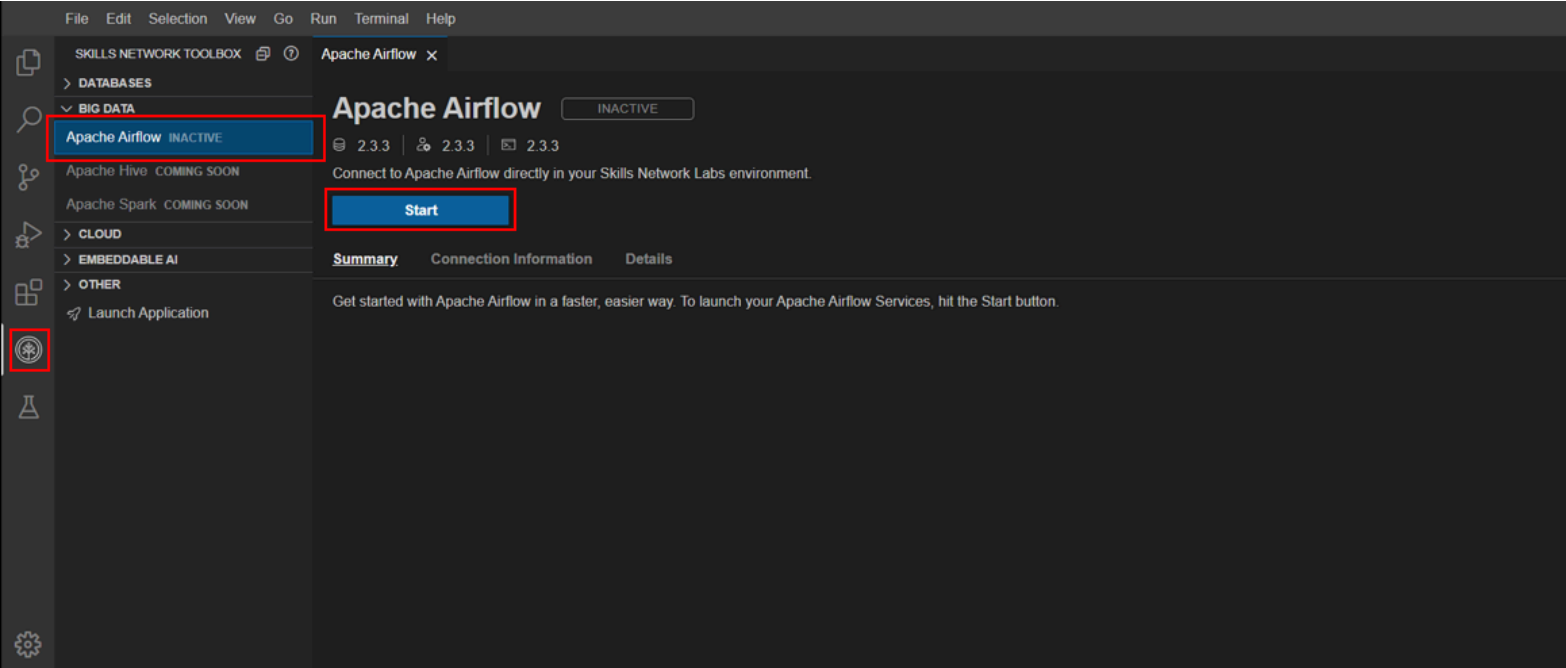
Skills Network Cloud IDE (based on Theia and Docker) provides an environment for hands on labs for course and project related labs. Theia is an open source IDE (Integrated Development Environment), that can be run on desktop or on the cloud. to complete this lab, we will be using the Cloud IDE based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

