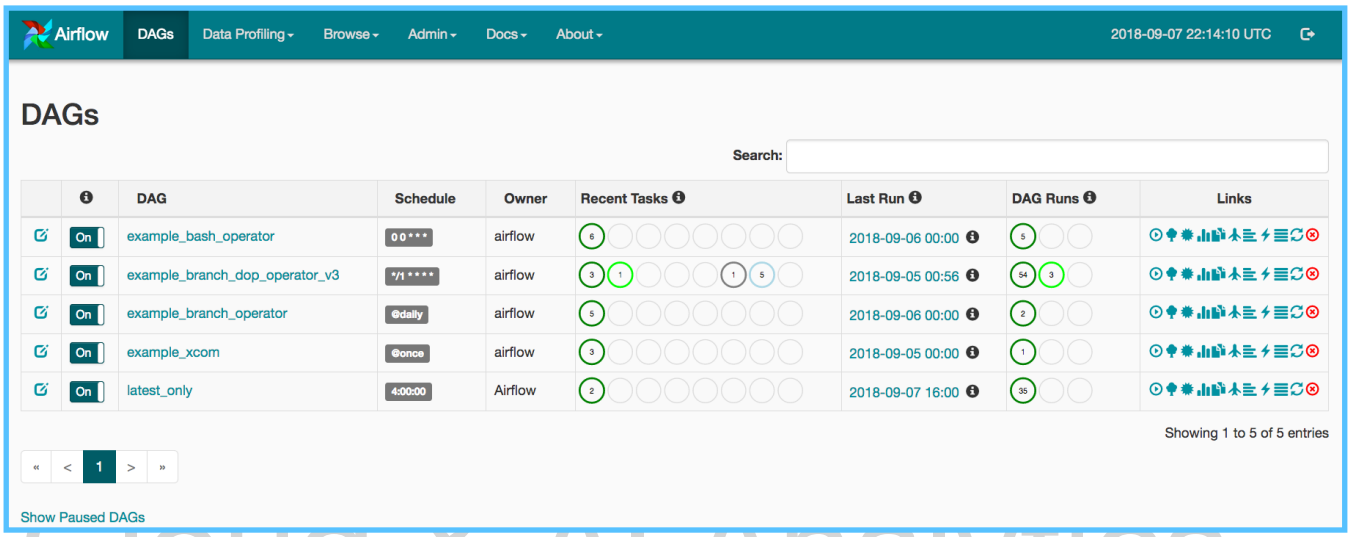
Overview on Apache Airflow

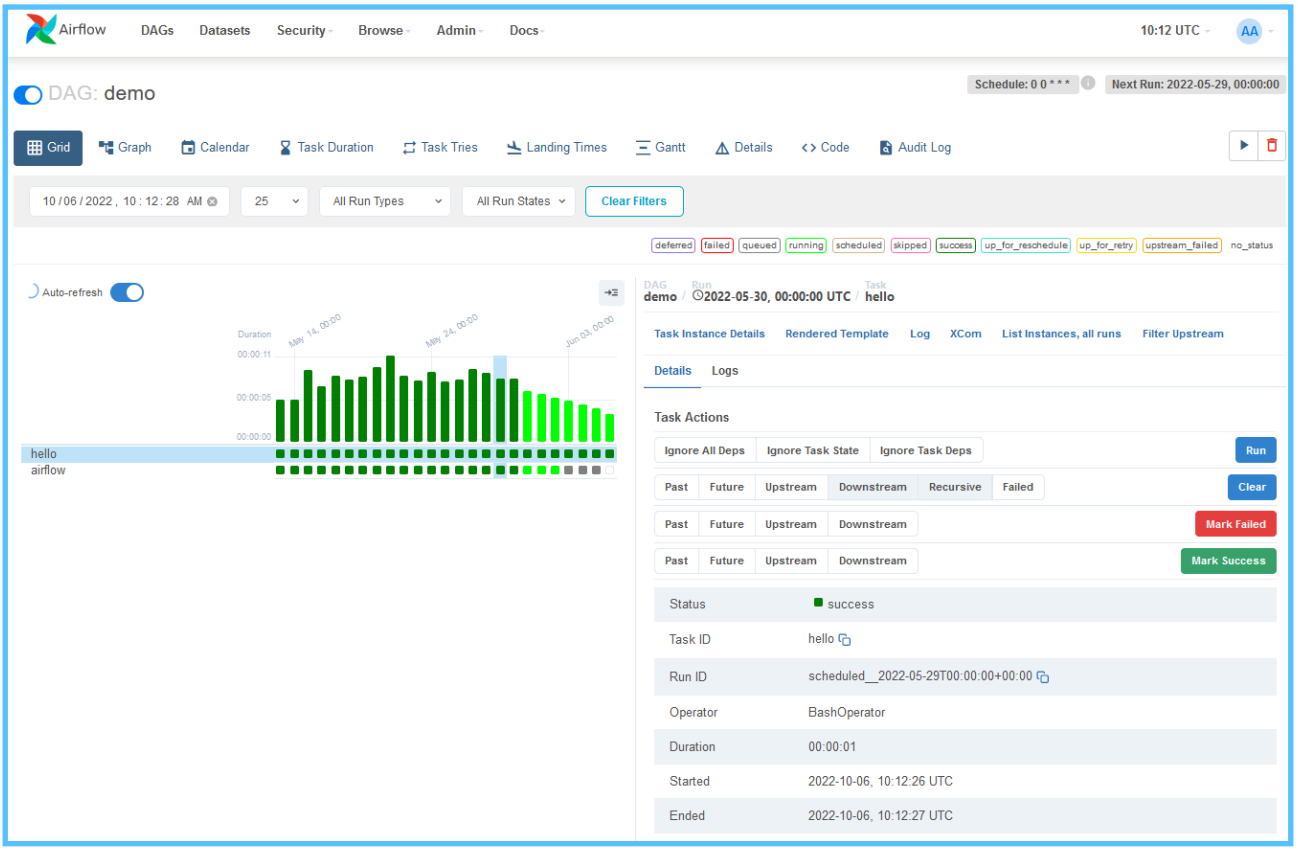
- *Airflow is an **open-source platform** for developing, scheduling, and monitoring batch-oriented workflows.*
- *It provides the **workflow management capabilities** that are integral to modern cloud-native data platforms.*
  - *It automates the execution of jobs, coordinates dependencies between tasks, and gives organizations a central point of control for monitoring and managing workflows.*
  - *It is especially used for creating and managing complex workflows — like the data pipelines that crisscross cloud and on-premises environments.*
- *Airflow is **Python-based framework** enables you to build workflows connecting with virtually any technology.*
- *Airflow is a **dynamic platform**, since anything that can be done with Python code can be done on Airflow.*



- *It provides elasticity, scalability, elegant and extensible.*
- *It provides a web interface that helps to manage the state of your workflows.*

The main characteristics of airflow are

- ***Dynamic -*** *Airflow pipelines are configuration as code (Python), allowing for dynamic pipeline generation. This allows for writing code that instantiates pipelines dynamically.*

