Assignment 5 - CI/CD - GitOps

- Sangram Jagtap
- Omkar Nagarkar
- Purvil Patel
- Atharva Jadhav

GitOps is a paradigm or a set of practices that emphasizes the use of Git as the source of truth for declarative infrastructure and applications. With Git at the center of the delivery pipelines, every change is submitted as a commit, which then goes through a version control system for approval before being applied to the infrastructure. It combines the version control system for code with the automation to apply that code to the infrastructure. This approach allows for improved collaboration, increased deployment speed and frequency, enhanced reliability and stability, and better compliance and auditing.

Benefits that organizations can realize by adopting GitOps:

Increased Deployment Speed and Frequency:
Automation and integrated collaboration mechanisms greatly increase the speed and frequency of deployments.

Increased Reliability: With Git's revert/rollback capabilities, it's straightforward to roll back to the last known good configuration, reducing recovery time.

Improved Stability: Audit logs are automatically created for all changes, ensuring greater system stability.

Consistency and Standardization: Environments are defined as code, allowing for consistent deployment across different environments.

Improved Collaboration:
GitOps keeps the entire state of your environment under source control, making it easier to collaborate on infrastructure changes.

Ease of Adoption: Git is widely used and understood, making it easier for developers to adopt GitOps practices.

Greater Efficiency: Teams don't have to switch tools to manage updates, increasing efficiency.

More Robust Security: Using Git for infrastructure definitions enhances security and makes all changes auditable.

Case Study:

Improving Developer Experience at a Financial Services Company with GitOps

Organization:
A large, heavily regulated financial services company.

Challenge:
The company needs to make it easier for their developers to work effectively so they can create valuable products quickly, but they also have to make sure they follow rules and keep track of everything properly.

Solution:
The company implemented GitOps to manage their infrastructure and application deployment processes.

Implementation:
1. They kept a record of their system's setup in Git. This meant that every modification was recorded and could be reviewed later.
2. Any updates to their system were done automatically by making commits in Git.
3. They utilized a software that took whatever was saved in Git and put it into use automatically, making the whole setup process smoother and faster.

Benefits Realized:
1. Increased Visibility:
   Keeping infrastructure configuration in Git allowed new starters or colleagues from other teams to quickly become acquainted with the infrastructure.
2. Improved DORA Metrics:
   GitOps helped improve key DevOps Research and Assessment (DORA) metrics, which are crucial indicators of a team's performance in software system development.
   a. Deployment Frequency: More frequent and successful releases to production.
   b. Lead Time for Changes: Reduced time from commit to production.
   c. Change Failure Rate: Lower percentage of deployments causing a failure in production.
   d. Time to Restore Service: Quicker recovery from failures in production.
3. Enhanced Developer Experience:
   The GitOps framework provided a repeatable pattern to automate tasks, thereby improving the overall developer experience.
4. Disaster Recovery:

In the event of a complete cluster failure, pointing a rebuilt or new cluster to the Git repository allowed for quick restoration of resources, improving the Time to Restore Service metric.

Key Principles:

1. The company focused on key principles rather than specific tooling, emphasizing the importance of Git as the only required tool.
2. They chose the best tools for the job, preferring flexibility over alignment with a specific vendor.

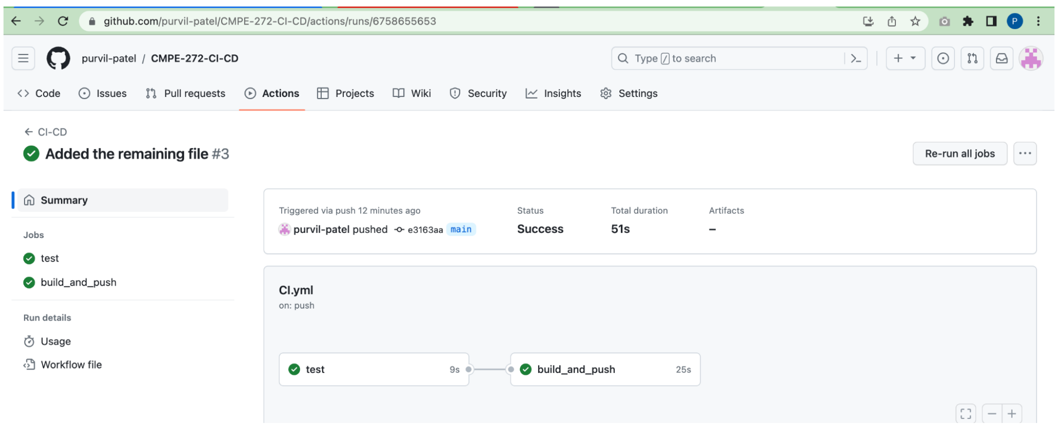
Task 2:

