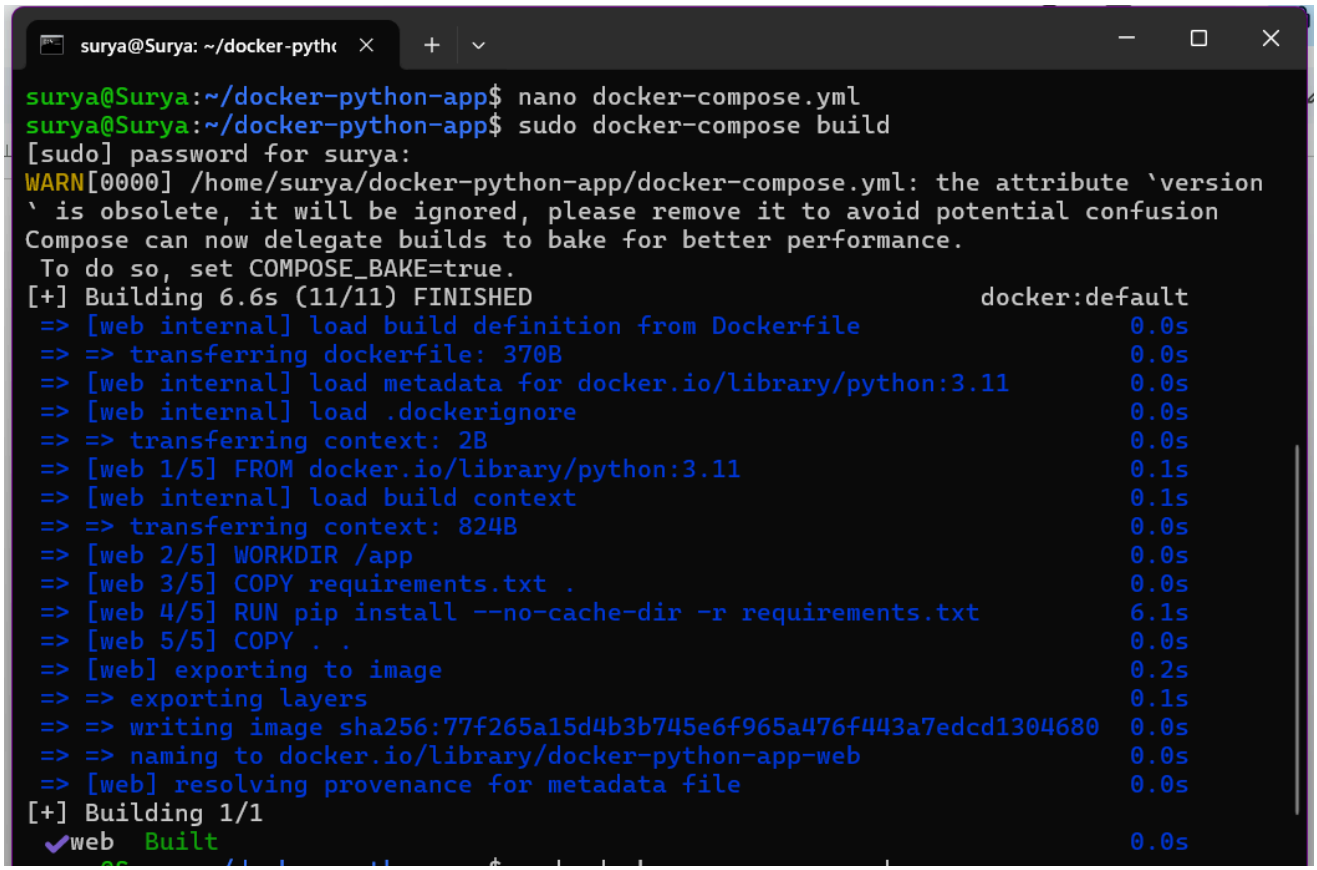
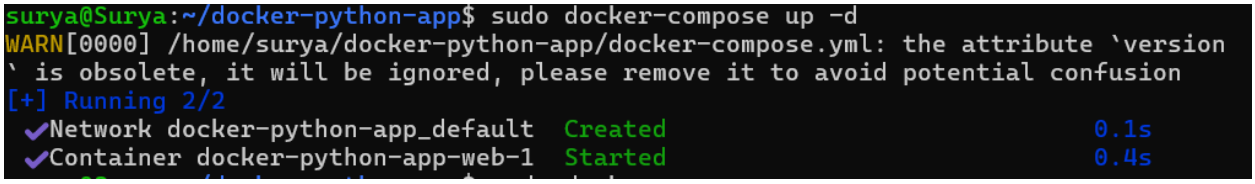
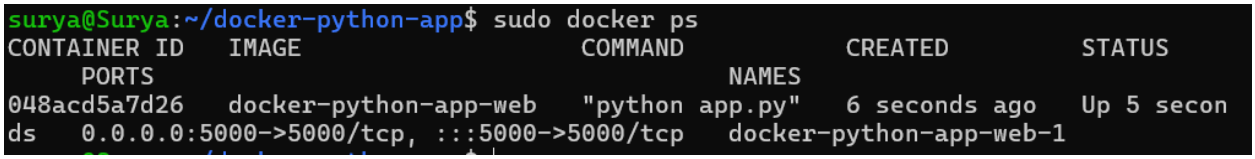
DevOps Task 2

