CI/CD Pipeline Implementation and Results Report for Quotis

CSCC01 A2

August 7, 2024

1 Introduction

This document details the Continuous Integration and Continuous Deployment (CI/CD) pipeline designed for the Quotis project, using GitLab for CI/CD operations and Docker for containerization.

2 System Design

2.1 Components

- **Source Code Repository:** GitLab
- **CI/CD Platform:** GitLab CI
- **Containerization Tool:** Docker
- **Deployment Target:** N/A

2.2 Workflow Implementation

2.2.1 Build Stage

Packages source code into a Docker container, installs dependencies, and pushes the image to Docker Hub.

2.2.2 Deploy Stage

Pulls the Docker image from Docker Hub and deploys it to a specified production environment.

2.2.3 Test Stage

Executes automated tests within the deployed application to verify functionality and performance.

2.2.4 Sync Stage

Synchronizes the codebase from GitLab to GitHub to maintain consistency across platforms.

3 Configuration Files

3.1 Dockerfile

```
FROM node:18
WORKDIR /usr/src/app
COPY package*.json ./
RUN npm install --legacy-peer-deps
RUN npm install jest --global
COPY . .
EXPOSE 3000
CMD ["npm", "start"]
```

3.2 docker-compose.yml

```
version: '3.8'
services:
  webapp:
    image: ji24077/quotis:1.0.0
    ports:
      - "80:3000"
    environment:
      NODE_ENV: production
    restart: always
  test:
    image: ji24077/quotis:1.0.0
    environment:
      NODE_ENV: test
    command: npm test
    depends_on:
      - webapp
```

3.3 .gitlab-ci.yml

```
stages:
  - build
  - deploy
  - test
  - sync

build-deploy-test:
  image: docker:latest
  stage: build
  services:
    - docker:dind
  script:
    - echo "$DOCKER_PASSWORD" | docker login -u "$DOCKER_USER" --
   password-stdin
```

```
    - docker build -t $DOCKER_USER/quotis:1.0.0 -f CICD/Dockerfile .
    - docker push $DOCKER_USER/quotis:1.0.0
    - docker pull $DOCKER_USER/quotis:1.0.0
    - docker run -d -p 80:3000 --name webapp $DOCKER_USER/quotis:1.0.0
    - sleep 120
    - docker logs webapp
    - docker exec webapp npm test
  environment:
    name: production

sync_to_github:
  stage: sync
  script:
    - git config --global user.email "yhs24077@gmail.com"
    - git config --global user.name "ji24077"
    - git remote add github https://ji24077:$PAT@github.com/UofT-UTSC-CS-sandbox/final-term-project-quotis.git
    - git fetch github
    - git checkout --track github/main
    - git push github main
```

4 Results and Observations

4.1 Build and Deployment

The build and deployment stages executed successfully as evidenced by the logs generated during the build process, which indicated that the Docker container was properly configured, built, and deployed. The build process pulled the necessary base image and installed all required dependencies without any errors.

4.1.1 Build Log

```
$ docker build -t $DOCKER_USER/quotis:1.0.0 -f CICD/Dockerfile .
#1 [internal] load build definition from Dockerfile
...
#12 exporting to image
#12 exporting layers 4.9s done
#12 writing image sha256:2
   f429c400cf58daaa5db6afe48b6c0d7dd963585e83b344bd28016e97bc54127 done
#12 naming to docker.io/ji24077/quotis:1.0.0 done
```

4.2 Testing

The testing phase ran automatically after the deployment. The tests verified the basic functionality of the application. The 'jest' framework was used to execute the test script, and it passed successfully, confirming the operational integrity of the application.

4.2.1 Test Output

```
$ docker exec webapp npm test
> quotis@1.0.0 test
> jest
```

```
PASS CICD/__tests__/App.test.js
      True is true (3 ms)
Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        0.379 s
Ran all test suites.
```

4.3 Sync to GitHub

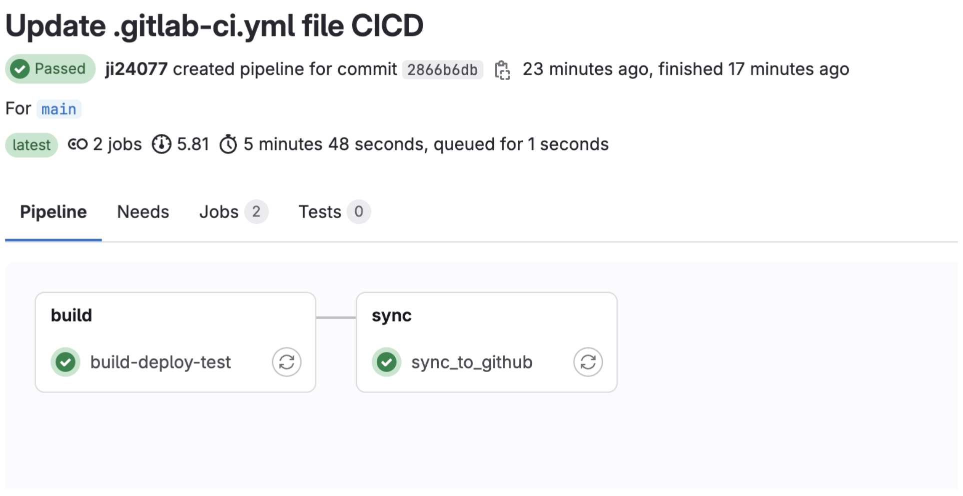
The synchronization stage pushed changes from GitLab to GitHub, ensuring all branches and commits are consistent across both repositories. The process concluded without any issues, maintaining up-to-date code on both platforms.

4.3.1 Sync Log

```
$ git push github main
Everything up-to-date
```

4.4 Pipeline Execution Screenshot

A screenshot of the GitLab CI/CD pipeline execution is included below to demonstrate the successful completion of all stages. This visual evidence supports the robustness of the implemented CI/CD pipeline.



Update .gitlab-ci.yml file CICD

Passed ji24077 created pipeline for commit 2866b6db 23 minutes ago, finished 17 minutes ago

For main

latest 2 jobs 5.81 5 minutes 48 seconds, queued for 1 seconds

Figure 1: Successful execution of the CI/CD pipeline in GitLab