Introduction to Airflow

INTRODUCTION TO AIRFLOW IN PYTHON

Mike Metzger
Data Engineer

datacamp

What is data engineering?

Data engineering is:

- Taking any action involving data and turning it into a reliable, repeatable, and maintainable process.

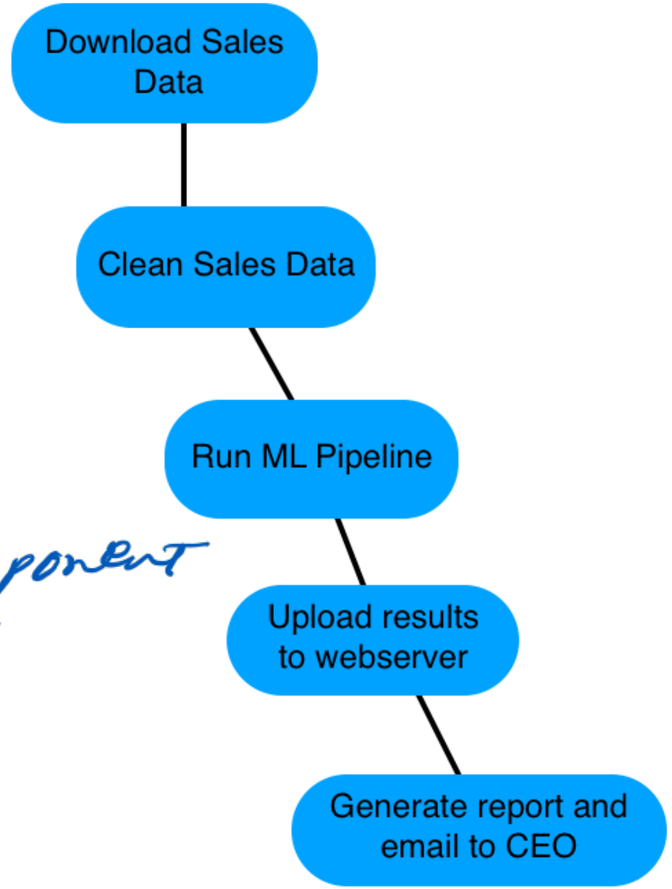
Data engineering refers to the building of systems to enable the collection and usage of data.

↳ This system should be (i) maintainable
(ii) Reliable
(iii) Repeatable.

What is a workflow?

A *workflow* is:

- A set of steps to accomplish a given data engineering task
  - Such as: downloading files, copying data, filtering information, writing to a database, etc
- Of varying levels of complexity from 2-3 to 100's component in one workflow.
- A term with various meaning depending on context → The context for above is data engineering



What is Airflow? → scalable to infinite tasks.

Airflow is a platform to program workflows, including:

- Creation → author
- Scheduling
- Monitoring

Apache Airflow → Backfilling ?

Airflow allows to define almost any workflow in python code, no matter how complex.

Use Case : (i) Business Operation / Data pipeline
(ii) ETL
(iii) infra management like AWS / spin / manage / tear down.
(iii) Use for MLOps / MLOps pipeline.

Airflow continued...

- Can implement programs from any language, but workflows are written in Python
- Implements workflows as DAGs: Directed Acyclic Graphs
- Accessed via code, command-line, or via web interface / REST API

Apache Airflow

Note: Do not use Airflow for infinitely running event based workflows.
it was built for finite batch workflow.

[1] https://airflow.apache.org/docs/stable/

Other workflow tools

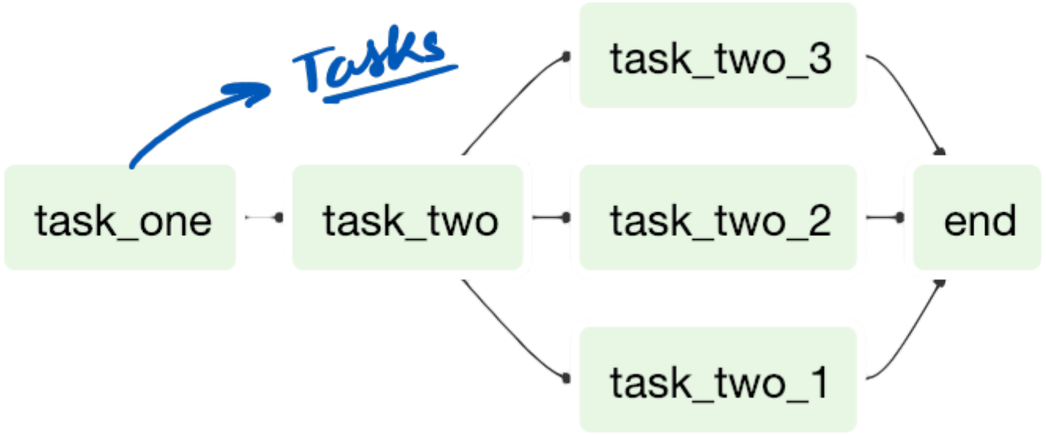
Other tools:

- Luigi → Spotify's
- SSIS → Microsoft
- Bash scripting

Quick introduction to DAGs

A *DAG* stands for *Directed Acyclic Graph*

- In Airflow, this represents the set of tasks that make up your workflow.
- Consists of the tasks and the dependencies between tasks.
- Created with various details about the DAG, including the name, start date, owner, etc.
- Further depth in the next lesson.



Dag → set of task → workflow → tasks and dependencies bw tasks.

DAG code example

Simple DAG definition:

In any Python code a dag referred by this

```
etl_dag = DAG(
    dag_id='etl_pipeline',
    default_args={"start_date": "2024-01-08"}
)
```

when using Airflow shell command dag will be reffred by this.

Running a workflow in Airflow

Running a simple Airflow task

```
airflow tasks test <dag_id> <task_id> [execution_date]
```

Using a DAG named *example-etl*, a task named *download-file* on 2024-01-10:

```
airflow tasks test example-etl download-file 2024-01-10
```

Let's practice!