Build of docker-compose

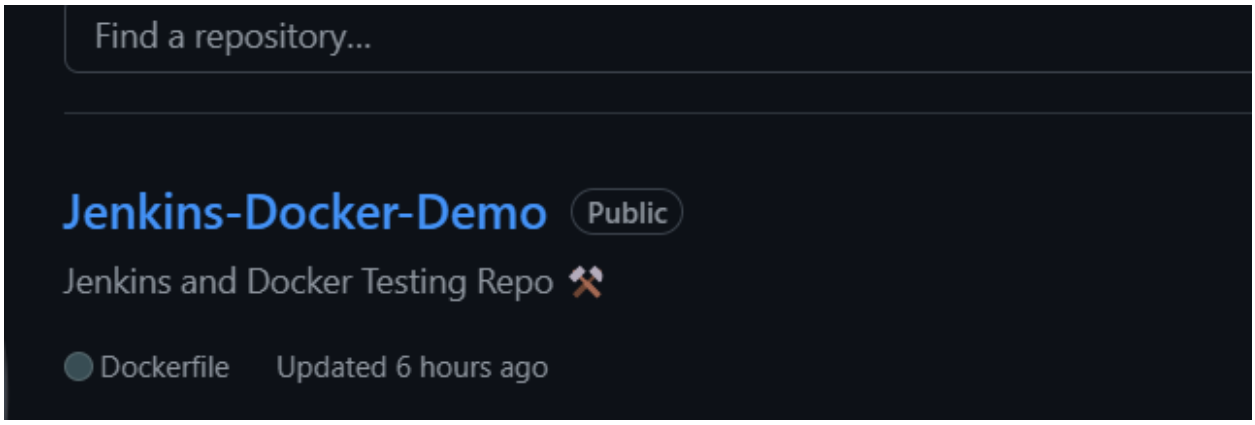






Step 1: Create a GitHub Repository

1. Click on New Repository
2. Set the repository name as: `Jenkins-Docker-Demo`
3. Select Public
4. Click on Create Repository (https://github.com/Surya-2k4/Jenkins-Docker-Demo.git)



Step 2: Generate a Personal Access Token (PAT)

1. Go to: GitHub Developer Settings
2. Navigate to:
   Personal Access Tokens → Tokens (classic) → Generate New Token
3. Set a **Note**: `Jenkins Docker Pipeline`
4. Select the following **Scopes**:
   - `repo`
   - `workflow`
   - `admin:repo_hook`
5. Generate the token.

Token:
`ghp_pY2u7AEWKIb2r0MgfPbDIDohlIgMKT0g****`

Step 3: Start and Enable Jenkins

1. Open the terminal.
2. Enable Jenkins to start at boot

```
sudo systemctl enable jenkins

sudo systemctl start jenkins
```

```
surya@Surya:/mnt/c/Users/SURYA$ sudo systemctl enable jenkins
[sudo] password for surya:
Synchronizing state of jenkins.service with SysV service script with /lib/systemd/system
d-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable jenkins
surya@Surya:/mnt/c/Users/SURYA$ sudo systemctl start jenkins
surya@Surya:/mnt/c/Users/SURYA$
```

