IBM Developer
SKILLS NETWORK

Hands-on Lab: Monitoring a DAG

Estimated time needed: **20** minutes

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 tree 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

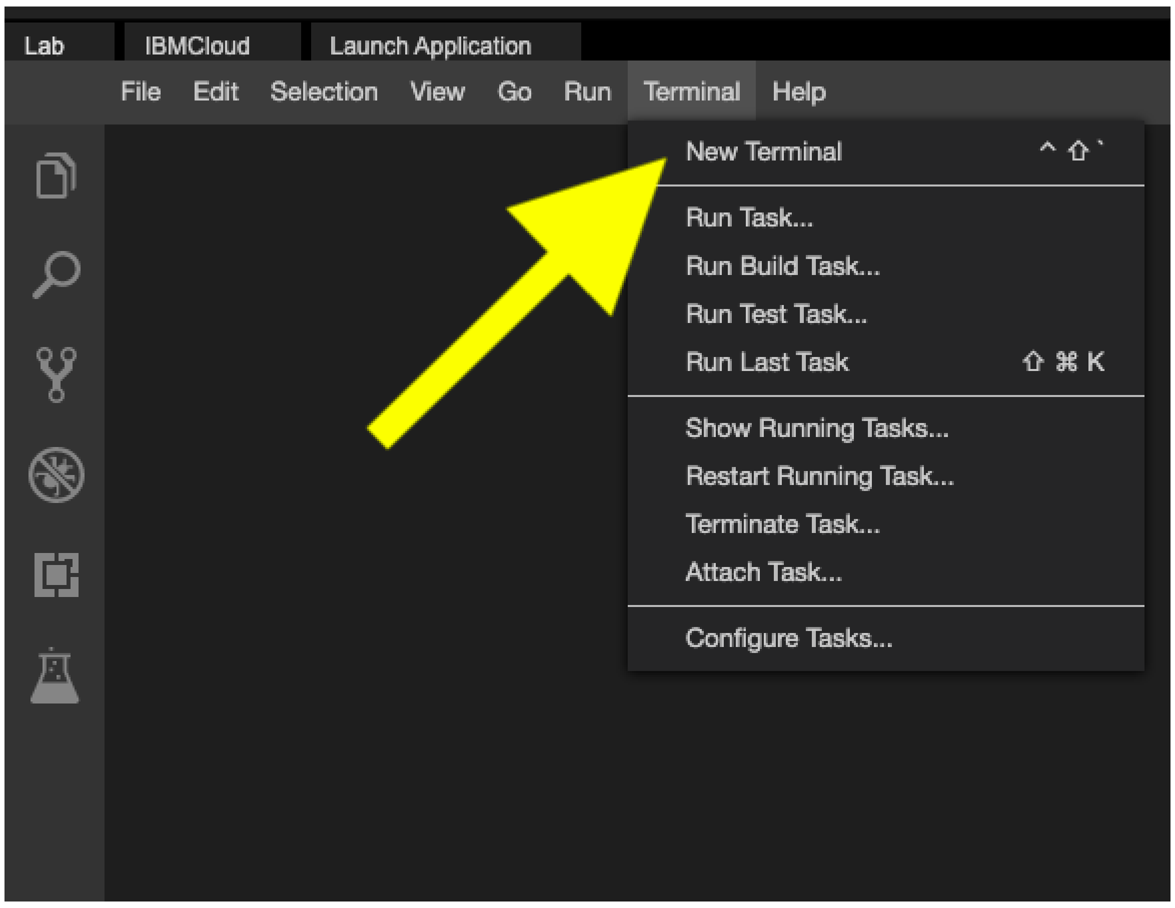
Skills Network Cloud IDE (based on Theia and Docker) provides an environment for hands on labs for course and project related labs. Theia is an open source IDE (Integrated Development Environment), that can be run on desktop or on the cloud. to complete this lab, we will be using the Cloud IDE based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

Important Notice about this lab environment

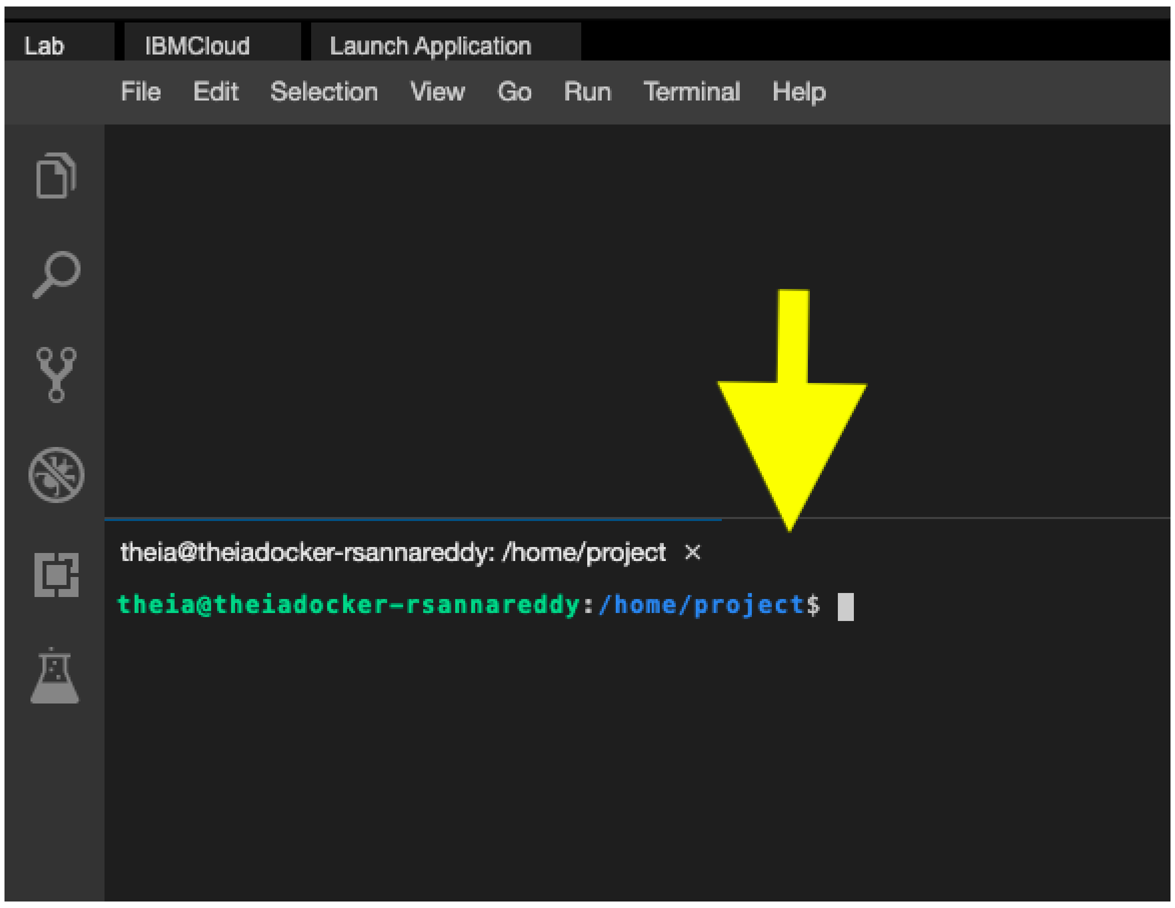
Please be aware that sessions for this lab environment are not persistent. A new environment is created for you every time you connect to this lab. Any data you may have saved in an earlier session will get lost. To avoid losing your data, please plan to complete these labs in a single session.

Exercise 1 - Getting the environment ready

Step 1.1. Open a new terminal by clicking on the menu bar and selecting **Terminal**->**New Terminal**, as shown in the image below.



