Exercise: Setup Your Local Environment for DAG Development & Testing

Prerequisites

- Basic understanding of Apache Airflow.
- An active GCP project with the necessary APIs enabled.
- VSCode and Conda are installed.

Task 1: Setting Up a Conda Environment

Hints (1)

1. Create a new Conda environment and activate it

```
conda create -n airflow_env python=3.11
conda activate airflow_env
```

2. Install Apache Airflow with the Google provider

```
pip install apache-airflow[google]
```

3. Verify the installation by running `airflow` in the terminal. You should see the airflow command's help output

Task 2: Initialize Apache Airflow

Hints (2)

1. Initialize the Airflow database and configure airflow.cfg

```
airflow db init
```

2. Open the `airflow.cfg` in a text editor. You might need to adjust the `dags_folder` and `base_log_folder` to your project directory. Make sure to use absolute paths

Task 3: Authenticate with GCP

Hints (3)

1. Activate the service account using `gcloud`

```
gcloud auth activate-service-account --key-file=PATH_TO_YOUR_KEY_FILE
```

2. Set the `GOOGLE_APPLICATION_CREDENTIALS` environment variable in your shell profile (.bashrc, .zshrc, etc.) or in the VSCode terminal

```
export GOOGLE_APPLICATION_CREDENTIALS="PATH_TO_YOUR_KEY_FILE"
```

3. Verify the authentication by running a gcloud command, e.g., `gcloud projects list`

Task 4: Test Your DAG Locally

Hints (4)

1. Place your DAG Python file in the `dags` folder specified in your `airflow.cfg`

2. Test individual tasks within your DAG using the `airflow tasks test` command

```
airflow tasks test [DAG_ID] [TASK_ID] [EXECUTION_DATE]
```

3. You should see the task executing and logging output in your terminal

Note

- Ensure your Python file with the DAG is error-free and uses correct references and IDs.
- Keep in mind to use the exact path to your key file and keep the file secure.
- Always deactivate your conda environment after usage by running `conda deactivate`.

Happy Coding & Testing!