Exercise 9: Configuring Astro cloud deployments with Github

Introduction

Integrating Astro Cloud with GitHub allows seamless deployment of Airflow projects directly from a Git repository. This enables teams to manage Airflow DAGs, plugins, and dependencies using Git's version control, ensuring streamlined collaboration and deployment.

This lab will guide you through configuring Astro Cloud to deploy DAGs from a GitHub repository and running a DAG using GitHub-triggered updates.

Objectives

- Set up a GitHub repository for your Airflow project.
- Connect Astro Cloud with GitHub to enable automatic deployments.
- Test the integration by deploying and running a sample DAG from GitHub.

Preparation

Prerequisites

- Familiarity with Airflow DAGs and task dependencies.
- Basic understanding of Github UI and Git cli commands.
- Docker installed and running.
- Python (3.7 or above) and pip installed.
- Active Astronomer Cloud account and workspace.
- Astro CLI installed on your local machine
    - Verify using bash command astro version

Setup Git

- Visit https://github.com/ to create a github account using personal email id
- Download and install git from https://git-scm.com/downloads/win
    - Verify running command on bash : git

Let's Get Started

Part 1: Environment Setup

Step 1. Authenticate Astro CLI:

- Authenticate your CLI with Astronomer Cloud:
    i. Use command astro login
    ii. Press Enter

Step 2. Project setup

- Ensure you have an initialized Airflow project. If not, initialize a new project using:
    i. Create a new directory and navigate into it:
        1. `mkdir astro_cloud`
        2. `cd astro_cloud`
        3. `astro dev init`

Part 2: Deploying to Astronomer Cloud

Step 1. Add a new Dag

- Navigate to the `dags/` directory and create a file named `git_dag`.py

```
from airflow import DAG
```

```
from airflow.operators.bash import BashOperator

from airflow.utils.dates import days_ago



default_args = {

    'owner': 'airflow',

    'retries': 1,

    'retry_delay': timedelta(minutes=5),

}



with DAG(

    'git_deployment_example',

    default_args=default_args,

    description='A simple DAG deployed via GitHub integration',

    schedule_interval='@daily',

    start_date=days_ago(1),

    catchup=False,

) as dag:



    hello_task = BashOperator(

        task_id='print_hello',

        bash_command='echo "Hello from Astro Cloud deployment!"',

    )



    goodbye_task = BashOperator(
```

```
    task_id='print_goodbye',

    bash_command='echo "Goodbye from Astro Cloud deployment!"',

)



hello_task >> goodbye_task
```

- Save the file and navigate to the project root.

Step 2. Setup Github repository

Log in to your GitHub account and create a new repository (e.g., `astro-cloud-deployment`).

Step 3. Initialize local git project

- Initialize Git in your local project directory and push the project to GitHub:
  i. git init
  ii. git add .
  iii. git commit -m "Initial commit for Astro deployment"
  iv. git branch -M main
  v. git remote add origin https://github.com/<your-username>/astro-cloud-deployment.git
  vi. git push -u origin main

Step 3. Connect Astro cloud with Github

- Log in to your Astronomer Cloud account.
- Navigate to your Deployment and select the Deployments tab.
- Click on Connect GitHub Repository and authorize Astronomer Cloud to access your GitHub account.

- Select the repository you created (astro-cloud-deployment) and specify the branch (main).
- Save the configuration.

Step 4. Verify deployment

- Log in to the Astronomer Cloud UI and navigate to your deployment.
- Click on Trigger Git deploy
- Verify if the git code is getting deployed in the deployment history

Step 5. Monitor and Manage Workflows

- Verify if we have updated dag in the deployment
- Trigger the DAG from the Astronomer Cloud UI.
- Monitor task execution
- Check the logs of the dag.

Step 6. Update deployment

- Update code in one of the dags
- Use below git command
    i. git add dags/<<dag_name>>.py
    ii. git commit -m "Updated goodbye task command"
    iii. git push
- Monitor the Astronomer Cloud UI for updated code after GitHub push and Trigger Git deploy
- Verify that the DAG appears in the **DAGs** tab of the Astronomer Cloud UI.
- Try Roll back to the previous version of your dags/project

Conclusion

In this lab, you learned:

- Advanced CI/CD practices using GitHub Actions for automated testing and deployment.
- Experiment with deploying plugins and custom Python packages via GitHub.
- Scale the workflow to handle multi-environment and multi-branch deployments.