This will open a new terminal at the bottom of the screen.



```
theia@theiadocker-rsannareddy:/home/project$
```

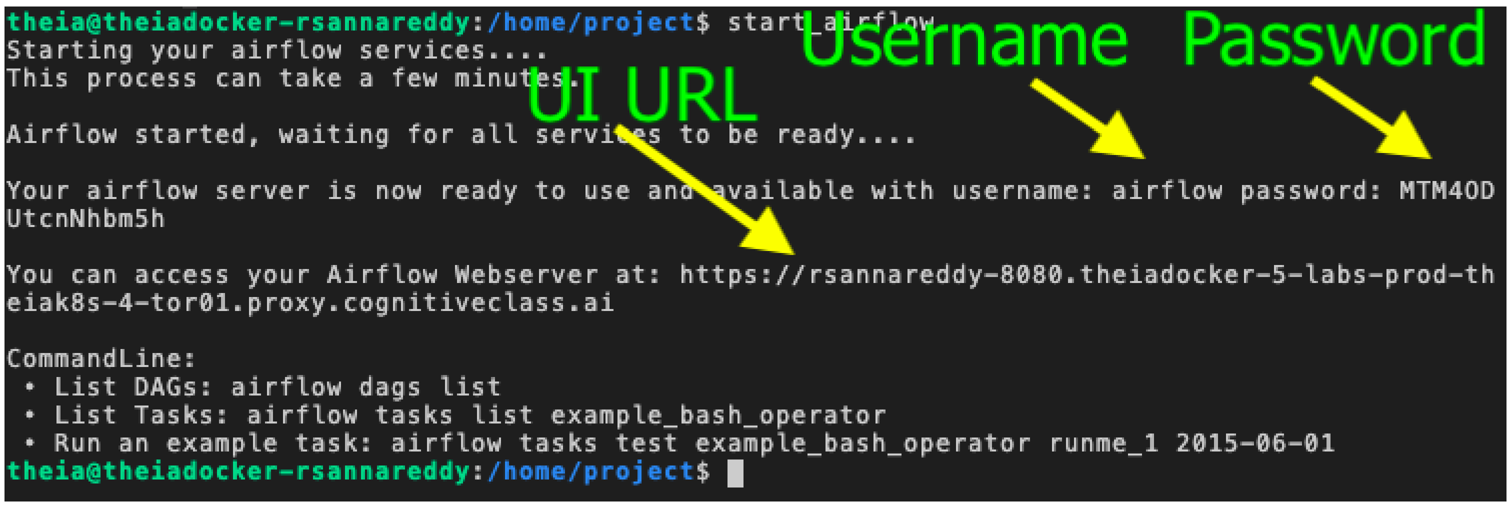
Run the commands below on the newly opened terminal. (You can copy the code by clicking on the little copy button on the bottom right of the codeblock below and then paste it, wherever you wish.)

Start Apache Airflow in the lab environment.

```
start_airflow
```

Please be patient, it will take a few minutes for airflow to get started.

When airflow starts successfully, you should see an output similar to the one below:

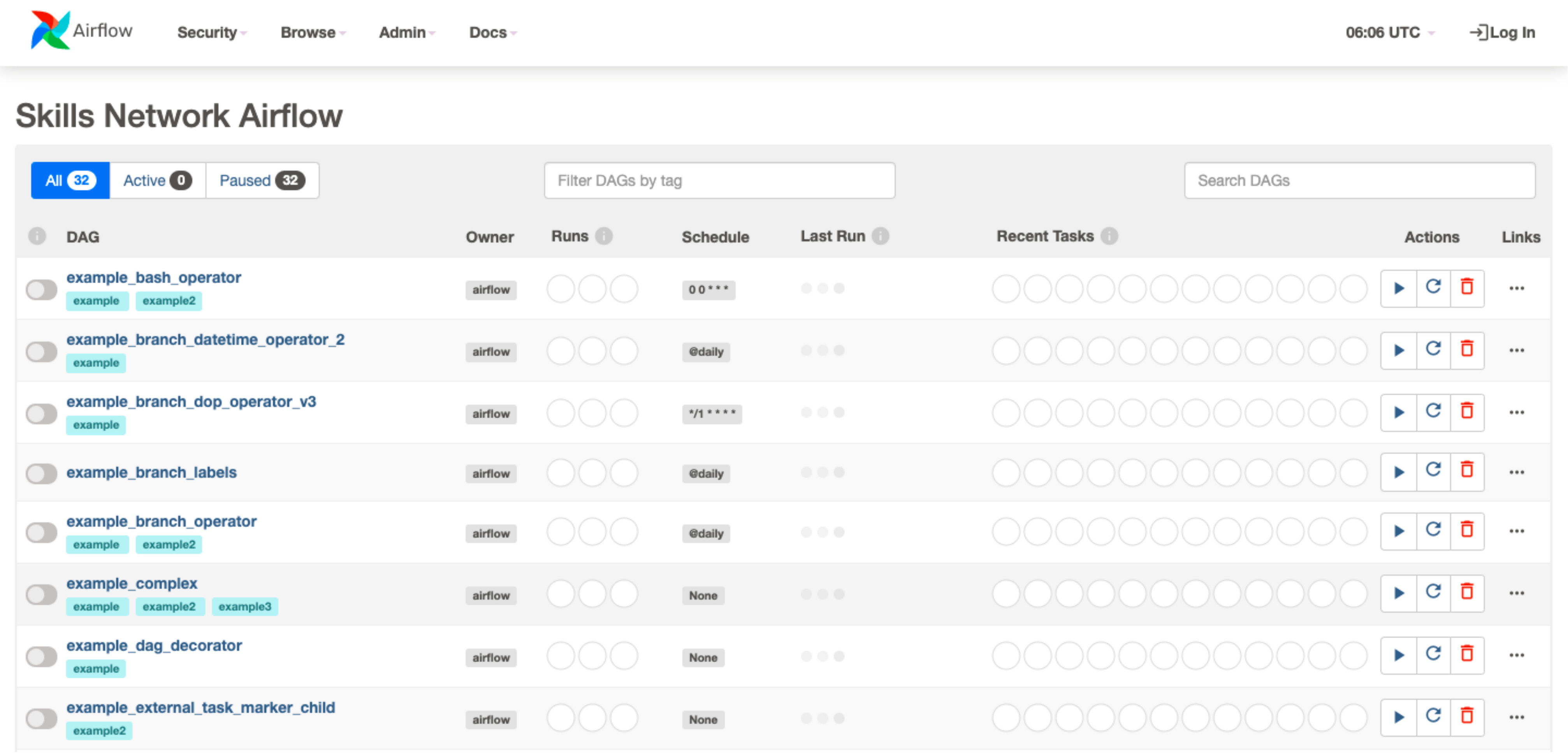


```
theia@theiadocker-rsannareddy:/home/project$ start_airflow
Starting your airflow services....
This process can take a few minutes.

Airflow started, waiting for all services to be ready....

Your airflow server is now ready to use and available with username: airflow password: MTM4OD
UtcnNhbm5h

You can access your Airflow Webserver at: https://rsannareddy-8080.theiadocker-5-labs-prod-th
eiak8s-4-tor01.proxy.cognitiveclass.ai

CommandLine:
 • List DAGs: airflow dags list
 • List Tasks: airflow tasks list example_bash_operator
 • Run an example task: airflow tasks test example_bash_operator runme_1 2015-06-01
theia@theiadocker-rsannareddy:/home/project$
```

Step 1.2. Open the Airflow Web UI

Copy the Web-UI URL and paste it on a new browser tab. Or your can click on the URL by holding the control key (Command key in case of a Mac).

