**MLOPS**

**Assignment: 01**

**Raffay Ahmed i21-1674**

**Ahmad Usama i20-2655**

**Submitted to: Sir Pir Sami Ullah**

**Report Analysis:**

Link: https://github.com/raffay464/i211674\_i202655\_Ass01.git

### **Introduction**

This report explains how a GitHub repository was set up with a structured branching strategy and workflow automation to ensure code quality and proper testing before merging.

### **Branching Strategy**

The repository has three branches:

* **Dev:** For development and new features.
* **Test:** For validating features before production.
* **Master:** For final, stable code.

Branch protection rules prevent direct commits to master, allowing only merges from test.

### **Code Quality Check (Flake8 & GitHub Actions)**

A GitHub Actions workflow (code\_quality.yml) runs on dev to:

1. Check out the repo.
2. Set up Python 3.11. (3.x for the latest version)
3. Install dependencies.
4. Run **Flake8** for code quality.

Merging to dev is only allowed if this check passes.

### **Feature Testing (GitHub Actions)**

When merging dev into test, an automated workflow runs unit tests to validate features. If tests fail, the merge is blocked.

### **Enforcing Testing Before Merging to Master**

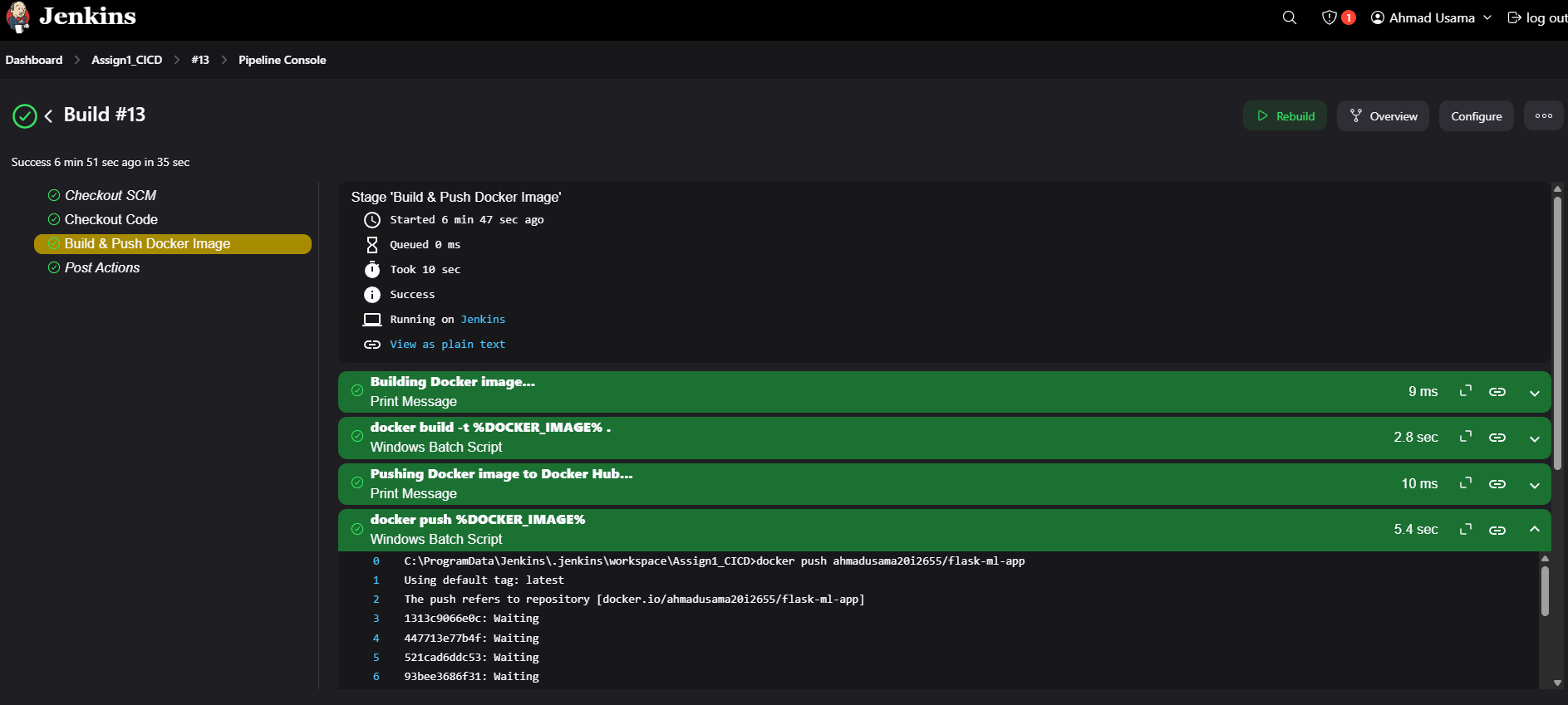
Branch protection is enabled on master:

