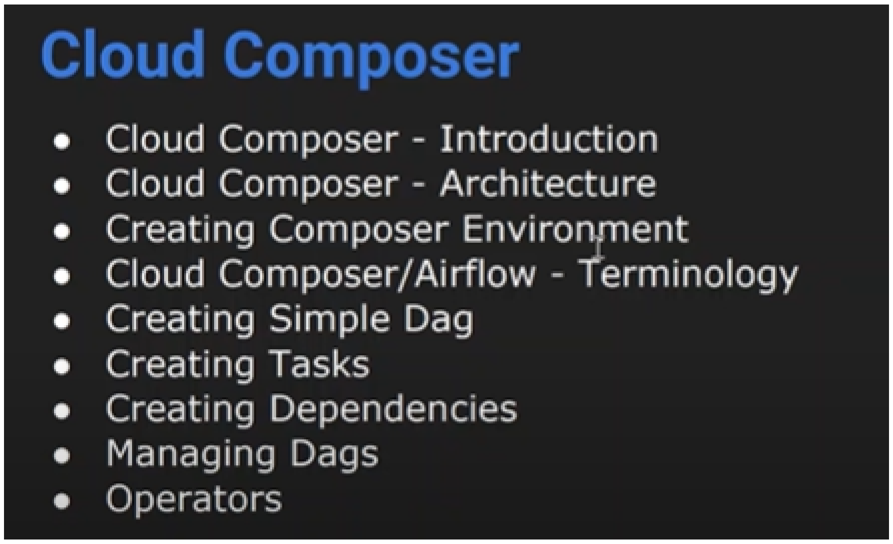
Create Cloud Composer - Google CLoud

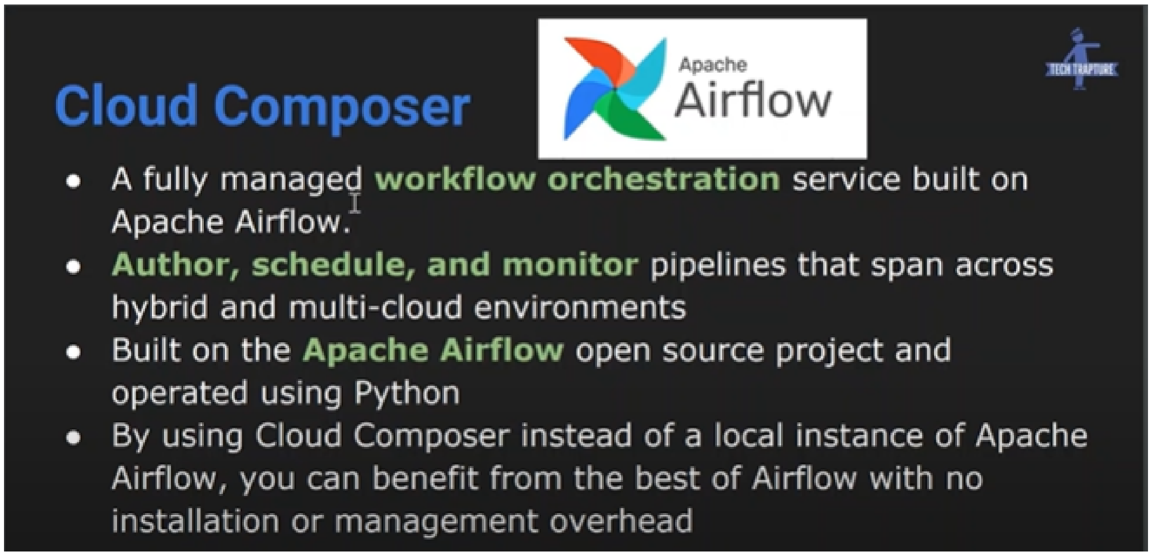
1. We'll learn



Cloud Composer

- Cloud Composer - Introduction
- Cloud Composer - Architecture
- Creating Composer Environment
- Cloud Composer/Airflow - Terminology
- Creating Simple Dag
- Creating Tasks
- Creating Dependencies
- Managing Dags
- Operators

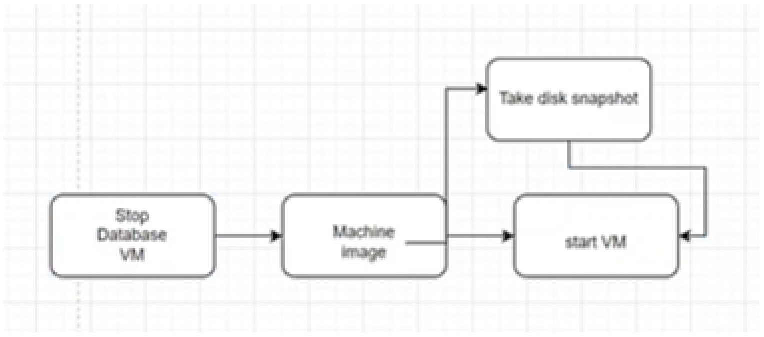
2. What is a cloud composer?