You should land at a page that looks like this:



Skills Network Airflow

DAG	Owner	Runs	Schedule	Last Run	Recent Tasks	Actions	Links
example_bash_operator example example2	airflow		0 0 * * *			▶ ↻ 🗑	···
example_branch_datetime_operator_2 example	airflow		@daily			▶ ↻ 🗑	···
example_branch_dop_operator_v3 example	airflow		*/1 * * * *			▶ ↻ 🗑	···
example_branch_labels	airflow		@daily			▶ ↻ 🗑	···
example_branch_operator example example2	airflow		@daily			▶ ↻ 🗑	···
example_complex example example2 example3	airflow		None			▶ ↻ 🗑	···
example_dag_decorator example	airflow		None			▶ ↻ 🗑	···
example_external_task_marker_child example2	airflow		None			▶ ↻ 🗑	···

Exercise 2 - 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.

Step 2.1. Using Menu->`File`->`New File` create a new file named `dummy_dag.py`.

Step 2.2. 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': 'Ramesh Sannareddy',
    'start_date': days_ago(0),
    'email': ['ramesh@somemail.com'],
    'email_on_failure': False,
    'email_on_retry': False,
    '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
```

Submitting a DAG is as simple as copying the DAG python file into `dags` folder in the `AIRFLOW_HOME` directory.

Step 2.3. Open a terminal and run the command below to submit the DAG that was created in the previous exercise.

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

Step 2.4. Verify that our DAG actually got submitted.

Run the command below to list out all the existing DAGs.

```
airflow dags list
```

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

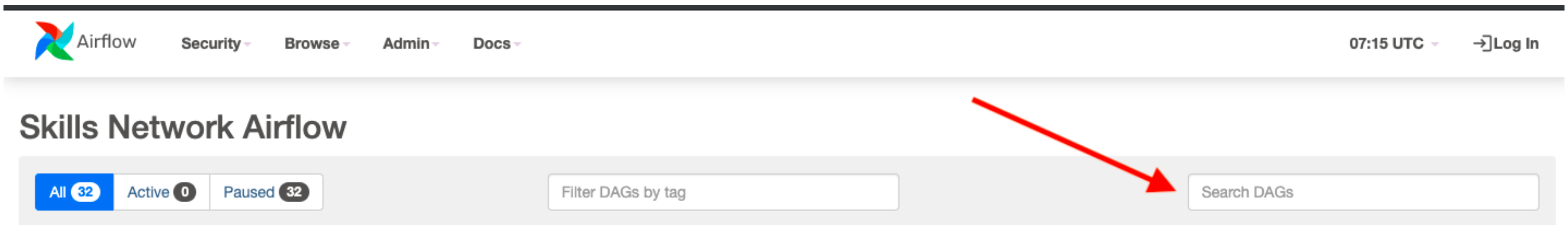
Step 2.5. Run the command below to list out all the tasks in `dummy_dag`.