Important notice about this lab environment

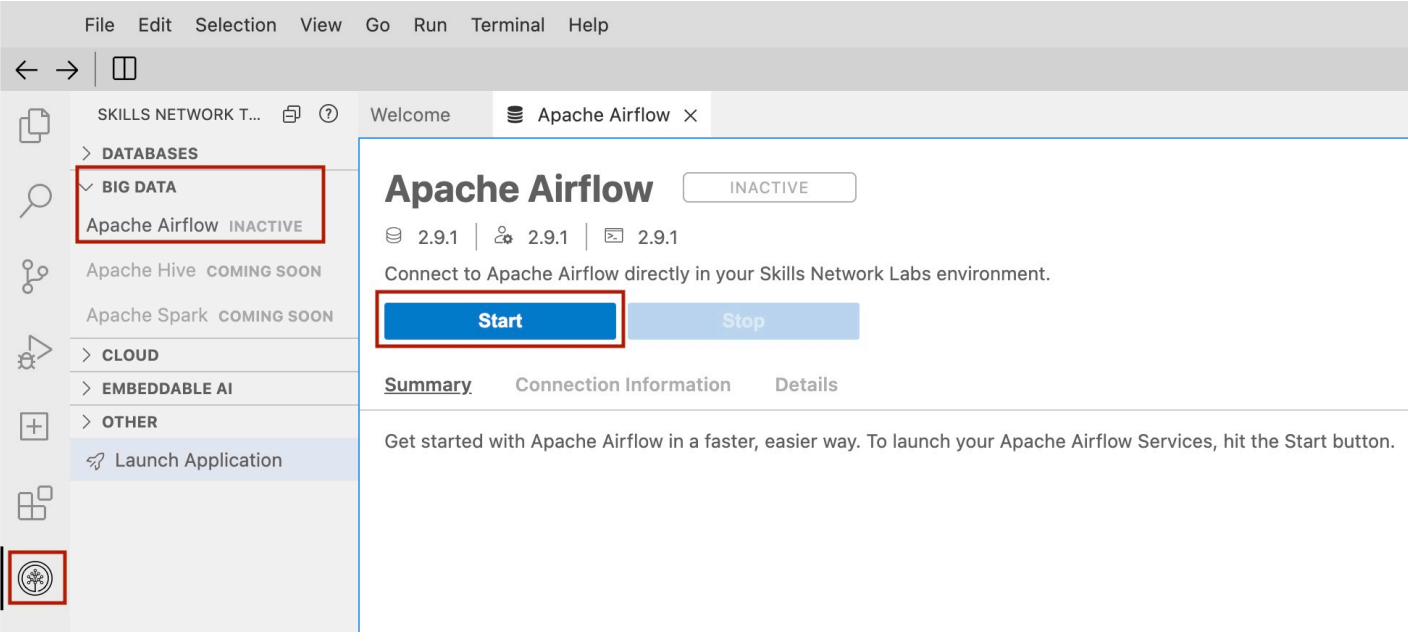
Please be aware that sessions for this lab environment are not persistent. A new environment is created for you every time you connect to this lab. Any data you may have saved in an earlier session will get lost. To avoid losing your data, please plan to complete these labs in a single session.

