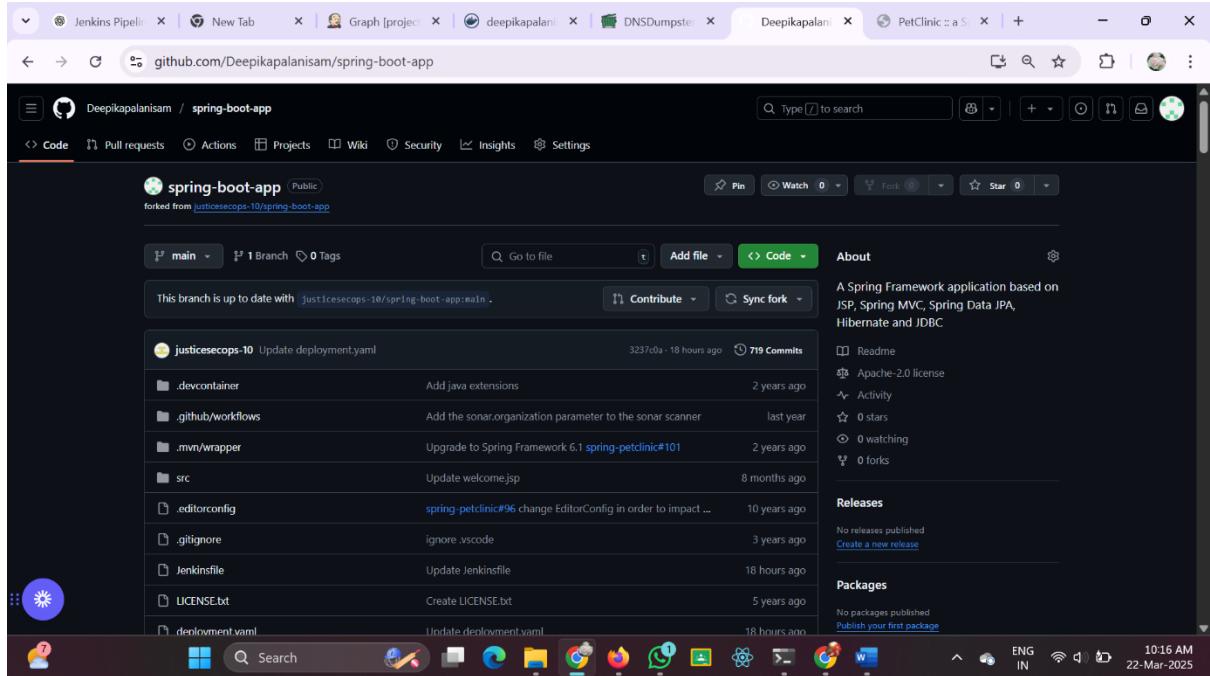


## DAY -5

Github – push the given files



## PIPELINE SCRIPT

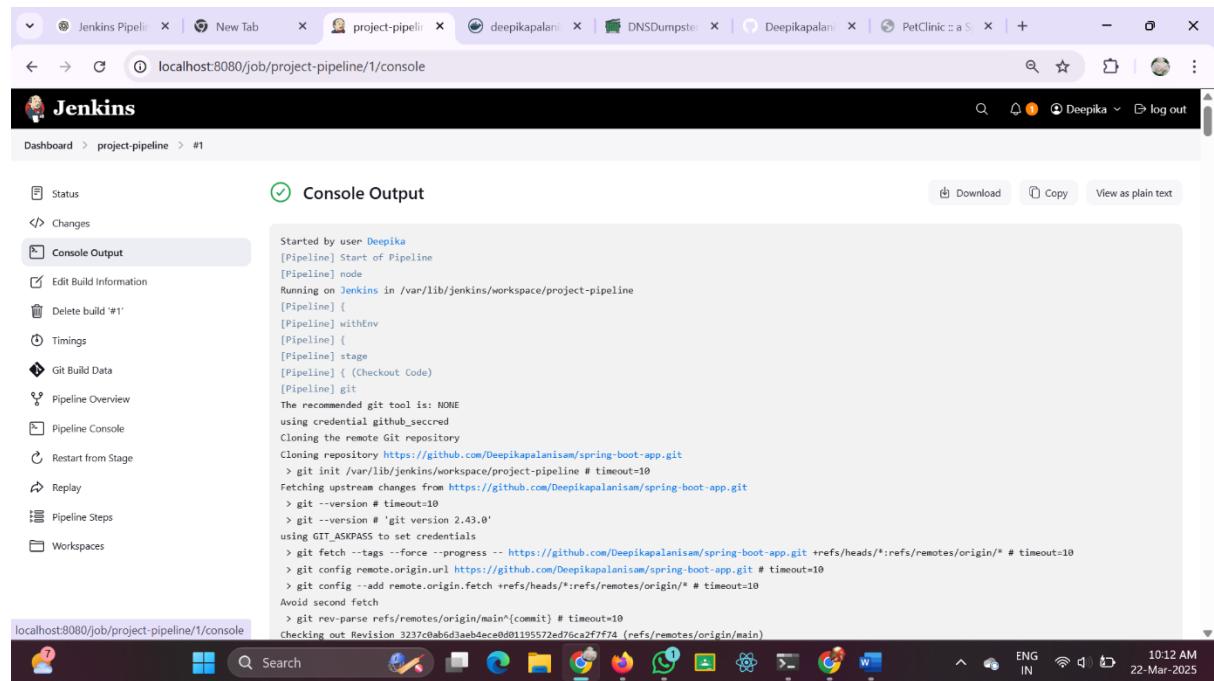
```
pipeline {  
    agent any  
  
    environment {  
  
        DOCKER_IMAGE = "deepikapalanisamy23/social-app"  
  
        DOCKER_TAG = "latest"  
  
        DOCKER_CREDENTIALS_ID = "docker"  
  
        GITHUB_CREDENTIALS_ID = "github_seccred"  
  
        KUBECONFIG = "/var/lib/jenkins/.kube/config"  
  
        MAVEN_HOME = "/opt/maven" // Set the correct Maven path  
  
    }  
  
    stages {  
  
        stage('Checkout Code') {  
  
            steps {  

```

```
        git credentialsId: GITHUB_CREDENTIALS_ID, url:  
        'https://github.com/Deepikapalanisam/spring-boot-app.git', branch: 'main'  
    }  
}  
  
stage('Build Application') {  
    steps {  
        script {  
            sh "${MAVEN_HOME}/bin/mvn clean package -DskipTests"  
        }  
    }  
}  
  
stage('Run Maven Tests') {  
    steps {  
        script {  
            catchError(buildResult: 'SUCCESS', stageResult: 'FAILURE') {  
                sh "${MAVEN_HOME}/bin/mvn test"  
            }  
        }  
    }  
}  
  
stage('Build Docker Image') {  
    steps {  
        script {  
            sh "docker build -t ${DOCKER_IMAGE}:${DOCKER_TAG} ."  
        }  
    }  
}
```