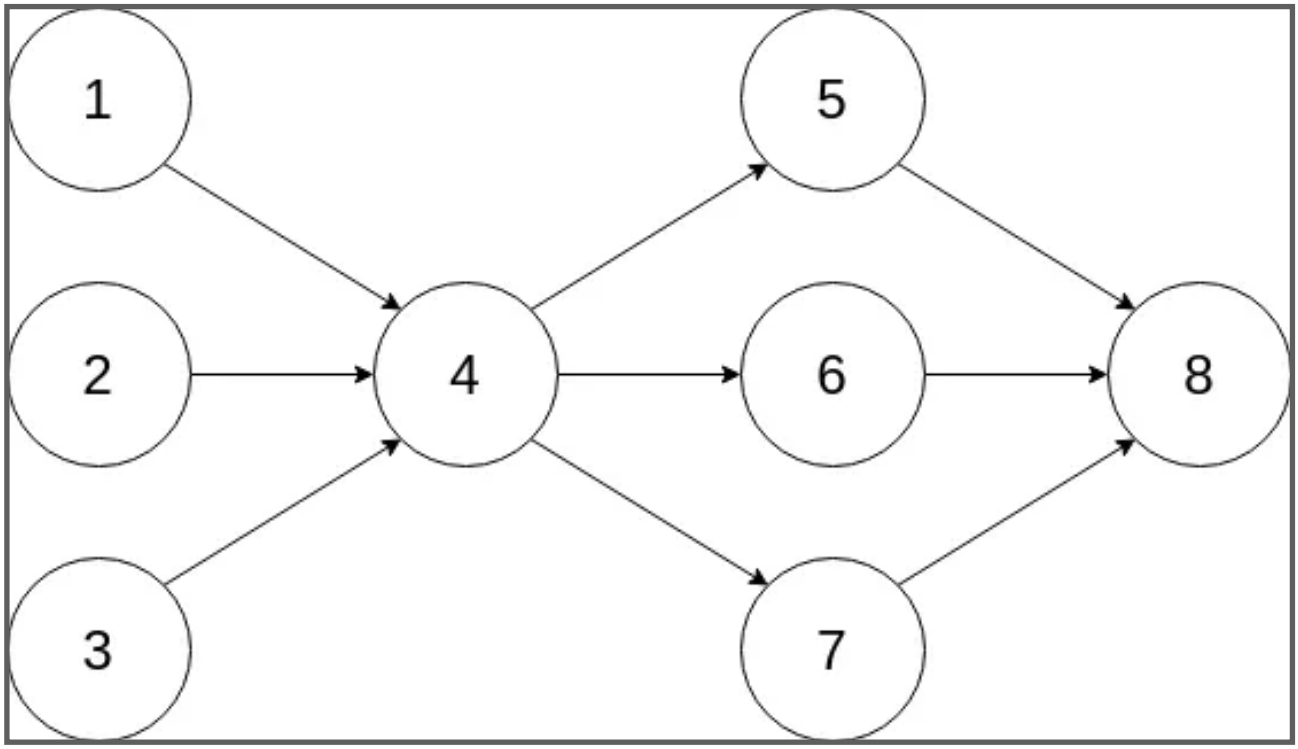


- ***Extensible -*** *Easily define your own operators, executors and extend the library so that it fits the level of abstraction that suits your environment.*
- ***Elegant -*** *Airflow pipelines are lean and explicit. Parameterizing your scripts is built into the core of Airflow using the powerful Jinja templating engine.*

- ***Scalable -*** *Airflow has a modular architecture and uses a message queue to orchestrate an arbitrary number of workers.*
- ***Flexible -*** *Workflow parameterization is built-in leveraging the Jinja templating engine.*
- *Airflow workflows are directed acyclic graphs (DAGs) of tasks.*

What is DAG(Directed Acyclic Graphs)?

- *A **directed acyclic graph (DAG)** is a conceptual representation of a series of activities.*



- *DAGs are often used to visually represent the relationships between your data models and dependencies between different events.*
- *Directed Acyclic Graph has two important features:*
  - ***<u>Directed Edges</u>*** *In Directed Acyclic Graph, each edge has a direction, meaning it goes from one vertex (node) to another.*

This direction signifies a one-way relationship or dependency between nodes.

- ***Acyclic*** *The term "acyclic" indicates that there are no cycles or closed loops within the graph. In other words, you cannot traverse a sequence of directed edges and return to the same node, following the edge directions.*
- *Formation of cycles is prohibited in DAG.*

- *The order of the activities is depicted by a graph, which is visually presented as a set of circles, each representing an activity, some of which are connected by lines, representing the flow from one activity to another.*

