ASSIGNMENT – 2 (Apache Airflow)

TRAINEE NAME: Swathi Baskaran

Apache Airflow is an open-source workflow orchestration tool. It helps us to define, schedule and monitor workflows as code. This was first developed by Airbnb. It later became a part of Apache.

- Apache Airflow code is written in Python, which is easy to read and also supports version control.
- Minimal knowledge of Python comprising the basics will be sufficient to handle Airflow.
- Defines tasks and dependencies using DAGs.
- It has a web UI to track runs, check logs, and retry failed tasks.
- It has a large library of operators/hooks.
- It can be integrated with third party applications such as Spark, Hadoop, Snowflake, Databricks, GCP, AWS, and Azure.

DAG (Directed Acyclic Graphs) are used to represent the workflow. DAGs tells Apache Airflow what tasks to run, when to run them, and in what order. DAGs contain tasks and dependencies between them.

Airflow is used for several purposes. A few of them are:

- Scheduling ETL pipelines that extract data from multiple sources, and run Spark jobs or other data transformations.
- Machine learning model training
- Automated generation of reports
- Backups and other Devops tasks

Airflow is most commonly used to automate machine learning tasks.

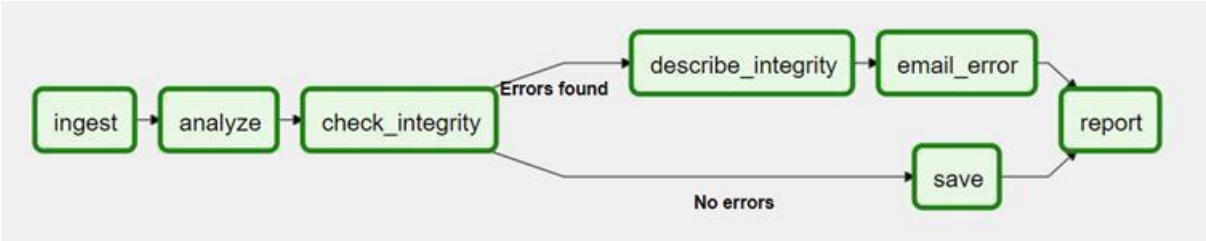
The DAG runs through a series of Tasks, which may be subclasses of Airflow's BaseOperator, including:

- **Operator:** Predefined tasks that can be strung together quickly
- **Sensors:** A type of operator that waits for external events to occur

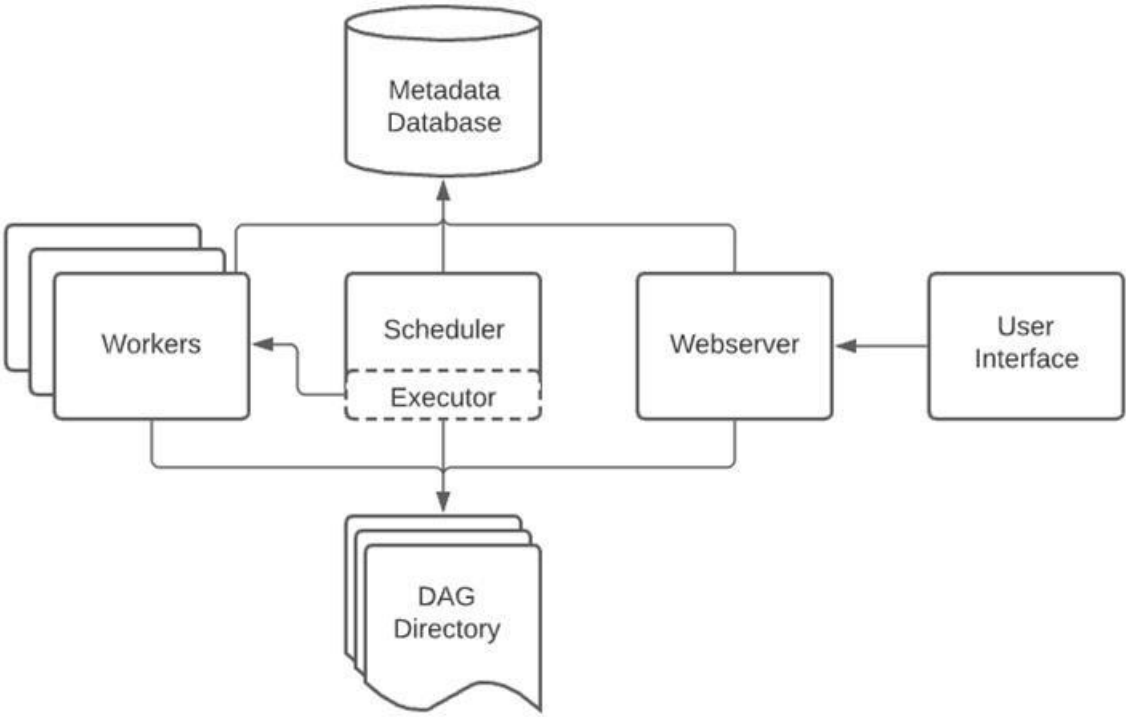
- **TaskFlow:** A custom Python function packaged as a task, which is decorated with @tasks

Operators are the building blocks of Apache Airflow, as they define how the tasks run and what they do.

Airflow Architecture:



Other typical components of an Airflow architecture include a database to store state metadata, a web server used to inspect and debug Tasks and DAGs, and a folder containing the DAG files.



- The **DAG Directory** contains all the DAGs and tasks that have to be performed.
- **Scheduler**, which acts as the brain reads the DAGs from the DAG Directory at regular intervals and schedules the tasks. It is the component that decides when the tasks should be run. When the time for the task has come, it passes it to the executor.
- The **executor** works along with the scheduler. This component decides where the task should run. It assigns the tasks to the workers accordingly.
- The **workers** are the components that perform the task that is being assigned to them by the executor.
- The **Metadata database** contains all the logs, transactions and the metadata of the system. When a scheduler reads a DAG from the DAG Directory, it is updated in the Metadata database. When a task is completed, it is also updated in the Metadata database. To know about the state of the system, it is sufficient to check the Metadata database.
- The **web server** contains all the details of the system including the tasks that are scheduled, running, completed and so on.
- The **User Interface (UI)** is what enables the user to know about the state of the system and their tasks.