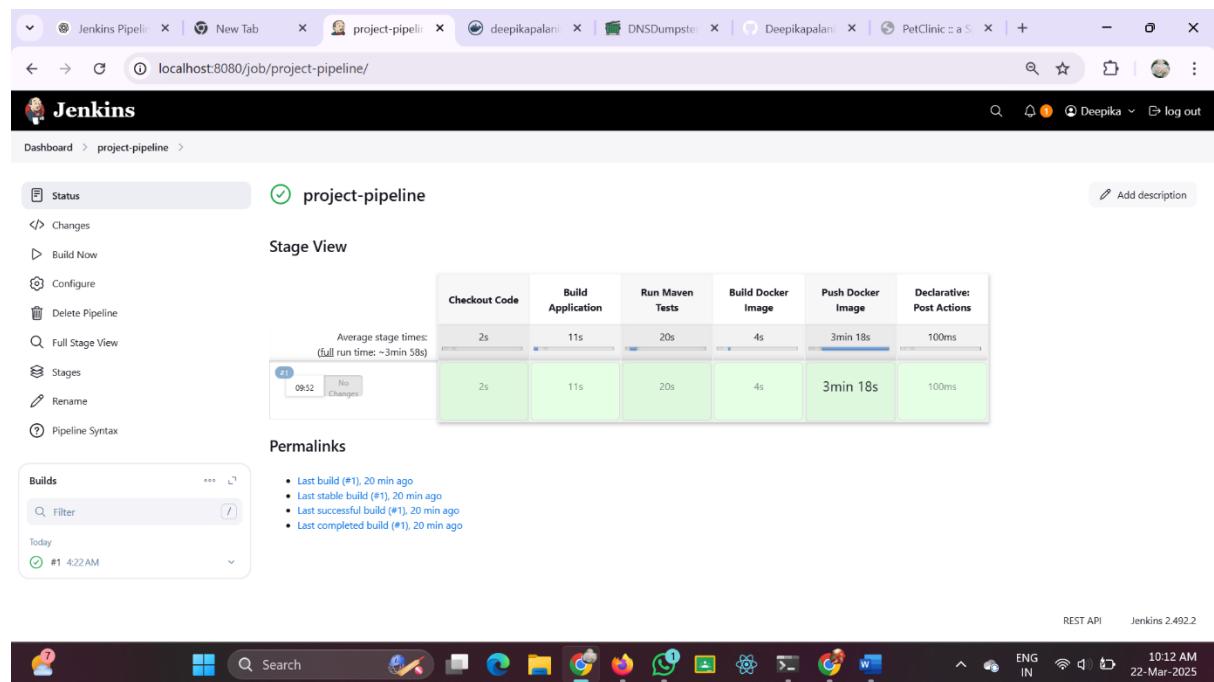
```
stage('Push Docker Image') {  
    steps {  
        withDockerRegistry([credentialsId: DOCKER_CREDENTIALS_ID, url: '']) {  
            sh "docker push ${DOCKER_IMAGE}:${DOCKER_TAG}"  
        }  
    }  
}  
  
// Uncomment this stage when Kubernetes deployment is ready  
  
// stage('Deploy to Kubernetes') {  
//     steps {  
//         script {  
//             sh ''  
//             chmod +x scripts/deploy.sh  
//             ./scripts/deploy.sh  
//             ''  
//         }  
//     }  
// }  
}  
post {  
    success {  
        echo "✅ Deployment Successful!"  
    }  
    failure {  
        echo "❌ Deployment Failed!"  
    }  
}
```

## BUILD PIPELINE



The screenshot shows the Jenkins Pipeline console output for build #1. The output details the steps taken to clone the repository from GitHub and fetch upstream changes.

```
Started by user Deepika
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/project-pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Checkout Code)
[Pipeline] git
The recommended git tool is: NONE
using credential github_secrecd
Cloning the remote Git repository
Cloning repository https://github.com/Deepikapalanisam/spring-boot-app.git
> git init /var/lib/jenkins/workspace/project-pipeline # timeout=10
Fetching upstream changes from https://github.com/Deepikapalanisam/spring-boot-app.git
> git --version # timeout=10
> git -version # git version 2.43.0'
using GIT_ASKPASS to set credentials
> git fetch --tags --force --progress -- https://github.com/Deepikapalanisam/spring-boot-app.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/Deepikapalanisam/spring-boot-app.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 3237c0ab6d3ae84ce8d01195572ed76ca2fxf74 (refs/remotes/origin/main)
```



The screenshot shows the Jenkins Pipeline stage view for the project-pipeline. It displays the execution times for each stage: Checkout Code (2s), Build Application (11s), Run Maven Tests (20s), Build Docker Image (4s), Push Docker Image (3min 18s), and Declarative: Post Actions (100ms). The stage for Push Docker Image is currently running.

Stage	Average stage time
Checkout Code	2s
Build Application	11s
Run Maven Tests	20s
Build Docker Image	4s
Push Docker Image	3min 18s
Declarative: Post Actions	100ms

**Permalinks**

- Last build (#1), 20 min ago
- Last stable build (#1), 20 min ago
- Last successful build (#1), 20 min ago
- Last completed build (#1), 20 min ago

Jenkins 2.492.2

OUTPUT : localhost:9098

qcewc wedcdwcdw

spring