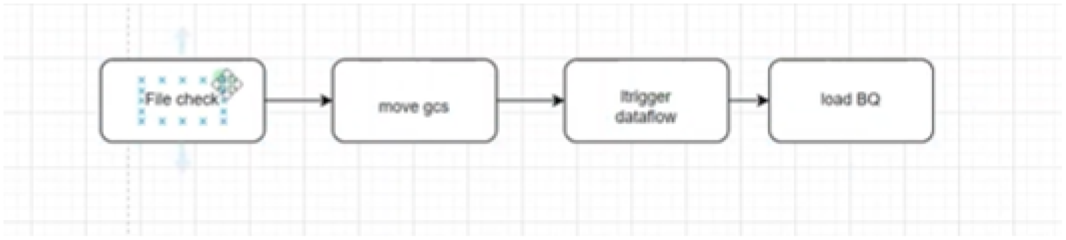
Cloud Composer

- A fully managed **workflow orchestration** service built on Apache Airflow.
- **Author, schedule, and monitor** pipelines that span across hybrid and multi-cloud environments
- Built on the **Apache Airflow** open source project and operated using Python
- By using Cloud Composer instead of a local instance of Apache Airflow, you can benefit from the best of Airflow with no installation or management overhead

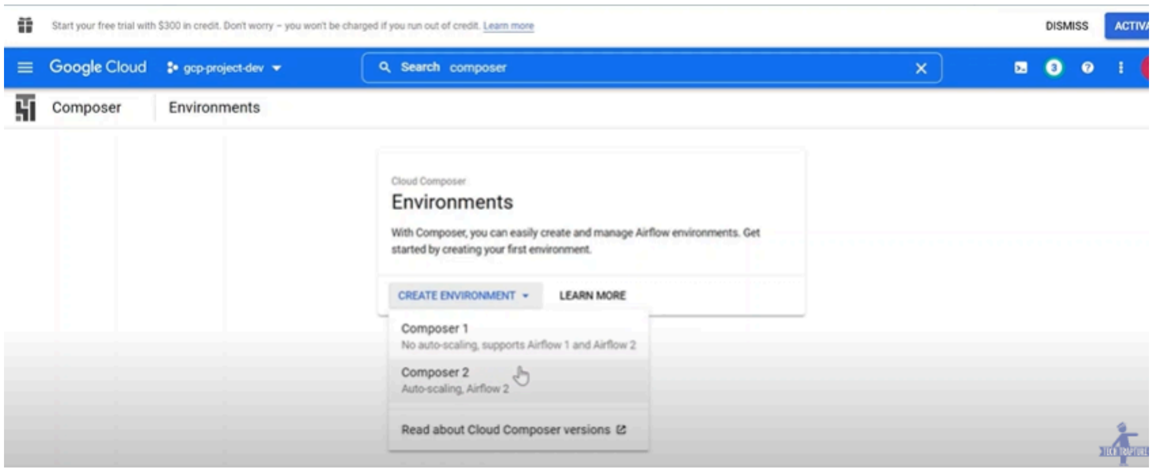
3. So if we want to perform multiple tasks…and if we want to add dependencies bw them…then airflow will helps us to add dependencies



4. Another example…



here if we have a file…then we move to GCS…and after that dataflow triggers and copies the file to bigquery
5. Lets see how can we create CC
6. First we have to enable CLoud composer API…and it is better to create composer using a new service account…and we give editor role
7. We can create service account by going into IAM and Admin roles
8. Now inside the composer…we will go ahead with composer2 (thru service account)


Start your free trial with $300 in credit. Don't worry – you won't be charged if you run out of credit. Learn more
DISMISS
ACTIVA
Google Cloud
gcp-project-dev
Search composer
Composer
Environments

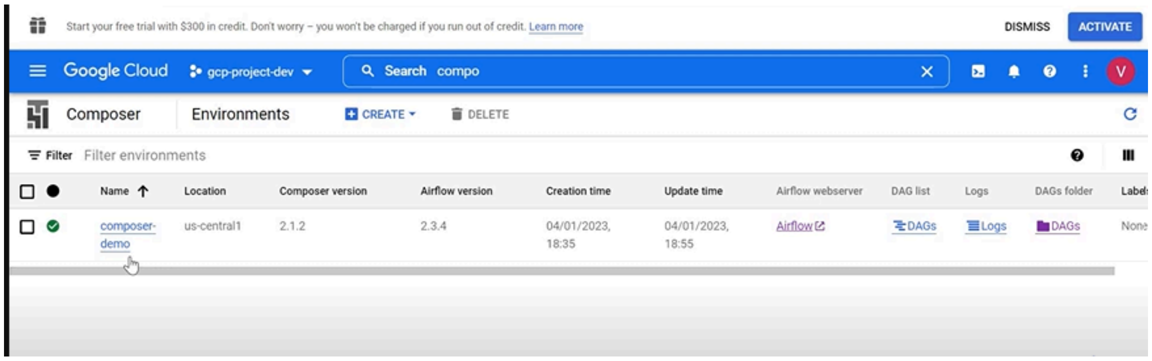
Cloud Composer

Environments

With Composer, you can easily create and manage Airflow environments. Get started by creating your first environment.