```
airflow tasks list dummy_dag
```

You should see 3 tasks in the output.

Exercise 3 - Search for a DAG

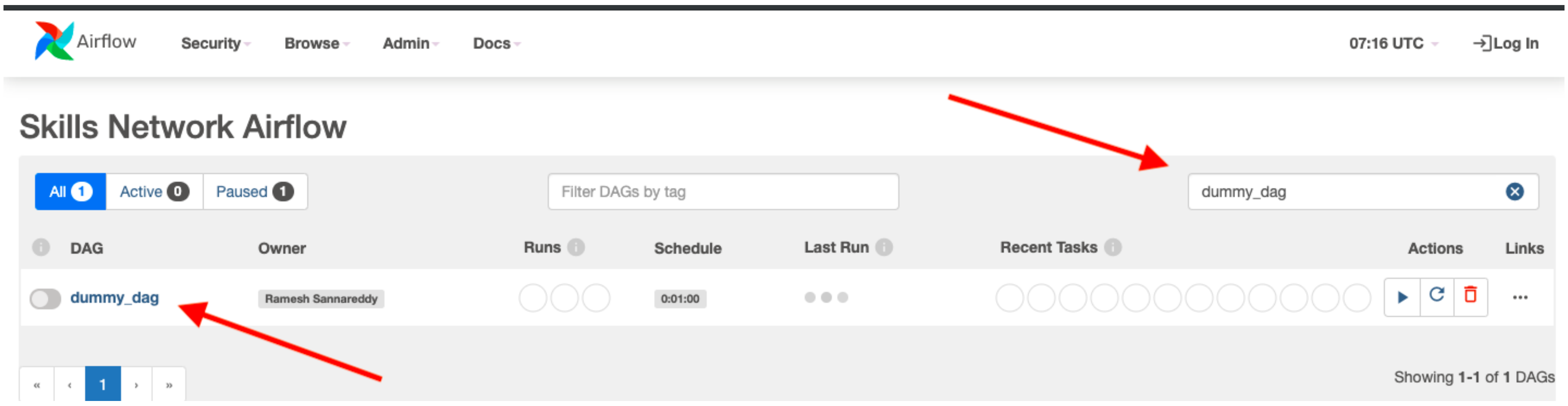
In the Web-UI, identify the `Search DAGs` text box as shown in the image below.



Type `dummy_dag` in the text box and press enter.

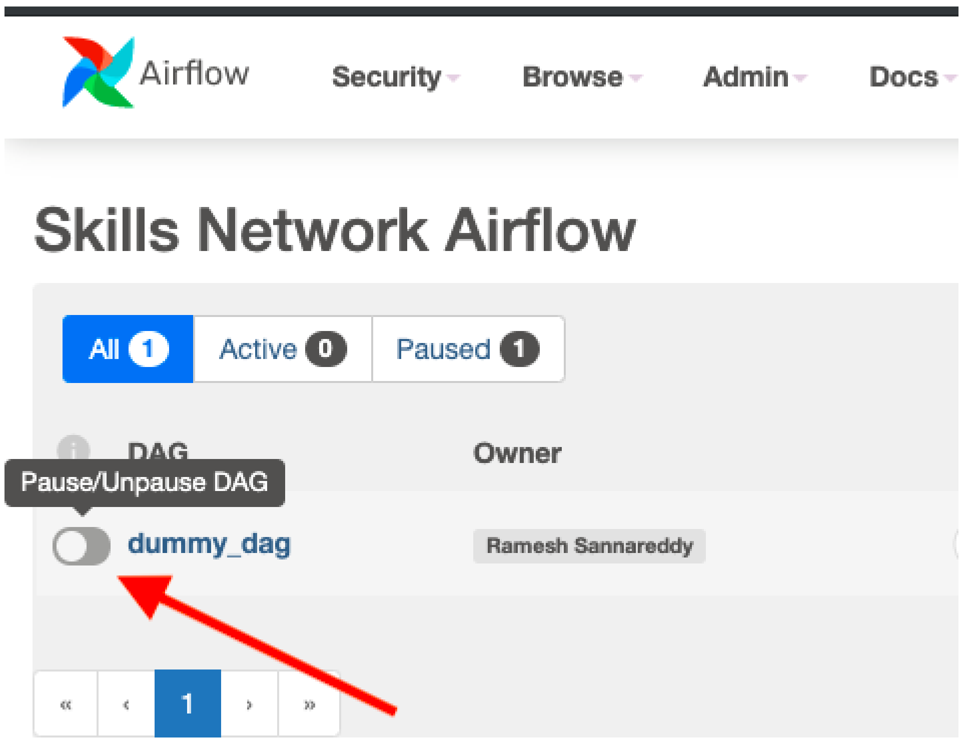
Note: It may take a couple of minutes for the dag to appear here. If you do not see your DAG, please give it a minute and try again.

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

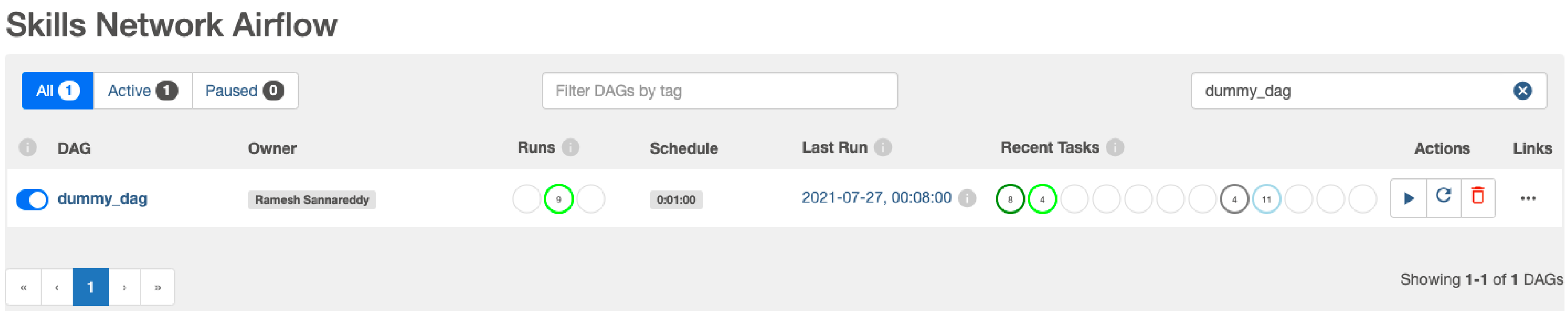


Exercise 4 - Pause/Unpause a DAG

Unpause the DAG using the Pause/Unpause button.



You should see the status as shown in the image below after you unpause the DAG.



Skills Network Airflow

DAG	Owner	Runs	Schedule	Last Run	Recent Tasks	Actions	Links
dummy_dag	Ramesh Sannareddy	9	0:01:00	2021-07-27, 00:08:00	8 4 4 11		...

