Final Project Day-6

Java Application Deployment in Minikube

Kubectl pipeline code:

```
pipeline {
    agent any

    stages {
        stage('SCM') {
            steps {
                git branch: 'main', url:
'https://github.com/PraneshC2005/DevOps_simple-web-app.git'
            }
        }
        stage('Build-clean') {
            steps{
                sh 'mvn clean'
            }
        }
         stage('Build-validate') {
            steps{
```

```
            sh 'mvn validate'
        }
    }
     stage('Build-compile') {
        steps{
            sh 'mvn compile'
        }
    }
     stage('Build-test') {
        steps{
            sh 'mvn test'
        }
    }
     stage('Build-package') {
        steps{
            sh 'mvn package'
        }
    }
    stage('build to images') {
        steps {
        script{
           sh "docker build -t praneshc/webapplication ."
        }
        }
```

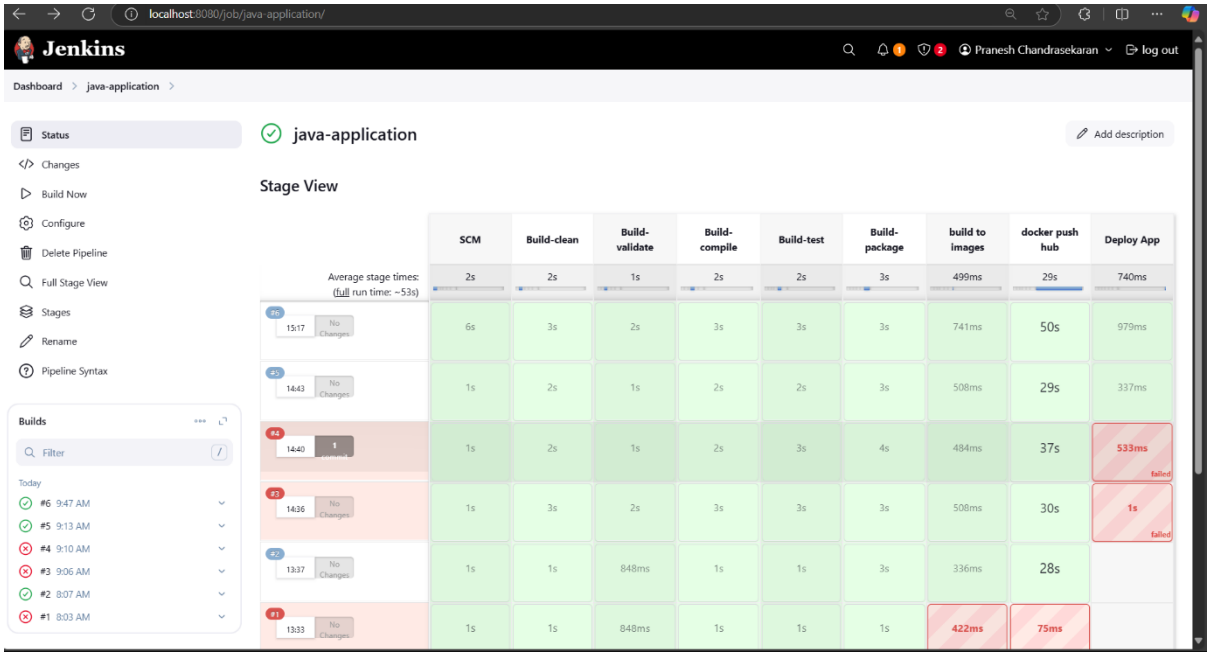
```
        }
        stage('docker push hub') {
            steps {
            script{
              withDockerRegistry(credentialsId: 'cred-2', url:
'https://index.docker.io/v1/') {
                sh 'docker push praneshc/webapplication'
            }
            }
            }
        }
            stage('Deploy App') {
        steps {
            withKubeConfig(caCertificate: '', clusterName: 'minikube',
contextName: 'minikube', credentialsId: 'kube-cred', namespace: '',