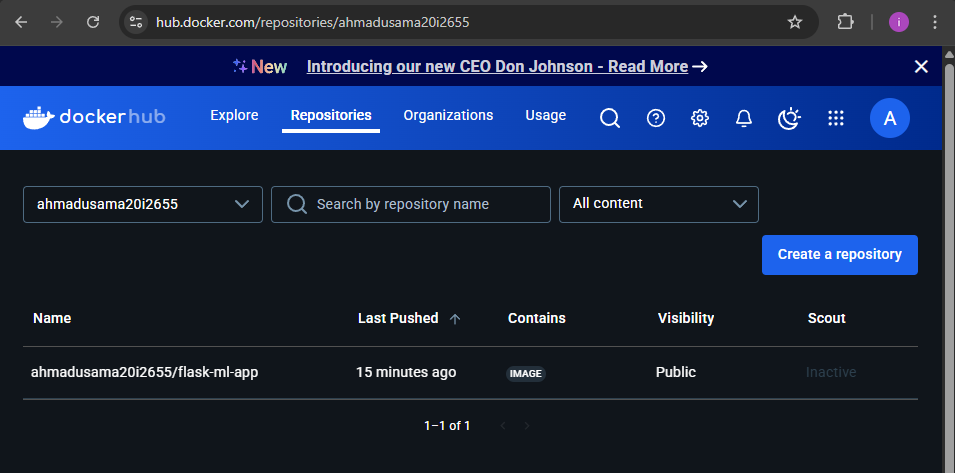
* **Require status checks before merging.**
* **Run Unit Tests** workflow must pass.
* **Only allow merges from test** (no direct commits).

**Deployment with Jenkins & Docker**

### **Jenkins Job Execution & Docker Integration**

* Jenkins Multibranch Pipeline was configured to trigger on merges to main.
* The Jenkinsfile was updated to:
  + Build a Docker image for the Flask application.
  + Push the image to Docker Hub under the repository ahmadusama20i2655/flask-ml-app.

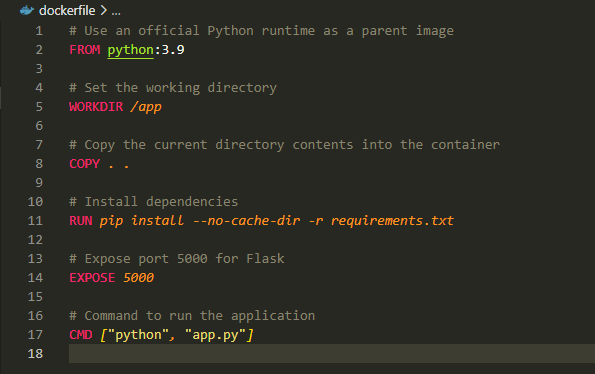
****



### **Dockerfile for Containerization**

A **Dockerfile** was created to define how the Flask application should be containerized. The key steps in the Docker build process are:

1. **Pull Python base image** (python:3.9).
2. **Set the working directory** inside the container.
3. **Copy application files** into the container.
4. **Install dependencies** using requirements.txt.
5. **Expose port 5000** and run the Flask app.



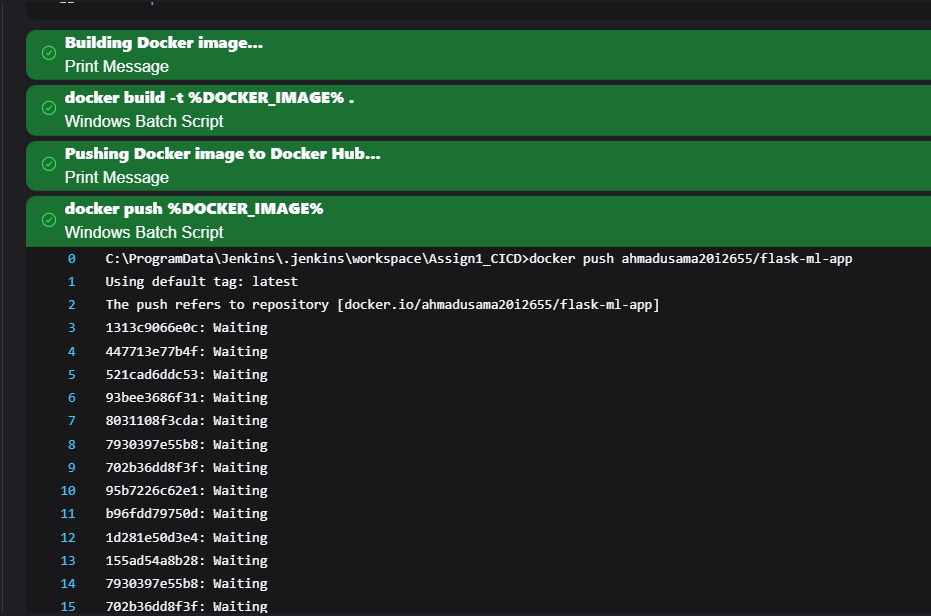
### **Docker Image Push to Docker Hub**

* **Jenkins was configured** with Docker credentials for authentication.

After a successful build, the image was pushed to **Docker Hub** using:  
docker build -t ahmadusama20i2655/flask-ml-app .

docker push ahmadusama20i2655/flask-ml-app

* Verified the image in **Docker Hub**.



### **Admin Notification on Deployment Completion**

* **Jenkins Extended E-mail Notification Plugin** was used to send deployment success emails.
* The email is triggered **after the image is pushed to Docker Hub**.
* SMTP was configured using **Gmail’s SMTP server (smtp.gmail.com:465)**.