Exercise 1: Start Apache Airflow

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

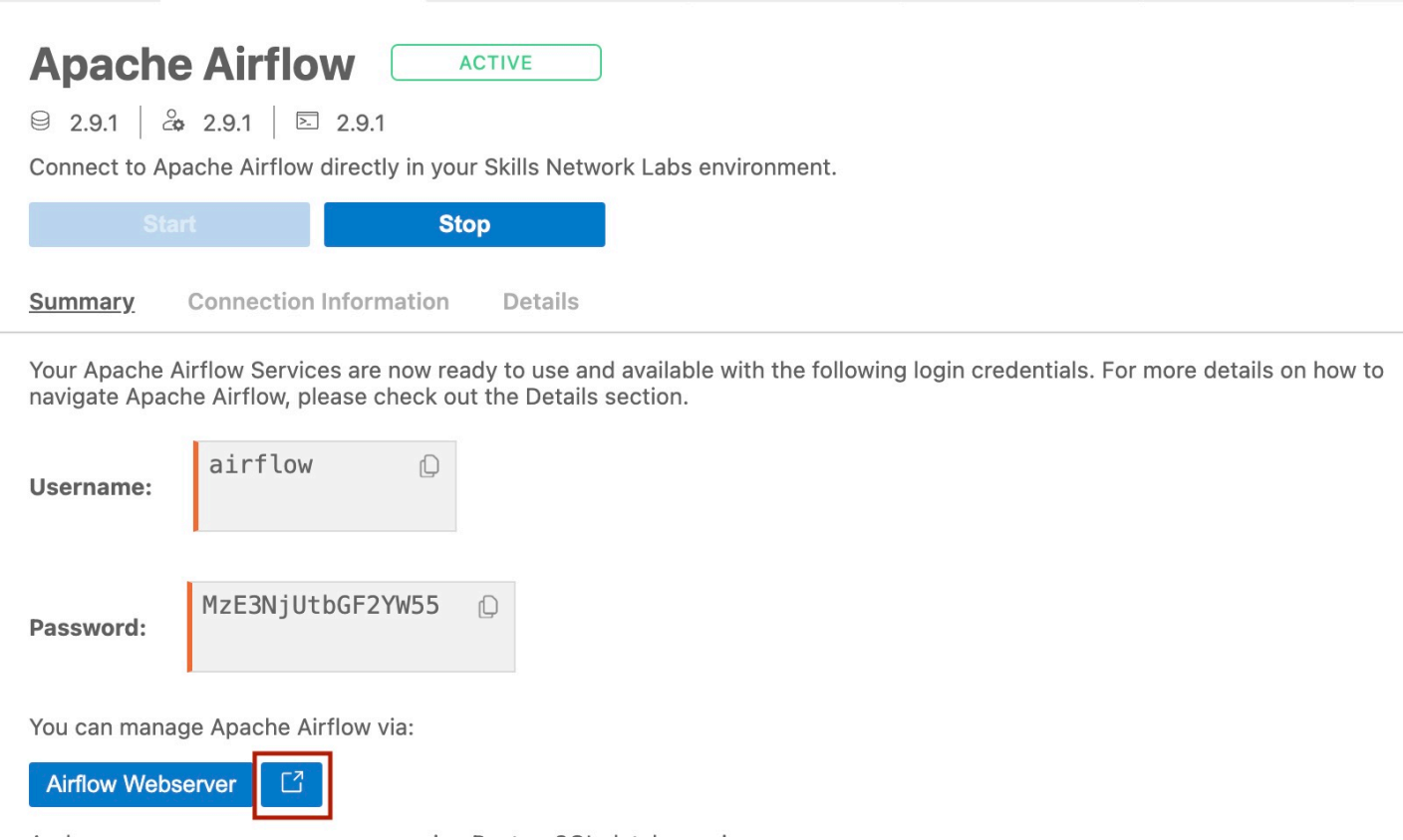


Note: Please be patient, it will take a few minutes for Airflow to get started.



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.



You should land at a page that looks like this.

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

```
1. 1
2. 2
3. 3
4. 4
5. 5
6. 6
7. 7
8. 8
9. 9
10. 10
11. 11
12. 12
13. 13
14. 14
15. 15
16. 16
17. 17
18. 18
19. 19
20. 20
21. 21
22. 22
23. 23
24. 24
25. 25
26. 26
27. 27
28. 28
29. 29
30. 30
31. 31
32. 32
33. 33
34. 34
35. 35
36. 36
37. 37
38. 38
39. 39
40. 40
41. 41
42. 42
43. 43
44. 44
45. 45
46. 46
47. 47
48. 48
49. 49
50. 50
51. 51
52. 52
53. 53
54. 54
55. 55

1. # import the libraries
2.
```

```

3. from datetime import timedelta
4. # The DAG object; we'll need this to instantiate a DAG
5. from airflow import DAG
6. # Operators; we need this to write tasks!
7. from airflow.operators.bash_operator import BashOperator
8. # This makes scheduling easy
9. from airflow.utils.dates import days_ago
10.
11. #defining DAG arguments
12.
13. # You can override them on a per-task basis during operator initialization
14. default_args = {
15.     'owner': 'Your name',
16.     'start_date': days_ago(0),
17.     'email': ['your email'],
18.     'retries': 1,
19.     'retry_delay': timedelta(minutes=5),
20. }
21.
22. # defining the DAG
23. dag = DAG(
24.     'dummy_dag',
25.     default_args=default_args,
26.     description='My first DAG',
27.     schedule_interval=timedelta(minutes=1),
28. )
29.
30. # define the tasks
31.
32. # define the first task
33.
34. task1 = BashOperator(
35.     task_id='task1',
36.     bash_command='sleep 1',
37.     dag=dag,
38. )
39.
40. # define the second task
41. task2 = BashOperator(
42.     task_id='task2',
43.     bash_command='sleep 2',
44.     dag=dag,
45. )
46.
47. # define the third task
48. task3 = BashOperator(
49.     task_id='task3',
50.     bash_command='sleep 3',
51.     dag=dag,
52. )
53.
54. # task pipeline
55. task1 >> task2 >> task3

```

Copied!

3. Set the AIRFLOW_HOME directory.

```

1. 1
1. export AIRFLOW_HOME=/home/project/airflow

```

Copied! Executed!

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.

```

1. 1
1. cp dummy_dag.py $AIRFLOW_HOME/dags

```

Copied!

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

```

1. 1
1. airflow dags list

```

Copied!

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

```

1. 1
1. airflow dags list | grep dummy_dag

```

Copied!

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

```

1. 1
1. airflow tasks list dummy_dag

```

Copied!

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.