restrictKubeConfigAccess: false, serverUrl:
'https://192.168.39.226:8443') {
            sh 'kubectl apply -f Deployment.yml --validate=false'
            }


            }


            }
    }
}
```

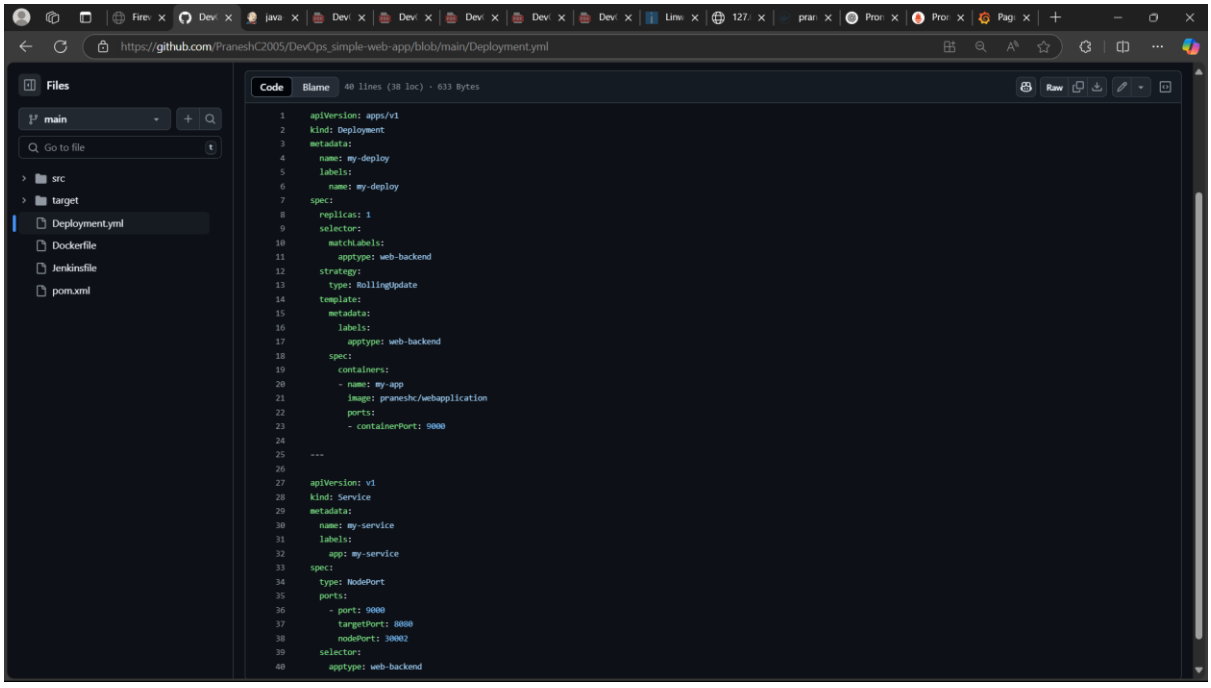
Deployment.yml:

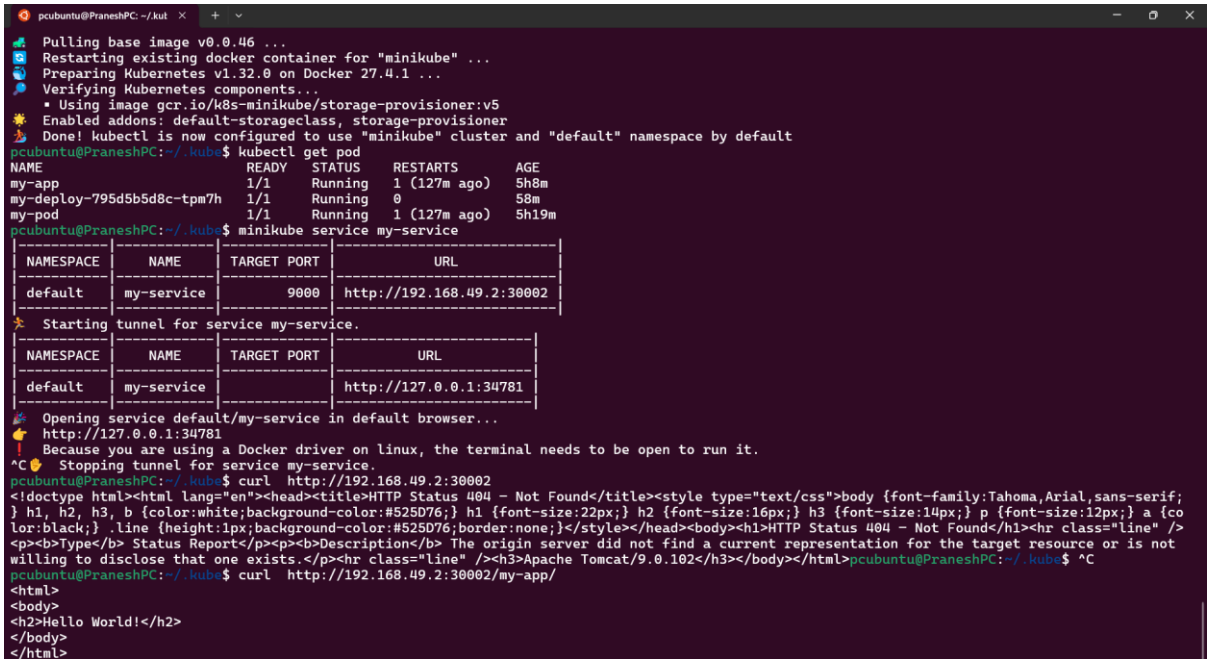
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deploy
  labels:
    name: my-deploy
spec:
  replicas: 1
  selector:
    matchLabels:
      apptype: web-backend
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        apptype: web-backend
    spec:
      containers:
      - name: my-app
        image: praneshc/webapplication
        ports:
        - containerPort: 9000
```

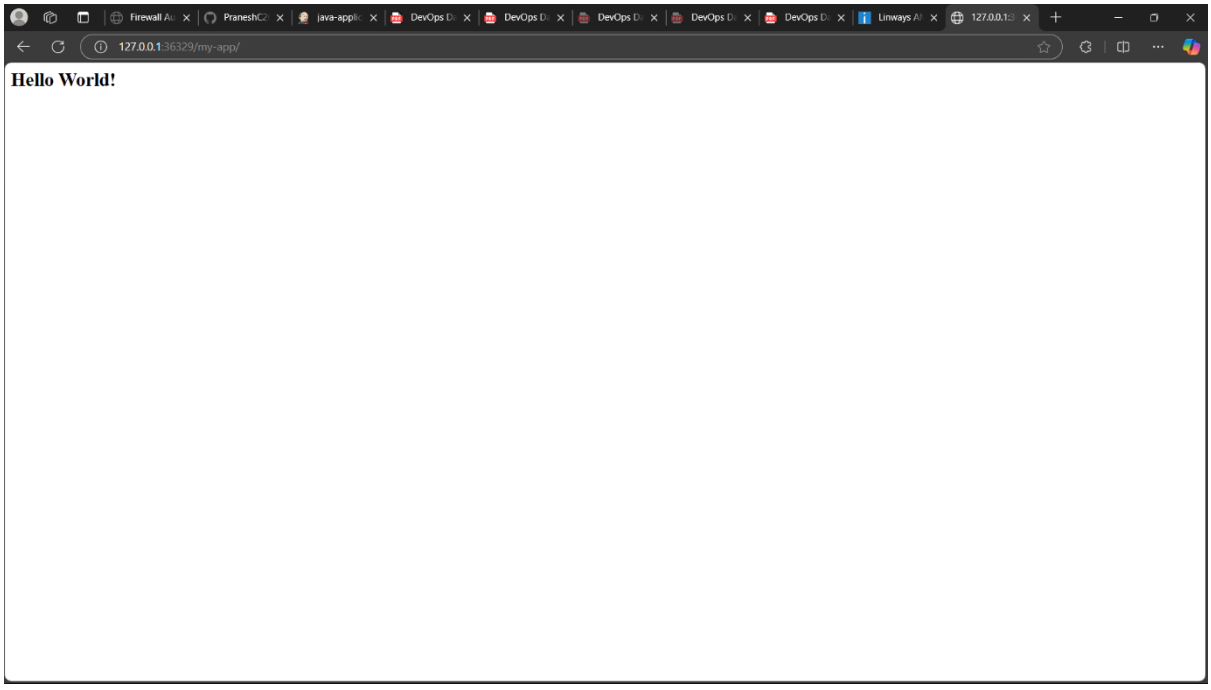
```
---


apiVersion: v1
kind: Service
metadata:
  name: my-service
  labels:
    app: my-service
spec:
  type: NodePort
  ports:
    - port: 9000
      targetPort: 8080
      nodePort: 30002
  selector:
    apptype: web-backend
```

Output:


localhost:8080/job/java-application/
Jenkins
Pranesh Chandrasekaran
log out
Dashboard
java-application
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
Stages
Rename
Pipeline Syntax
Builds
Filter
Today
#6 9:47 AM
#5 9:13 AM
#4 9:10 AM
#3 9:06 AM
#2 8:07 AM
#1 8:03 AM
java-application
Add description
Stage View
SCM
Build-clean
Build-validate
Build-compile
Build-test
Build-package
build to images
docker push hub
Deploy App
Average stage times:
(full run time: ~53s)
2s
2s
1s
2s
2s
3s
499ms
29s
740ms
15:17