Step 4: Access Jenkins

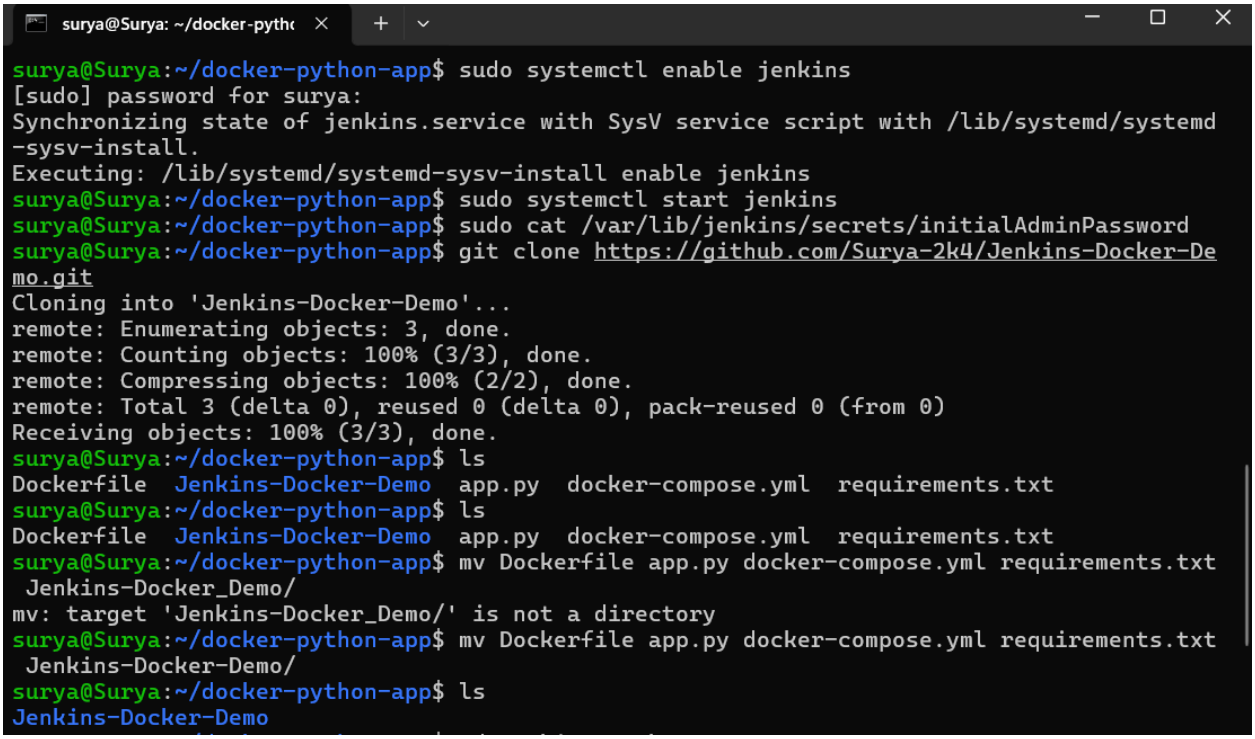
1. Open your browser:
   http://localhost:8080
2. Log in with the following credentials:
   - **Username:** Admin
   - **Password:** 316b8fd8d958454e928ed60**********

Step 5:

In Jenkins, create a new project by selecting Pipeline and entering the repo URL: https://github.com/Surya-2k4/Jenkins-Docker-Demo.git. Use the main branch for the pipeline source.

git clone https://github.com/Surya-2k4/Jenkins-Docker-Demo.git

cd Jenkins-Docker-Demo



Step 6:

List the files using `ls` and verify that `app.py`, `Dockerfile`, `requirements.txt`, and `compose-docker.yml` are present. Move all required project files into the repo folder using:

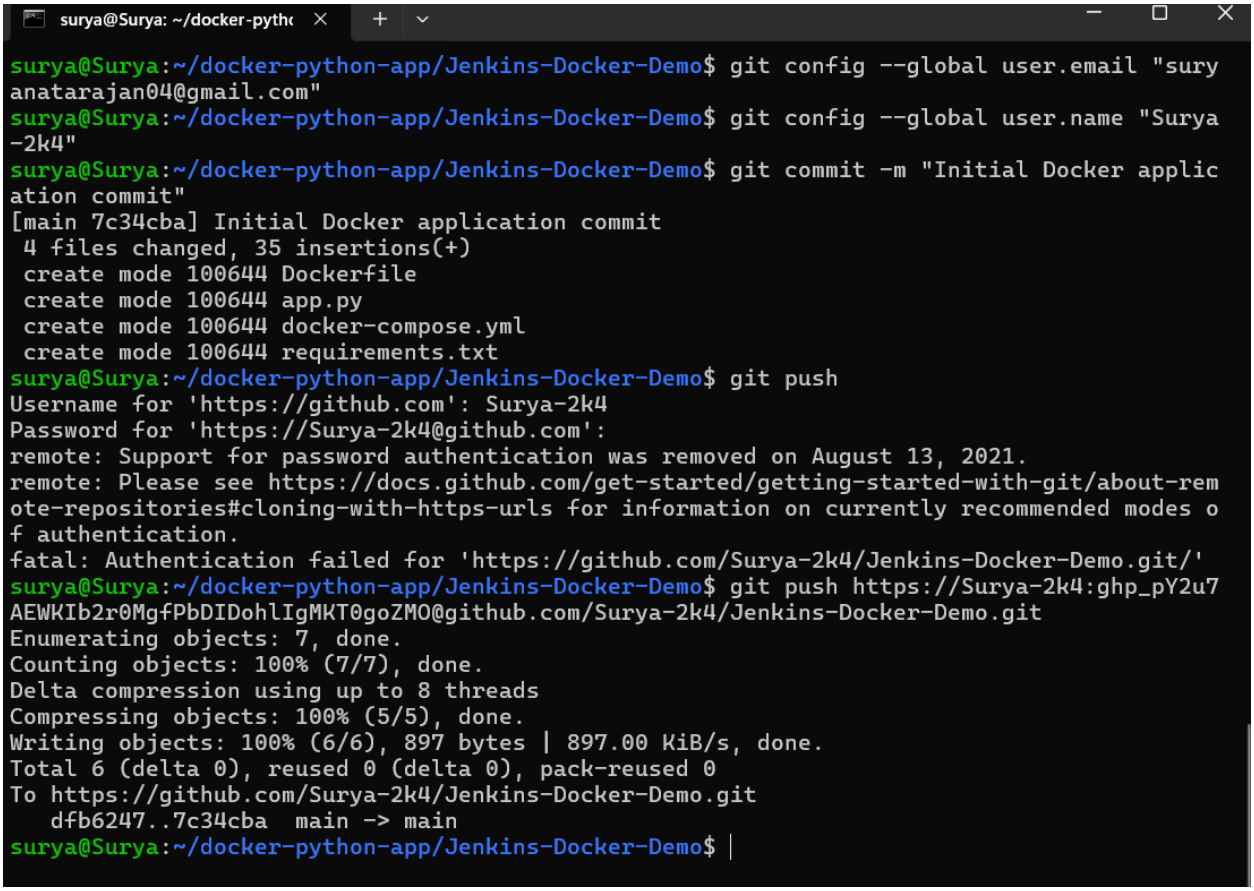
```
mv /path/to/your/files/* Jenkins-Docker-Demo/
```