Skills Network Airflow

All **59** Active **0** Paused **59**

Running **0** Failed **0**

Filter DAGs by tag

☒ Auto-refresh

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:

Skills Network Airflow

All **1** Active **0** Paused **1**

Running **0** Failed **0**

Filter DAGs by tag

☒ Auto-refresh

DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks
<input checked="" type="checkbox"/> dummy_dag	Your name	<div><div></div><div></div><div></div><div></div></div>	0:01:00	2024-06-06, 00:00:00		<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>

Exercise 5: Pause/Unpause a DAG

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

Skills Network Airflow

All **1** Active **0** Paused **1**

Running **0** Failed **0**

Filter DAGs by tag

☒ Auto-refresh

DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks	Actions	Links
<input checked="" type="checkbox"/> dummy_dag	Your name	<div><div></div><div></div><div></div><div></div></div>	0:01:00	2024-06-06, 00:00:00		<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>		...

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

Skills Network Airflow

All **1** Active **0** Paused **1**

Running **0** Failed **0**

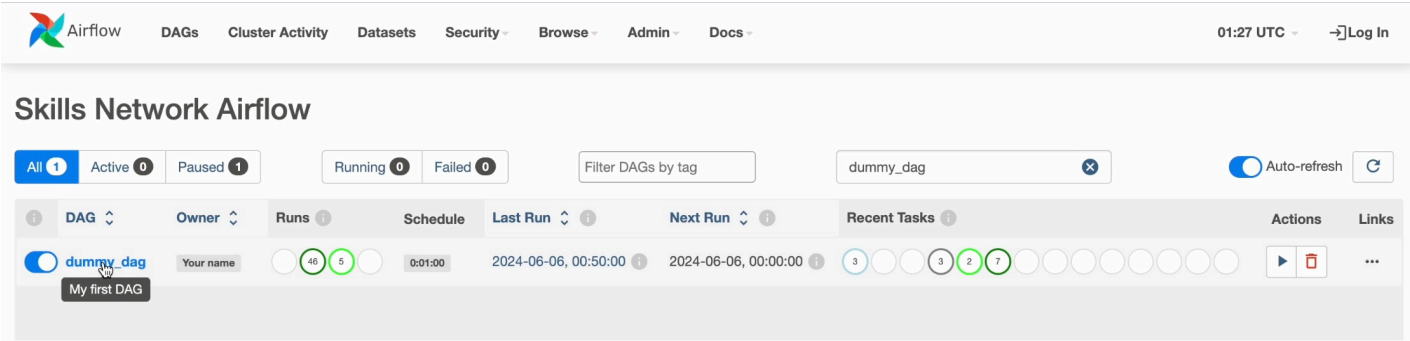
Filter DAGs by tag

☒ Auto-refresh

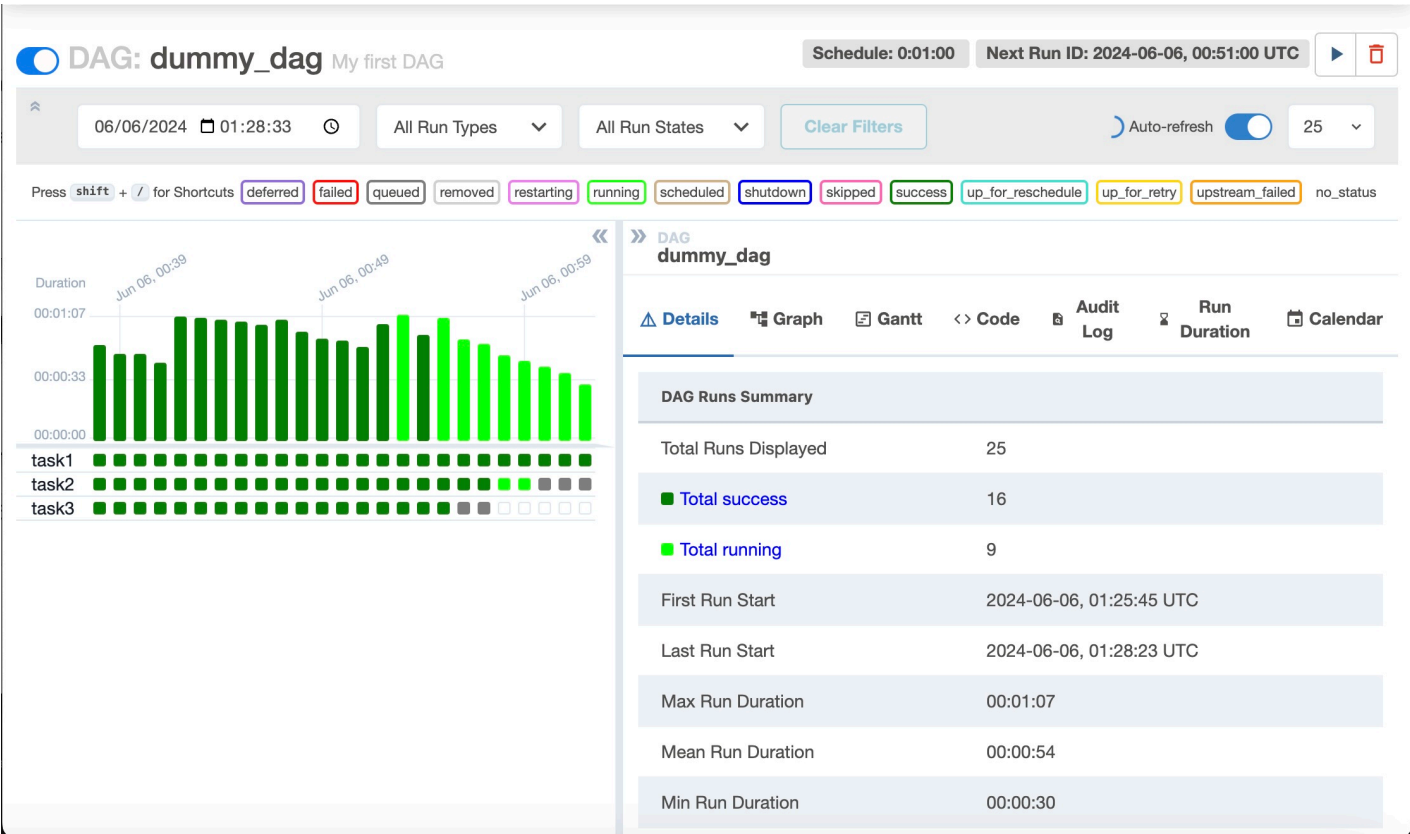
DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks	Actions	Links
<input checked="" type="checkbox"/> dummy_dag	Your name	<div><div></div><div></div><div></div><div></div><div></div></div>	0:01:00	2024-06-06, 00:11:00	2024-06-06, 00:00:00	<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>		...

