Task – 4

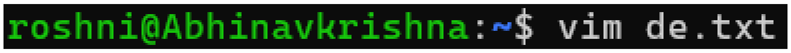
Deploying a sample application using Kubernetes

Introduction

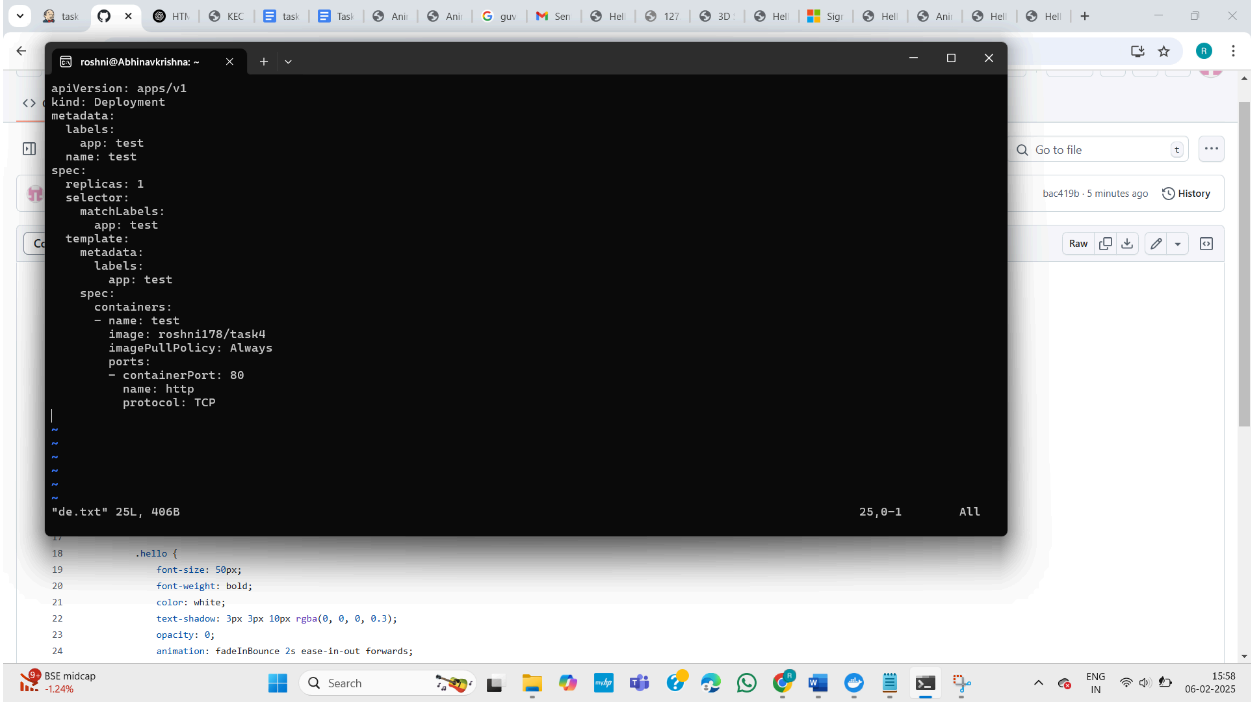
This guide outlines the steps to deploy a sample application using Kubernetes. It covers setting up a Deployment and a Service configuration, applying them with kubectl, and accessing the deployed application through Minikube. Additionally, it includes best practices for monitoring, scaling, and troubleshooting Kubernetes deployments.

- **Set Up Kubernetes Environment** – Install and start Minikube and kubectl.
- **Create a Deployment** – Define a YAML file to manage application pods.
- **Expose the Application** – Configure a Service to enable network access.
- **Apply Configurations** – Use kubectl apply to deploy resources.
- **Verify Deployment** – Check pod status, logs, and resource utilization.
- **Access the Application** – Retrieve the Minikube service URL for interaction.
- **Scale the Deployment** – Adjust replicas for better availability and load balancing.
- **Monitor and Troubleshoot** – Use kubectl logs, kubectl describe, and dashboards to resolve issues.

