Airflow Automation Task

Your mission is to build a basic workflow using Apache Airflow.

What is Airflow?
https://airflow.apache.org/docs/apache-airflow/2.5.1/index.html

The workflow (DAG) needs to create a file according to a user parameter and print it to the console

Requirements:

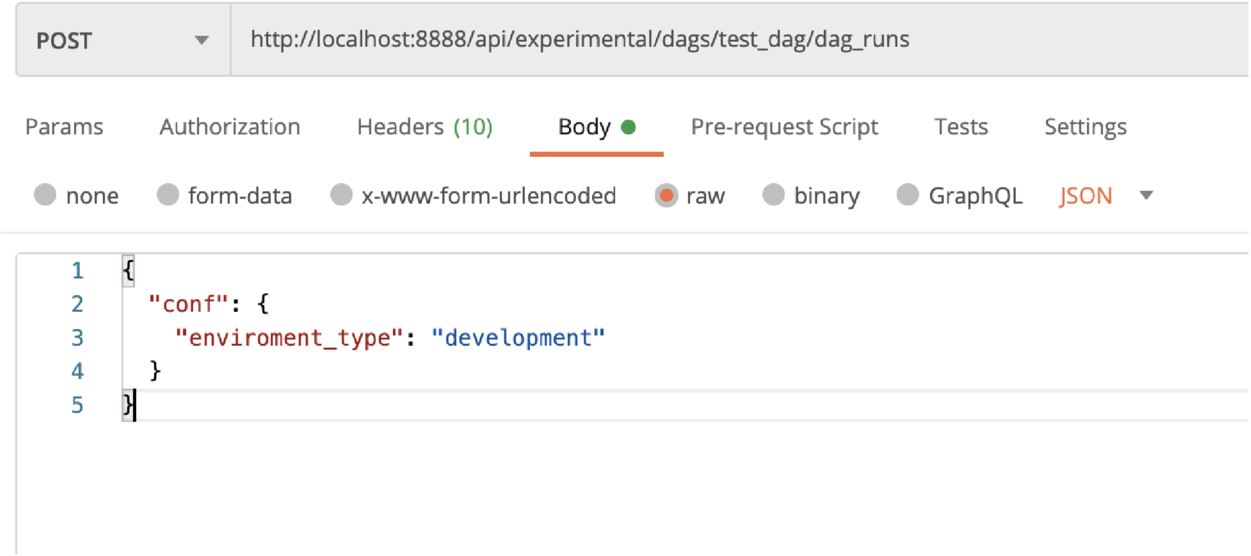
- Input -
    - The workflow should be triggered by REST API POST request with an environment type parameter provided in the body request https://airflow.apache.org/docs/apache-airflow/stable/stable-rest-api-ref.html
    - “environment_type” parameter is mandatory, with the following allowed values:
        - *development*
        - *production*

      For any different input value, the workflow should fail
- Output -
    - In case of *environment_type* = *‘development`* -
        - printing "hello ciValue from development branch" into a local file named "civalue_development _<timestamp>.txt"
    - In case of *environment_type* = *`production’* -
        - printing "hello ciValue from production branch" into a local file named "civalue_production_<timestamp>.txt"
    - The content of the written file should be printed to console
    - Response for POST request - default apache workflow response

Example of POST request using curl:

```
> curl -X POST 'http://localhost:8080/api/v1/dags/<dag_id>/dagRuns' \
        -H 'Content-Type: application/json' \
        --user "airflow:airflow" \
        -d '{
                "conf":{"environment_type": "development"}
            }'
```

Example of POST request using POSTMAN:



POST http://localhost:8888/api/experimental/dags/test_dag/dag_runs

```
{
  "conf": {
    "enviroment_type": "development"
  }
}
```

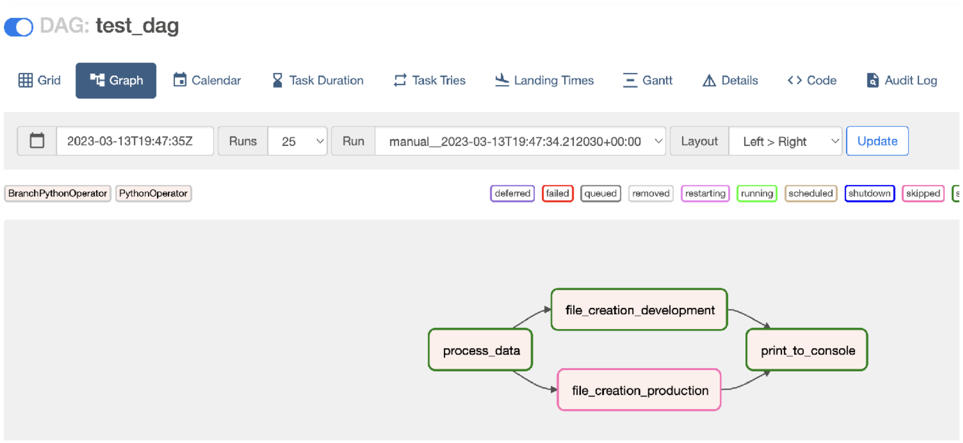
Example of successful response:

{"execution_date":"2020-11-11T18:45:05+00:00","message":"Created <DagRun test_dag @ 2020-11-11 18:45:05+00:00: manual__2020-11-11T18:45:05+00:00, externally triggered: True>"}

Workflow steps:

1. enviroment_branch - Parses the environment type parameter given by the user and chooses the correct workflow branch to execute
2. file_creation -
   a. file_creation_development - printing "hello ciValue from development branch" into a local file named "civalue_development _<timestamp>.txt"
   b. file_creation_production - printing "hello ciValue from production branch" into a local file named "civalue_production_<timestamp>.txt"
3. print_to_console - Reads the newly created file and prints its content to the console

Example of how the workflow looks like:



DAG: test_dag

2023-03-13T19:47:35Z

Runs 25

Run manual__2023-03-13T19:47:34.212030+00:00

Layout Left > Right

Notes:

1. The environment setup is based on Airflow 2.5.1, so make sure you work with the right documentation
2. There are a lot of amazing Airflow operators, but in this task you are only allowed to use "PythonOperator" and "BranchPythonOperator"

Environment setup guide:

1. Install Python 3.7 - https://www.python.org/downloads/release/python-370/
2. Python IDE of your choice (PyCharm is recommended https://www.jetbrains.com/pycharm/download/)
3. Install Docker Engine - https://docs.docker.com/desktop/
4. Install REST API tool of your choice (POSTMAN is recommended https://www.postman.com/downloads)
5. Follow this guide to initialize a local airflow instance: https://airflow.apache.org/docs/apache-airflow/stable/howto/docker-compose/index.html#running-airflow-in-docker
6. Disable example dags, by editing the docker-compose.yaml file, or in any other way.
7. Go to http://localhost:8080/. You should be able to access the Airflow web application.

The setup installation should take up to 1 hour and is **not** part of the test. If you have any issues setting up the environment please contact us.

References:

1. Apache Airflow - https://airflow.apache.org/docs/apache-airflow/2.5.1/tutorial/index.html
2. Docker logs for basic debugging - https://docs.docker.com/config/containers/logging/