CREATE ENVIRONMENT
LEARN MORE
Composer 1
No auto-scaling, supports Airflow 1 and Airflow 2
Composer 2
Auto-scaling, Airflow 2
Read about Cloud Composer versions

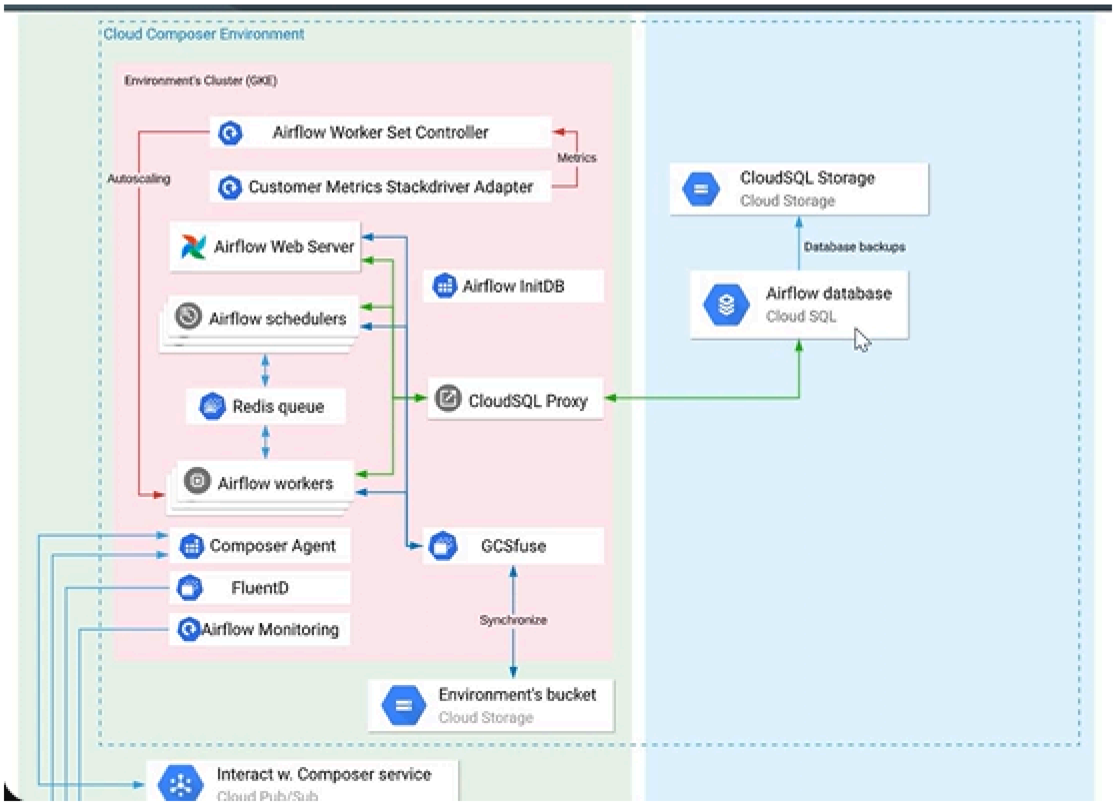
9. Now we create rhe composer with minimum specification and give our service account…and we create it….it will take 15 mins to complete

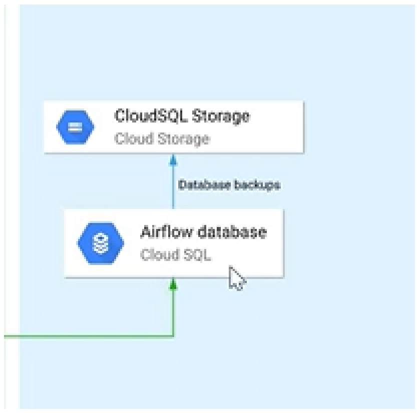

Start your free trial with $300 in credit. Don't worry – you won't be charged if you run out of credit. Learn more
DISMISS
ACTIVATE
Google Cloud
gcp-project-dev
Search compo
Composer
Environments
CREATE
DELETE
Filter
Filter environments

Name	Location	Composer version	Airflow version	Creation time	Update time	Airflow webserver	DAG list	Logs	DAGs folder	Label
composer-demo	us-central1	2.1.2	2.3.4	04/01/2023, 18:35	04/01/2023, 18:55	Airflow	DAGs	Logs	DAGs	None