1. **You are free to choose the software tooling of your choice**
    a. We’ve used the github action to create the CICD
2. **There must be at least one integration for an automated trigger to build/deploy your application.**
    a. This workflow is going to execute when we commit or create the pull request on the main branch.
    b. Also we’ve Created the 2 stages test and built in our pipeline. Once the test stage is going to pass then only the build stage is going to execute.
3. **Explain your decisions on choosing the software tooling and share any challenges that you may face while setting up an automated pipeline.**
    a. **Integrated with GitHub:** GitHub Actions is natively integrated with GitHub, eliminating the need for third-party integrations or additional sign-ups. This makes setting up CI/CD pipelines for repositories hosted on GitHub straightforward.
    b. **Easy Setup:** With GitHub Actions, you can set up CI/CD workflows directly within your repository using YAML files. There's no need to set up and maintain separate servers or infrastructure as with Jenkins.
    c. **Built-in Secret Management:** GitHub provides a secure way to store and use encrypted secrets in your workflows, ensuring sensitive data like API keys or credentials are kept safe.
    d. **Granular Permissions:** With GitHub Actions, you can set permissions at the workflow, job, or step level, providing fine-grained control over who can execute or modify specific parts of the CI/CD process.
    e. **Cost-effective:** For public repositories, GitHub Actions provides a generous amount of free build minutes. Even for private repositories, the pricing is competitive, especially considering the integrated nature of the service.

4. **Capture screenshots of your build pipeline in action and explain what is happening at each step.**

Create the CI.yml file that controls the workflow of the pipeline: https://github.com/purvil-patel/CMPE-272-CI-CD/blob/main/.github/workflows/CI.yml

When we commit or create PR on the main branch then it will run first the test job and the build job.



Added the remaining file #3

Triggered via push 12 minutes ago — purvil-patel pushed e3163aa main

Status: Success

Total duration: 51s

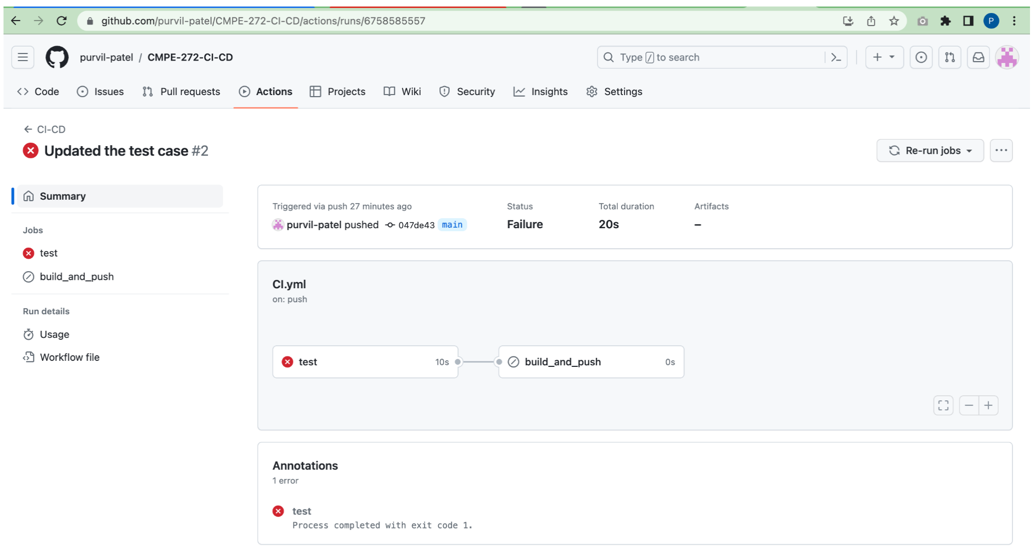
Artifacts: –

Jobs: test, build_and_push

CI.yml
on: push

test 9s → build_and_push 25s

When the Test job fails then the build job won't run.