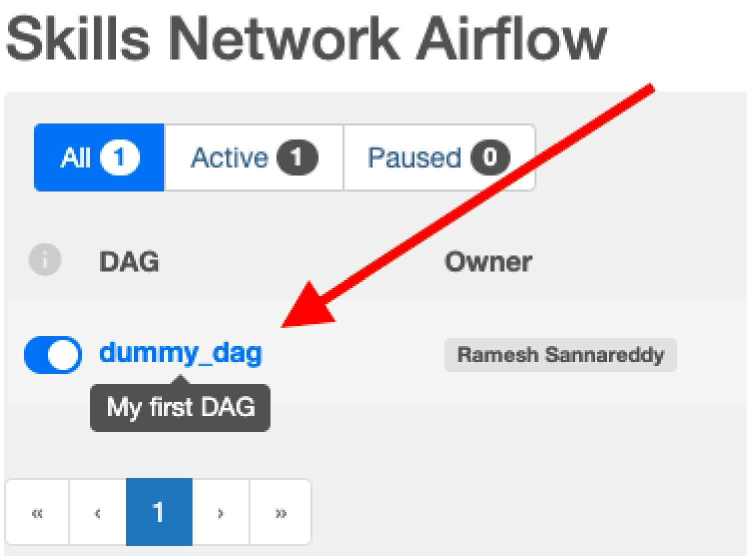
Showing **1-1** of **1** DAGs

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.

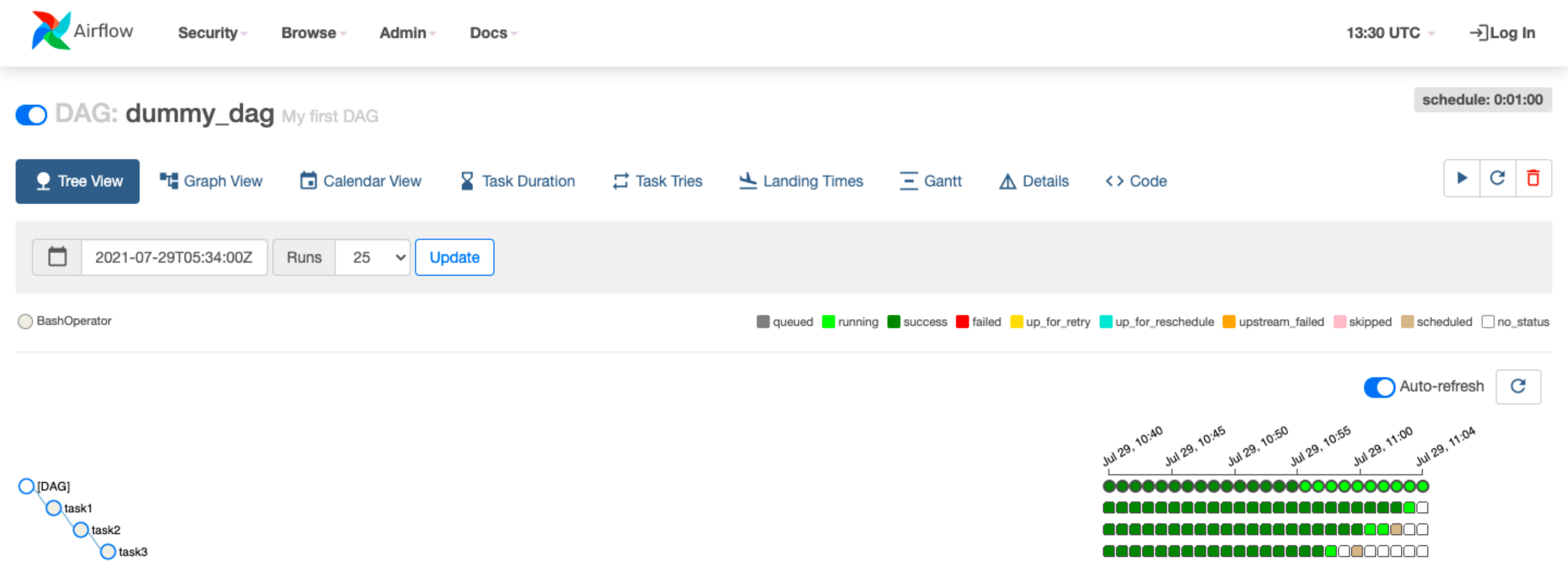
Exercise 5 - DAG - Detailed view

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



Skills Network Airflow

You will land a page that looks like this.



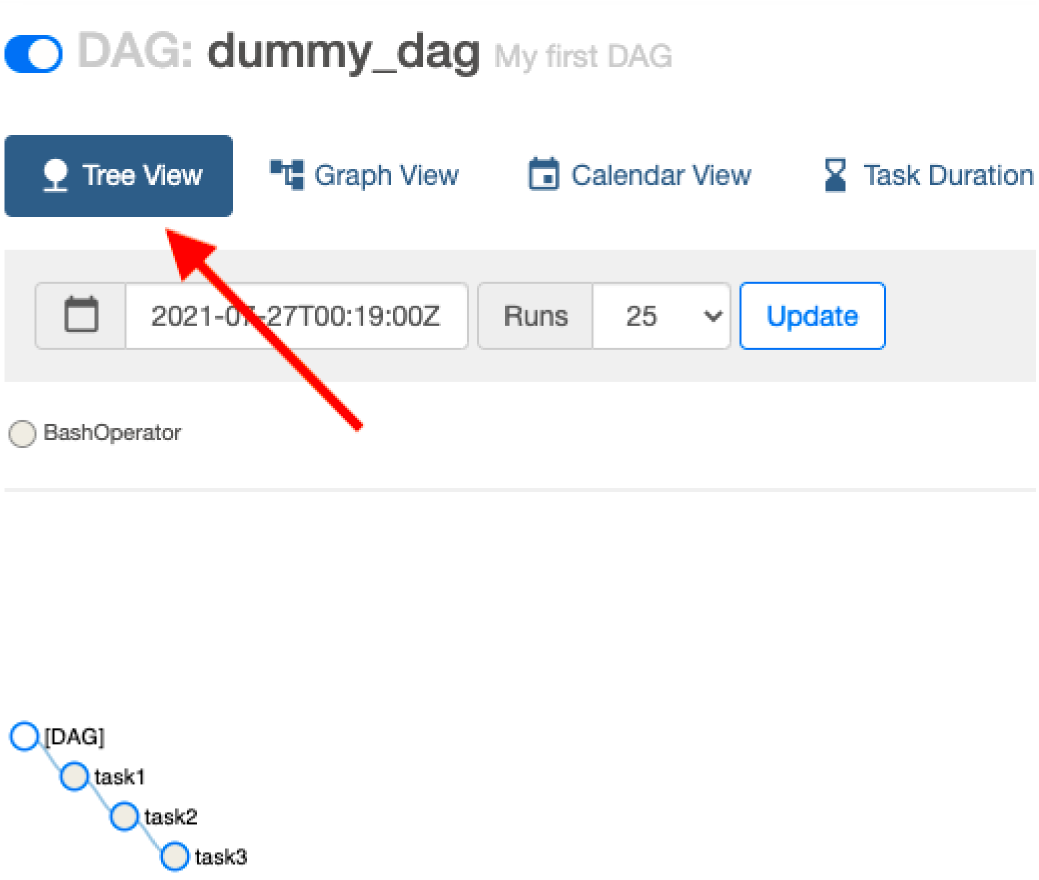
DAG: dummy_dag My first DAG

schedule: 0:01:00

2021-07-29T05:34:00Z Runs 25

Exercise 6 - Explore tree view of DAG

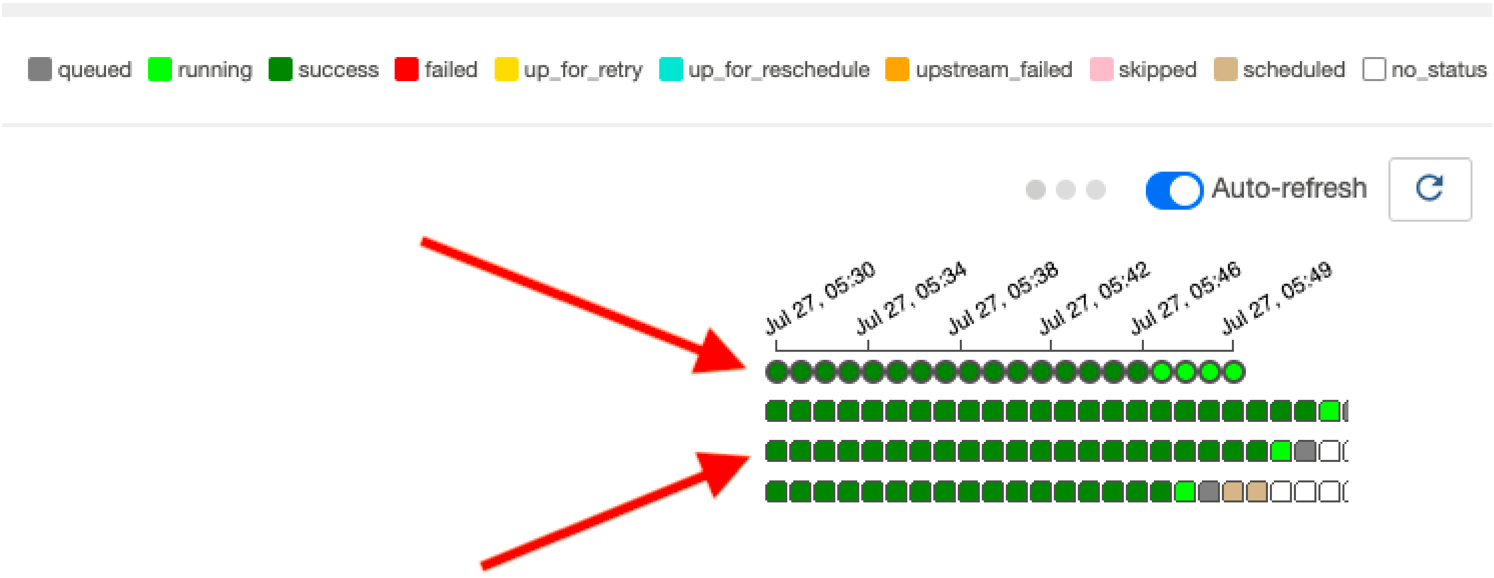
Click on the `Tree View` button to open the Tree view.



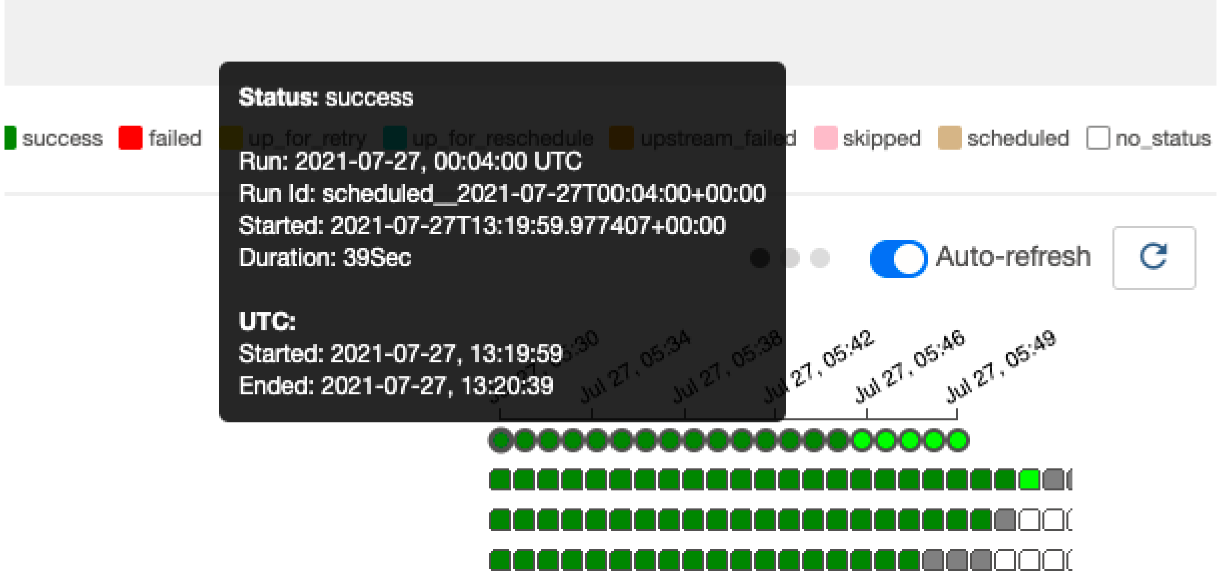
Click on the `Auto Refresh` button to switch on the auto refresh feature.

The tree view shows your DAG tasks in the form of a tree as seen in the image above.

It also shows the DAG run and task run status as seen below.



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