INTRODUCTION TO AIRFLOW IN PYTHON

Airflow DAGs

INTRODUCTION TO AIRFLOW IN PYTHON

Mike Metzger
Data Engineer

datacamp

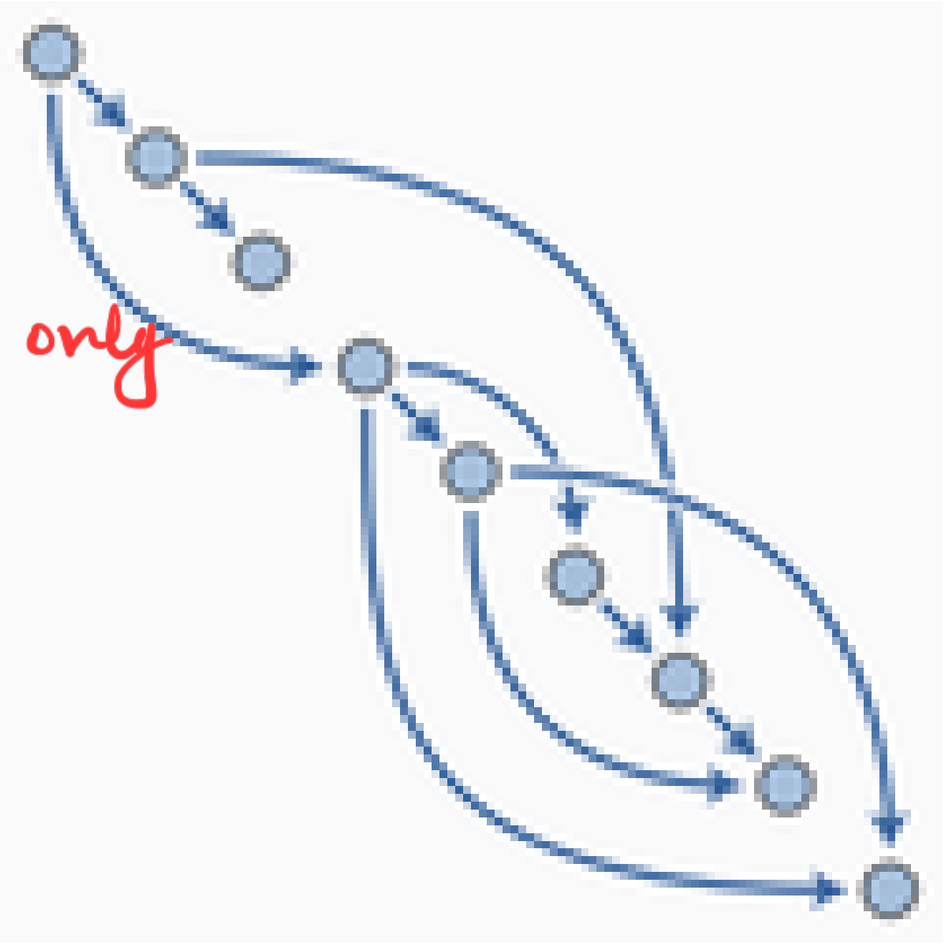
What is a DAG?

DAG, or *Directed Acyclic Graph*:

- *Directed*, there is an inherent flow representing dependencies between components.
- *Acyclic*, does not loop / cycle / repeat.
- *Graph*, the actual set of components.
- Seen in Airflow, Apache Spark, dbt

provides context How to order the running of components

one component run only once per run.



[1] https://en.m.wikipedia.org/wiki/Directed_acyclic_graph

DAG in Airflow

Within Airflow, DAGs:

- Are written in Python (but can use components written in other languages).
- Are made up of components (typically *tasks*) to be executed, such as operators, sensors, etc.
- Contain dependencies defined explicitly or implicitly.
  - ie, Copy the file to the server before trying to import it to the database service.

Define a DAG

Example DAG:

```
from airflow import DAG

from datetime import datetime
default_arguments = {
  'owner': 'jdoe',
  'email': 'jdoe@datacamp.com',
  'start_date': datetime(2020, 1, 20)
}

with DAG('etl_workflow', default_args=default_arguments ) as etl_dag:
```

these are optional but provide power to define the run time behaviour of airflow.

↳ dag object

↳ alias.

Define a DAG (before Airflow 2.x)

Example DAG:

```
from airflow import DAG

from datetime import datetime
default_arguments = {
  'owner': 'jdoe',
  'email': 'jdoe@datacamp.com',
  'start_date': datetime(2020, 1, 20)
}

etl_dag = DAG('etl_workflow', default_args=default_arguments )
```

↳ older method before 2.0

DAGs on the command line

Using `airflow`:

- The `airflow` command line program contains many subcommands.
- `airflow -h` for descriptions.
- Many are related to DAGs.
- `airflow dags list` to show all recognized DAGs.

Command line vs Python

Use the command line tool to:

- Start Airflow processes
- Manually run DAGs / Tasks
- Get logging information from Airflow

Use Python to:

- Create a DAG
- Edit the individual properties of a DAG

Let's practice!

INTRODUCTION TO AIRFLOW IN PYTHON

Airflow webUI is useful in developing and administering workflows on the airflow platform.

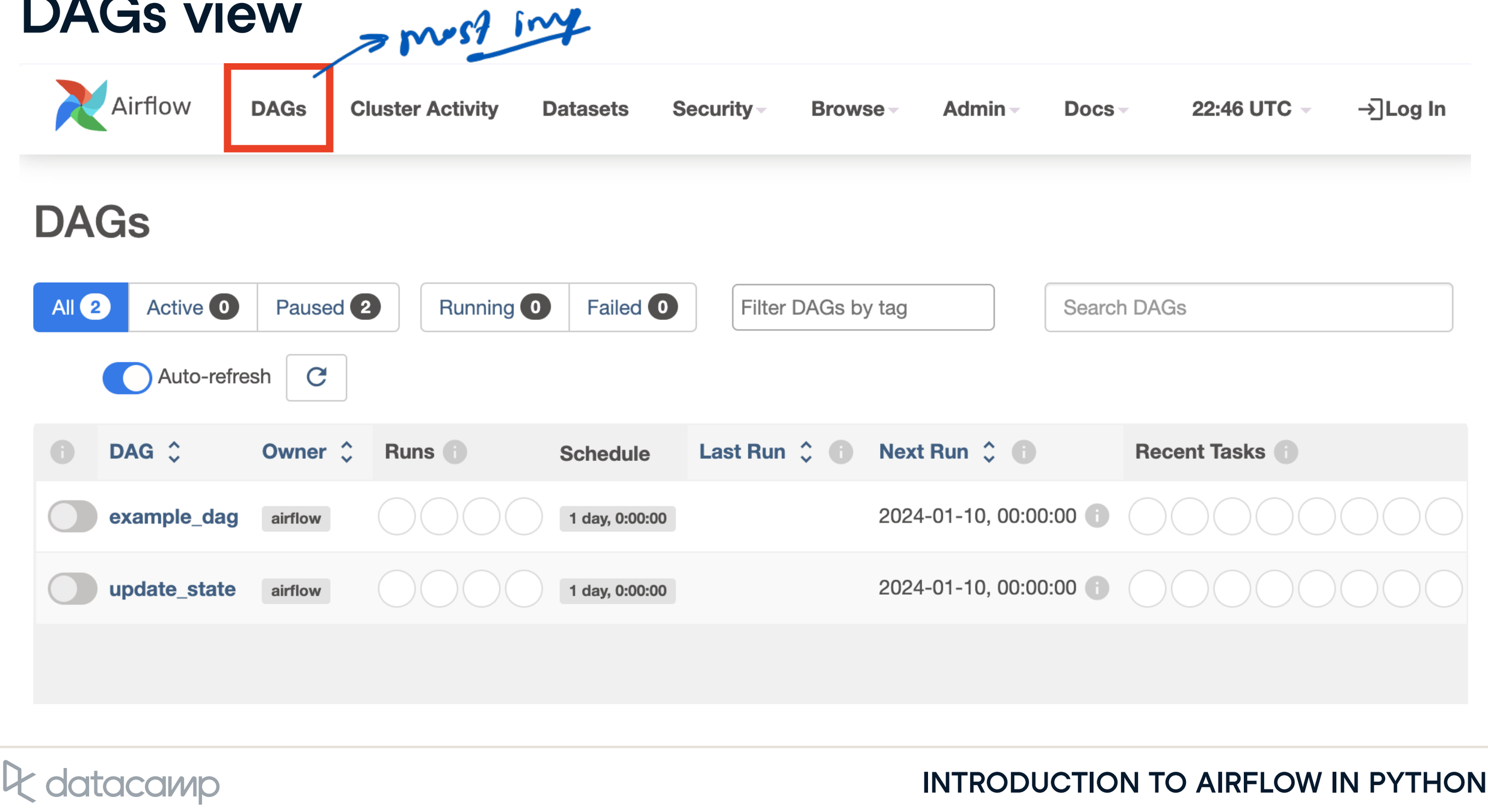
→ made up of several primary page groups.

