

TASK 4 – Kubernetes Using Shell Script

Step 1: MiniKube

Start the minikube using minikube start command

```
vijay@LAPTOP-KFMKT43R:~$ minikube start
🐳 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: default-storageclass, storage-provisioner
🏁 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

Step 2: Folder Creation

Create a folder named task4

```
vijay@LAPTOP-KFMKT43R:~$ mkdir task4
```

Step 3: New Yaml File

Create a new vim file named devops.yaml

```
vijay@LAPTOP-KFMKT43R:~/task4$ vim devops.yaml
```

Step 4: Yaml file

Enter the yaml file code using the insert

```

apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: springboot-app
    name: springboot-app
spec:
  replicas: 1
  selector:
    matchLabels:
      app: springboot-app
  template:
    metadata:
      labels:
        app: springboot-app
    spec:
      containers:
        - name: my-springboot-app
          image: vishal15276t/petclinic
          imagePullPolicy: Always
          ports:
            - containerPort: 8080
              name: http
              protocol: TCP
# service type loadbalancer
---
apiVersion: v1
kind: Service
metadata:
  labels:
    app: springboot-app
    k8s-app: springboot-app
    name: springboot-app
spec:
  ports:
    - name: http
      port: 8080
      protocol: TCP
      targetPort: 8080

```

Step 5: Apply

Apply the changes made in the devops.yaml file

```

vijay@LAPTOP-KFMKT43R:~/task4$ kubectl apply -f devops.yaml
deployment.apps/springboot-app created

```

Step 6: Get Pods

Get the pods information to check if it is running or not.

```

vijay@LAPTOP-KFMKT43R:~/task4$ kubectl get pods
NAME                                READY   STATUS              RESTARTS   AGE
pet-75fbddbcfd-ptxr6               1/1     Running             2 (23m ago) 17h
pet1-664d6b6749-58pdz              1/1     Running             1 (23m ago) 17h
r1-576d6445f7-298wt               1/1     Running             3 (23m ago) 22h
springboot-app-7b985f9bc8-j86bb    0/1     ContainerCreating   0           4s

```

Step 7: Service

Open the service springboot-app in the browser

```

vijay@LAPTOP-KFMKT43R:~/task4$ minikube service springboot-app

```

NAMESPACE	NAME	TARGET PORT	URL
default	springboot-app	http/8080	http://192.168.49.2:32310

```

🚀 Starting tunnel for service springboot-app.

```

NAMESPACE	NAME	TARGET PORT	URL
default	springboot-app		http://127.0.0.1:33391

```

🌐 Opening service default/springboot-app in default browser...
👉 http://127.0.0.1:33391
! Because you are using a Docker driver on linux, the terminal needs to be open to run it.

```

Step 8: Output

The output is shown in the browser in the localhost url present