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.

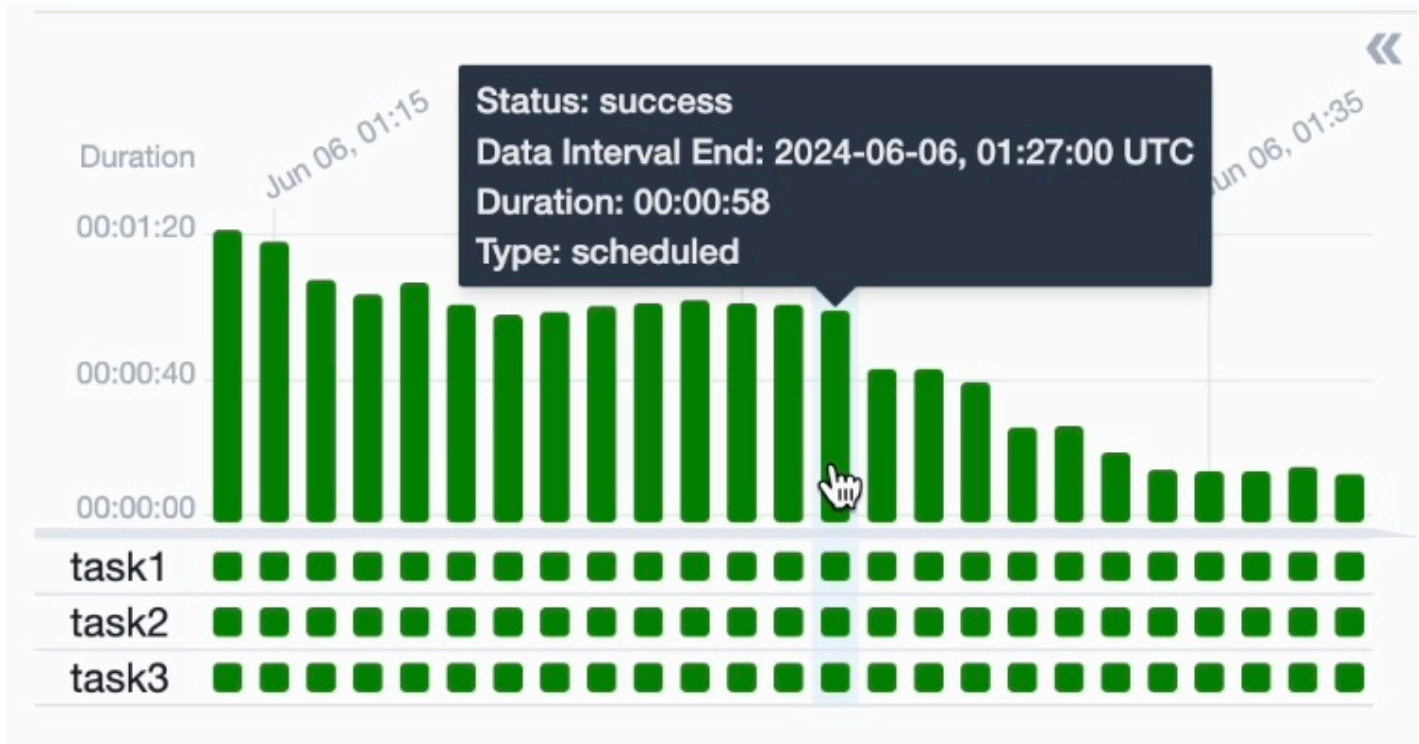


2. You will land on a DAG details page showing the default grid view with the three tasks listed.

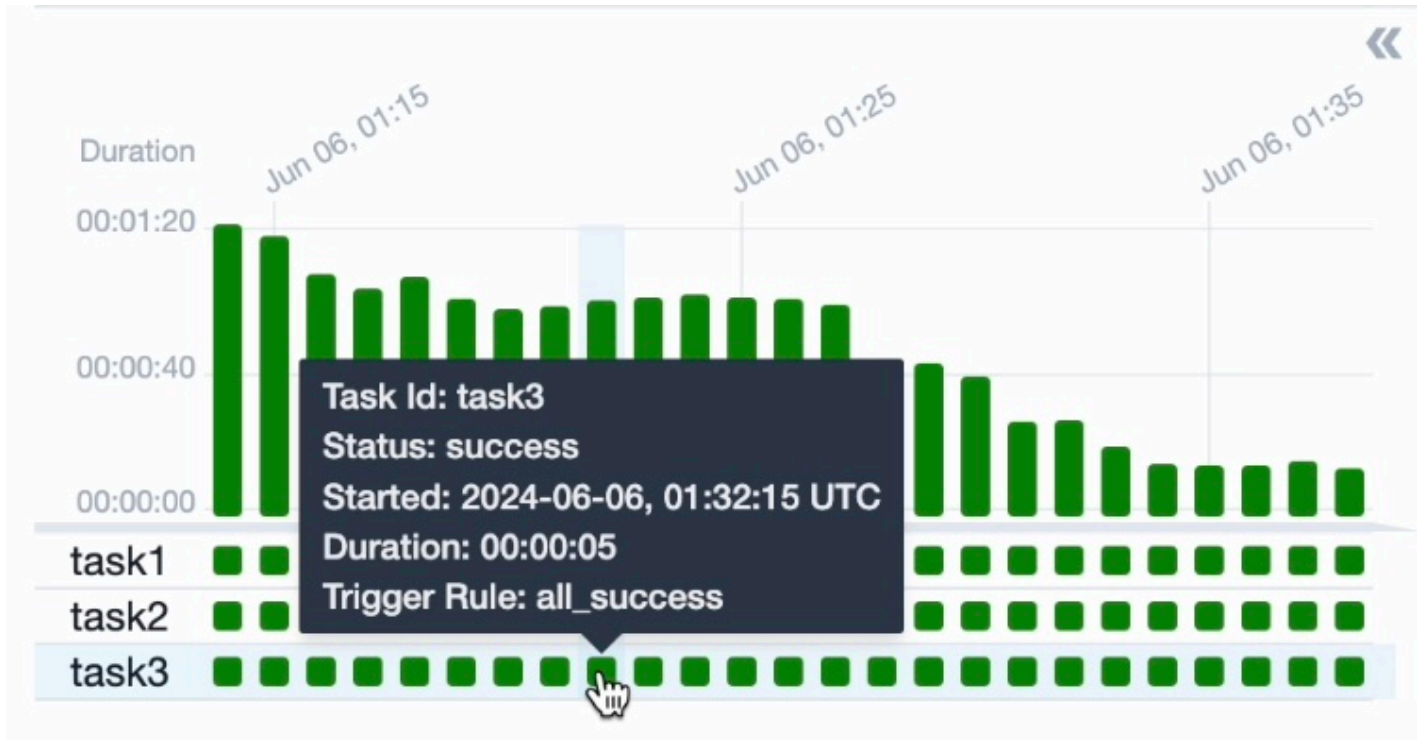


The Grid view shows your DAG tasks in the form of grids as seen in the image. You will observe the Auto Refresh button switched on by default on the right corner.

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.



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.



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.

