Name: Abinaya B T

Roll no: 22ISR001

Deploying a Simple HTML Website Using Kubernetes

Introduction

Deployment of simple HTML website using Docker, Kubernetes, and Minikube. The deployment involves containerizing an HTML file using Nginx, pushing the Docker image to a registry, and orchestrating it with Kubernetes.

Steps to Deploy

1. Create Kubernetes Deployment

• Create deployment.yaml:

```
abinaya@ABINAYA:~$ vim deployment.yaml
abinaya@ABINAYA: ~
apiVersion: apps/v1
kind: Deployment
metadata:
 name: sample-app
spec:
 replicas: 2
 selector:
   matchLabels:
     app: sample-app
 template:
   metadata:
     labels:
      app: sample-app
     containers:
     - name: sample-app
       image: abinayabalusamy/bookbarter
       ports:
       - containerPort: 80
"deployment.yaml" 19L, 339B
```

2. Create Kubernetes Service

3. Deploy the Application in Kubernetes

```
abinaya@ABINAYA:~  

abinaya@ABINAYA:~  

minikube v1.35.0 on Ubuntu 24.04 (amd64)

Using the docker driver based on existing profile

Starting "minikube" primary control-plane node in "minikube" cluster

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: storage-provisioner, default-storageclass

Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default abinaya@ABINAYA:~  

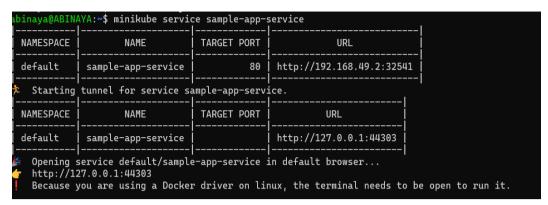
minikube

type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

```
^Cabinaya@ABINAYA:~kubectl apply -f deployment.yami
deployment.apps/sample-app configured
```

```
abinaya@ABINAYA:~$ kubectl apply -f service.yaml
service/sample-app-service created
```

4. Access the Application



5.Output