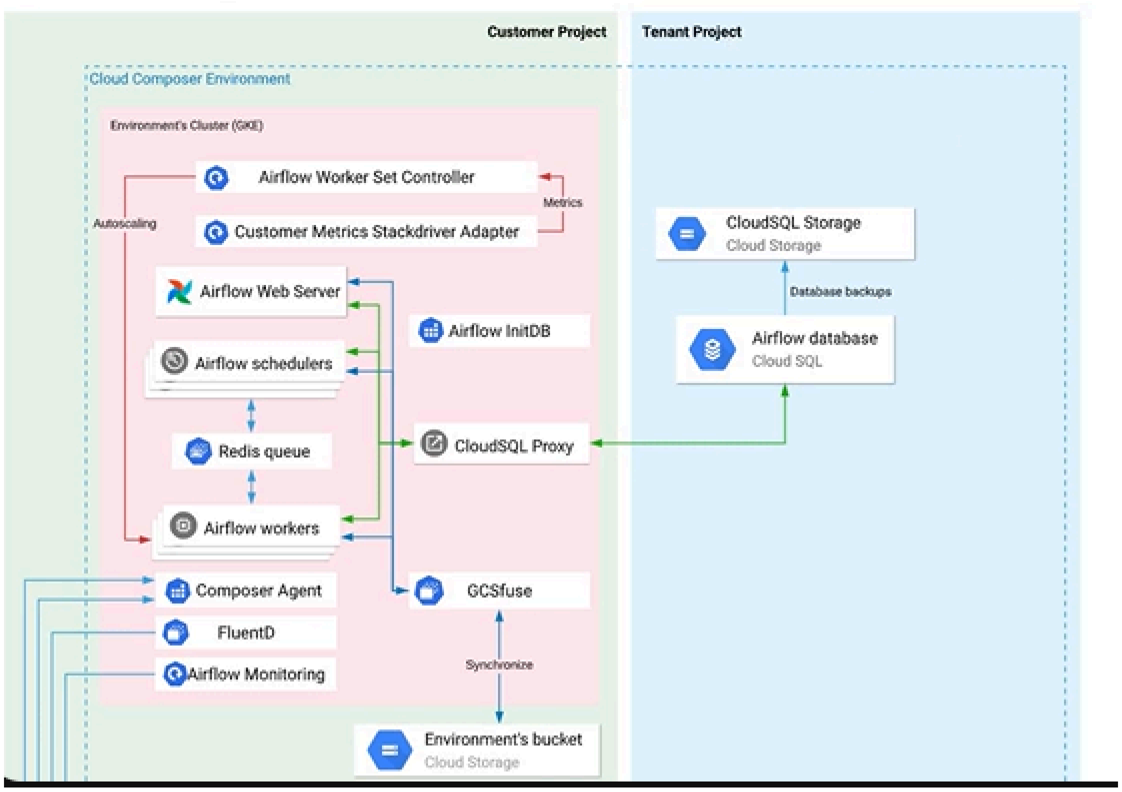
Cloud Composer Architecture

1. When ever we create composer…it create many services backend

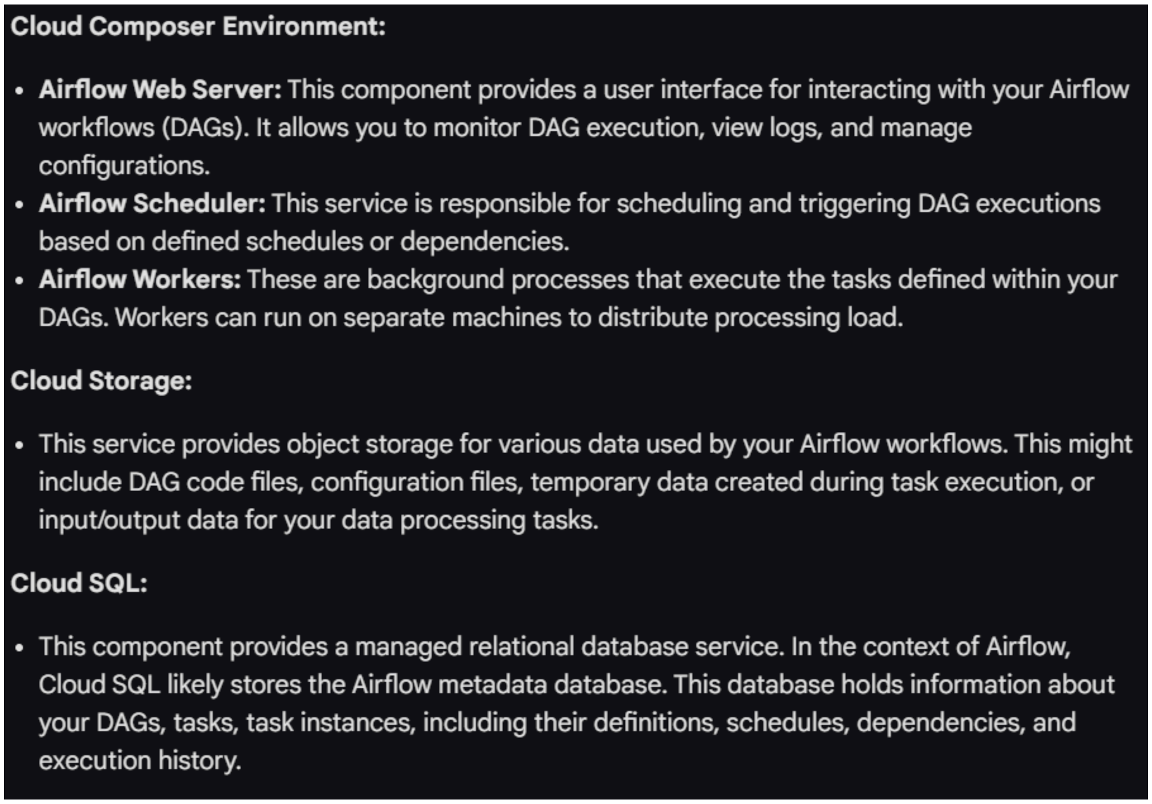


2. Here we have  ..so whenever we create a composer..it create a DB which is cloud SQL instance for this composer

3. These 2 services are created by tenant project …which is managed by google..so we dont get to see them



4. Cloud SQL proxy connects the customer project with Airflow DB



Cloud Composer Environment:

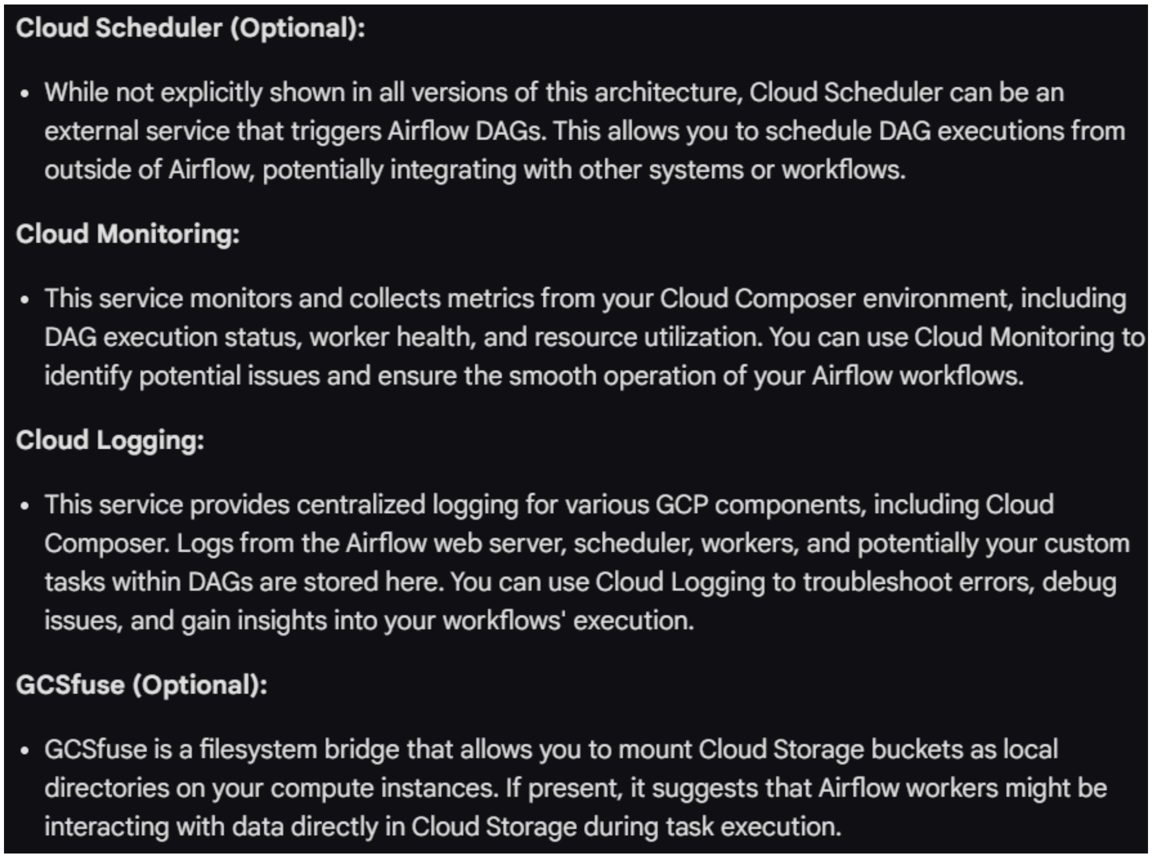
- **Airflow Web Server:** This component provides a user interface for interacting with your Airflow workflows (DAGs). It allows you to monitor DAG execution, view logs, and manage configurations.
- **Airflow Scheduler:** This service is responsible for scheduling and triggering DAG executions based on defined schedules or dependencies.
- **Airflow Workers:** These are background processes that execute the tasks defined within your DAGs. Workers can run on separate machines to distribute processing load.

