

Instructions for Using the BigQuery-PySpark-MongoDB Data Engineering Project

This Data engineering project involves the use of Docker, Apache Airflow, Terraform, PySpark, BigQuery, and MongoDB. The project is designed for building a robust data pipeline with two microservices and ingest big amazon_review dataset from local host and save it into BigQuery and processes dataset with pyspark and loads data into MongoDB for using sentiment analysis ML app.

Project Overview

The project includes two microservices and components to handle data ingestion, preprocessing, and processing. This includes:

- **Docker Compose setup** for containerized services.
- **Apache Airflow DAGs** for managing workflows.
- **Terraform** configurations for infrastructure setup.
- **PySpark** scripts for data processing.

Credentials File

The project requires access to Google Cloud services for BigQuery. The credentials file contains sensitive information such as service account keys for Google Cloud. This file is ignored in the repository using .gitignore to prevent accidental commits of sensitive data.

Tip: If you want to create an account in BigQuery and MongoDB, generate your credentials JSON file for BigQuery and set the MONGODB_URI for MongoDB. Then, replace my credentials file with yours. Just don't forget to update the JSON file name in the ingestion and preprocessing Python code accordingly.

How to Use:

1- Firstly you should download this file Google Cloud Service Account (big-test-449715-2b0e9010365e.json) from

(https://drive.google.com/drive/u/4/folders/16qyKNRDmMrKSCbH7vv9_51Ajmy5Byf_A) google drive address and paste it into these folders: microservices_data ingestion, microservices_data preprocessing, terraform.

2- Secondly you should download the dataset from

(https://drive.google.com/drive/u/4/folders/16qyKNRDmMrKSCbH7vv9_51Ajmy5Byf_A) and then paste it into microservices_data ingestion folder.

3- And you should create two images from Ingestion DockerFile and Preprocessing DockerFile with the following commands , just don't forget firstly go to the path of each microservices and then run these command on the right path:

run this command in the path of Ingestion folder

```
# 3.1- docker build -t ingestion-microservice .
```

run this command in the path of preprocessing folder

```
# 3.2- docker build -t preprocess_micro .
```

4- And then run the docker desktop and build a docker-copmose and up it with the following command:

run these command in the path of Bigquery-Pyspark-MongoDB

```
# 4.1- docker compose build
```

```
# 4.2- docker-compose up
```

Or you can use this alternative command instead of two commands

```
# docker-compose up --build
```

5- And then you can see all logs that shows ingest data from local server and save it into BigQuery and then with another microservices get a part of dataset into Pyspark and preprocess it and then save the cleaned-data into MongoDB.

screenshots from different steps of project

Creating Image docker for ingestion

```
C:\Windows\System32\cmd.e x + v
tion>docker build -t ingestion-microservice .
[+] Building 55.3s (12/12) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile                0.1s
=> => transferring dockerfile: 608B                                0.1s
=> [internal] load metadata for docker.io/library/python:3.8-slim 1.3s
=> [internal] load .dockerignore                                   0.0s
=> => transferring context: 2B                                       0.0s
=> [1/7] FROM docker.io/library/python:3.8-slim@sha256:1d52838af602b4b5a831beb13a0e4d073280665ea7be7f69ce2382f29 0.0s
=> => resolve docker.io/library/python:3.8-slim@sha256:1d52838af602b4b5a831beb13a0e4d073280665ea7be7f69ce2382f29 0.0s
=> [internal] load build context                                   7.0s
=> => transferring context: 174.78MB                                  7.0s
=> CACHED [2/7] WORKDIR /app                                       0.0s
=> [3/7] COPY requirements.txt .                                    0.2s
=> [4/7] RUN pip install --no-cache-dir -r requirements.txt        27.3s
=> [5/7] COPY ingest.py .                                          0.1s
=> [6/7] COPY amazon_review.csv .                                  0.4s
=> [7/7] COPY big-test-449715-2b0e9010365e.json credentials.json 0.1s
=> exporting to image                                              18.7s
=> => exporting layers                                              14.4s
=> => exporting manifest sha256:4709459c50e53262e32a16b9ace115e7d2751c75b9c81cddb43f97d0da7a514e 0.0s
=> => exporting config sha256:4959784b7e6988deb4b7cba56592d311abf76a8ed3dcace7c1bf59066c7c6393 0.0s
=> => exporting attestation manifest sha256:e986080d939962edc55f4c0a62d0ace343993164032e6db243d1bbb16a0ec9fd 0.0s
=> => exporting manifest list sha256:a15587cd1328721a6e865b3213c0921e19ecde0579cc1c88f2369c0a2b8491e7 0.0s
=> => naming to docker.io/library/ingestion-microservice:latest    0.0s
=> => unpacking to docker.io/library/ingestion-microservice:latest 4.2s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/tpy8pp5bmxsqteqkdv16ttxl7
D:\IU Semesters\Semester\Data Engineering-Project\Clone-github-project\Bigquery-Pyspark-MongoDB\microservices_data ingestion>
```