1. Deployment script file:



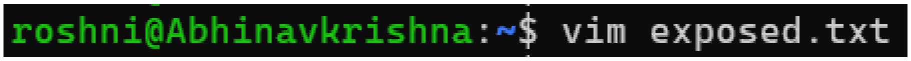
```
roshni@Abhinavkrishna:~$ vim de.txt
```



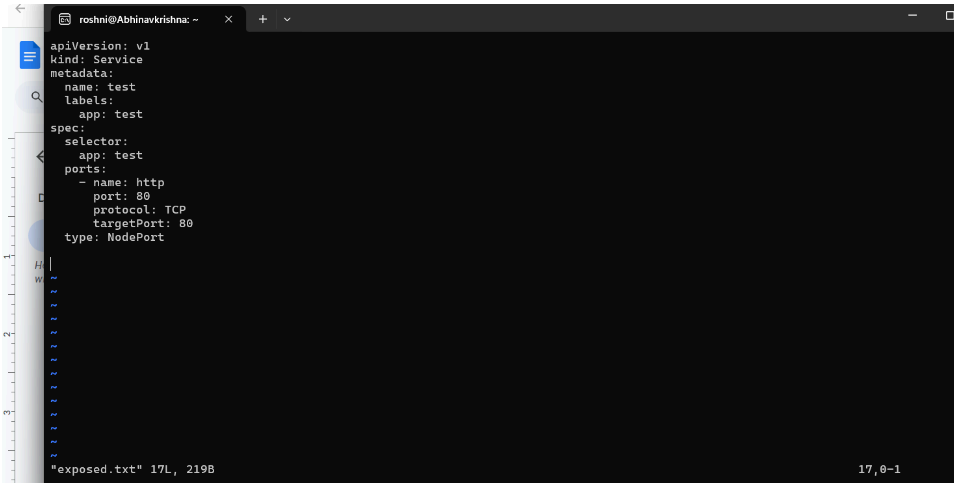
```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: test
  name: test
spec:
  replicas: 1
  selector:
    matchLabels:
      app: test
  template:
    metadata:
      labels:
        app: test
    spec:
      containers:
      - name: test
        image: roshni178/task4
        imagePullPolicy: Always
        ports:
        - containerPort: 80
          name: http
          protocol: TCP

~
~
~
~
~
~
"de.txt" 25L, 406B                                                      25,0-1        All
```

```
        .hello {
            font-size: 50px;
            font-weight: bold;
            color: white;
            text-shadow: 3px 3px 10px rgba(0, 0, 0, 0.3);
            opacity: 0;
            animation: fadeInBounce 2s ease-in-out forwards;
```

2. Expose script file:

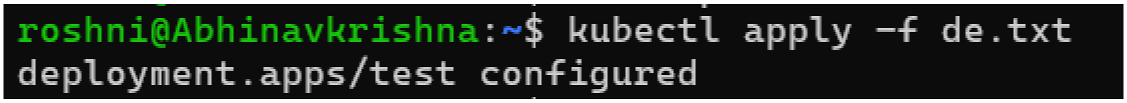


```
roshni@Abhinavkrishna:~$ vim exposed.txt
```



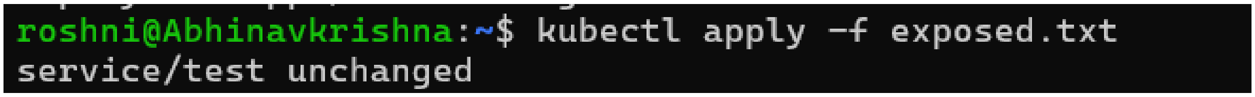
```
apiVersion: v1
kind: Service
metadata:
  name: test
  labels:
    app: test
spec:
  selector:
    app: test
  ports:
    - name: http
      port: 80
      protocol: TCP
      targetPort: 80
  type: NodePort

~
~
~
~
~
~
~
~
~
~
~
~
~
~
"exposed.txt" 17L, 219B                                                 17,0-1
```

3. To run deployment script file:



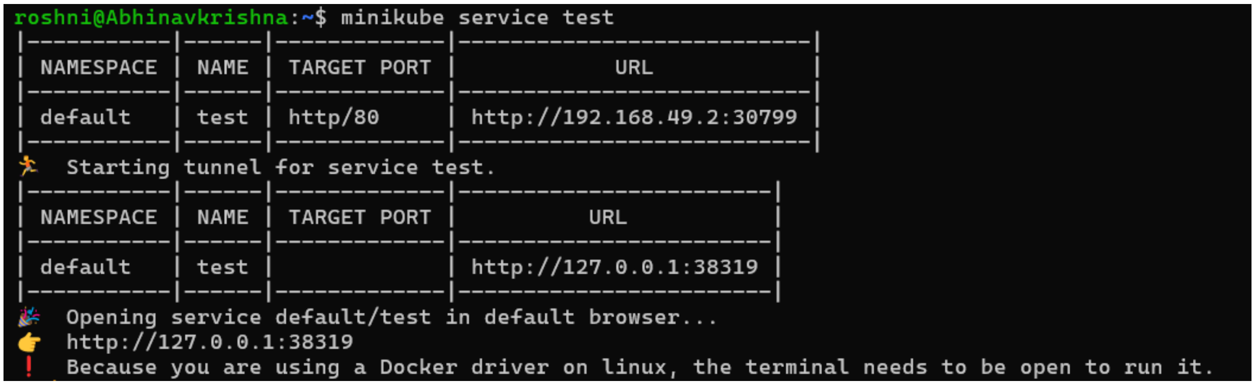
```
roshni@Abhinavkrishna:~$ kubectl apply -f de.txt
deployment.apps/test configured
```