Cloud Storage:

- This service provides object storage for various data used by your Airflow workflows. This might include DAG code files, configuration files, temporary data created during task execution, or input/output data for your data processing tasks.

Cloud SQL:

- This component provides a managed relational database service. In the context of Airflow, Cloud SQL likely stores the Airflow metadata database. This database holds information about your DAGs, tasks, task instances, including their definitions, schedules, dependencies, and execution history.


Cloud Scheduler (Optional):

- While not explicitly shown in all versions of this architecture, Cloud Scheduler can be an external service that triggers Airflow DAGs. This allows you to schedule DAG executions from outside of Airflow, potentially integrating with other systems or workflows.

Cloud Monitoring:

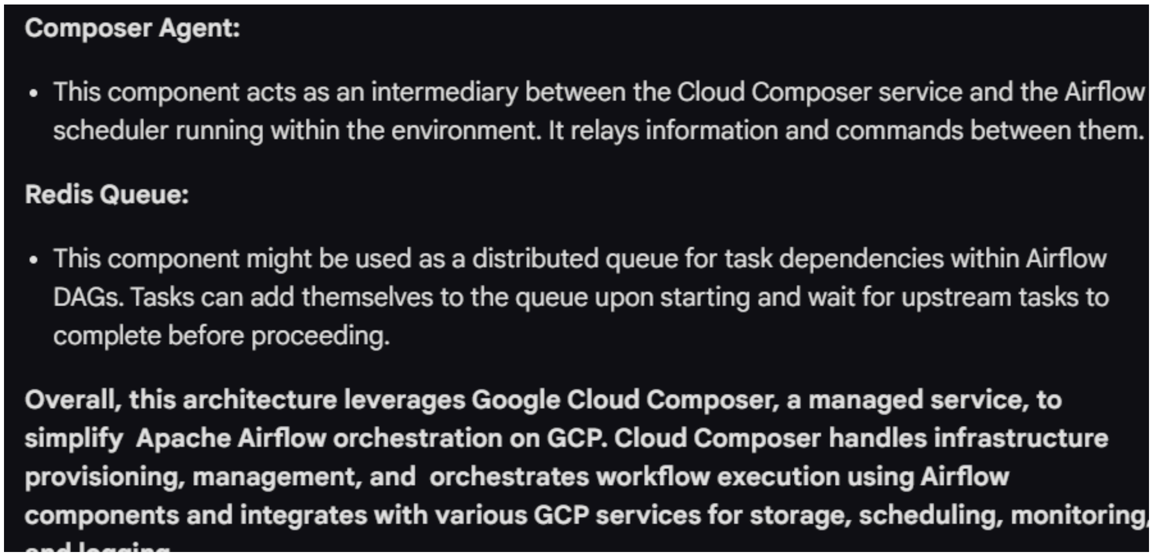
- This service monitors and collects metrics from your Cloud Composer environment, including DAG execution status, worker health, and resource utilization. You can use Cloud Monitoring to identify potential issues and ensure the smooth operation of your Airflow workflows.

Cloud Logging:

- This service provides centralized logging for various GCP components, including Cloud Composer. Logs from the Airflow web server, scheduler, workers, and potentially your custom tasks within DAGs are stored here. You can use Cloud Logging to troubleshoot errors, debug issues, and gain insights into your workflows' execution.

GCSfuse (Optional):

- GCSfuse is a filesystem bridge that allows you to mount Cloud Storage buckets as local directories on your compute instances. If present, it suggests that Airflow workers might be interacting with data directly in Cloud Storage during task execution.


Composer Agent:

- This component acts as an intermediary between the Cloud Composer service and the Airflow scheduler running within the environment. It relays information and commands between them.

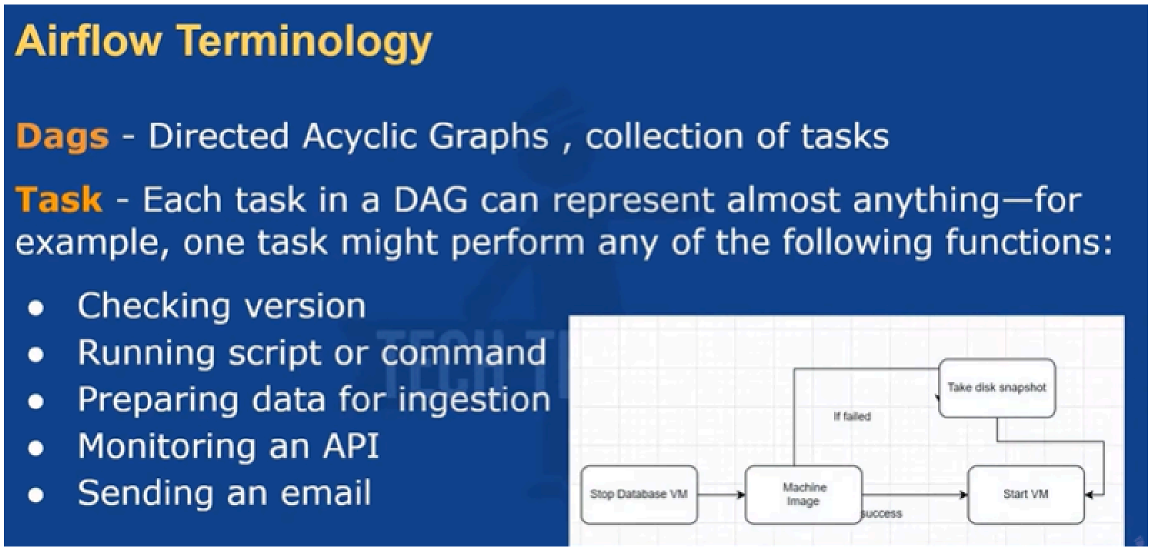
Redis Queue:

- This component might be used as a distributed queue for task dependencies within Airflow DAGs. Tasks can add themselves to the queue upon starting and wait for upstream tasks to complete before proceeding.

Overall, this architecture leverages Google Cloud Composer, a managed service, to simplify Apache Airflow orchestration on GCP. Cloud Composer handles infrastructure provisioning, management, and orchestrates workflow execution using Airflow components and integrates with various GCP services for storage, scheduling, monitoring, and logging

Create first DAG in Cloud Composer

1. Airflow Terminolgy



Airflow Terminology

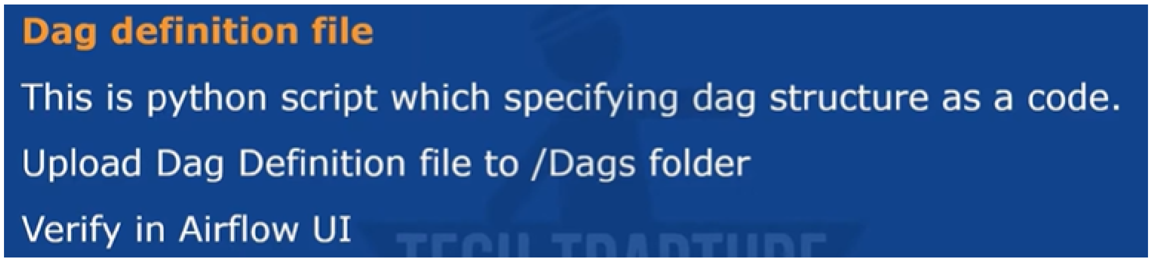
Dags - Directed Acyclic Graphs , collection of tasks

Task - Each task in a DAG can represent almost anything—for example, one task might perform any of the following functions:

- Checking version
- Running script or command
- Preparing data for ingestion
- Monitoring an API
- Sending an email

Take disk snapshot
If failed
Stop Database VM
Machine Image
success
Start VM

2. What we need to create a DAG?
   We need DAG definition file..which is a python script



Dag definition file

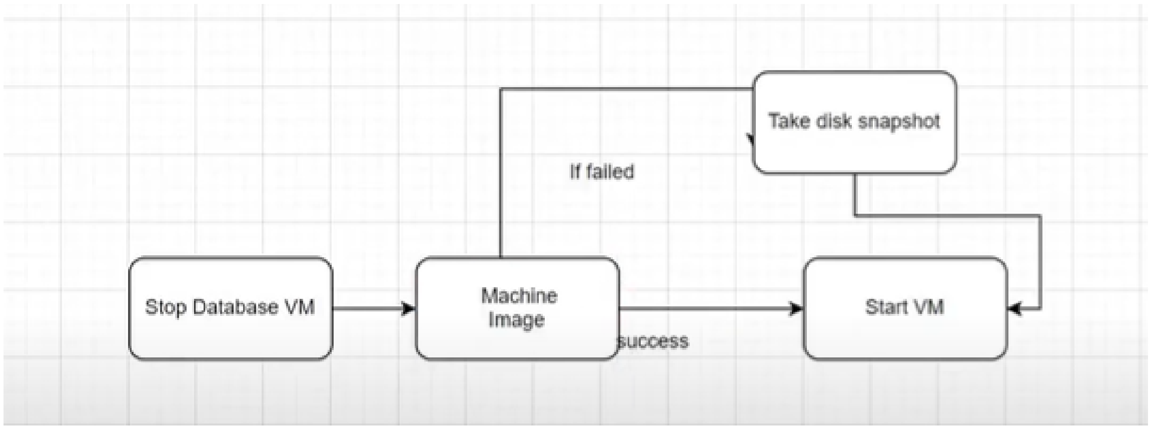
This is python script which specifying dag structure as a code.