Creating Image docker for preprocessing

```
D:\IU Semesters\Semester\Data Engineering-Project\Clone-github-project\Bigquery-Pyspark-MongoDB\microservices_data preprocessing>docker build -t preprocess_micro .
[+] Building 365.3s (12/12) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile                0.1s
=> => transferring dockerfile: 768B                                0.1s
=> [internal] load metadata for docker.io/library/python:3.9       1.5s
=> [internal] load .dockerignore                                   0.0s
=> => transferring context: 2B                                       0.0s
=> [1/7] FROM docker.io/library/python:3.9@sha256:bc2e05bca883473850fc3b7c134c28ab822be73126balce29517d9e8b7f3 121.7s
=> => resolve docker.io/library/python:3.9@sha256:bc2e05bca883473850fc3b7c134c28ab822be73126balce29517d9e8b7f370 0.0s
=> => sha256:6e02a90e58aec58d3d8f4549cb6a82a2cc1db075b8a26a48fe1d0f065b52d86 19.85MB / 19.85MB 9.6s
=> => sha256:f6d72b00aefcbea513baa839d4f1bcebb51c434d4f9602410df434bc71e233c8e 6.16MB / 6.16MB 2.8s
=> => sha256:353e14e5cc47664fba714a7da288001d98427c785494847ac773f5cc88199451 211.35MB / 211.35MB 115.7s
=> => sha256:255774e0027b72d2327719e78dbad5ad8c9cf446d055e45be7fc149418470bae 64.40MB / 64.40MB 52.5s
=> => sha256:f299e06712452fd49405fe52fb66dc0bbdd14cc8d8342baa8b2741df89dd465d 250B / 250B 0.3s
=> => sha256:091e8249475f42de217265c501e0186f8a3ea7490ef7f51458c30db91fb3cac 24.01MB / 24.01MB 19.5s
=> => sha256:7cd785773db44087e20a679ce5439222e505475eed5b99f191eb2cda51729ab 48.47MB / 48.47MB 82.1s
=> => extracting sha256:7cd785773db44087e20a679ce5439222e505475eed5b99f191eb2cda51729ab 1.6s
=> => extracting sha256:091e8249475f42de217265c501e0186f8a3ea7490ef7f51458c30db91fb3cac 0.6s
=> => extracting sha256:255774e0027b72d2327719e78dbad5ad8c9cf446d055e45be7fc149418470bae 2.3s
=> => extracting sha256:353e14e5cc47664fba714a7da288001d98427c785494847ac773f5cc88199451 4.7s
=> => extracting sha256:f6d72b00aefcbea513baa839d4f1bcebb51c434d4f9602410df434bc71e233c8e 0.4s
=> => extracting sha256:6e02a90e58aec58d3d8f4549cb6a82a2cc1db075b8a26a48fe1d0f065b52d86 0.6s
=> => extracting sha256:f299e06712452fd49405fe52fb66dc0bbdd14cc8d8342baa8b2741df89dd465d 0.0s
=> [internal] load build context                                   0.1s
=> => transferring context: 4.88kB                                    0.0s
=> [2/7] RUN apt-get update && apt-get install -y default-jdk     70.8s
=> [3/7] WORKDIR /app                                             0.1s
=> [4/7] COPY requirements.txt .                                    0.0s
=> [5/7] RUN pip install --no-cache-dir -r requirements.txt        130.1s
=> [6/7] COPY preprocess_container.py                             0.1s
=> [7/7] COPY big-test-449715-2b0e9010365e.json credentials.json 0.1s
=> exporting to image                                              40.5s
=> => exporting layers                                              29.1s
=> => exporting manifest sha256:233e4c6e07b81afa6f0144c69383ade81b994a0e47cd656c136f1d7b1e51154 0.0s
=> => exporting config sha256:5b901ca9711fcd0c6f83a749961cff8ef8ed866d3c5affe8cd8588295a433b9c 0.0s
=> => exporting attestation manifest sha256:4de324088c42ad66625751a887c1736bf8582f261e53eff29a4dd243bb4778eaa4 0.0s
=> => exporting manifest list sha256:07d08dcd4680b621fa92b5c080cf8882ce7f519ea7d276e755bdf52cc24fa 0.0s
=> => naming to docker.io/library/preprocess_micro:latest          0.0s
=> => unpacking to docker.io/library/preprocess_micro:latest       11.2s
```