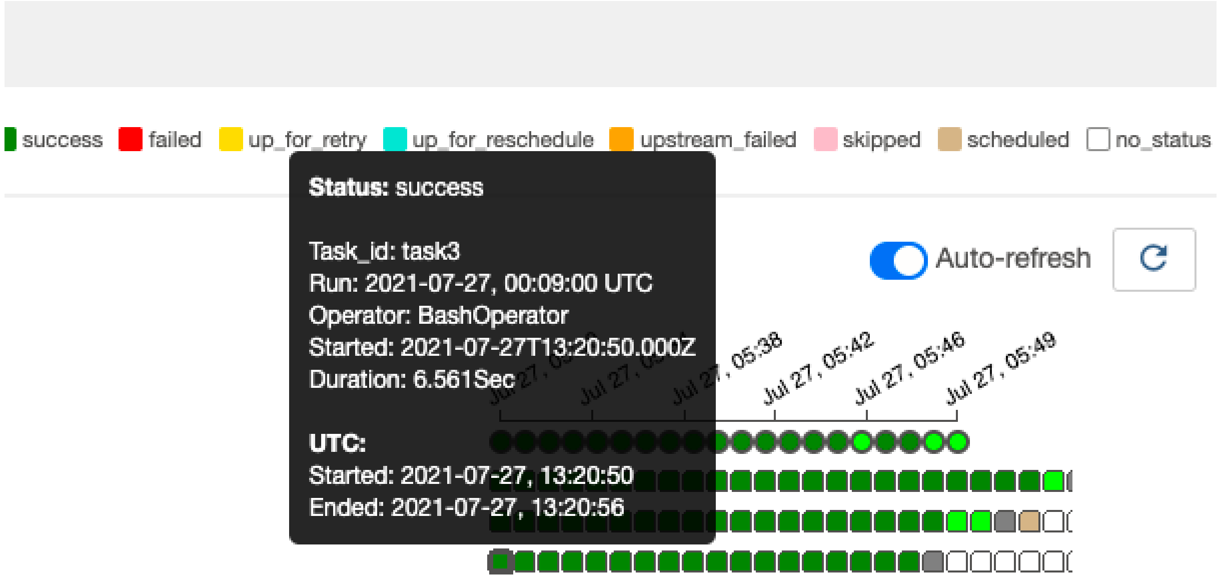


Status: success

Run: 2021-07-27, 00:04:00 UTC
Run Id: scheduled__2021-07-27T00:04:00+00:00
Started: 2021-07-27T13:19:59.977407+00:00
Duration: 39Sec

UTC:
Started: 2021-07-27, 13:19:59
Ended: 2021-07-27, 13:20:39

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.



Status: success

Task_id: task3
Run: 2021-07-27, 00:09:00 UTC
Operator: BashOperator
Started: 2021-07-27T13:20:50.000Z
Duration: 6.561Sec

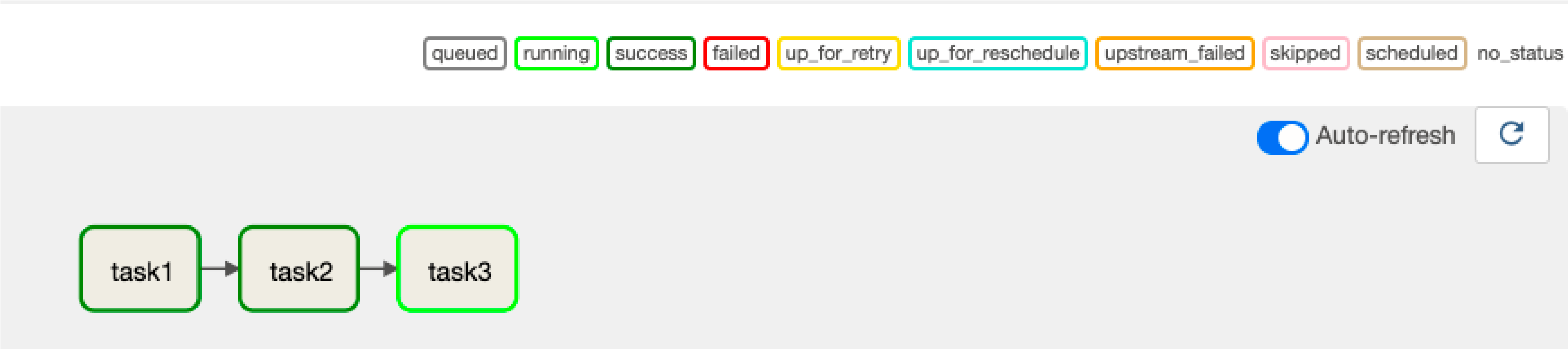
UTC:
Started: 2021-07-27, 13:20:50
Ended: 2021-07-27, 13:20:56

Exercise 7 - Explore graph view of DAG

Click on the `Graph View` button to open the graph view.

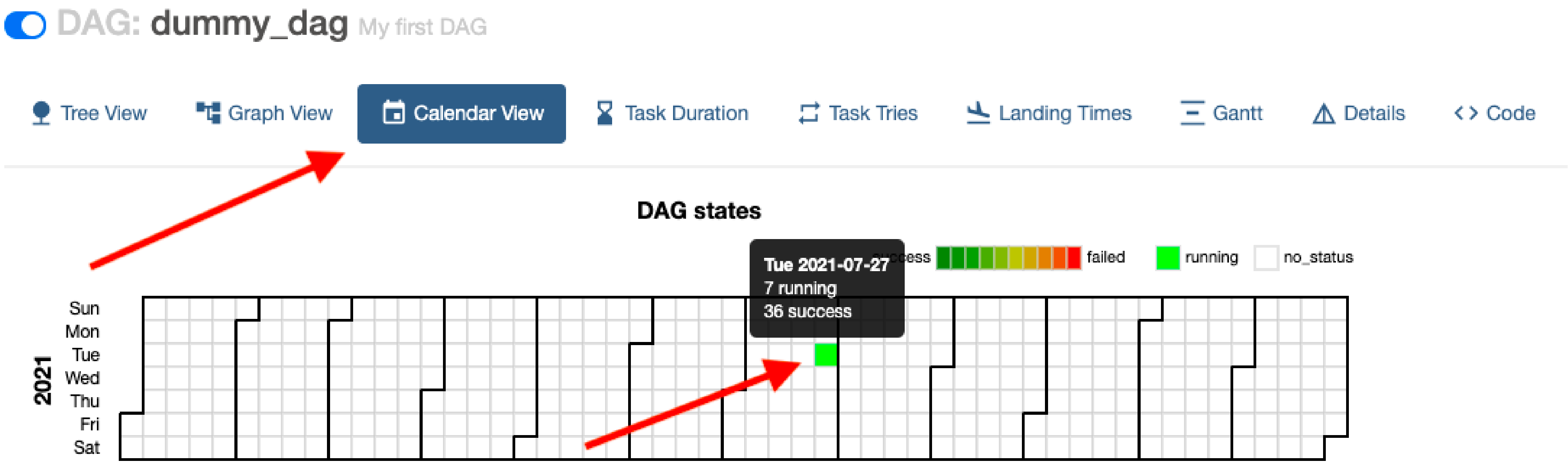
Click on the `Auto Refresh` button to switch on the auto refresh feature.

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

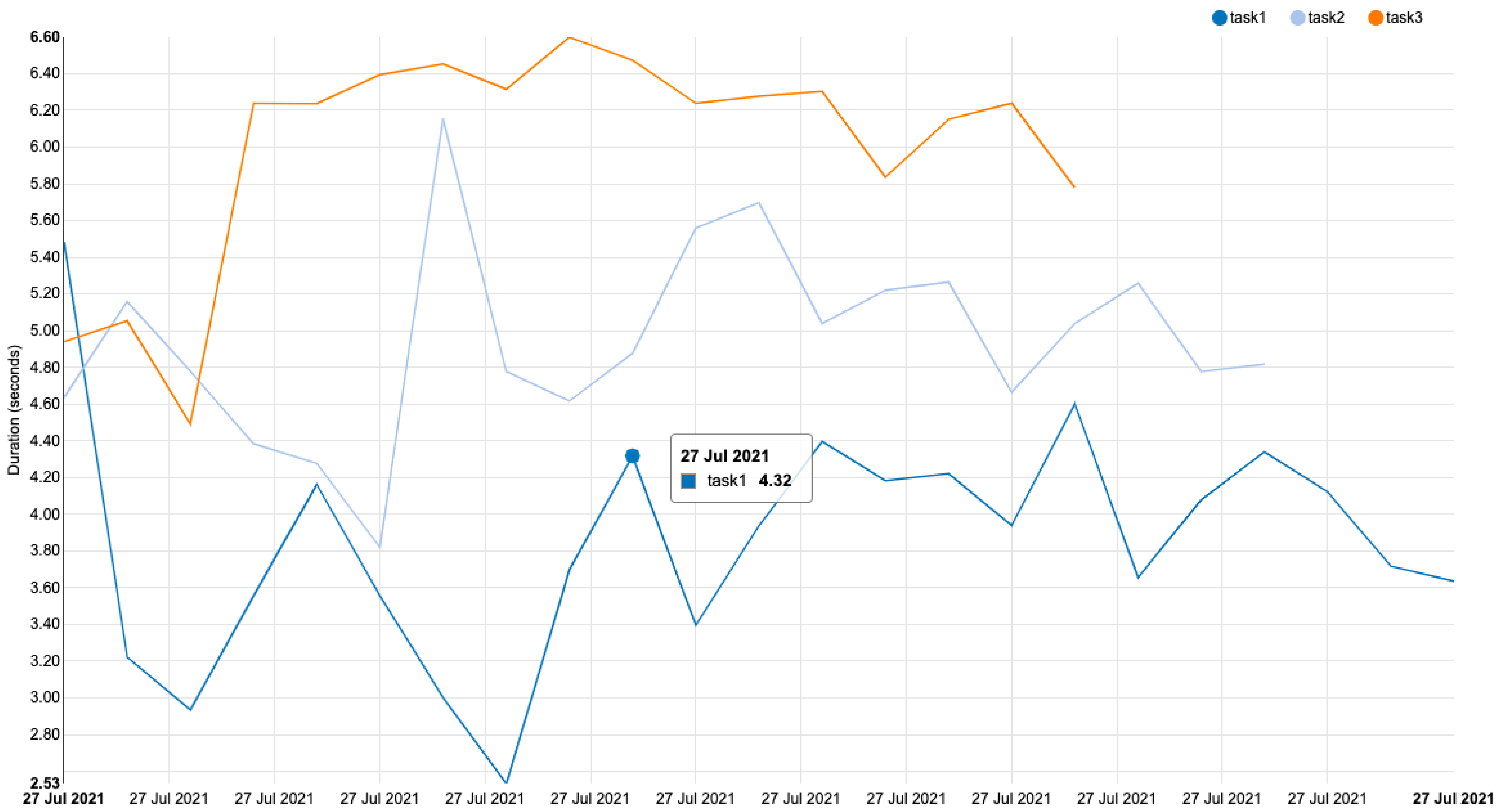