4. To run exposed script file:



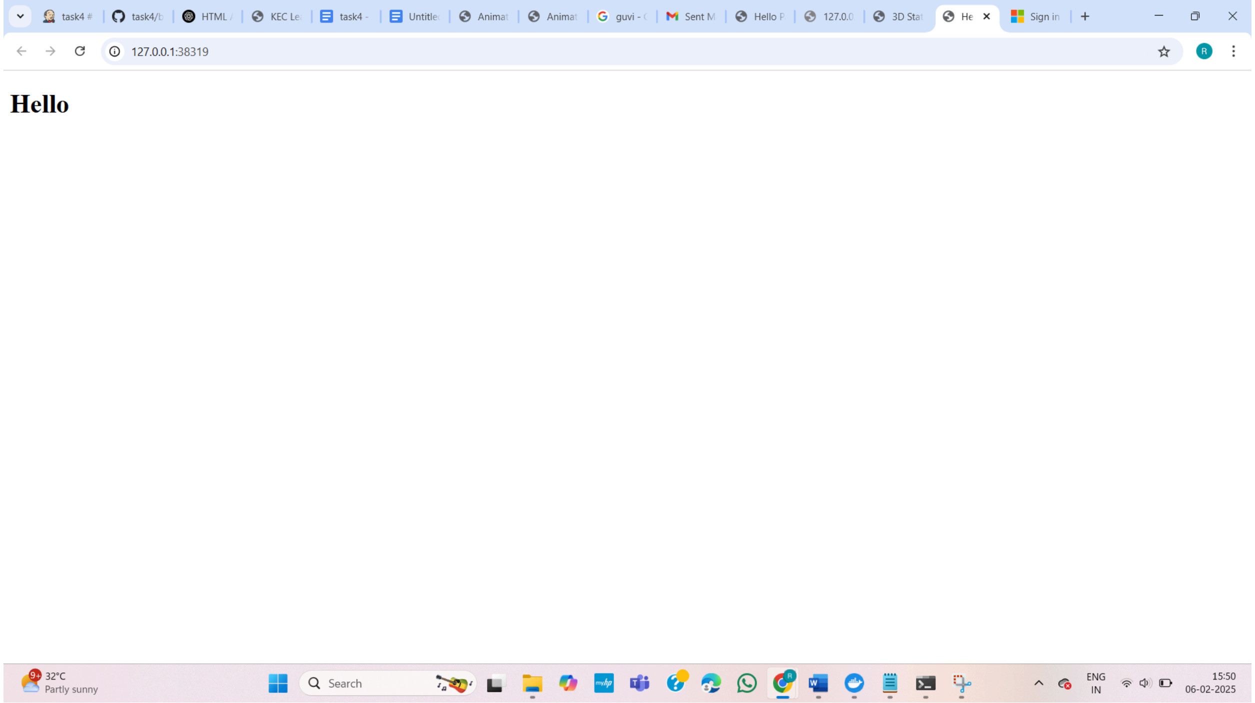
```
roshni@Abhinavkrishna:~$ kubectl apply -f exposed.txt
service/test unchanged
```

5. To start the service:



```
roshni@Abhinavkrishna:~$ minikube service test
|-----------|------|-------------|---------------------------|
| NAMESPACE | NAME | TARGET PORT |            URL            |
|-----------|------|-------------|---------------------------|
| default   | test | http/80     | http://192.168.49.2:30799 |
|-----------|------|-------------|---------------------------|
🏃  Starting tunnel for service test.
|-----------|------|-------------|------------------------|
| NAMESPACE | NAME | TARGET PORT |          URL           |
|-----------|------|-------------|------------------------|
| default   | test |             | http://127.0.0.1:38319 |
|-----------|------|-------------|------------------------|
🎉  Opening service default/test in default browser...
👉  http://127.0.0.1:38319
❗  Because you are using a Docker driver on linux, the terminal needs to be open to run it.
```

6. Chrome output:



Hello