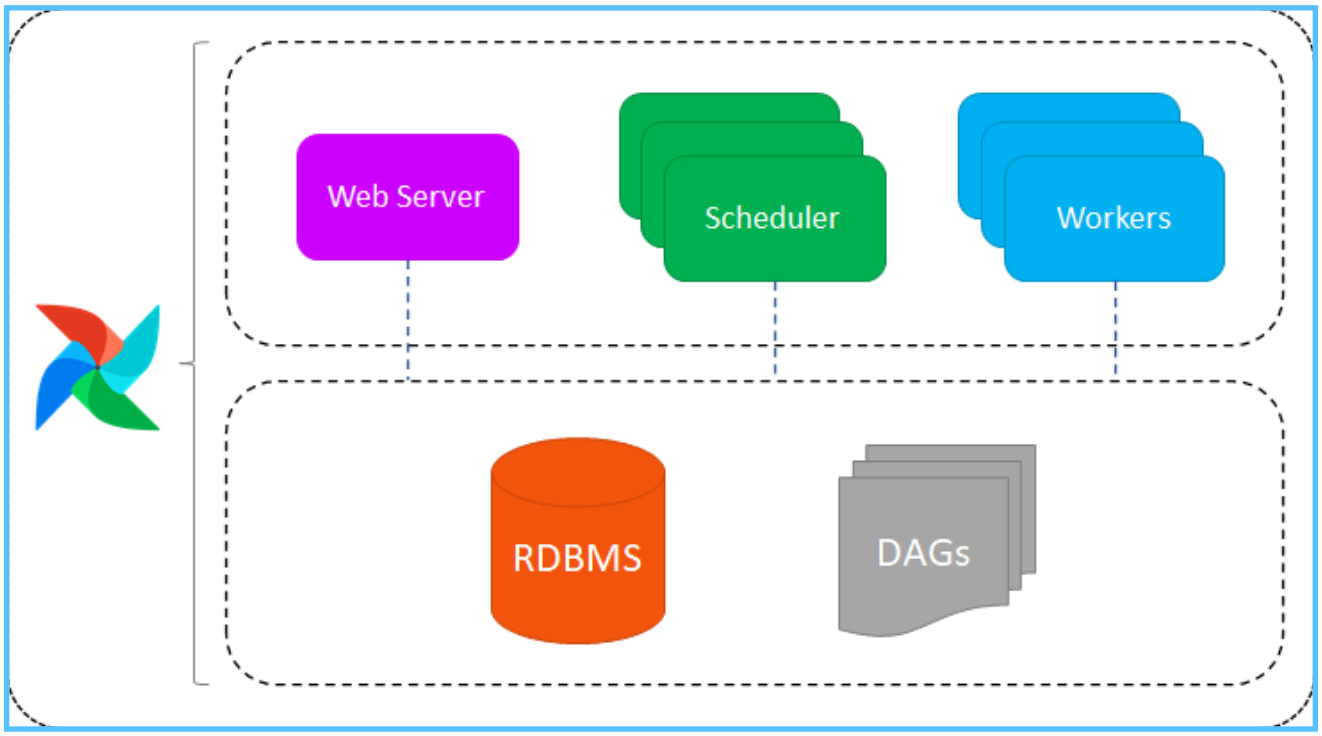
Features of Apache Airflow

- **Easy to Use**
  - *If you have a bit of python knowledge, you are good to go and deploy on Airflow.*
- **Open Source**
  - *It is free and open-source with a lot of active users.*
- **Robust Integrations**
  - *It will give you ready to use operators so that you can work with Google Cloud Platform, Amazon AWS, Microsoft Azure, etc.*
- **Use Standard Python to code**
  - *You can use python to create simple to complex workflows with complete flexibility.*
- **Amazing User Interface**

- *You can monitor and manage your workflows.*
- *It will allow you to check the status of completed and ongoing tasks.*

- **Deferrable Operators**
  - *Accommodate long-running tasks with deferrable operators and triggers that run tasks asynchronously, making more efficient use of resources.*

Components of Apache Airflow



- **DAG**
  - It is the Directed Acyclic Graph – a collection of all the tasks that you want to run which is organized and shows the relationship between different tasks.
  - It is defined in a python script.
- **Web Server**

  - It is the user interface built on the Flask.
  - It allows us to monitor the status of the DAGs and trigger them.
- **Metadata Database**
  - Airflow stores the status of all the tasks in a database and do all read/write operations of a workflow from here.
- **Scheduler**
  - As the name suggests, this component is responsible for scheduling the execution of DAGs.
  - It retrieves and updates the status of the task in the database.
- **Tasks**
  - Each node in a DAG represents a task.
  - It is a representation of a sequence of tasks to be performed, which constitutes a pipeline.
  - The represented jobs are defined by the operators.
- **Operators**
  - The operators are the building blocks of the Airflow platform.
  - They are used to determine the work done.
  - It can be an individual task (node of a DAG), defining how the task will be executed.
  - The DAG ensures that the operators are scheduled and executed in a specific order, while the operators define the jobs to be executed at each step of the process.
- **Hooks**

    - On Airflow, Hooks allow interfacing with third-party systems.
    - They allow the connection between APIs and external databases like Hive, S3, GCS, MySQL, and Postgres.
- **Plugins**
    - Airflow plugins can be described as a combination of Hooks and Operators.
    - They are used to accomplish specific tasks involving an external application.
- **Connections**
    - Connections allow Airflow to store information, allowing it to connect to external systems such as API credentials or tokens.
    - They are managed directly from the platform's user interface.
    - The data is encrypted and stored as metadata in a Postgres or MySQL database.

When is Airflow used for?

- Airflow can be used for any **batch data pipeline**, so its use cases are as numerous as they are diverse.
    - Due to its scalability, this platform particularly excels at orchestrating tasks with complex dependencies on multiple external systems.
- By writing pipelines in code and using the various plugins available, it is possible to integrate Airflow with any dependent systems from a unified platform for orchestration and monitoring.

Benefits of Apache Airflow

- **Ease of use**—you only need a little python knowledge to get started.
- **Open-source community**—Airflow is free and has a large community of active users.
- **Integrations**—ready-to-use operators allow you to integrate Airflow with cloud platforms (Google, AWS, Azure, etc).
- **Coding with standard Python—**you can create flexible workflows using Python with no knowledge of additional technologies or frameworks.
- **Graphical UI**—monitor and manage workflows, check the status of ongoing and completed tasks.