>> DAG dummy_dag

[Details](#) [Graph](#) [Gantt](#) [Code](#) [Audit Log](#) [Run Duration](#) [Calendar](#)

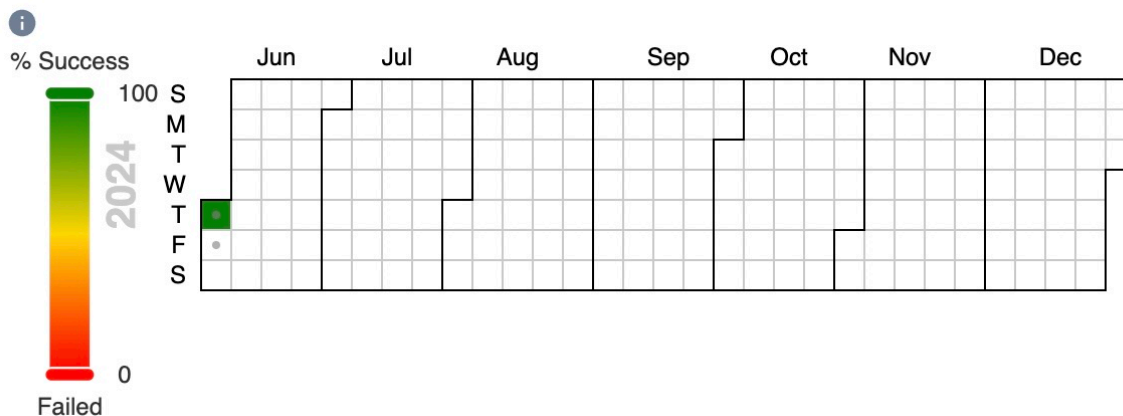


Exercise 8: Calender view

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

>> DAG dummy_dag

[Details](#) [Graph](#) [Gantt](#) [Code](#) [Audit Log](#) [Run Duration](#) [Calendar](#)