6s
3s
2s
3s
3s
3s
741ms
50s
979ms
14:43
1s
2s
1s
2s
2s
3s
508ms
29s
337ms
14:40
1s
2s
1s
2s
3s
4s
484ms
37s
533ms
failed
14:36
1s
3s
2s
3s
3s
3s
508ms
30s
1s
failed
13:37
1s
1s
848ms
1s
1s
3s
336ms
28s
13:33
1s
1s
848ms
1s
1s
1s
422ms
75ms


https://github.com/PraneshC2005/DevOps_simple-web-app/blob/main/Deployment.yml
Files
main
Go to file
src
target
Deployment.yml
Dockerfile
Jenkinsfile
pom.xml
Code
Blame
40 lines (38 loc) · 633 Bytes
Raw
apiVersion: apps/v1
kind: Deployment
metadata:
name: my-deploy
labels:
name: my-deploy
spec:
replicas: 1
selector:
matchLabels:
apptype: web-backend
strategy:
type: RollingUpdate
template:
metadata:
labels:
apptype: web-backend
spec:
containers:
- name: my-app
image: praneshc/webapplication
ports:
- containerPort: 9000
---
apiVersion: v1
kind: Service
metadata:
name: my-service
labels:
app: my-service
spec:
type: NodePort
ports:
- port: 9000
targetPort: 8080
nodePort: 30002
selector:
apptype: web-backend


pcubuntu@PraneshPC: ~/.kut
Pulling base image v0.0.46 ...
Restarting existing docker container for "minikube" ...
Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
Verifying Kubernetes components...
▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
Enabled addons: default-storageclass, storage-provisioner
Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
pcubuntu@PraneshPC:~/.kube$ kubectl get pod
NAME READY STATUS RESTARTS AGE
my-app 1/1 Running 1 (127m ago) 5h8m
my-deploy-795d5b5d8c-tpm7h 1/1 Running 0 58m