Airflow web interface

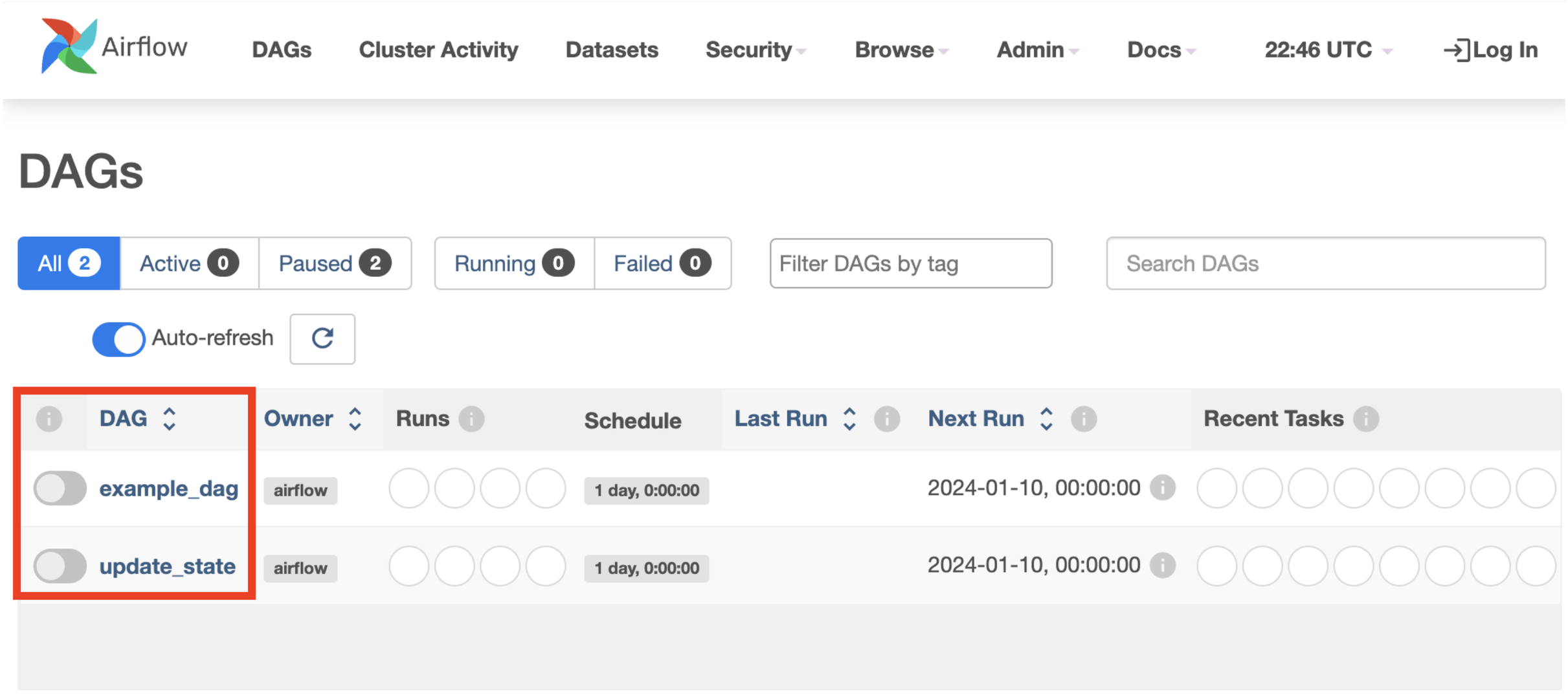
INTRODUCTION TO AIRFLOW IN PYTHON

Mike Metzger
Data Engineer

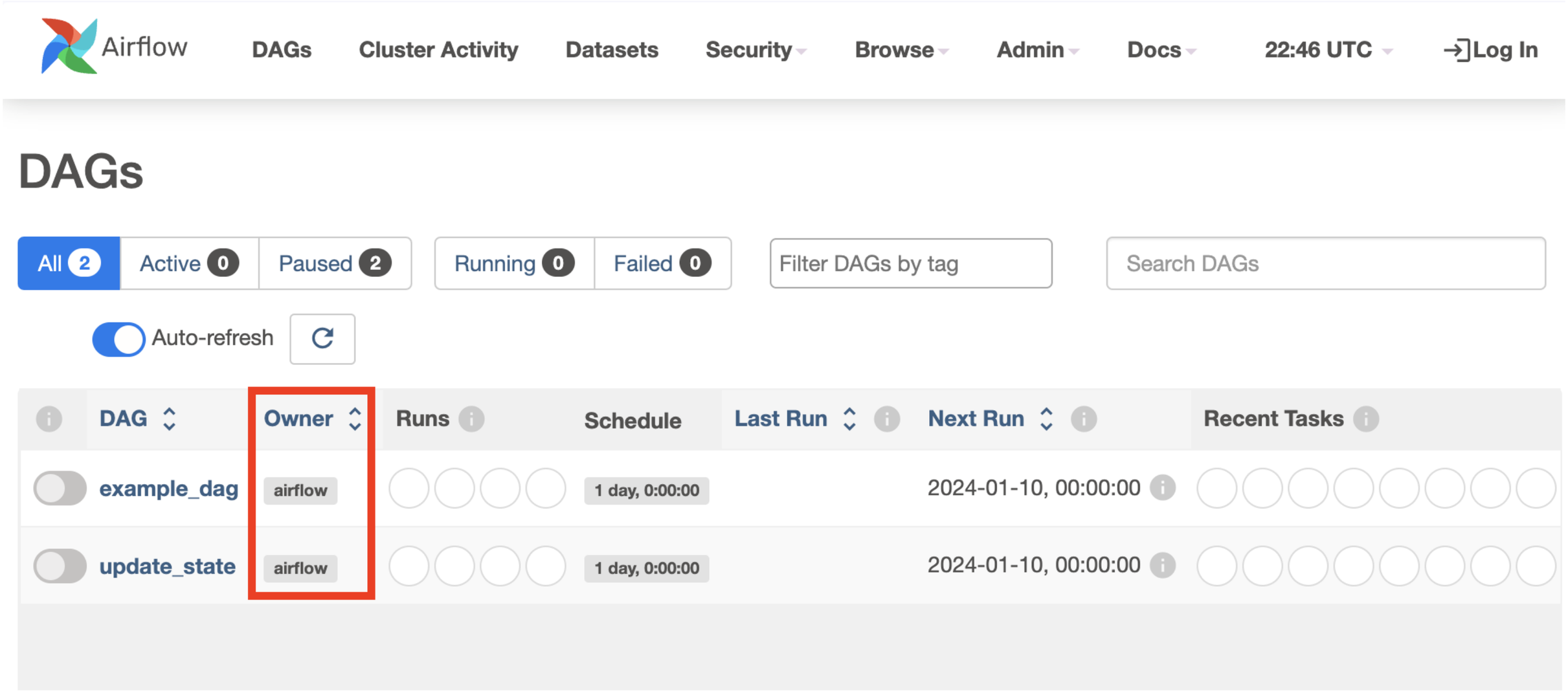
DAGs view


most imp
Airflow
DAGs
Cluster Activity
Datasets
Security
Browse
Admin
Docs
22:46 UTC
Log In
DAGs
All 2
Active 0
Paused 2
Running 0
Failed 0
Filter DAGs by tag
Search DAGs
Auto-refresh
DAG
Owner
Runs
Schedule
Last Run
Next Run
Recent Tasks
example_dag
airflow
1 day, 0:00:00
2024-01-10, 00:00:00
update_state
airflow
1 day, 0:00:00
2024-01-10, 00:00:00

