Experiment -3.2

Student Name: Chayan Gope
UID: 22BDO10036
Branch: AIT-CSE-DevOps
Section/Group: 22BCD-1(A)
Semester: 5
Date of Performance: 21-10-24
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:

- **Creating the Node.js application**
  1. Start the node application using the **npm init** command.
  2. Create the **index.js** file.

```
var express = require('express'); var app = express(); app.get('/', function
(req, res) { res.send('{ "response": "Hey There! My name is Chayan
Gope." }');
}); app.get('/about', function (req, res) { res.send('{ "response": "Hello
World! this is the 'about' section." }');
}); app.get('/student', function (req, res) { res.send('{ "response":
"This is the 'student' info section." }');
```

```
});app.listen(process.env.PORT ||
3000);module.exports = app;
```

- The **package.json** file will contain the following information.

```
{
  "name": "node-app",
  "description": "Node App for Exp9",
  "version": "0.0.1",
  "private": true,
  "dependencies": {
      "express": "4.17.1"
  },
  "devDependencies": {
      "mocha": "9.1.1",
      "supertest": "6.1.6"
  },
  "scripts": {
      "start": "node index.js",
      "test": "./node_modules/.bin/mocha ./test/test.js"
  }
}
```

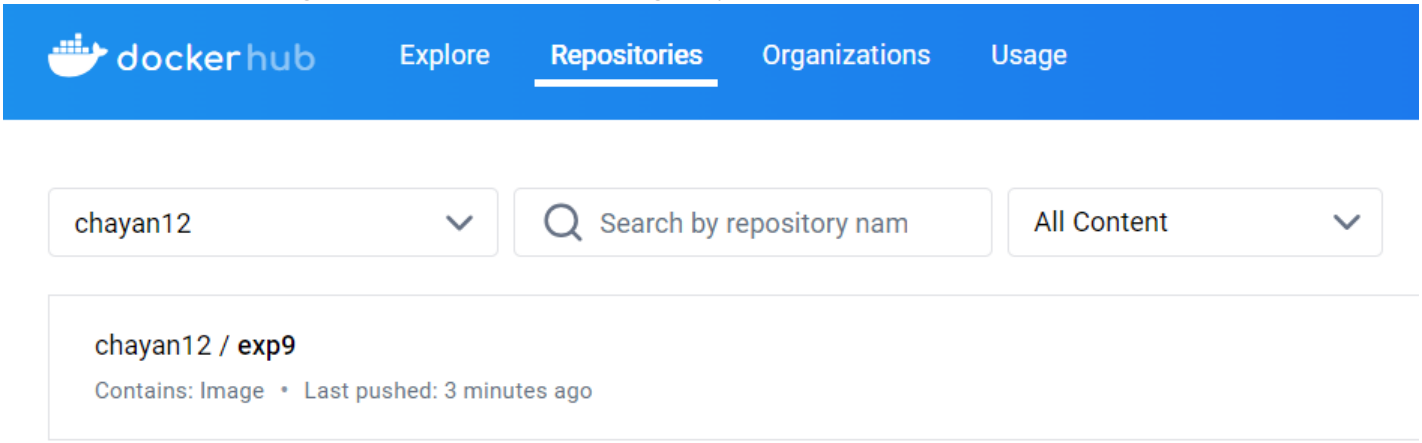
- The content of the **Dockerfile** is as follows:

```
FROM node:lts

WORKDIR /usr/src/app

COPY package*.json ./

RUN npm install

COPY . .

EXPOSE 4000

CMD [ "node", "index.js" ]
```

- **Build** the image using docker.

```
chayan@chayan-virtual-machine:~/Desktop/exp9$ docker images
REPOSITORY                  TAG       IMAGE ID       CREATED          SIZE
chayan12/exp9               latest    d61246678e08   4 minutes ago    1.13GB
node                        lts       4c2ff0421257   2 weeks ago      1.1GB
```

- **Push** the image on the docker hub registry.



- Create the Kubernetes cluster and configure the **deployment** and **service YAML** files.

```
chayan@chayan-virtual-machine:~/Desktop/exp9$ 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 "minikube" cluster
🚜  Pulling base image v0.0.45 ...
🔄  Restarting existing docker container for "minikube" ...
🐳  Preparing Kubernetes v1.31.0 on Docker 27.2.0 ...
🔎  Verifying Kubernetes components...
    ▪ Using image docker.io/kubernetesui/dashboard:v2.7.0
    ▪ Using image docker.io/kubernetesui/metrics-scraper:v1.0.8
    ▪ Using image registry.k8s.io/ingress-nginx/controller:v1.11.2
    ▪ Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.3
    ▪ Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.3
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
💡  Some dashboard features require the metrics-server addon. To enable all features please run:

        minikube addons enable metrics-server

🔎  Verifying ingress addon...
🌟  Enabled addons: default-storageclass, storage-provisioner, dashboard, ingress
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default

chayan@chayan-virtual-machine:~/Desktop/exp9$ kubectl apply -f deployment.yaml
deployment.apps/nodeapp-deployment created
chayan@chayan-virtual-machine:~/Desktop/exp9$ kubectl apply -f deploymentservice.yaml
service/nodeapp-service created
```

- Check whether the pods and services are running or not using the following commands:
    1. kubectl get service
    2. kubectl get pods

```
chayan@chayan-virtual-machine:~/Desktop/exp9$ kubectl get pods
NAME                                 READY   STATUS
nginx-web-5cb57cfb4b-r9qhg           1/1     Running
nodeapp-deployment-847f485468-tbzls   0/1     ContainerCreating

chayan@chayan-virtual-machine:~/Desktop/exp9$ kubectl get services
NAME              TYPE           CLUSTER-IP       EXTERNAL-IP   POR
kubernetes        ClusterIP      10.96.0.1        <none>        443
nginx-web         NodePort       10.100.207.227   <none>        80:
nodeapp-service   LoadBalancer   10.106.117.85    <pending>     500
```

- In order to access the application on the web browser directly, you can use the following command which exposes our service to our local machine:
    1. minikube service <service-name>

        a. minikube service nodeapp-service

```
chayan@chayan-virtual-machine:~/Desktop/exp9$ minikube service nodeapp-service
|-----------|-----------------|-------------|---------------------------|
| NAMESPACE |      NAME       | TARGET PORT |            URL            |
|-----------|-----------------|-------------|---------------------------|
| default   | nodeapp-service |        5000 | http://192.168.49.2:31110 |
|-----------|-----------------|-------------|---------------------------|
```

Learning outcomes (What I have learnt):

1. I have learnt the concept of containerization and virtualization.

2. I have learnt about orchestration and orchestration tools.

3. I have learnt about Kubernetes and its architecture.

4. I have learnt 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.			