

Task 6: Jenkins Pipeline – Docker Build and Push to Docker Hub

1. Objective:-

The objective of this task is to create a **Jenkins Declarative Pipeline** that:

1. Pulls the latest code from GitHub.
2. Builds a Docker image from the application code.
3. Pushes the Docker image to Docker Hub under the repository shafith04/jenkins-demo.

2. Purpose

- To implement **Continuous Integration and Continuous Deployment (CI/CD)**.
- To avoid manual steps of building and pushing Docker images.
- To ensure that **any code change** triggers an automatic pipeline, reducing errors and saving time.
- To demonstrate **Jenkins pipeline skills** with Docker and GitHub integration.

3. Tools Used:-

Tool	Purpose
Jenkins	Automates pipeline (CI/CD)
GitHub	Version control and source code repository
Git	Containerization of the application
Ubuntu	Server environment for Jenkins and Docker

4. Project Steps

Step 1: Create GitHub Repository

- Created repository: jenkins-docker-pipeline
- Added project files:
 - app.js (Node.js sample app)
 - package.json
 - Dockerfile
 - Jenkinsfile

Step 2: Write Application Code:

```
javascript

// app.js
console.log("Hello from Jenkins Docker Pipeline!");
```

- Initialized Node.js project using `npm init -y`.
- Verified app runs locally:

```
bash

node app.js
```

Output:

```
csharp

Hello from Jenkins Docker Pipeline!
```

Step 3: Create Dockerfile:-

```
dockerfile

FROM node:18-alpine
WORKDIR /app
COPY package*.json ./
RUN npm install --only=production
COPY . .
CMD ["node", "app.js"]
```

Step 4: Build Docker Image Locally:-

```
bash

docker build -t shafith04/jenkins-demo .
docker run shafith04/jenkins-demo
```

Step 5: Push Docker Image to Docker Hub:-

```
bash

docker login -u shafith04
docker push shafith04/jenkins-demo
```

- Image is now available at: Docker Hub.

Step 6: Create Jenkins Pipeline:-

- Jenkinsfile added to project:

```
groovy

pipeline {
    agent any

    environment {
        DOCKER_HUB_USER = 'shafith04'
        IMAGE_NAME = 'jenkins-demo'
    }

    stages {
        stage('Checkout Code') {
            steps {
                git branch: 'main', url: 'https://github.com/mohamedshafith04/jenkins-docker-pipeline.git'
            }
        }

        stage('Build Docker Image') {
            steps {
                sh 'docker build -t $DOCKER_HUB_USER/$IMAGE_NAME .'
            }
        }

        stage('Login to Docker Hub') {
            steps {
                withCredentials([string(credentialsId: 'dockerhub-pass', variable: 'DOCKERHUB_PASS')]) {
                    sh 'echo $DOCKERHUB_PASS | docker login -u $DOCKER_HUB_USER --password-stdin'
                }
            }
        }
    }
}
```

```

    }
  }
}

stage('Push Docker Image') {
  steps {
    sh 'docker push $DOCKER_HUB_USER/$IMAGE_NAME'
  }
}

post {
  success {
    echo '✔ Docker image built and pushed successfully!'
  }
  failure {
    echo '✗ Pipeline failed!'
  }
}
}

```

- Configured Jenkins job: Pipeline → SCM → GitHub repository.

Step 7: Run Jenkins Pipeline:-

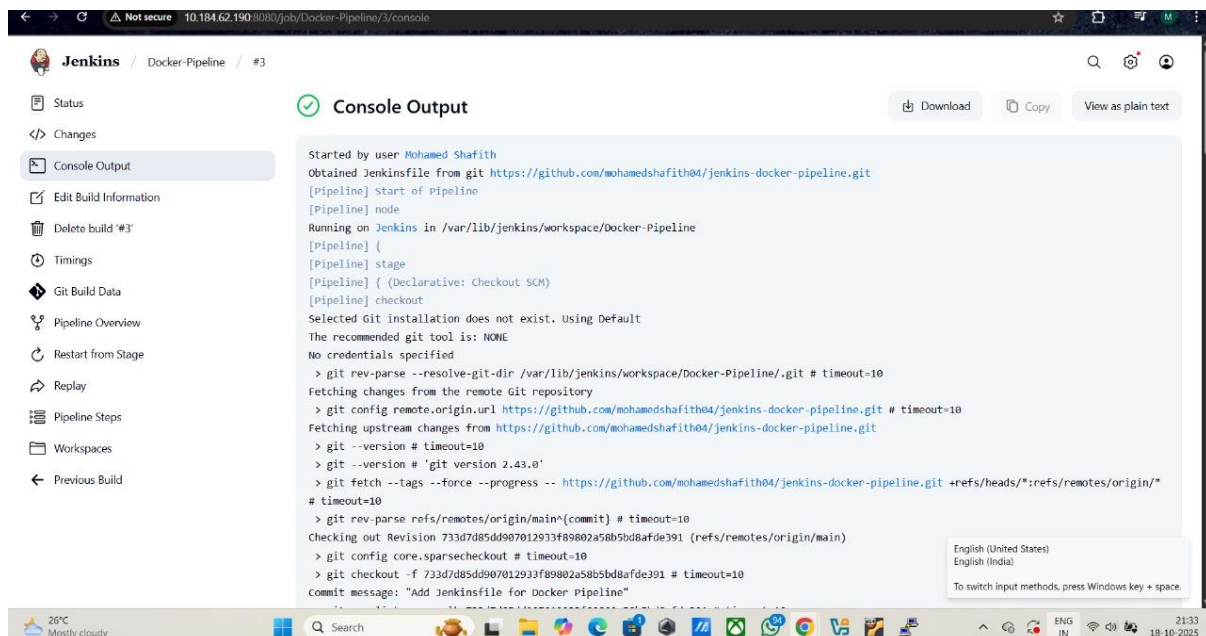
- Pipeline successfully ran:
 - Pulled latest code from GitHub
 - Built Docker image
 - Logged in to Docker Hub
 - Pushed image to Docker Hub
- Console output showed:

```
arduino

✓ Docker image built and pushed successfully!
```