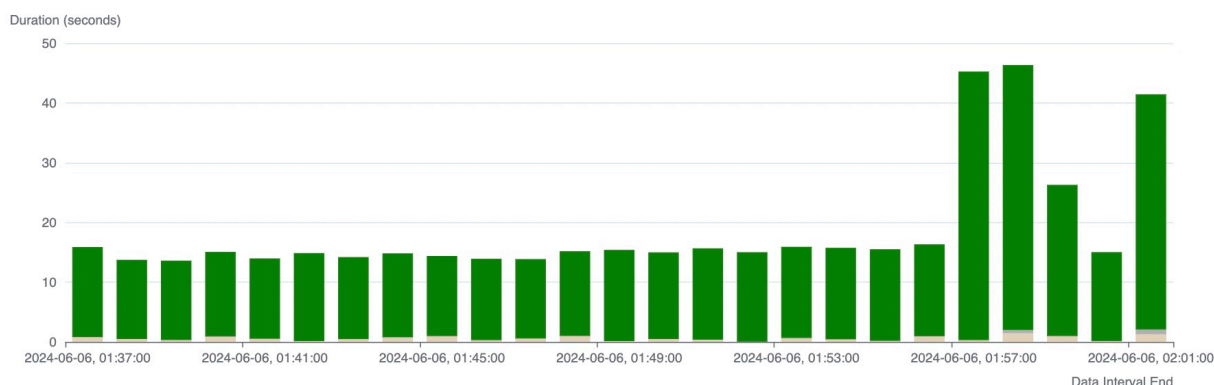
Exercise 9: DAG and Task Duration view

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

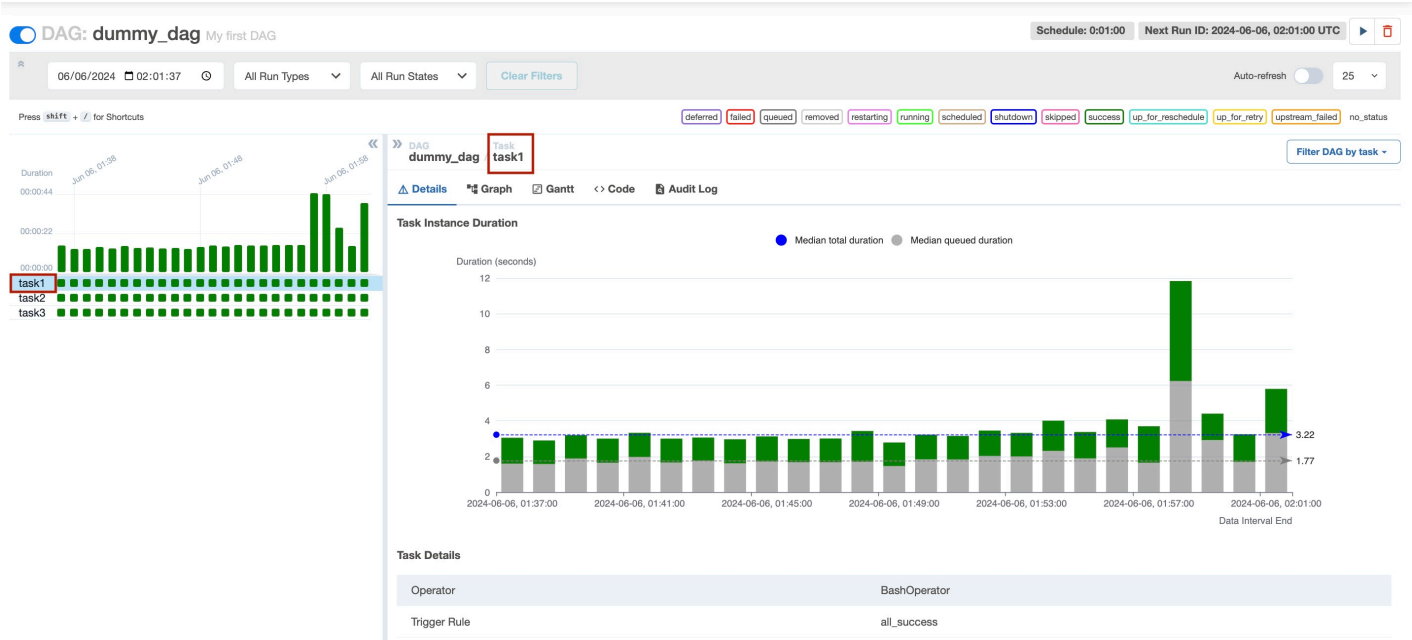
>> DAG dummy_dag

[Details](#) [Graph](#) [Gantt](#) [Code](#) [Audit Log](#) [Run Duration](#) [Calendar](#)

☒ Show Landing Times ⓘ



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



Exercise 10: Details view

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

» DAG
dummy_dag

- △ Details
- 📊 Graph
- 📅 Gantt
- < > Code
- 🔍 Audit Log
- 🕒 Run Duration
- 📅 Calendar

DAG Runs Summary

Total Runs Displayed25

■ Total success25

First Run Start2024-06-06, 01:32:20 UTC

Last Run Start2024-06-06, 01:54:00 UTC

Max Run Duration00:00:38

Mean Run Duration00:00:16

Min Run Duration00:00:13

DAG Summary

Total Tasks3

BashOperators3

DAG Details

Dag display namedummy_dag

Dag iddummy_dag

DescriptionMy first DAG

Fileloc/home/project/airflow/dags/dummy_dag.py

Has import errorsfalse

Has task concurrency limitsfalse

Is activetrue

Is pausedfalse

Exercise 11: Code view

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

>> DAG Run
 dummy_dag / 2024-06-06, 01:22:00 UTC
 Clear Mark state as...

Details Graph Gantt **<> Code** Audit Log

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

```

1 # import the libraries
2
3 from datetime import timedelta