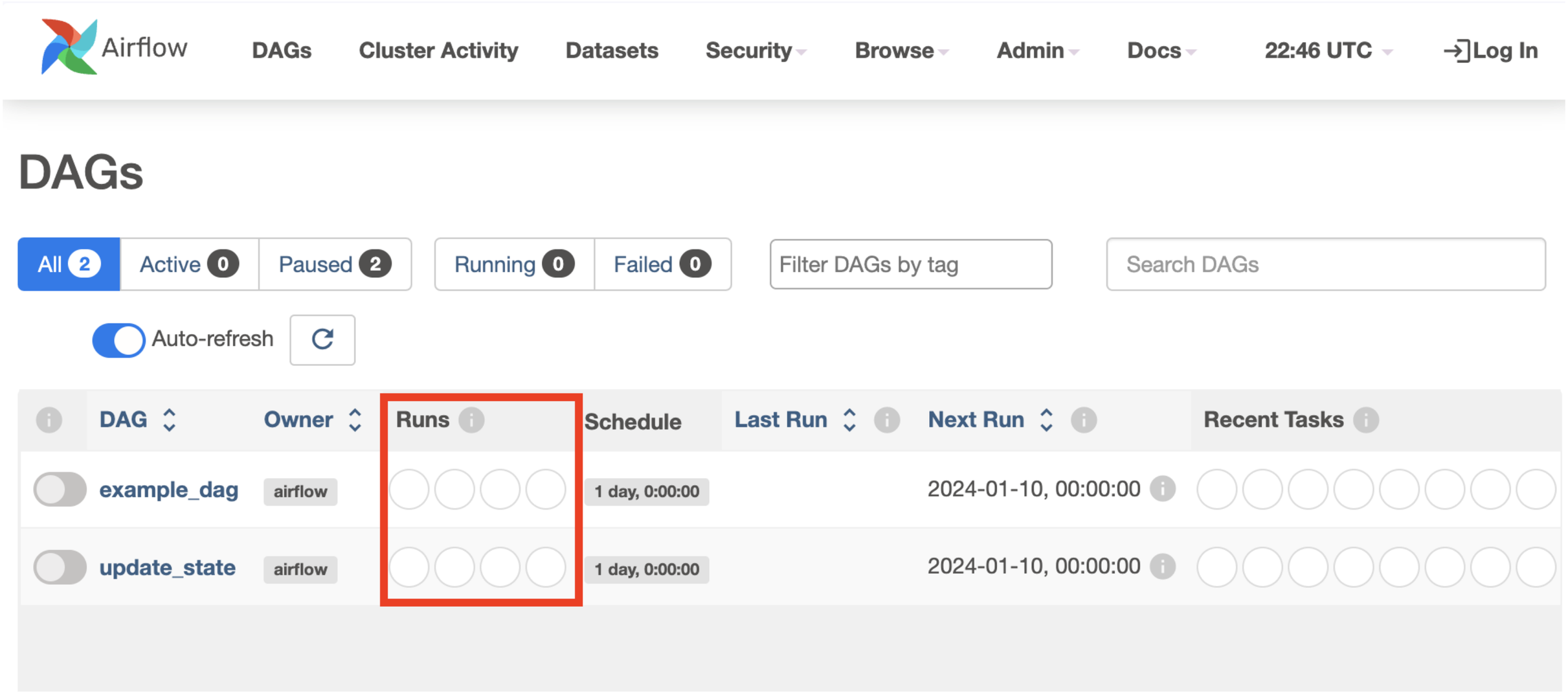
DAGs view DAGs



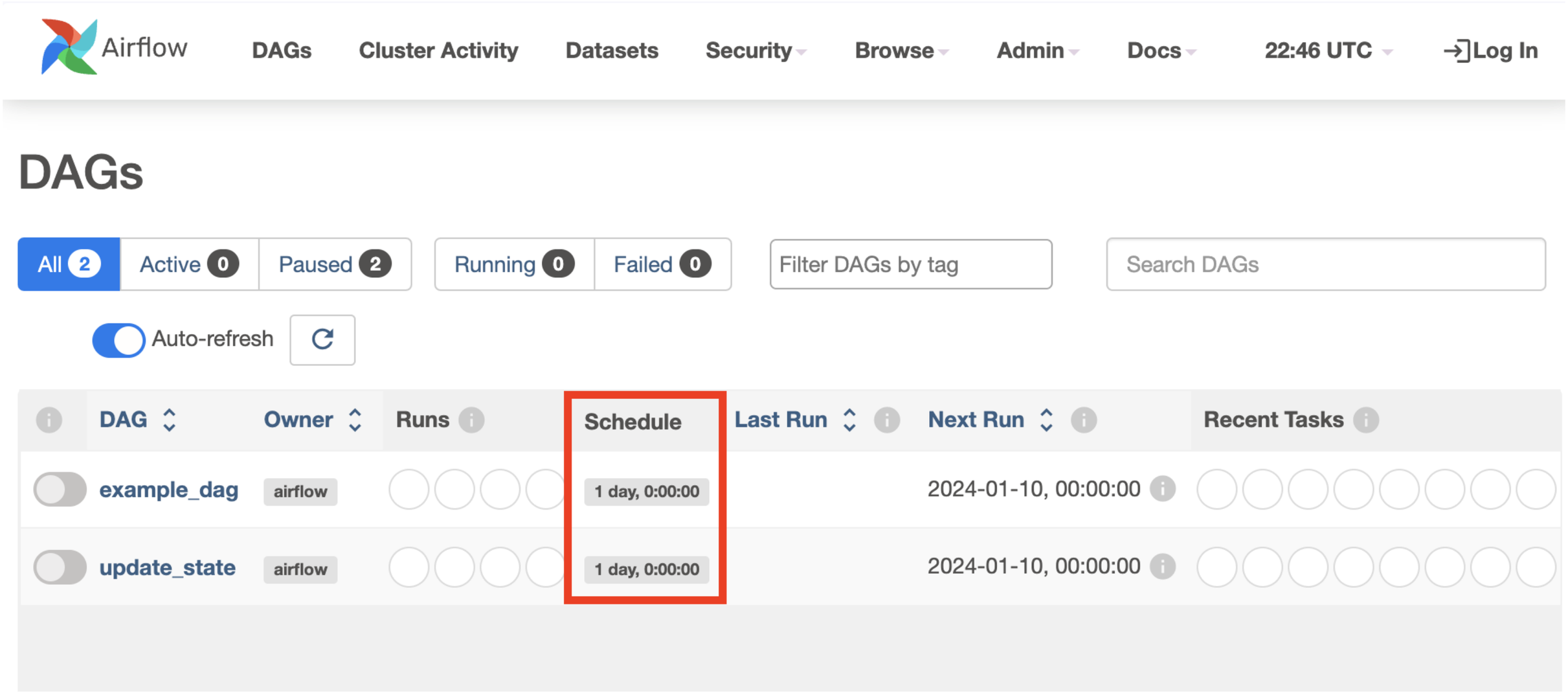
DAGs view owner



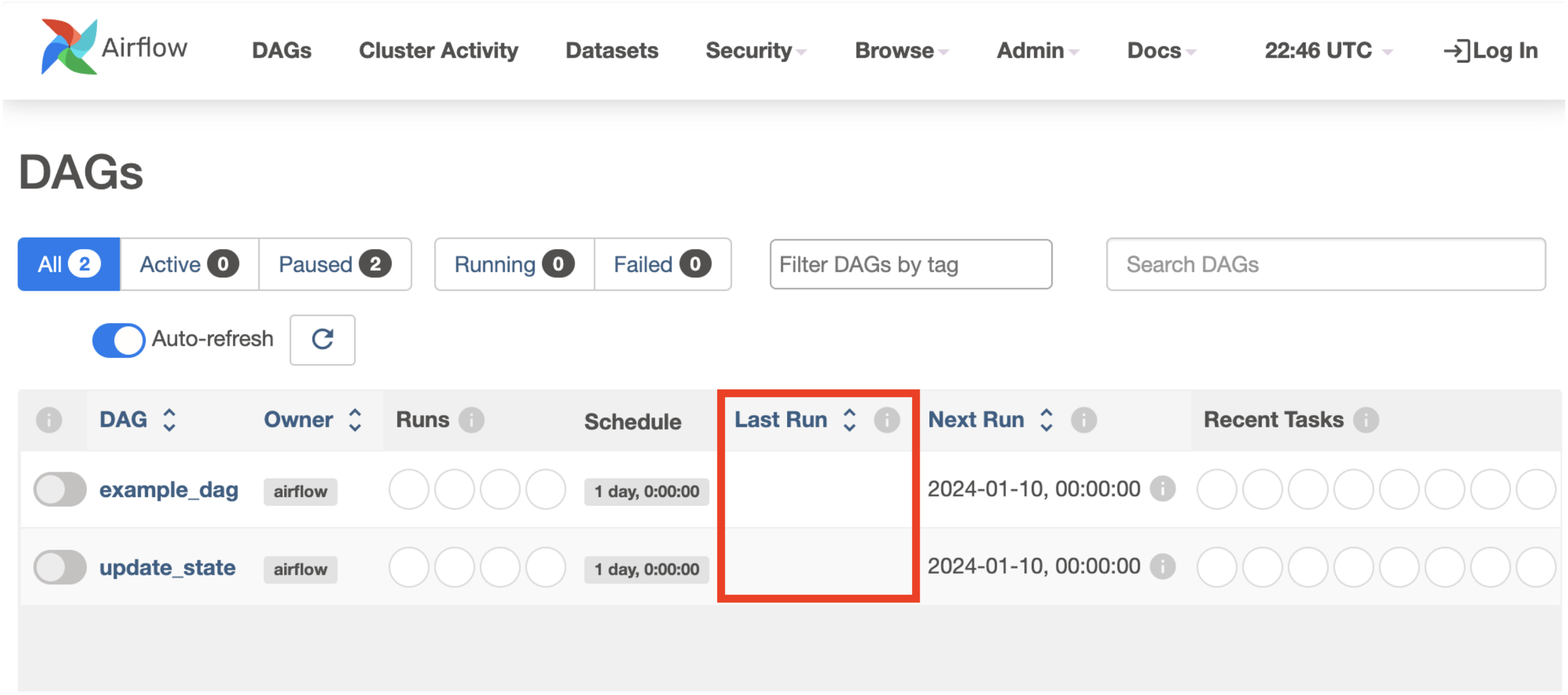