5. Screenshots (Placeholders):-

1. Jenkins Dashboard with Pipeline Job
(Insert screenshot of Docker-Pipeline job)



2. Pipeline Console Output:-

(Insert screenshot showing stages successful: Checkout → Build → Push)

Fig:(1)

```
← → 🔍 10.184.62.190:3080/job/Docker-Pipeline/3/consoletext
Started by user Mohamed Shafith
Obtained Jenkinsfile from git https://github.com/mohamedshafith04/jenkins-docker-pipeline.git
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/Docker-Pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Docker-Pipeline/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/mohamedshafith04/jenkins-docker-pipeline.git # timeout=10
Fetching upstream changes from https://github.com/mohamedshafith04/jenkins-docker-pipeline.git
> git --version # 'git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/mohamedshafith04/jenkins-docker-pipeline.git :refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main # timeout=10
Checking out Revision 733d7d85d907012933f89802a58b5bd8afde391 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 733d7d85d907012933f89802a58b5bd8afde391 # timeout=10
Commit message: "Add Jenkinsfile for Docker Pipeline"
> git rev-list --no-walk 733d7d85d907012933f89802a58b5bd8afde391 # timeout=10
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] withEnv
[Pipeline] stage
[Pipeline] (Checkout Code)
[Pipeline] git
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Docker-Pipeline/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/mohamedshafith04/jenkins-docker-pipeline.git # timeout=10
Fetching upstream changes from https://github.com/mohamedshafith04/jenkins-docker-pipeline.git
> git --version # 'git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/mohamedshafith04/jenkins-docker-pipeline.git :refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main # timeout=10
Checking out Revision 733d7d85d907012933f89802a58b5bd8afde391 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 733d7d85d907012933f89802a58b5bd8afde391 # timeout=10
> git branch -a -v --no-abbrev # timeout=10
> git checkout -B main 733d7d85d907012933f89802a58b5bd8afde391 # timeout=10
Commit message: "Add Jenkinsfile for Docker Pipeline"
[Pipeline]
[Pipeline] }
```

Fig:(2)

```
← → 🔍 10.184.62.190:3080/job/Docker-Pipeline/3/consoletext
+ docker login -u shafith04 --password=stain
WARNING! Your credentials are stored unencrypted in '/var/lib/jenkins/.docker/config.json'.
Configure a credential helper to remove this warning. See
https://docs.docker.com/go/credential-store/

login Succeeded
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] (Push Docker Image)
[Pipeline] sh
+ docker push shafith04/jenkins-demo
Using default tag: latest
The push refers to repository [docker.io/shafith04/jenkins-demo]
6d9e92b0b9c: Preparing
d6bff02b148d: Preparing
2ab0d67c3ad: Preparing
d8e9d34201c: Preparing
8214d9a78a7: Preparing
f3b408cd81c: Preparing
08000c18d1d: Preparing
08000c18d1d: Waiting
f3b408cd81c: Waiting
08000c18d1d: Waiting
d8e9d34201c: Layer already exists
8214d9a78a7: Layer already exists
f3b408cd81c: Layer already exists
08000c18d1d: Layer already exists
d6bff02b148d: Pushed
6d9e92b0b9c: Pushed
2ab0d67c3ad: Pushed
latest: digest: sha256:ef1b310ac65f28718b94de428289c906c9074db497a8baf3c8820c6a2af4974 size: 1987
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] (Declarative: Post Actions)
[Pipeline] echo
Docker image built and pushed successfully!
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

3.Docker Hub Repository:-

(Insert screenshot showing *shafith04/jenkins-demo* image with latest tag).

The screenshot shows the Docker Hub interface for the repository **shafith04/jenkins-demo**. The page is titled "Repositories / jenkins-demo / General". It indicates the repository was last pushed about 1 hour ago and has a size of 42.9 MB. The "Tags" section shows a single tag named **latest** with an OS of **linux** and a type of **Image**. The "Pulled" status is "less than 1 day" and the "Pushed" status is "about 1 hour". The "Docker commands" section shows the command `docker push shafith04/jenkins-demo:tagname`. The "Build with Docker Build Cloud" section is also visible.

Tag	OS	Type	Pulled	Pushed
latest	linux	Image	less than 1 day	about 1 hour

4. Offer Letter:-

As part of the internship documentation, the offer letter confirming the internship position was included.



September 21, 2025

Dear Mohamed Shafith,

We are pleased to offer you the opportunity to join NullClass as an **Cloud Technology Intern 21-09-2025 to 21-10-2025 (1 Months)**

This offer is conditional upon Annexure A: Terms and Conditions attached below . Upon fulfilling the internship criteria, you will embark on a journey of professional growth and real-world experience with us

Congratulations!

**Elavarasi,
COO**



6. Conclusion:-

- Task 6 demonstrates a **complete CI/CD pipeline** using Jenkins.
- Any code change in app.js triggers the **automatic build and push** of Docker images
- .
- Jenkins, GitHub, and Docker Hub integration ensures that deployment is **automated, fast, and error-free**.
- This task shows practical experience in **DevOps practices** and containerized deployments.