4 # The DAG object; we'll need this to instantiate a DAG
5 from airflow import DAG
6 # Operators; we need this to write tasks!
7 from airflow.operators.bash_operator import BashOperator
8 # This makes scheduling easy
9 from airflow.utils.dates import days_ago
10
11 #defining DAG arguments
12

```

Toggle Wrap

Exercise 12: Task logs

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

>> DAG Run Task
 dummy_dag / 2024-06-06, 01:22:00 UTC / task2
 Clear task Mark state as... Filter DAG by task

Details Graph Gantt **<> Code** Audit Log **Logs** XCom Task Duration

(by attempts)
 1

All Levels ▼ All File Sources ▼ ☐ Wrap Download See More ⋮

```

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.

Airflow
 DAGs Cluster Activity Datasets Security Browse Admin Docs 02:15 UTC Log In

DAG: dummy_dag My first DAG
 Schedule: 0:01:00 Next Run ID: 2024-06-06, 02:01:00 UTC
 ▶ 🗑️

06/06/2024 02:01:37
 All Run Types All Run States Clear Filters
 Auto-refresh ☐ 25
 Delete DAG

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

...rod-theiak8s-4-tor01.proxy.cognitiveclass.ai says

Are you sure you want to delete 'dummy_dag' now?

This option will delete ALL metadata, DAG runs, etc.

EXCEPT Log.

This cannot be undone.

Cancel

OK

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

© IBM Corporation. All rights reserved.