DAGs view runs



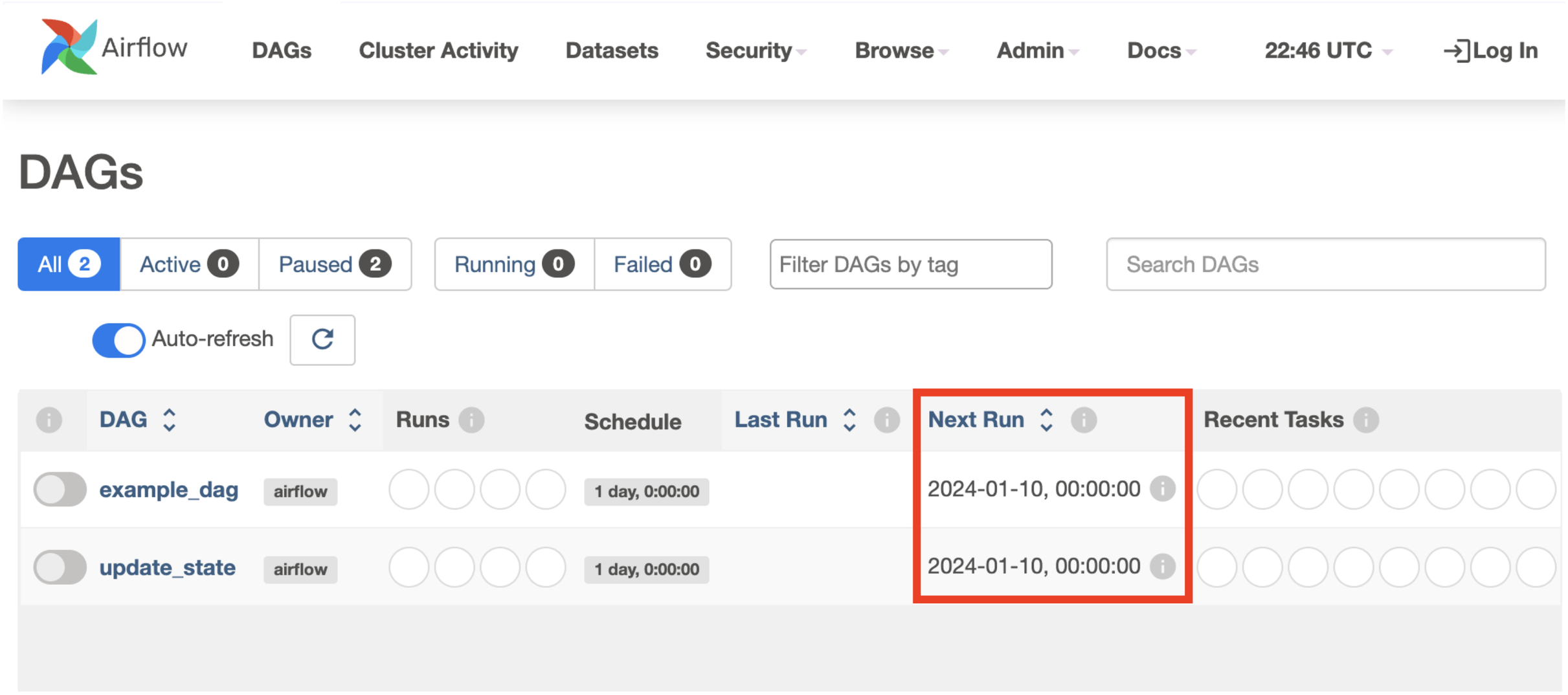
DAGs view schedule



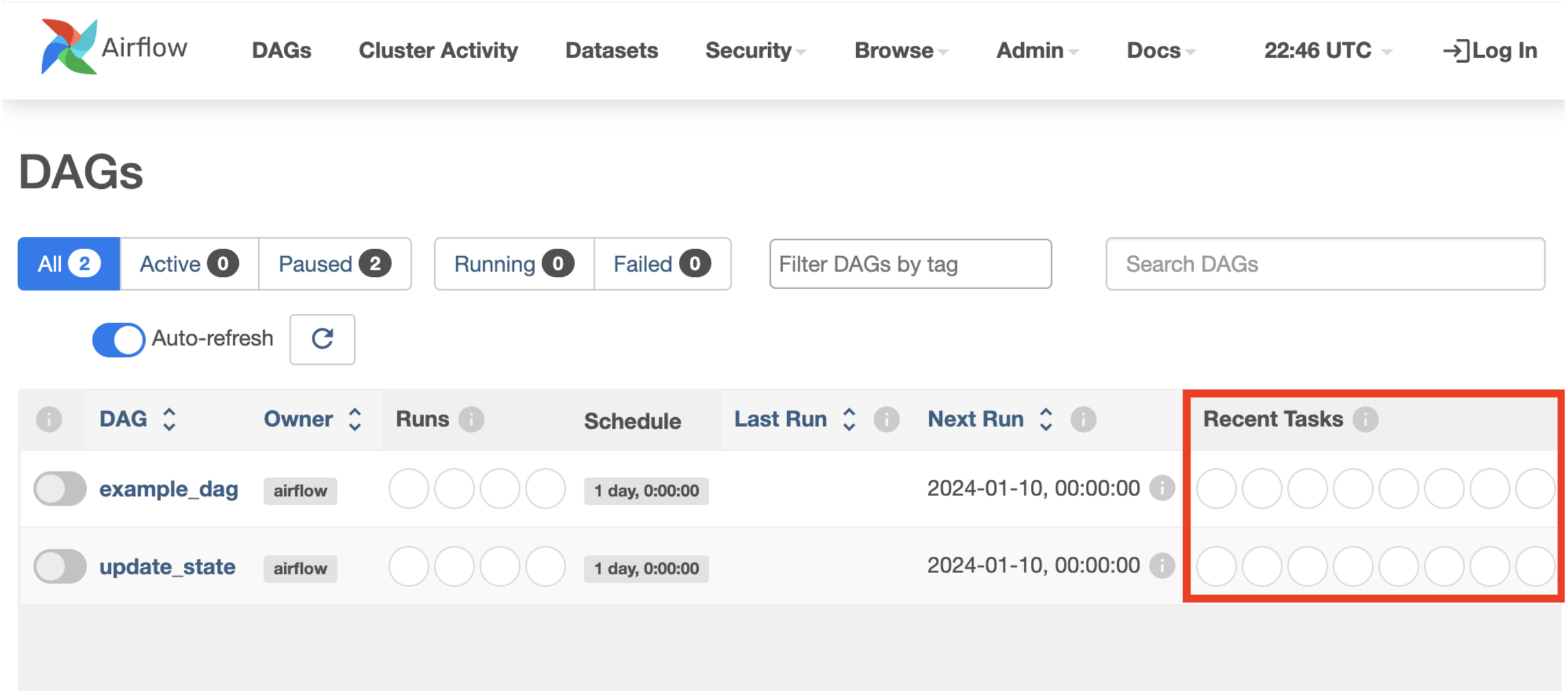
DAGs view last run