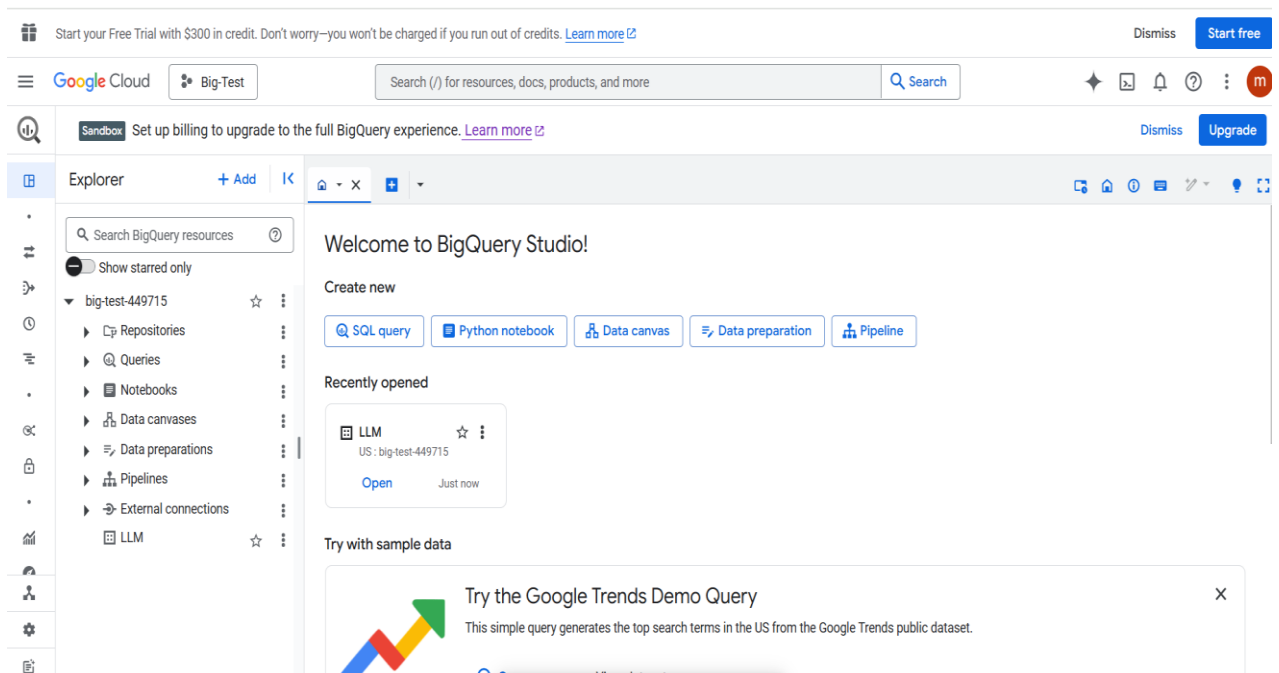
build docker container with docker compose file

```
Name                                Date modified  Type  Size
C:\Windows\System32\cmd.e x + v

Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

D:\IU Semesters\Semester\Data Engineering-Project\Clone-github-project\Bigquery-Pyspark-MongoDB>docker-compose up --build
time="2025-03-24T11:02:48+01:00" level=warning msg="D:\IU Semesters\Semester\Data Engineering-Project\Clone-github-p
project\Bigquery-Pyspark-MongoDB\docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please re
move it to avoid potential confusion"
[+] Running 1/7
- airflow-init [#####] 87.7MB / 251.2MB Pulling 27.1s
- airflow-scheduler Pulling 27.1s
- postgres [#####] 17.68MB / 151MB Pulling 27.1s
- airflow-worker Pulling 27.1s
- redis [ ] Pulling 27.1s
- flower Pulling 27.1s
- airflow-webserver [#####] 32.66kB / 32.66kB Pulling 27.1s
```

