TASK 5

Step 1: Clone the Git Repository

# Clone the repository

git clone https://github.com/original-repo/spring-framework.git

# Change to the project directory

cd spring-framework

# Add your GitHub repository as the remote

git remote set-url origin <https://github.com/GokulRavi26/java_project>

# Push the project to your own repo

git push origin main

Step 2: Install Maven

# Update package lists

sudo apt update

# Install Maven

sudo apt install maven -y

mvn -version

# Remove existing Maven

sudo apt remove maven -y

# Download the latest Maven version

cd /opt

sudowget https://downloads.apache.org/maven/maven-

3/3.9.6/binaries/apache-maven-3.9.6-bin.tar.gz

# Extract and move Maven

sudo tar -xvzf apache-maven-3.9.6-bin.tar.gz

sudo mv apache-maven-3.9.6 /opt/maven

# Set up environment variables

echo 'export M2\_HOME=/opt/maven' | sudo tee -a /etc/profile.d/maven.sh

echo 'export PATH=$M2\_HOME/bin:$PATH' | sudo tee -a

/etc/profile.d/maven.sh

source /etc/profile.d/maven.sh

Verify installation:

mvn -version

which mvn

If necessary, create a symbolic link:

sudo ln -s /opt/maven/bin/mvn /usr/bin/mvn

Step 3: Build the Project with Maven

Navigate to the Jenkins workspace and build the project:

cd /var/lib/jenkins/workspace/Spring-framework

# Clean and package the project (skipping tests)

mvn clean package -DskipTests

Step 4: Set Up Jenkins Pipeline

1. Open Jenkins Dashboard → Click New Item → Select Pipeline.

2. Go to Pipeline section and add the following script:

pipeline {  
    agent any  
    environment {  
        DOCKER\_IMAGE = "gokulravi26/social-app"  
        DOCKER\_TAG = "latest"  
        DOCKER\_CREDENTIALS\_ID = "docker-hub-creds"  
        GITHUB\_CREDENTIALS\_ID = "github"  
        KUBECONFIG = "/var/lib/jenkins/.kube/config"  
    }  
  
    stages {  
        stage('Checkout Code') {  
            steps {  
                git credentialsId: GITHUB\_CREDENTIALS\_ID, url: '<https://github.com/GokulRavi26/java_project.git>', branch: 'main'  
            }  
        }  
  
        stage('Build Application') {  
            steps {  
                script {  
                    sh '${MAVEN\_HOME}/bin/mvn clean package -DskipTests'  
                }  
            }  
        }  
  
        stage('Run Maven Tests') {  
            steps {  
                script {  
                    try {  
                        sh 'mvn test'  
                    } catch (Exception e) {  
                        echo "Tests failed, but proceeding..."  
                    }  
                }  
            }  
        }  
  
        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}"  
                }  
            }  
        }  
  
        // stage('Deploy to Kubernetes') {  
        //     steps {  
        //         script {  
        //             sh '''  
        //                 chmod +x scripts/deploy.sh  
        //                 ./scripts/deploy.sh  
        //             '''  
        //         }  
        //     }  
        // }  
    }  
    post {  
        success {  
            echo "Deployment Successful!"  
        }  
        failure {  
            echo "Deployment Failed!"  
        }  
    }

}

Step 5: Fix Permissions for Jenkins

Ensure Jenkins has the correct permissions:

sudochown -R jenkins:jenkins /var/lib/jenkins/workspace/Spring-framework

sudochmod -R 775 /var/lib/jenkins/workspace/Spring-framework

Restart Jenkins to apply changes:

sudosystemctl restart jenkins

Step 6 : Run and Debug the Pipeline

After setting up everything, go to Jenkins and trigger the build. If there

are any errors:

docker images # Check if the image exists

docker ps -a # Check running containers

docker logs <container\_id> # View container logs