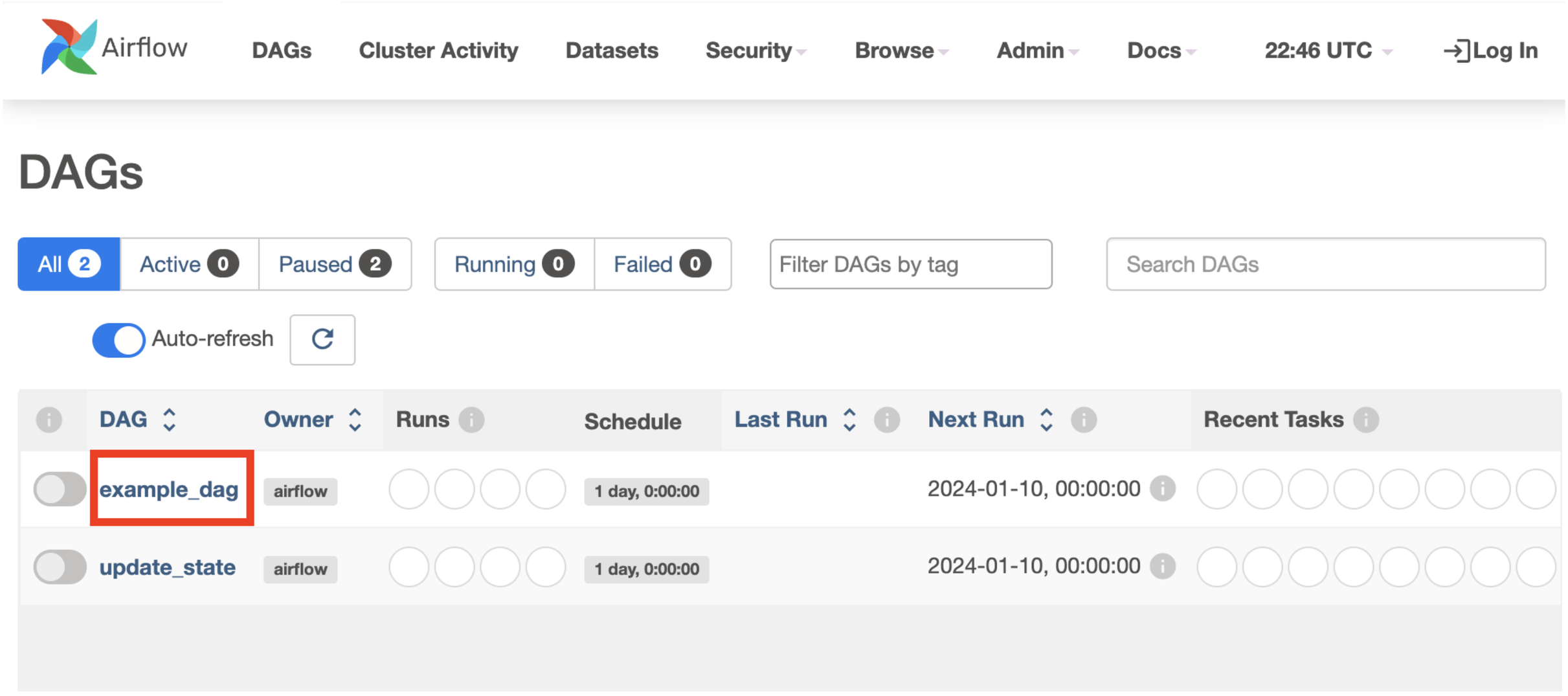
DAGs view next run



DAGs view recent tasks



DAGs view example_dag



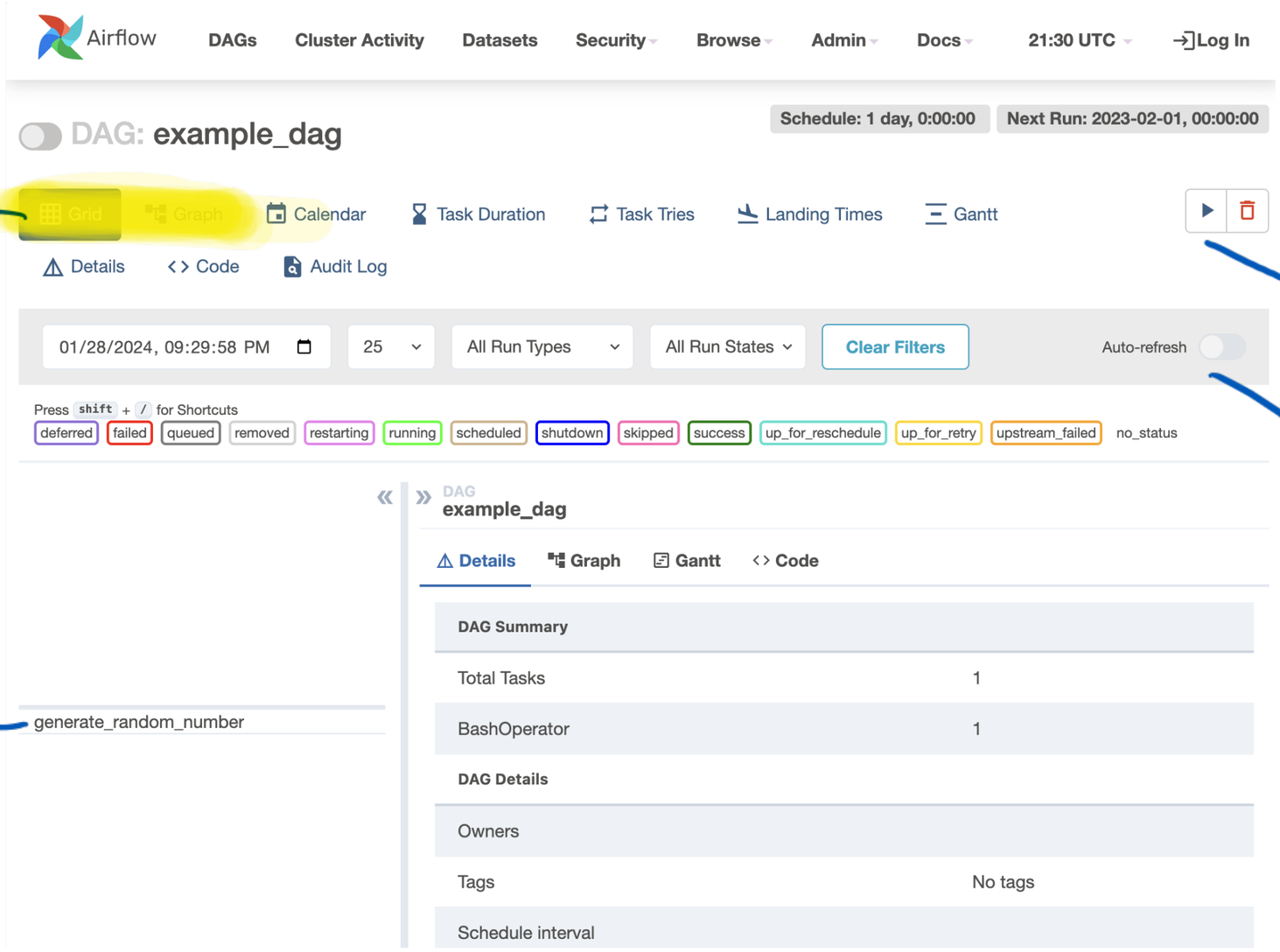