Bigquery account before ingesting data into LLM table



Log for ingesting big dataset from local server and save it into BigQuery account

Data Successfully loaded to review_amazon in Bigquery!

```
preprocessing_service | confs: [default]
airflow-webserver-1 | [2025-03-24 10:06:47 +0000] [49] [INFO] Handling signal: ttou
airflow-webserver-1 | [2025-03-24 10:06:47 +0000] [53] [INFO] Worker exiting (pid: 53)
airflow-webserver-1 | 127.0.0.1 - - [24/Mar/2025:10:06:52 +0000] "GET /health HTTP/1.1" 200 187 "-" "curl/7.64.0"
ingestion_service | /usr/local/lib/python3.8/site-packages/google/cloud/bigquery/_pandas_helpers.py:483: FutureWarn
ing: Loading pandas DataFrame into BigQuery will require pandas-gbq package version 0.26.1 or greater in the future. Tri
ed to import pandas-gbq and got: No module named 'pandas_gbq'
ingestion_service | warnings.warn(
ingestion_service | Data successfully loaded to review_amazon in BigQuery!
ingestion_service exited with code 0
preprocessing_service | found com.google.cloud.spark#spark-bigquery-with-dependencies_2.12;0.32.2 in central
preprocessing_service | downloading https://repo1.maven.org/maven2/com/google/cloud/spark/spark-bigquery-with-dependencies_2.12/0.32.2/spark-bigquery-with-dependencies_2.12-0.32.2.jar
```

Dataset is loaded in Bigquery

The screenshot shows the Google Cloud BigQuery interface. At the top, there's a header with 'Google Cloud' and 'Big-Test' project. A search bar is present. Below the header, the 'Explorer' panel on the left shows a tree view of resources. Under 'big-test-449715', there's a folder 'review_amazon' which is selected. The main panel shows the 'review_amazon' dataset with a table explorer view. The table has columns: 'Row', 'rating', 'title', and 'review'. It displays 5 rows of data. The 'review' column contains text reviews of products. At the bottom, there's a 'Job history' section.

| Row | rating | title | review |
|-----|--------|-----------------------------------|--|
| 1 | 1 | Batteries died within a year ... | I bought this charger in Jul 2003 and it worked OK for a while. The design is nice and convenient. However, after about a year, the batteries would not hold a charge. Might as well just get alkaline disposables, or look elsewhere for a charger that comes with batteries that have better staying power... |
| 2 | 1 | DVD Player crapped out after o... | I also began having the incorrect disc problems that I've read about on here. The VCR still works, but hte DVD side is useless. I understand that DVD players sometimes just quit on you, but after not even one year? To me that's a sign on bad quality. I'm giving up JVC after this as well. I'm stickin... |
| 3 | 1 | Incorrect Disc | I love the style of this, but after a couple years, the DVD is giving me problems. It doesn't even work anymore and I use my broken PS2 Now. I wouldn't recommend this, I'm just going to upgrade to a recorder now. I wish it would work but I guess I'm giving up on JVC. I really did like this one...be... |
| 4 | 1 | DVD menu select problems | I cannot scroll through a DVD menu that is set up vertically. The triangle keys will only select horizontally. So I cannot select anything on most DVD's besides play. No special features, no language select, nothing, just play. |
| 5 | 1 | Not an 'ultimate guide' | Firstly, I enjoyed the format and tone of the book (how the author addressed the reader). However, I did not feel that she imparted any insider secrets that the book promised to reveal. If you are just starting to research law school, and do not know all the requirements of admission, then this book ... |

mongoDB account before saving cleaned-data into preprocessed-review table

The screenshot shows the MongoDB Atlas interface. The top navigation bar includes 'Atlas', 'mostafa's O...', 'Access Manager', and 'Billing'. The main content area is titled 'MOSTAFA'S ORG - 2025-03-01 > PROJECT 0 > DATABASES'. It shows a 'Pspark-Cluster' with version 8.0.5 and region AWS Frankfurt (eu-central-1). The 'Collections' tab is selected, showing a list of collections: 'sample_mflix', 'comments', 'embedded_movies', 'movies', 'sessions', 'theaters', and 'users'. The 'sample_mflix' collection is expanded, showing the 'comments' collection. The 'comments' collection has a storage size of 6.34MB, logical data size of 11.4MB, total documents of 41079, and index size of 1.92MB. A query is shown: 'Type a query: { field: 'value' }'. The query results are displayed as JSON documents.

```
{ "_id": ObjectId("5a9427648b0beebe6957abd"), "name": "John Bishop", "email": "john_bishop@fakegmail.com", "movie_id": ObjectId("573a1391f29213caabdc6f98"), "text": "Accusamus qui distinctio ut ab saepe tenetur. Quae optio aut eius dele-", "date": "1972-04-16T14:52:53.000+00:00" }
```

