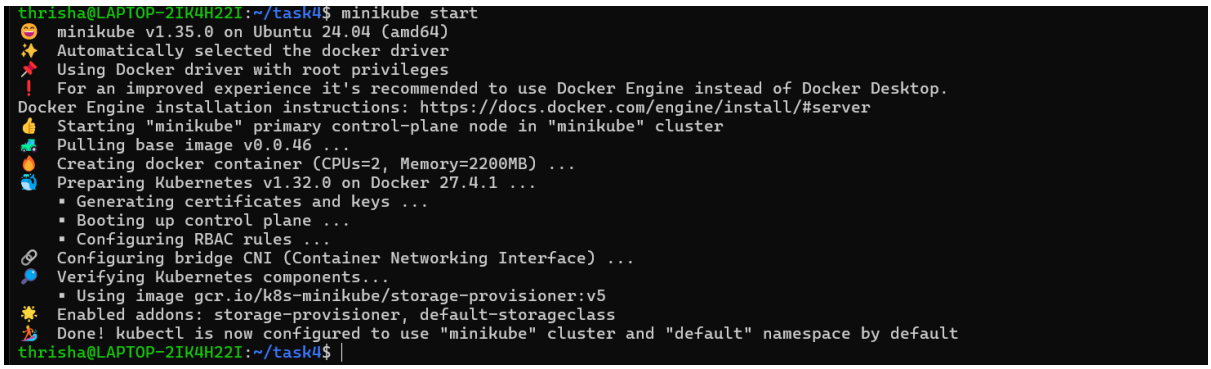
TASK 4 – Kubernet Using Shell Script

Step 1: MiniKube

Start the minikube using minikube start command



Step 2: Folder Creation

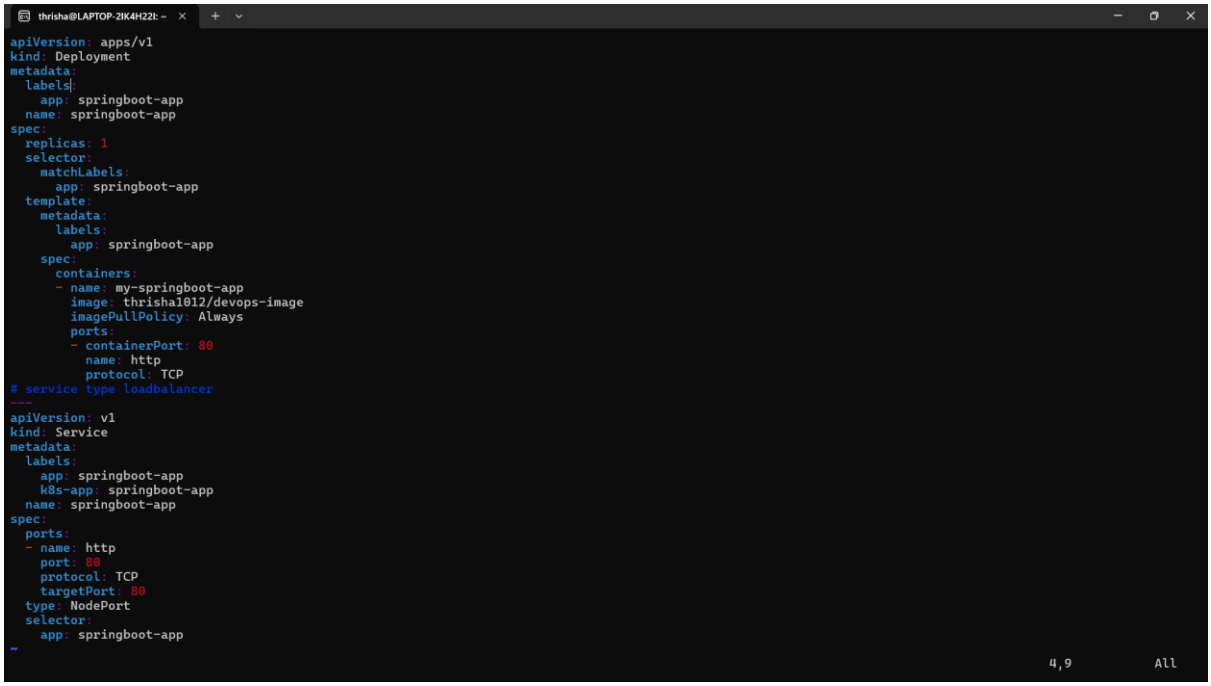
Create a folder named task4

Step 3: New Yaml File

Create a new vim file named scripts.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: thrisha1012/devops-image
        imagePullPolicy: Always
        ports:
        - containerPort: 80
          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: 80
    protocol: TCP
    targetPort: 80
  type: NodePort
  selector:
    app: springboot-app
~
                                                                    4,9           All
```

```
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: thrisha1012/devops-image
        imagePullPolicy: Always
        ports:
        - containerPort: 80
          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: 80
    protocol: TCP
    targetPort: 80
  type: NodePort
  selector:
    app: springboot-app
~
                                                                    4,9           All
```

Step 5: Apply

Apply the changes made in the devops.yaml file

```
thrisha@LAPTOP-2IK4H22I:~/task4$ vim script.yaml
thrisha@LAPTOP-2IK4H22I:~/task4$ kubectl apply -f script.yaml
deployment.apps/springboot-app created
service/springboot-app created
thrisha@LAPTOP-2IK4H22I:~/task4$ kubectl get pods
NAME                              READY   STATUS              RESTARTS   AGE
springboot-app-96bd6f6dd-dcsms    0/1     ContainerCreating   0          14s
```

Step 6: Get Pods

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

```
thrisha@LAPTOP-2IK4H22I:~/task4$ kubectl get pods
NAME                              READY   STATUS    RESTARTS   AGE
springboot-app-96bd6f6dd-dcsms   1/1     Running   0          58s
```

Step 7: Service

Open the service springboot-app in the browser

```
thrisha@LAPTOP-2IK4H22I:~/task4$ minikube service springboot-app
|-----------|----------------|-------------|------------------------|
| NAMESPACE |      NAME      | TARGET PORT |          URL           |
|-----------|----------------|-------------|------------------------|
| default   | springboot-app | http/80     | http://192.168.58.2:31879 |
|-----------|----------------|-------------|------------------------|
🏃  Starting tunnel for service springboot-app.
|-----------|----------------|-------------|------------------------|
| NAMESPACE |      NAME      | TARGET PORT |          URL           |
|-----------|----------------|-------------|------------------------|
| default   | springboot-app |             | http://127.0.0.1:36223 |
|-----------|----------------|-------------|------------------------|
🎉  Opening service default/springboot-app in default browser...
👉  http://127.0.0.1:36223
❗  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