Upload Dag Definition file to /Dags folder

Verify in Airflow UI

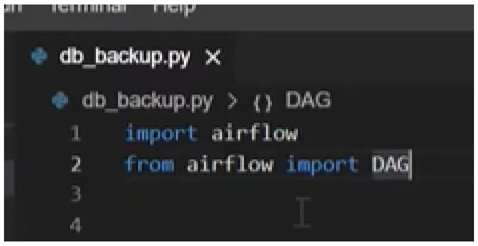
   If our syntax is correct..then it will appear in airflow UI

3. Lets see how we can create our first dag
4. Here we will design this airflow

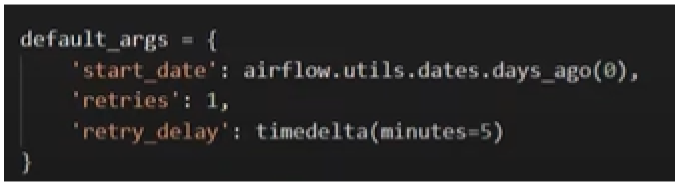


5. First we will create a python script in cloud shell

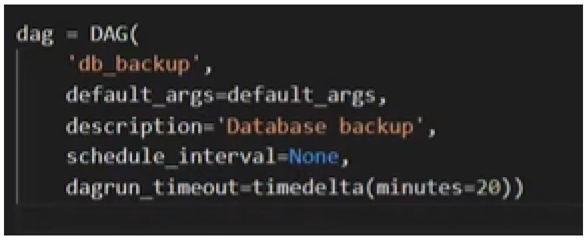
6. Whenever we are writing a dag file…we need to import the required libraries



```
import airflow
from airflow import DAG
```

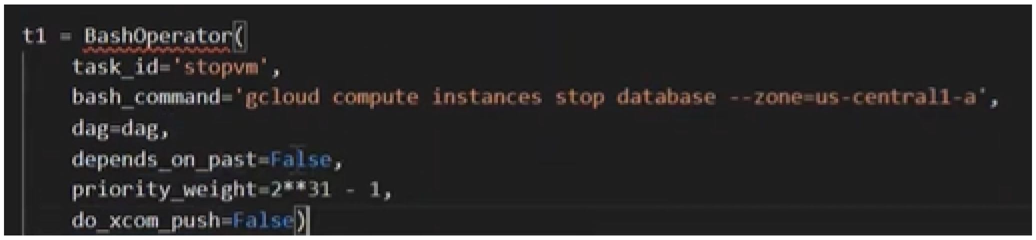
7. Next we have to give default argos 

```
default_args = {
    'start_date': airflow.utils.dates.days_ago(0),
    'retries': 1,
    'retry_delay': timedelta(minutes=5)
}
```

8. To initialize the dag we use 

```
dag = DAG(
    'db_backup',
    default_args=default_args,
    description='Database backup',
    schedule_interval=None,
    dagrun_timeout=timedelta(minutes=20))
```

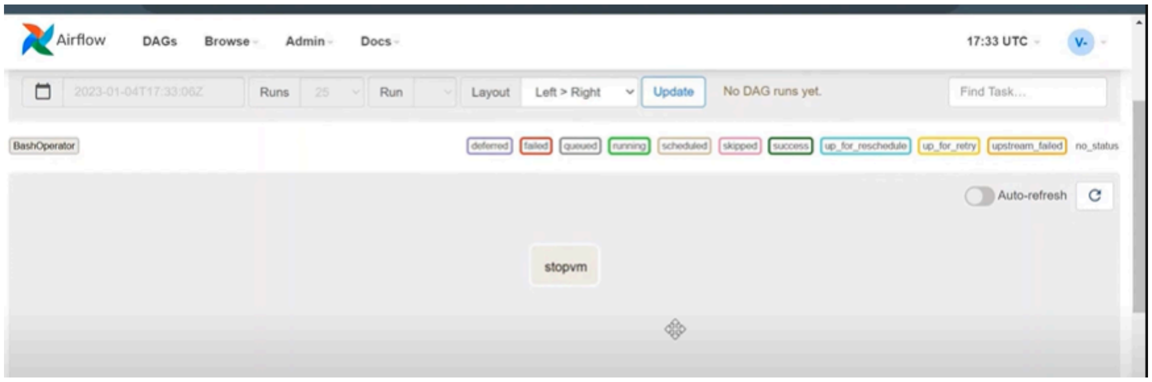
9. Next we will create the task



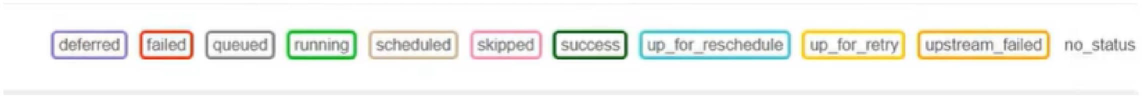
```
t1 = BashOperator(
    task_id='stopvm',
    bash_command='gcloud compute instances stop database --zone=us-central1-a',
    dag=dag,
    depends_on_past=False,
    priority_weight=2**31 - 1,
    do_xcom_push=False)
```

here in bash_command…we have given a command to stop the db

10. Next we'll upload this file in composer..



11. SO here we have our first task
12. We can get task status from here



13. Here we can see our DAG runs…and our first task is successful

14. For airflow refer project video