DAG detail view → info about dag itself

several views of dags are available.

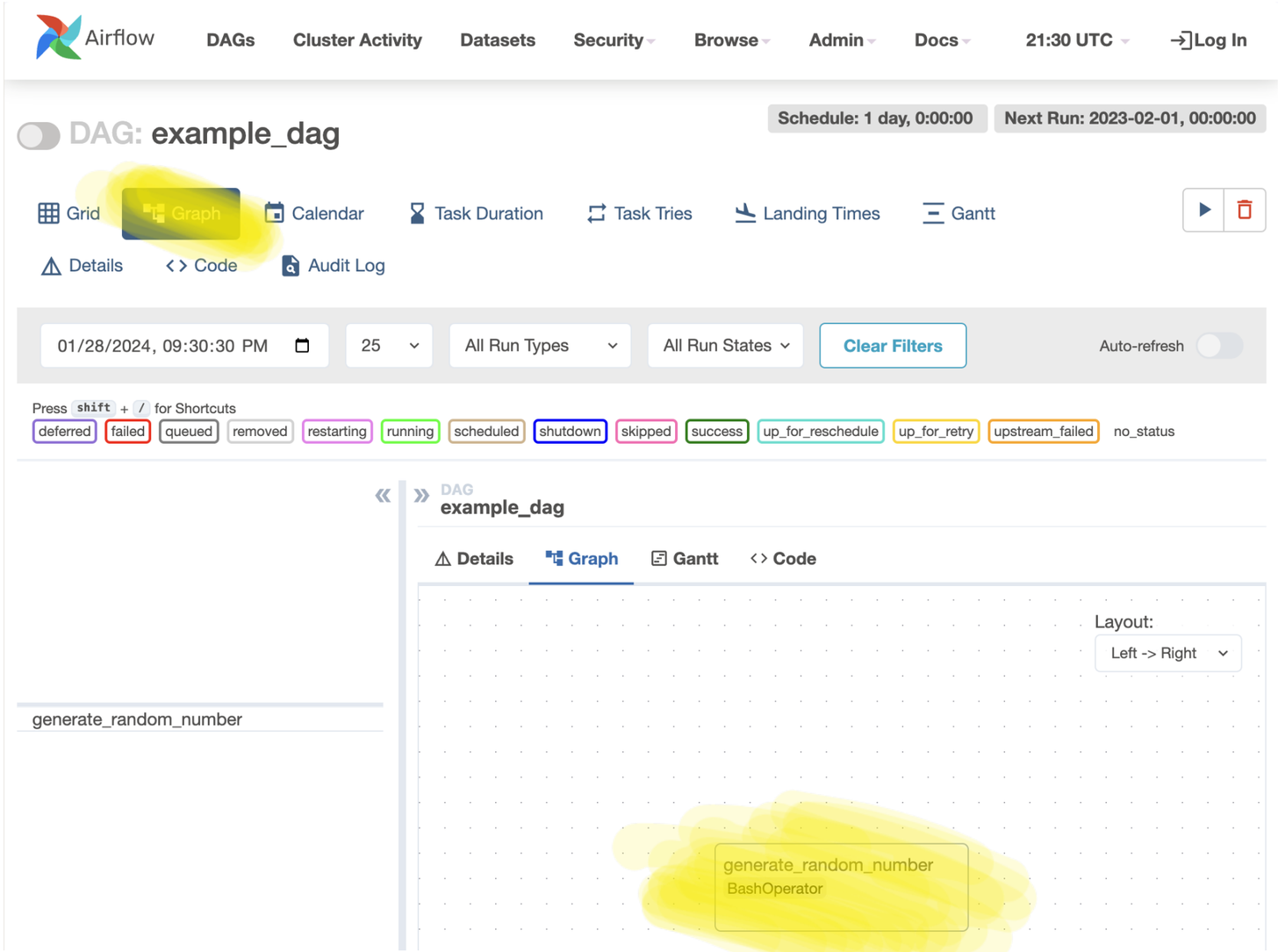
start/delete

refresh.

