TASK 4 – Kubernet Using Shell Script

Step 1: MiniKube

Start the minikube using minikube start command

```
vishal@LAPTOP-U45BV05I:~$ minikube start
😄  minikube v1.35.0 on Ubuntu 24.04 (amd64)
✨  Using the docker driver based on existing profile

❗  The requested memory allocation of 2200MiB does not leave room for system overhead (total system memory: 2901MiB). You may face stability is
sues.
💡  Suggestion: Start minikube with less memory allocated: 'minikube start --memory=2200mb'

👍  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
```

Step 2: Folder Creation

Create a folder named task4

```
vishal@LAPTOP-U45BV05I:~$ mkdir task4
```

Step 3: New Yaml File

Create a new vim file named devops.yaml

```
vishal@LAPTOP-U45BV05I:~/task4$ vim devops.yaml
```

Step 4: Yaml file

Enter the yaml file code using the insert

```yaml
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

```
vishal@LAPTOP-U45BV05I:~/task4$ kubectl apply -f devops.yaml
deployment.apps/springboot-app configured
service/springboot-app unchanged
```

Step 6: Get Pods

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

```
vishal@LAPTOP-U45BV05I:~/task4$ kubectl get pods
NAME                              READY   STATUS    RESTARTS      AGE
petclinic-6dc86bd65-gz7th         1/1     Running   1 (38m ago)   17h
r1-7b886b659-f2sv6                1/1     Running   2 (38m ago)   22h
r2-f784c9f59-7f7g9                1/1     Running   2 (38m ago)   22h
springboot-app-6cc66b9c5f-wm6fh   1/1     Running   0             4s
springboot-app-6ffb9888cf-l87fq   1/1     Running   0             22m
```

Step 7: Service

Open the service springboot-app in the browser

```
vishal@LAPTOP-U45BV05I:~/task4$ minikube service springboot-app
|-----------|----------------|-------------|---------------------------|
| NAMESPACE |      NAME      | TARGET PORT |            URL            |
|-----------|----------------|-------------|---------------------------|
| default   | springboot-app | http/80     | http://192.168.49.2:32472 |
|-----------|----------------|-------------|---------------------------|
🏃  Starting tunnel for service springboot-app.
|-----------|----------------|-------------|------------------------|
| NAMESPACE |      NAME      | TARGET PORT |          URL           |
|-----------|----------------|-------------|------------------------|
| default   | springboot-app |             | http://127.0.0.1:43723 |
|-----------|----------------|-------------|------------------------|
🎉  Opening service default/springboot-app in default browser...
👉  http://127.0.0.1:43723
❗  Because you are using a Docker driver on linux, the terminal needs to be open to run it.
^C✋  Stopping tunnel for service springboot-app.
```

Step 8: Output

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