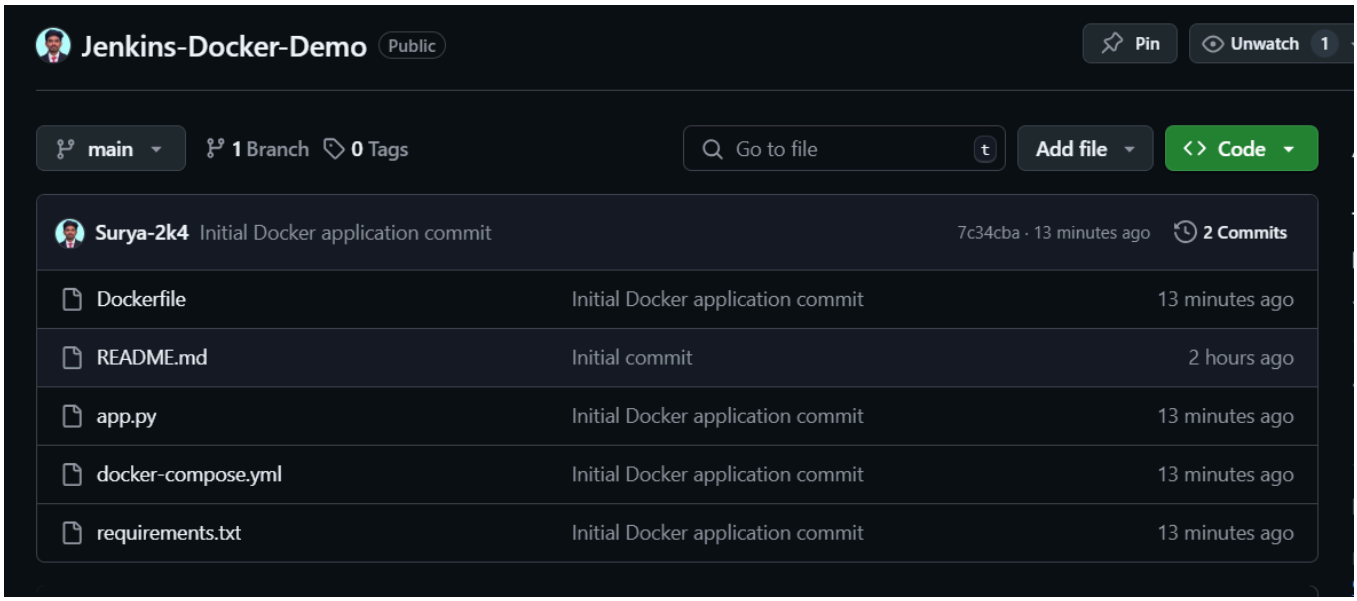
Add and push the files to GitHub:

```
git add .
```

```
git commit -m "Initial Docker application commit"
```

```
git push https://Surya-2k4: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@github.com/Surya-2k4/Jenkins-Docker-Demo.git
```





Step 7: Create a DockerHub account at DockerHub with:

- **Username:** surya2k4
- **Password:** *Pass*

```
surya@Surya:~/docker-python-app/Jenkins-Docker-Demo$ docker login -u surya2k4
Password:
WARNING! Your password will be stored unencrypted in /home/surya/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
surya@Surya:~/docker-python-app/Jenkins-Docker-Demo$  sudo usermod -aG docker jenkins
[sudo] password for surya:
surya@Surya:~/docker-python-app/Jenkins-Docker-Demo$ sudo systemctl restart jenkins
surya@Surya:~/docker-python-app/Jenkins-Docker-Demo$
```

In Jenkins, go to Manage Jenkins → Manage Credentials → System → Global Credentials. Add new credentials with:

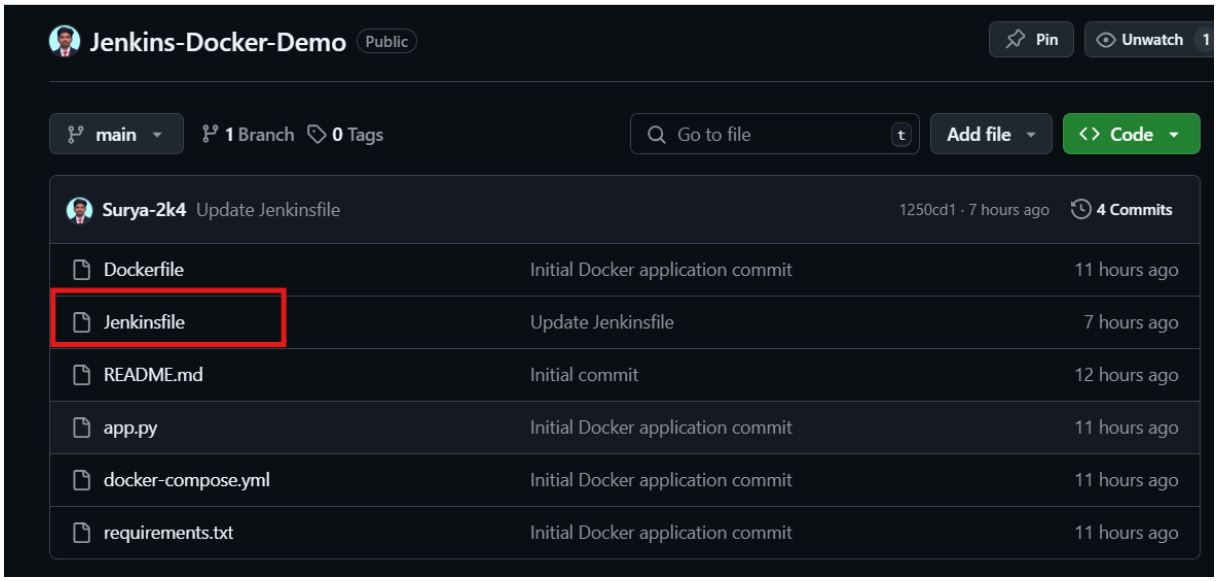
- **Username:** surya2k4
- **Password:** *(docker hub pass)*
- **ID:** docker