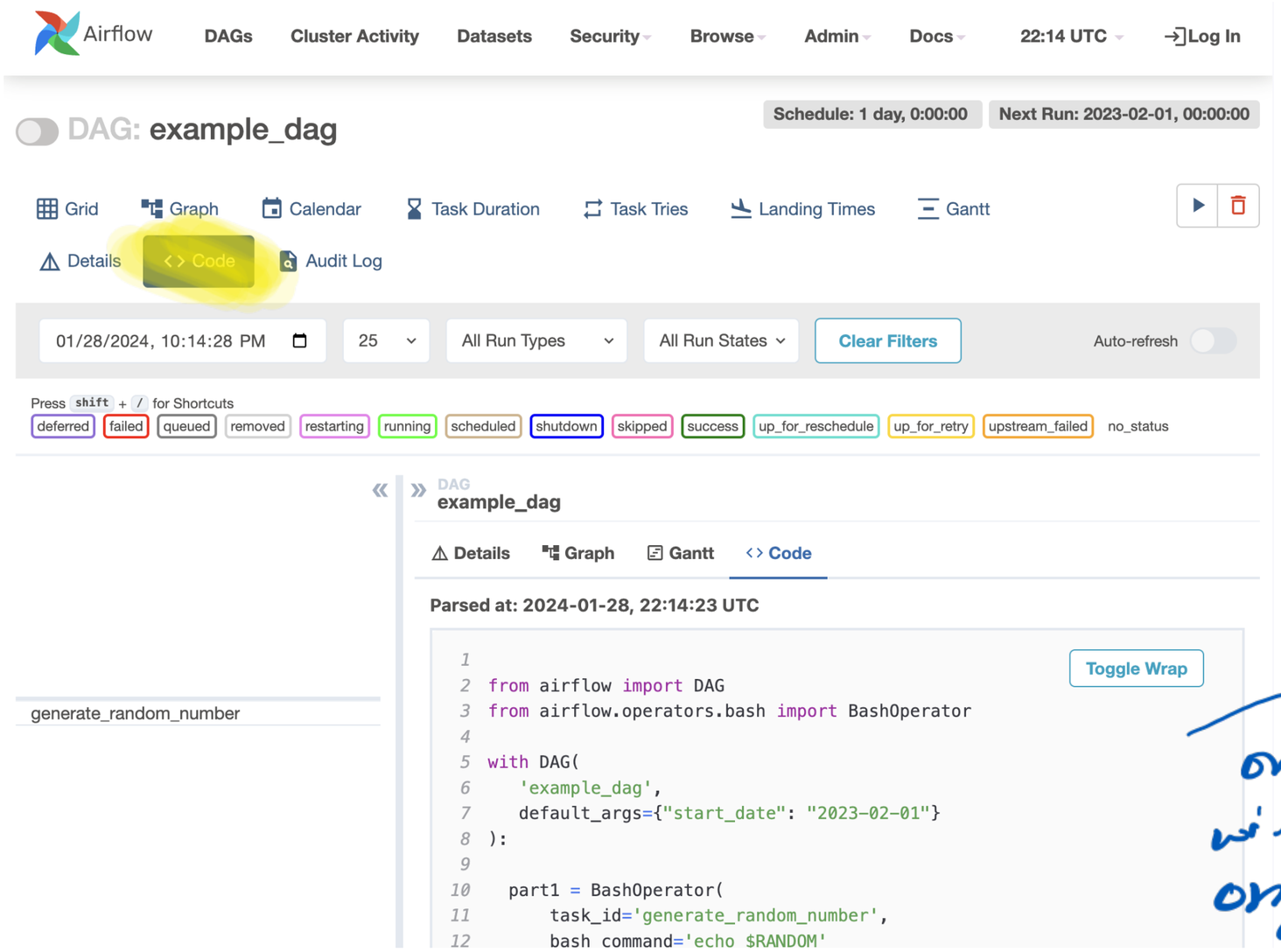
only one task



DAG graph view

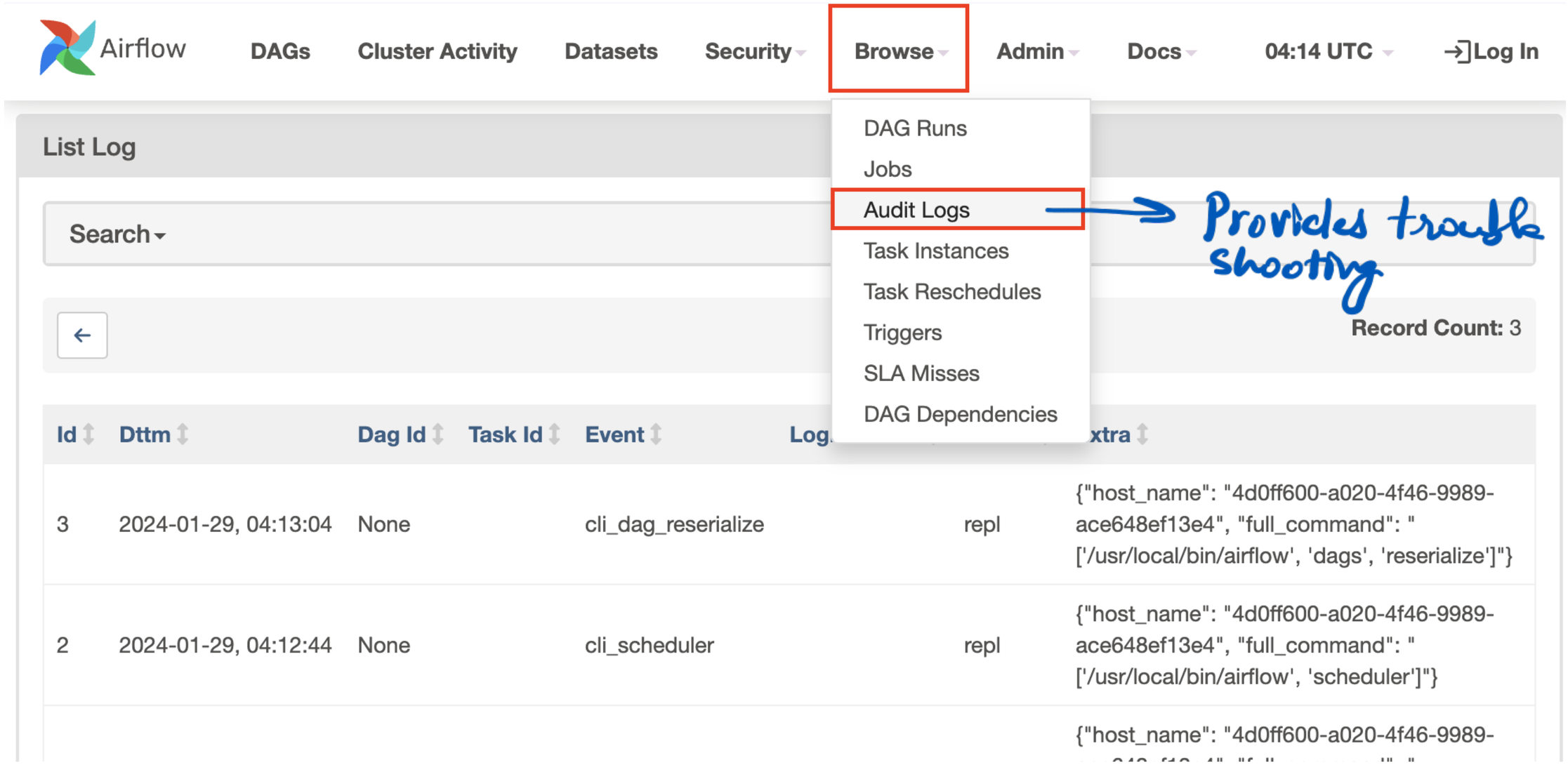


DAG code view



it is read only, any change will done through original script only

Audit logs


Airflow
DAGs
Cluster Activity
Datasets
Security
Browse
Admin
Docs
04:14 UTC
Log In
DAG Runs
Jobs
Audit Logs
Task Instances
Task Reschedules
Triggers
SLA Misses
DAG Dependencies
Provides trouble shooting
List Log
Search
Record Count: 3
Id
Dttm
Dag Id
Task Id
Event
Log
Extra
3
2024-01-29, 04:13:04
None
cli_dag_reserialize
repl
{"host_name": "4d0ff600-a020-4f46-9989-ace648ef13e4", "full_command": "['/usr/local/bin/airflow', 'dags', 'reserialize']"}
2
2024-01-29, 04:12:44
None
cli_scheduler
repl
{"host_name": "4d0ff600-a020-4f46-9989-ace648ef13e4", "full_command": "['/usr/local/bin/airflow', 'scheduler']"}
{"host_name": "4d0ff600-a020-4f46-9989-

Web UI vs command line

In most cases:

- Equally powerful depending on needs
- Web UI is easier
- Command line tool may be easier to access depending on settings

Let's practice!

INTRODUCTION TO AIRFLOW IN PYTHON