Updated the test case #2

Triggered via push 27 minutes ago — purvil-patel pushed 047de43 main

Status: Failure

Total duration: 20s

Artifacts: –

Jobs: test, build_and_push

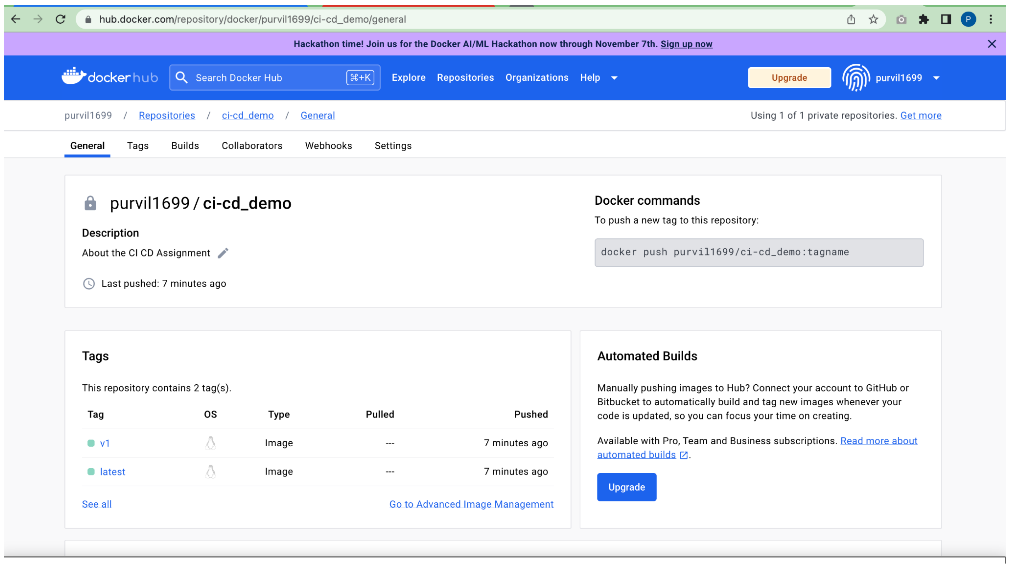
CI.yml
on: push

test 10s → build_and_push 0s

Annotations
1 error

test
```
Process completed with exit code 1.
```

Docker Hub Repo



hub.docker.com/repository/docker/purvil1699/ci-cd_demo/general

Hackathon time! Join us for the Docker AI/ML Hackathon now through November 7th. Sign up now

purvil1699 / Repositories / ci-cd_demo / General

Using 1 of 1 private repositories. Get more

General | Tags | Builds | Collaborators | Webhooks | Settings

purvil1699 / ci-cd_demo

Description

About the CI CD Assignment

Last pushed: 7 minutes ago

Docker commands

To push a new tag to this repository:

```
docker push purvil1699/ci-cd_demo:tagname
```

Tags

This repository contains 2 tag(s).

Tag	OS	Type	Pulled	Pushed
v1		Image	---	7 minutes ago
latest		Image	---	7 minutes ago

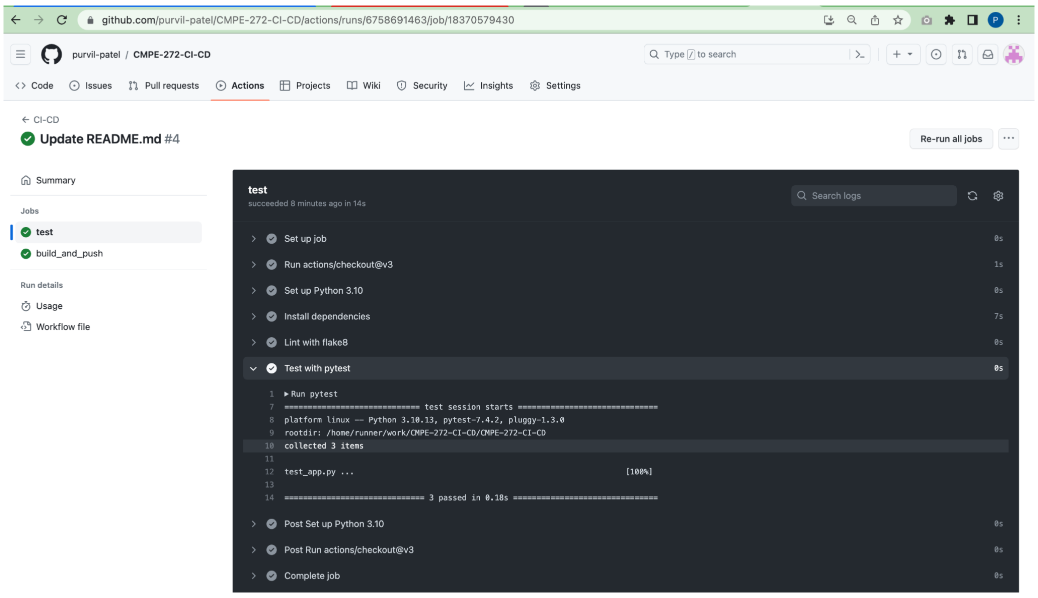
See all | Go to Advanced Image Management

Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. Read more about automated builds.

Python Testcases passed



github.com/purvil-patel/CMPE-272-CI-CD/actions/runs/6758691463/job/18370579430

purvil-patel / CMPE-272-CI-CD

← CI-CD

Update README.md #4

Jobs: test, build_and_push

test
succeeded 8 minutes ago in 14s

```
Set up job                          0s
Run actions/checkout@v3             1s
Set up Python 3.10                  0s
Install dependencies                7s
Lint with flake8                    0s
Test with pytest                    0s
▶Run pytest
============================= test session starts ==============================
platform linux -- Python 3.10.13, pytest-7.4.2, pluggy-1.3.0
rootdir: /home/runner/work/CMPE-272-CI-CD/CMPE-272-CI-CD
collected 3 items

test_app.py ...                                                          [100%]

============================== 3 passed in 0.18s ===============================
Post Set up Python 3.10             0s
Post Run actions/checkout@v3        0s
Complete job                        0s
```

A working demo is embedded in the readme.