Data loaded from Bigquery into pyspark and cleand and save it into mongoDB

```
w.providers.google.common.hooks.leveldb.LevelDBHook' from 'apache-airflow-providers-google' package: No module named 'ai
rflow.providers.google.common.hooks.leveldb.LevelDBHook'
airflow-webserver-1 [2025-03-24 10:07:18 +0000] [49] [INFO] Handling signal: ttou
airflow-webserver-1 [2025-03-24 10:07:18 +0000] [54] [INFO] Worker exiting (pid: 54)
airflow-webserver-1 127.0.0.1 - - [24/Mar/2025:10:07:22 +0000] "GET /health HTTP/1.1" 200 187 "-" "curl/7.64.0"
preprocessing_service Loading data from BigQuery...
preprocessing_service Data loaded! Cleaning now...
preprocessing_service Connecting to MongoDB...
preprocessing_service Inserted 100 records into MongoDB
preprocessing_service PySpark session stopped
ingestion_service /usr/local/lib/python3.8/site-packages/google/cloud/bigquery/_pandas_helpers.py:483: FutureWarn
ing: Loading pandas DataFrame into BigQuery will require pandas-gbq package version 0.26.1 or greater in the future. Tri
ed to import pandas-gbq and got: No module named 'pandas_gbq'
ingestion_service warnings.warn(
ingestion_service Data successfully loaded to review_amazon in BigQuery!

ingestion_service exited with code 0
preprocessing_service exited with code 0
airflow-webserver-1 127.0.0.1 - - [24/Mar/2025:10:07:32 +0000] "GET /health HTTP/1.1" 200 187 "-" "curl/7.64.0"
airflow-webserver-1 [2025-03-24 10:07:41 +0000] [49] [INFO] Handling signal: ttin
```

mongoDB account after saved preprocessed-data into preprocessed-review table

The screenshot shows the MongoDB Atlas web interface. The left sidebar contains navigation options like Overview, Clusters, SERVICES, and SECURITY. The main panel displays the 'preprocessed-review' database with a collection named 'amazon_reviews'. The collection statistics show 100 documents and a storage size of 48KB. The 'Find' tab is active, showing a query filter and query results. The results display two document snippets with fields like '_id', 'rating', 'title', and 'review'.

Image dockers

The screenshot shows the Docker Desktop interface. The left sidebar has a menu with options like Containers, Images, Volumes, Builds, Docker Scout, and Extensions. The 'Images' tab is selected, showing a list of installed images. The list includes 'ingestion-microservice', 'preprocess_micro', 'redis', 'postgres', and 'apache/airflow'. Each entry shows its tag, image ID, creation time, and size. The bottom status bar indicates the engine is running with system resource usage.

| | Name | Tag | Image ID | Created | Size | Actions |
|--------------------------|------------------------|--------|--------------|----------------|-----------|---------|
| <input type="checkbox"/> | ingestion-microservice | latest | a15587cd1328 | 27 minutes ago | 845.73 MB | |
| <input type="checkbox"/> | preprocess_micro | latest | 07d48dcdda68 | 18 minutes ago | 3.23 GB | |
| <input type="checkbox"/> | redis | latest | bd41d55aae1e | 3 months ago | 172.77 MB | |
| <input type="checkbox"/> | postgres | 13 | 95b9e8830fff | 25 days ago | 599.72 MB | |
| <input type="checkbox"/> | apache/airflow | 2.0.2 | 519ea07c4b14 | 1 year ago | 1.22 GB | |

Image container

The screenshot shows the Docker Desktop application window. The top bar includes the Docker logo, a search bar, and a 'Sign in' button. The left sidebar contains navigation options: Containers, Images, Volumes, Builds, Docker Scout, and Extensions. The main area is titled 'Containers' and displays a table of running containers. Above the table, there are statistics for CPU and memory usage, a search bar, and a toggle for 'Only show running containers'. The table lists containers with columns for Name, Container ID, Image, Port(s), CPU (%), Last started, and Actions. The containers listed are: bigquery-pyspark-mongodb, ingestion_service, postgres-1, redis-1, airflow-init-1, airflow-scheduler-1, preprocessing_service, airflow-webserver-1, and airflow-worker-1. A 'Delete' button and playback controls are visible on the right side of the table.