Step 8: Create a Jenkinsfile in your project folder and replace the credentials of your own Docker and Github

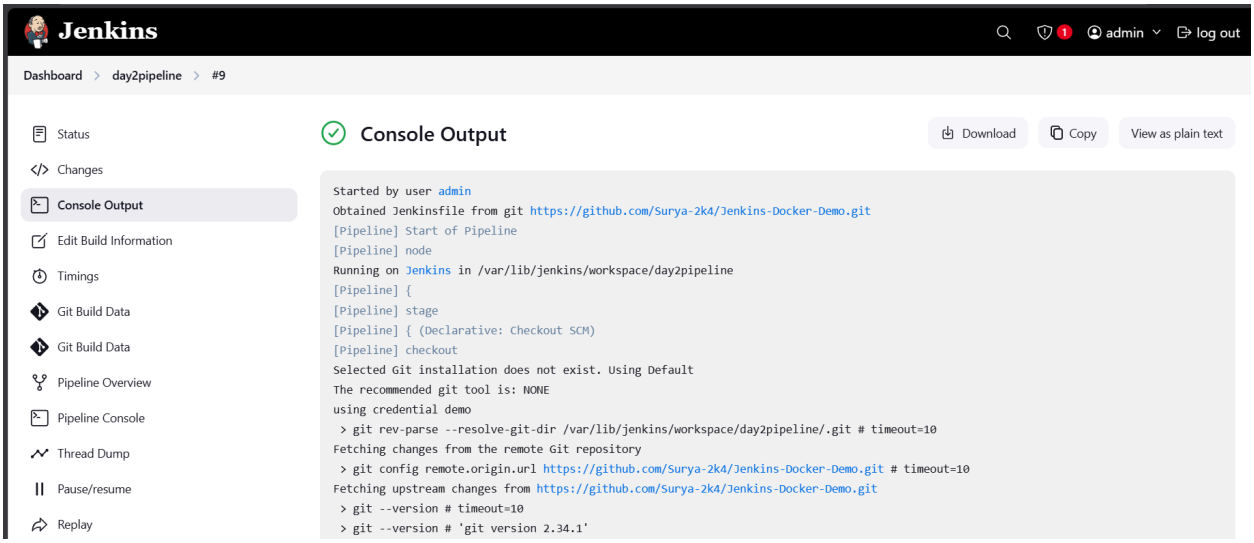
```
surya@Surya:~/docker-python-app/Jenkins-Docker-Demo$ cat Jenkinsfile
pipeline {
    agent any
    environment {
        DOCKER_IMAGE = "surya2k4/docker-app:latest"  // Docker image name
        CONTAINER_NAME = "docker-running-app"
        REGISTRY_CREDENTIALS = "docker"  // Jenkins credentials ID for Docker Hub
    }

    stages {
        stage('Checkout Code') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'demo', usernameVariable: 'GIT
_USER', passwordVariable: 'GIT_TOKEN')]) {
                    git url: "https://$GIT_USER:$GIT_TOKEN@github.com/Surya-2k4/Jenkins-Docker-
Demo.git", branch: 'main'
                }
            }
        }

        stage('Build Docker Image') {
            steps {
                script {
                    sh 'docker build -t ${DOCKER_IMAGE} .'   // Ensure the image is built with
the correct tag
                }
            }
        }
```

Push the Jenkinsfile to the repo:

```
git add Jenkinsfile

git commit -m "Added Jenkinsfile"

git push origin main
```



Step 9: In Jenkins, build the project and verify the console output.



Step 10: Test the application by visiting http://localhost:5001