**DevOps Assignments (Local Setup, No Cloud Required)**

**4. End-to-End CI/CD: GitHub Actions + Docker + K8s**

**Objective:** Set up a local pipeline that runs via GitHub Actions + deploys to

Minikube/kind.

**Tasks:**

**o** Create GitHub Actions workflow:

▪ Build and test app

▪ Build/push Docker image to Docker Hub (if available) or local registry

▪ Use kubectl + kubeconfig to deploy to Minikube

**o** Use secrets for credentials

**Setup instructions :**

Local pipeline with GitHub Actions, Docker, and Minikube.

## **GitHub Actions Workflow File ( .github/workflows/ci-cd.yml )**

ci-cd.yml

name: CI/CD

on:

push:

branches: [ main ] # Trigger on push to main branch

jobs:

build-test:

runs-on: ubuntu-latest

steps:

- uses: actions/checkout@v4

- name: Set up Node.js

uses: actions/setup-node@v4

with:

node-version: '18'

- name: Install dependencies

run: npm install

- name: Run tests

run: npm test

docker-build-push:

needs: build-test

runs-on: ubuntu-latest

steps:

- uses: actions/checkout@v4

- name: Build Docker image

run: docker build -t ${{ secrets.DOCKERHUB\_USERNAME }}/sample-app:${{ github.sha }} .

- name: Log in to Docker Hub

run: echo ${{ secrets.DOCKERHUB\_TOKEN }} | docker login -u ${{ secrets.DOCKERHUB\_USERNAME }} --password-stdin

- name: Push Docker image

run: docker push ${{ secrets.DOCKERHUB\_USERNAME }}/sample-app:${{ github.sha }}

deploy:

needs: docker-build-push

runs-on: ubuntu-latest

steps:

- uses: actions/checkout@v4

- name: Set up kubectl

uses: azure/setup-kubectl@v3

with:

version: 'latest'

- name: Deploy to Minikube

env:

KUBECONFIG: ${{ secrets.KUBECONFIG }}

run: |

kubectl set image deployment/sample-app \

sample-app=${{ secrets.DOCKERHUB\_USERNAME }}/sample-app:${{ github.sha }}

### **Jenkinsfile for automation**

pipeline {

agent any

tools {

nodejs 'Node18'

}

environment {

DOCKER\_IMAGE = 'my-node-app'

}

stages {

stage('Clone Repo') {

steps {

git branch: 'main', url: 'https://github.com/ramya-create/nodeJs-application.git'

}

}

stage('Install Dependencies') {

steps {

bat 'npm install'

}

}

stage('Build Docker Image') {

steps {

bat 'docker build -t my-node-app .'

}

}

// For run the docker

// stage('Run Docker Container') {

// steps {

// bat 'docker rm -f node-app || true'

// bat 'docker run -d --name node-app -p 3000:3000 my-node-app'

// }

// }

// }

// For k8s integratiom

stage('Check Kubernetes Cluster') {

steps {

bat 'kubectl config current-context'

bat 'kubectl cluster-info'

bat 'kubectl get nodes'

}

}

stage('Deploy to Kubernetes') {

steps {

script {

def deleteStatus = bat(script: 'kubectl delete -f deployment.yaml', returnStatus: true)

if (deleteStatus != 0) {

echo 'No existing deployment to delete or deletion failed, continuing...'

}

}

bat 'kubectl apply -f deployment.yaml'

}

}

stage('Expose Service') {

steps {

bat 'kubectl apply -f service.yaml'

}

}

stage('Verify Deployment') {

steps {

bat 'kubectl get pods'

bat 'kubectl get svc'

}

}

stage('Access App') {

steps {

bat 'minikube service my-node-service'

}

}

}

post {

always {

echo 'Cleaning workspace and releasing agent...'

cleanWs()

}

success {

echo '✅ Build and deployment successful!'

}

failure {

echo '❌ Build failed!'

}

}

}

#### **Trigger:**

This workflow runs **automatically** when code is **pushed to the main branch**.

### 1. Build and Test the App

### 2. Docker Build and Push

### 3. Deploy to Kubernetes (Minikube)

Deploy the updated image to Kubernetes by updating the deployment using:  
  
kubectl set image deployment/sample-app sample-app=<DOCKER\_IMAGE>

Uses KUBECONFIG secret for Kubernetes cluster access.