| Name | Container ID | Image | Port(s) | CPU (%) | Last started | Actions |
|--------------------------|--------------|------------------------|-----------|---------|----------------|----------------------------|
| bigquery-pyspark-mongodb | - | - | - | 11.89% | 10 seconds ago | [Stop] [Refresh] [Delete] |
| ingestion_service | 72ab9349fd42 | ingestion-microservice | | 1.19% | 10 seconds ago | [Stop] [Refresh] [Delete] |
| postgres-1 | 15d1f9b270b1 | postgres:13 | | 3.63% | 8 minutes ago | [Stop] [Refresh] [Delete] |
| redis-1 | 8f5571665b83 | redis:latest | 6379:6379 | 0.87% | 8 minutes ago | [Stop] [Refresh] [Delete] |
| airflow-init-1 | 872c4895e0f2 | apache/airflow:2.0.2 | | 0% | 8 minutes ago | [Start] [Refresh] [Delete] |
| airflow-scheduler-1 | fcc0024b9544 | apache/airflow:2.0.2 | | 4.72% | 8 minutes ago | [Stop] [Refresh] [Delete] |
| preprocessing_service | 5ec40cead35c | preprocess_micro | | 0% | 8 minutes ago | [Start] [Refresh] [Delete] |
| airflow-webserver-1 | c0d0b914e7a5 | apache/airflow:2.0.2 | 8080:8080 | 0.56% | 8 minutes ago | [Stop] [Refresh] [Delete] |
| airflow-worker-1 | ecba039bd6ac | apache/airflow:2.0.2 | | 0.64% | 8 minutes ago | [Stop] [Refresh] [Delete] |

Running Airflow

The screenshot shows the Apache Airflow web interface in a browser. The address bar shows 'localhost:8080/home'. The top navigation bar includes links for DAGs, Security, Browse, Admin, and Docs. The main content area is titled 'DAGs' and displays a table of DAGs. The table has columns for DAG, Owner, Runs, Schedule, Last Run, Recent Tasks, Actions, and Links. The DAGs listed include: data_pipeline, example_bash_operator, example_branch_operator, example_branch_dop_operator_v3, example_branch_operator, example_complex, example_dag_decorator, example_external_task_marker_child, example_external_task_marker_parent, example_kubernetes_executor, example_kubernetes_executor_config, example_nested_branch_dag, and example_passing_params_via_test_command. The interface also includes a search bar for DAGs and a status bar at the bottom showing the current time and date.

| DAG | Owner | Runs | Schedule | Last Run | Recent Tasks | Actions | Links |
|---|-----------|------|----------|----------|--------------|----------------------------|-------|
| data_pipeline | data_team | 0/0 | @@*+* | | 0/0 | [Start] [Refresh] [Delete] | ... |
| example_bash_operator | airflow | 0/0 | @@*+* | | 0/0 | [Start] [Refresh] [Delete] | ... |
| example_branch_dop_operator_v3 | airflow | 0/0 | @@*+* | | 0/0 | [Start] [Refresh] [Delete] | ... |
| example_branch_operator | airflow | 0/0 | @@*+* | | 0/0 | [Start] [Refresh] [Delete] | ... |
| example_complex | airflow | 0/0 | None | | 0/0 | [Start] [Refresh] [Delete] | ... |
| example_dag_decorator | airflow | 0/0 | None | | 0/0 | [Start] [Refresh] [Delete] | ... |
| example_external_task_marker_child | airflow | 0/0 | None | | 0/0 | [Start] [Refresh] [Delete] | ... |
| example_external_task_marker_parent | airflow | 0/0 | None | | 0/0 | [Start] [Refresh] [Delete] | ... |
| example_kubernetes_executor | airflow | 0/0 | None | | 0/0 | [Start] [Refresh] [Delete] | ... |
| example_kubernetes_executor_config | airflow | 0/0 | None | | 0/0 | [Start] [Refresh] [Delete] | ... |
| example_nested_branch_dag | airflow | 0/0 | @@*+* | | 0/0 | [Start] [Refresh] [Delete] | ... |
| example_passing_params_via_test_command | airflow | 0/0 | @@*+* | | 0/0 | [Start] [Refresh] [Delete] | ... |