**VIBUDESH R B**

**22CSR233**

**TASK 2:**

### ****Aim:****

To automate the process of checking out code from Git, installing dependencies, building the application, creating a container, and running the container using Jenkins.

### ****Steps to Run the Task:****

#### ****Step 1: Install Jenkins and Required Plugins****

If Jenkins is not installed:

sudo apt update

sudo apt install -y openjdk-11-jdk

wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -

sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

sudo apt update

sudo apt install -y jenkins

sudo systemctl start jenkins

sudo systemctl enable jenkins

* Install **Git**, **Docker**, and **Pipeline Plugin** in Jenkins.

### ****Step 2: Create a New Jenkins Job****

* Open Jenkins → **New Item** → Enter job name → Select **Pipeline** → Click **OK**.

### ****Step 3: Configure Git Webhook (Optional)****

* In **GitHub/GitLab**, go to repository settings → Webhooks → Add webhook.
* Set payload URL: http://localhost:8080/github-webhook/
* Select "Just the push event" → Save.

### ****Step 4: Define Jenkins Pipeline Script****

Go to **Pipeline** → Select **Pipeline script** and enter the following:

pipeline {

agent any

environment {

IMAGE\_NAME = "myapp"

CONTAINER\_NAME = "myapp\_container"

REPO\_URL = "https://github.com/your-repo.git"

}

stages {

stage('Checkout Code') {

steps {

git branch: 'main', url: "${REPO\_URL}"

}

}

stage('Install Dependencies') {

steps {

sh '''

sudo apt update

sudo apt install -y docker.io

'''

}

}

stage('Build Application') {

steps {

sh 'echo "Building application..."'

# Add build steps for your application (e.g., Maven, Gradle, npm install)

}

}

stage('Build Docker Image') {

steps {

sh '''

docker build -t ${IMAGE\_NAME}:latest .

'''

}

}

stage('Run Docker Container') {

steps {

sh '''

docker stop ${CONTAINER\_NAME} || true

docker rm ${CONTAINER\_NAME} || true

docker run -d --name ${CONTAINER\_NAME} -p 8080:8080 ${IMAGE\_NAME}:latest

'''

}

}

}

post {

success {

echo 'Deployment successful!'

}

failure {

echo 'Deployment failed!'

}

}

}

### ****Step 5: Save and Run the Pipeline****

* Click **Save** and **Build Now**.
* On the next push to Git, Jenkins will trigger the pipeline automatically.

### ****Step 6: Verify Deployment****

* Run docker ps on the server to check if the container is running.
* Open a web browser and go to: http://your-server-ip:8080 to check the deployed application.

















