Experiment -3.3

Student Name: Chayan Gope
UID: 22BDO10036
Branch: AIT-CSE-DevOps
Section/Group: 22BCD-1(A)
Semester: 5
Date of Performance: 28-10-24
Subject Name: Docker and Kubernetes
Subject Code: 22CSH-343

1. Aim/Overview of the practical:

Deploying a Node.js Application on Kubernetes with IBM Containers.

2. Apparatus: PC, Docker Engine, Kubernetes, Minikube, Ubuntu Linux

3. Steps for experiment/practical:

Step 1: Create YAML Manifests for the Pods

1. pod-a.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-a
spec:
  containers:
    - name: container-a
      image: nginx
      ports:
        - containerPort: 80
      volumeMounts:
        - name: static-content
          mountPath: /usr/share/nginx/html
  volumes:
    - name: static-content
      configMap:
        name: static-web-content
```

2. pod-b.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-b
spec:
  containers:
    - name: container-b
      image: nginx
      ports:
        - containerPort: 80
      volumeMounts:
        - name: static-content
          mountPath: /usr/share/nginx/html
  volumes:
    - name: static-content
      configMap:
        name: static-web-content
```

3. static-web-content.yaml

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: static-web-content
data:
  index.html: |
    <html>
      <head><title>Static Web Page</title></head>
      <body>
        <h1>Hi!! I am Chayan Gope</h1>
        <h3>Welcome to my default static web page!</h3>
      </body>
    </html>
```

Step 2: Apply the YAML Manifests to Create Pods and ConfigMap

```
chayan@chayanvm:~/Desktop/exp10$ minikube start
😄  minikube v1.34.0 on Ubuntu 22.04
✨  Using the docker driver based on existing profile
👍  Starting "minikube" primary control-plane node in "mir
🚜  Pulling base image v0.0.45 ...
🔄  Restarting existing docker container for "minikube" ..

chayan@chayanvm:~/Desktop/exp10$ alias kubectl="minikube kubectl --"
chayan@chayanvm:~/Desktop/exp10$ kubectl apply -f pod-a.yaml
    > kubectl.sha256:  64 B / 64 B [-------------------------] 100.00% ?
    > kubectl:  53.77 MiB / 53.77 MiB [--------------] 100.00% 4.65 MiB p
pod/pod-a created
chayan@chayanvm:~/Desktop/exp10$ kubectl apply -f pod-b.yaml
pod/pod-b created
chayan@chayanvm:~/Desktop/exp10$ kubectl apply -f static-web-content.yaml
configmap/static-web-content created
```

Step 3: Check the Status of the Pods

```
chayan@chayanvm:~/Desktop/exp10$ kubectl get pods
NAME                                  READY   STATUS             RESTARTS      AGE
nginx-web-5cb57cfb4b-r9qhg            1/1     Running            4 (11d ago)   12d
nodeapp-deployment-847f485468-tbzls   0/1     CrashLoopBackOff   9 (56s ago)   11d
pod-a                                 1/1     Running            0             105s
pod-b                                 1/1     Running            0             97s
```

Step 4: Enable Communication Between Pods

1. service-a.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: service-a
spec:
  selector:
    app: pod-a
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

2. service-b.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: service-b
spec:
  selector:
    app: pod-b
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

```
chayan@chayanvm:~/Desktop/exp10$ kubectl apply -f service-a.yaml
service/service-a created

chayan@chayanvm:~/Desktop/exp10$ kubectl apply -f service-b.yaml
service/service-b created
```

Step 5: Verify Communication

1. **Check the ClusterIP of the services:**

```
chayan@chayanvm:~/Desktop/exp10$ kubectl get svc service-a
NAME        TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)   AGE
service-a   ClusterIP   10.104.130.199   <none>        80/TCP    6m25s
chayan@chayanvm:~/Desktop/exp10$ kubectl get svc service-b
NAME        TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
service-b   ClusterIP   10.110.172.63   <none>        80/TCP    3m30s
```

2. **Access the services using the following command to ensure that they serve the static web content:**

```
chayan@chayanvm:~/Desktop/exp10$ kubectl exec -it pod-a -- /bin/sh
# curl http://localhost
<html>
  <head><title>Static Web Page</title></head>
  <body>
    <h1>Hi!! I am Chayan Gope</h1>
    <h3>Welcome to my default static web page!</h3>
  </body>
</html>
```

Kubernetes ensures immutability by maintaining the existing pods until the new ones are ready with the updated content.

Learning outcomes (What I have learned):

1. I have learned the concepts of containerization and virtualization.

2. I have learned about orchestration and orchestration tools.

3. I have learned about Kubernetes and its architecture.

4. I have learned the purpose of using microservice architecture over monolithic.

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			