my-pod 1/1 Running 1 (127m ago) 5h19m
pcubuntu@PraneshPC:~/.kube$ minikube service my-service
| NAMESPACE | NAME | TARGET PORT | URL |
| default | my-service | 9000 | http://192.168.49.2:30002 |
Starting tunnel for service my-service.
| NAMESPACE | NAME | TARGET PORT | URL |
| default | my-service | | http://127.0.0.1:34781 |
Opening service default/my-service in default browser...
http://127.0.0.1:34781
Because you are using a Docker driver on linux, the terminal needs to be open to run it.
^C Stopping tunnel for service my-service.
pcubuntu@PraneshPC:~/.kube$ curl http://192.168.49.2:30002
<!doctype html><html lang="en"><head><title>HTTP Status 404 – Not Found</title><style type="text/css">body {font-family:Tahoma,Arial,sans-serif;
} h1, h2, h3, b {color:white;background-color:#525D76;} h1 {font-size:22px;} h2 {font-size:16px;} h3 {font-size:14px;} p {font-size:12px;} a {co
lor:black;} .line {height:1px;background-color:#525D76;border:none;}</style></head><body><h1>HTTP Status 404 – Not Found</h1><hr class="line" />
Type Status Report Description The origin server did not find a current representation for the target resource or is not
willing to disclose that one exists. <hr class="line" /><h3>Apache Tomcat/9.0.102</h3></body></html>pcubuntu@PraneshPC:~/.kube$ ^C
pcubuntu@PraneshPC:~/.kube$ curl http://192.168.49.2:30002/my-app/
<html>
<body>
<h2>Hello World!</h2>
</body>
</html>


127.0.0.1:36329/my-app/
Hello World!