The calender view gives you an overview of all the dates when this DAG was run along with its status as a color code.



Tue 2021-07-27
7 running
36 success

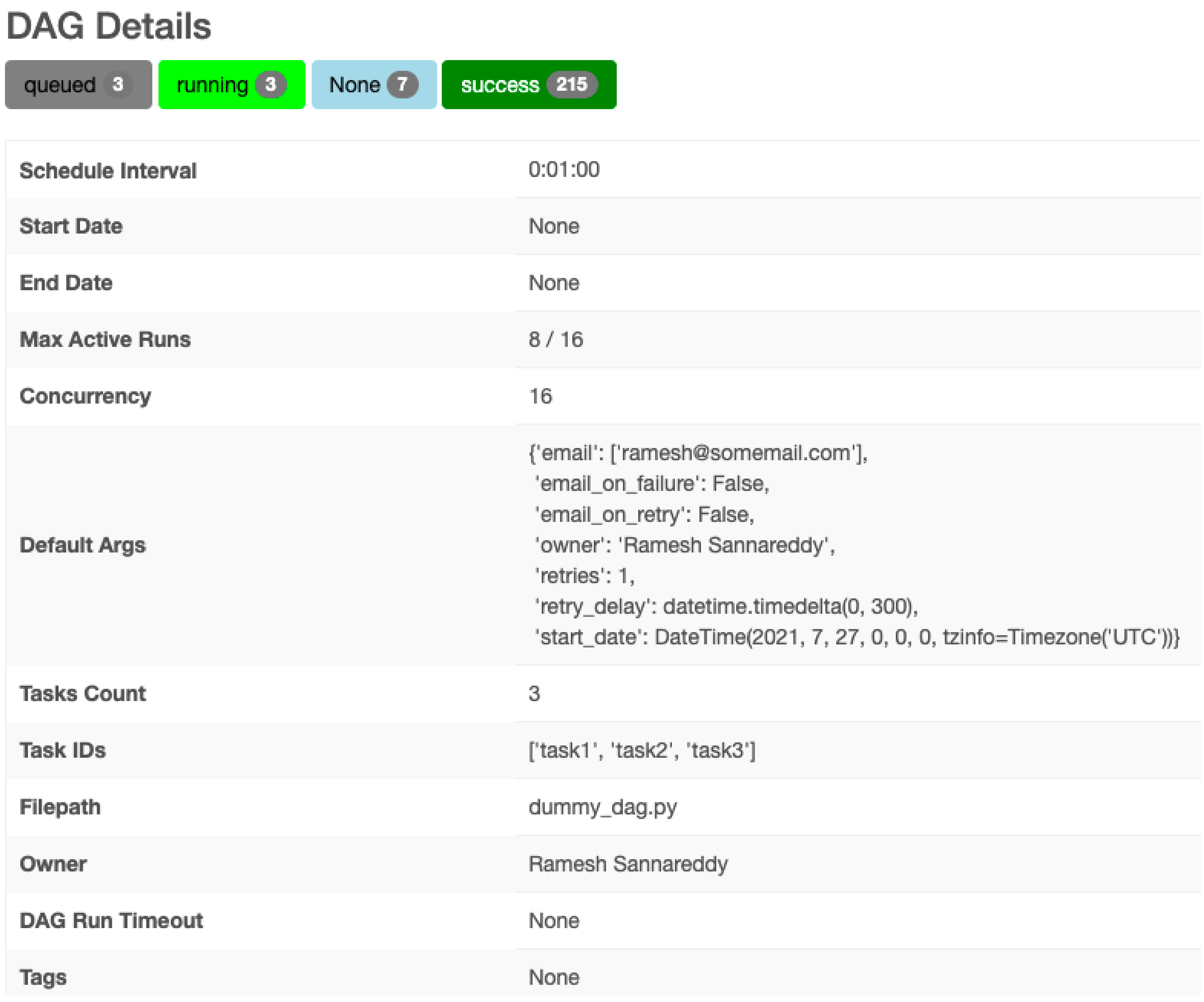
Exercise 9 - Task Duration view

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.



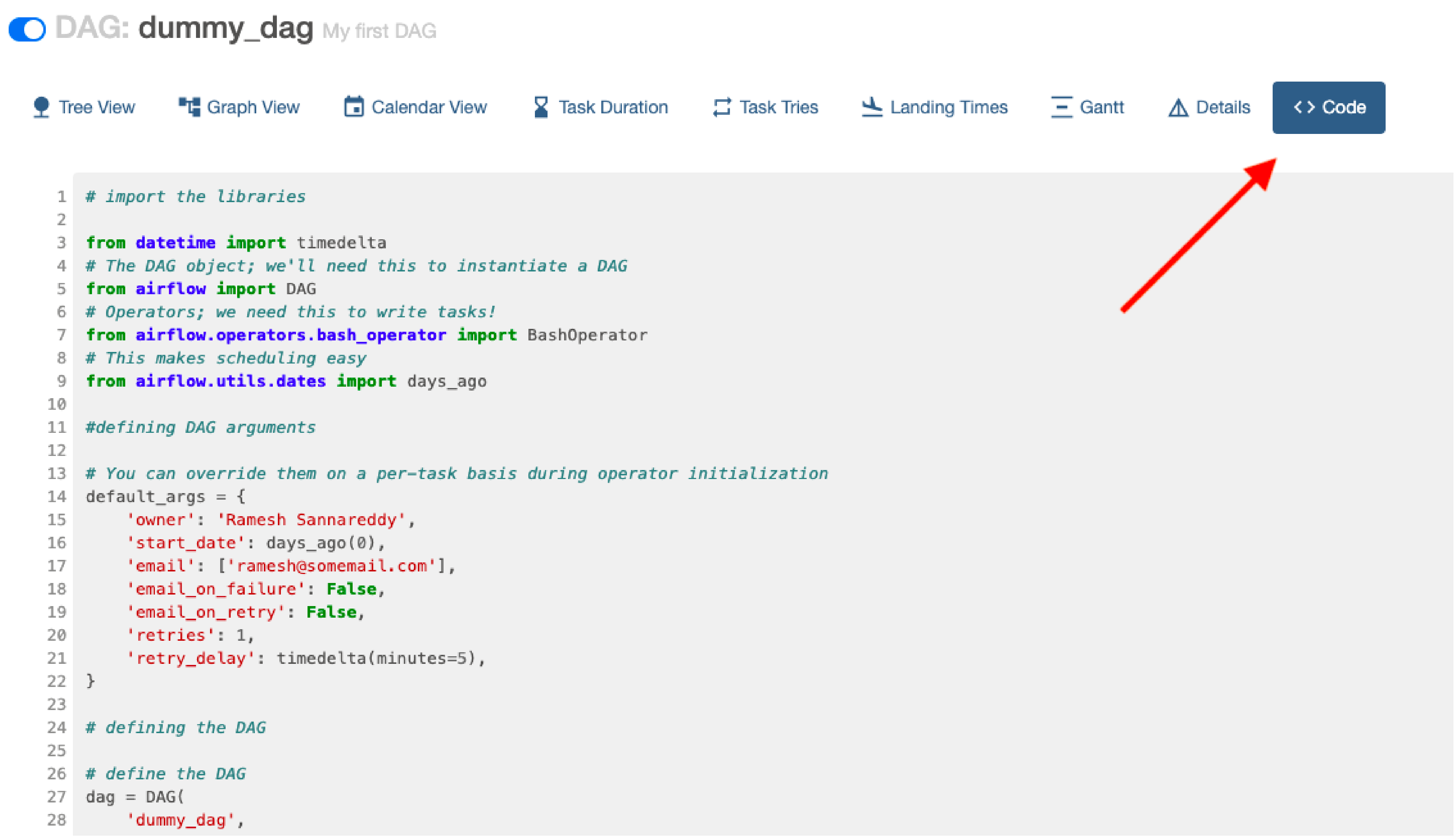
DAG Details

queued 3 | running 3 | None 7 | success 215

Schedule Interval	0:01:00
Start Date	None
End Date	None
Max Active Runs	8 / 16
Concurrency	16
Default Args	{'email': ['ramesh@somemail.com'], 'email_on_failure': False, 'email_on_retry': False, 'owner': 'Ramesh Sannareddy', 'retries': 1, 'retry_delay': datetime.timedelta(0, 300), 'start_date': DateTime(2021, 7, 27, 0, 0, 0, tzinfo=Timezone('UTC'))}
Tasks Count	3
Task IDs	['task1', 'task2', 'task3']
Filepath	dummy_dag.py
Owner	Ramesh Sannareddy
DAG Run Timeout	None
Tags	None

Exercise 11 - Code view

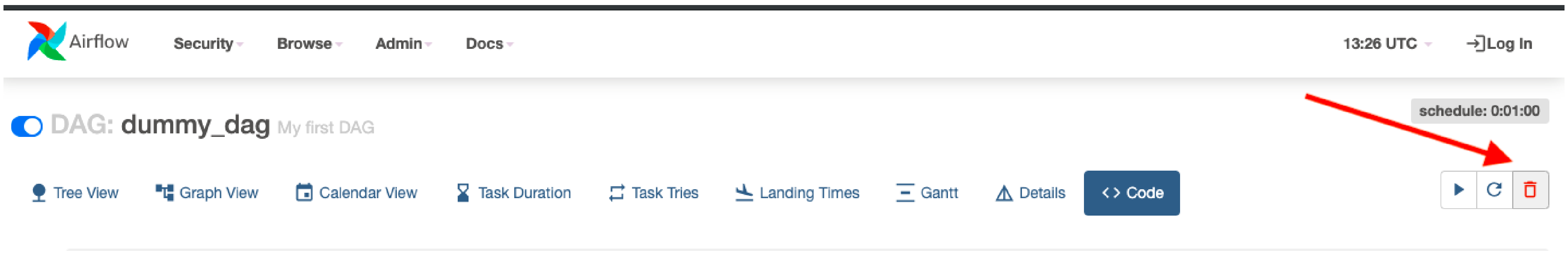
The Code view lets you view the code of the DAG.



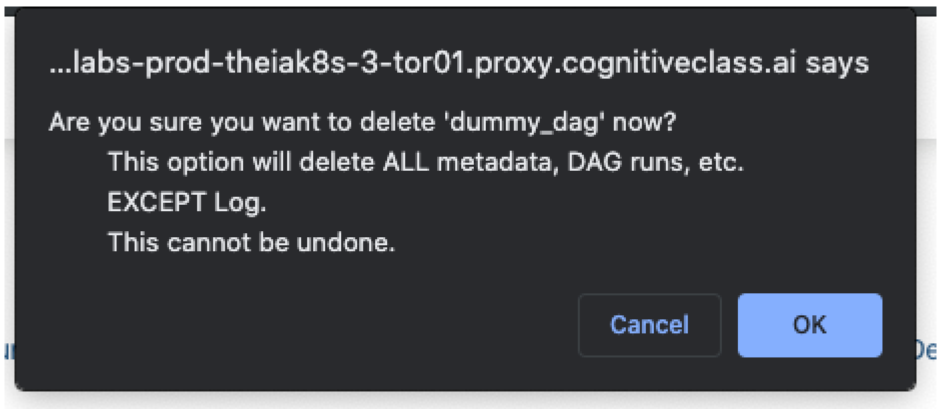
```
# import the libraries

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': 'Ramesh Sannareddy',
    'start_date': days_ago(0),
    'email': ['ramesh@somemail.com'],
    'email_on_failure': False,
    'email_on_retry': False,
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
}

# defining the DAG

# define the DAG
dag = DAG(
    'dummy_dag',
```

Exercise 12 - 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.



...labs-prod-theiak8s-3-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.

Practice exercises

1. Problem:

> *Unpause any existing DAG and monitor it.*

Authors

Ramesh Sannareddy

Other Contributors

Rav Ahuja

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2021-07-05	0.1	Ramesh Sannareddy	Created initial version of the lab