Experiment -3.2

Student Name: Ayush Pandey
Branch: CSE-DevOps
Semester: 5
Subject Name: Docker and Kubernetes

UID: 22BDO10038
Section/Group: 22BCD-1(A)
Date of Performance: 21-10-24
Subject Code: 22CSH-343

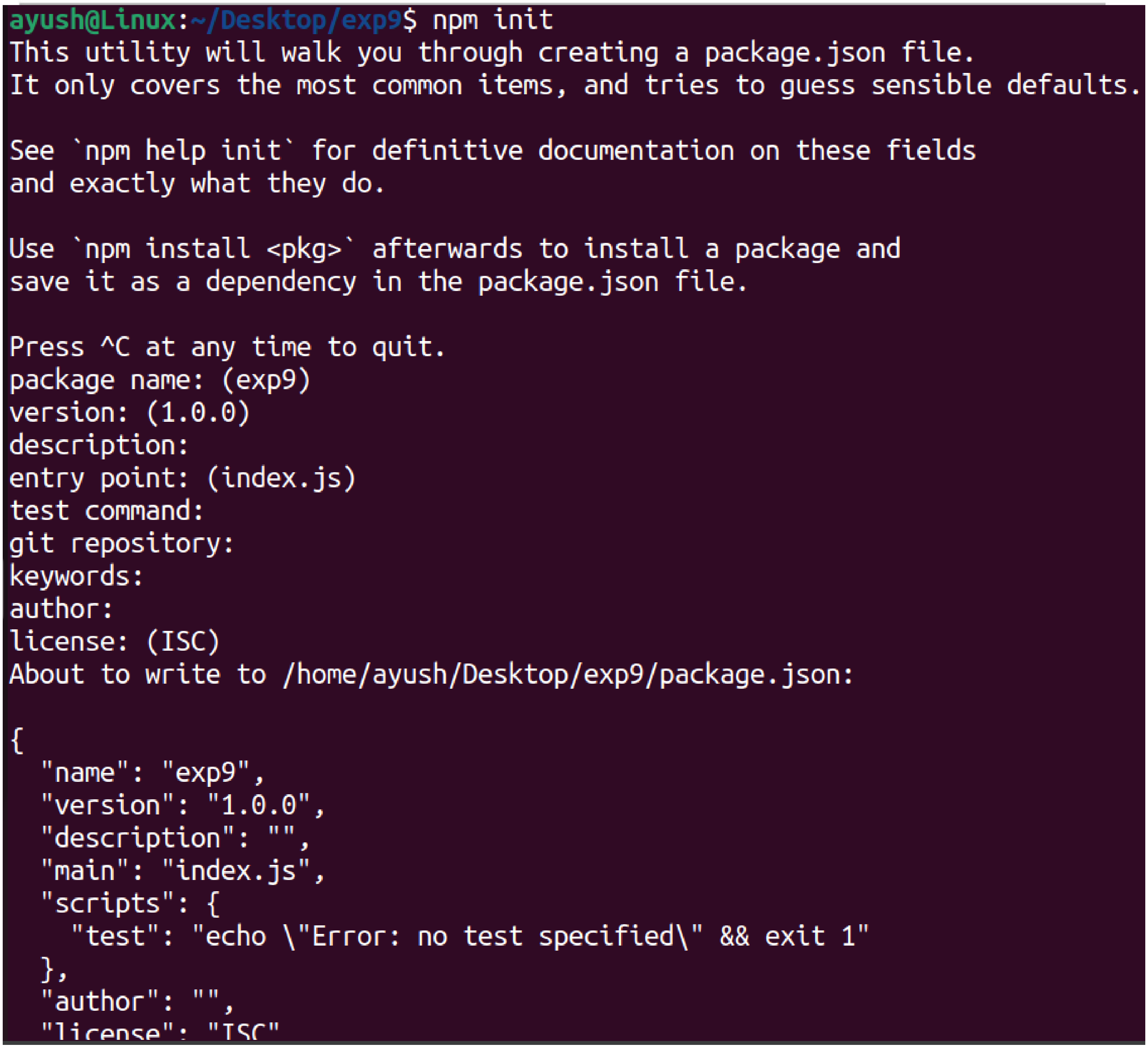
1. Aim/Overview of the practical:

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

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

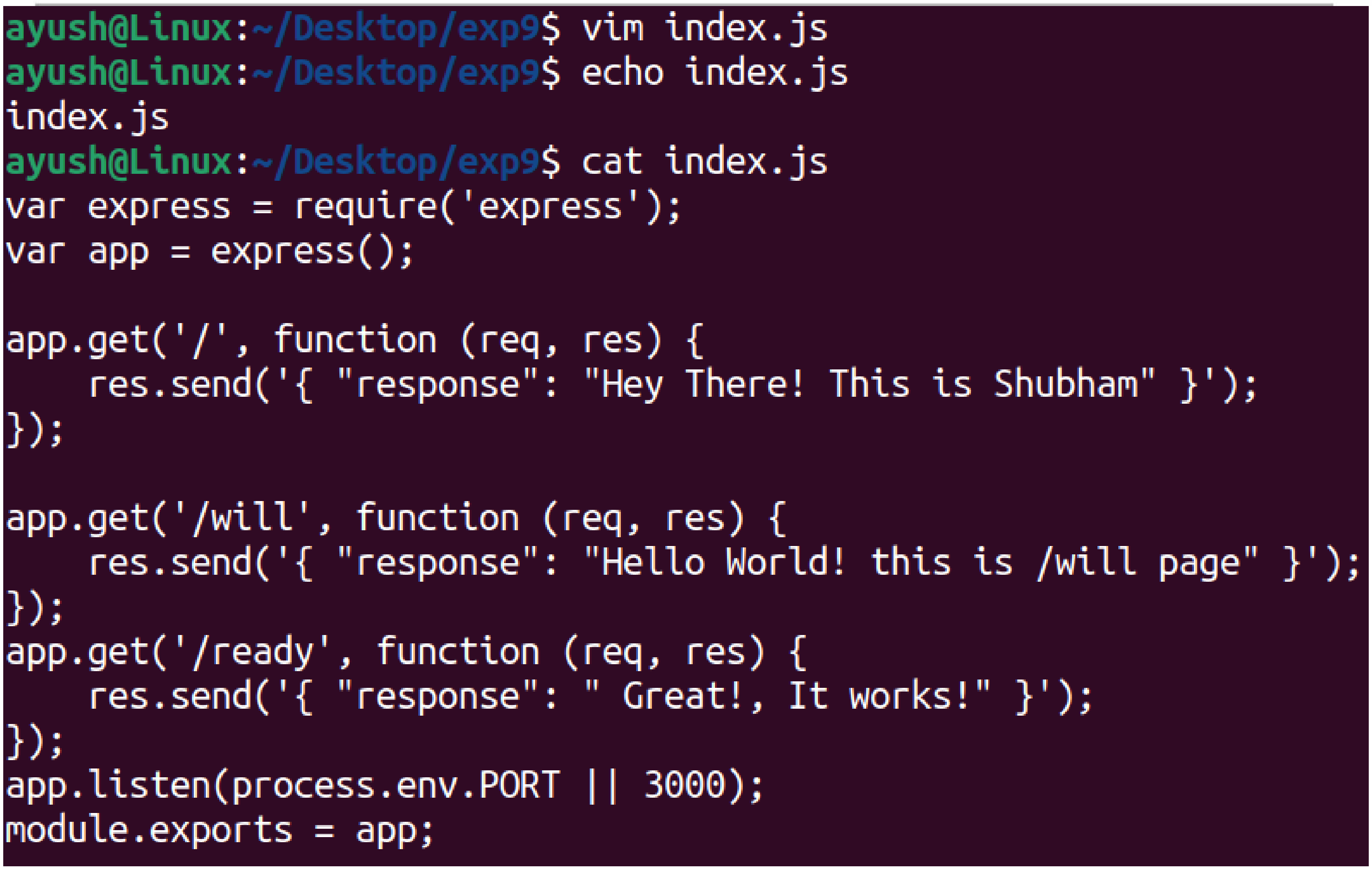
3. Steps for experiment/practical:

- **Creating the Node.js application**
  1. Start the node application using the **npm init** command.



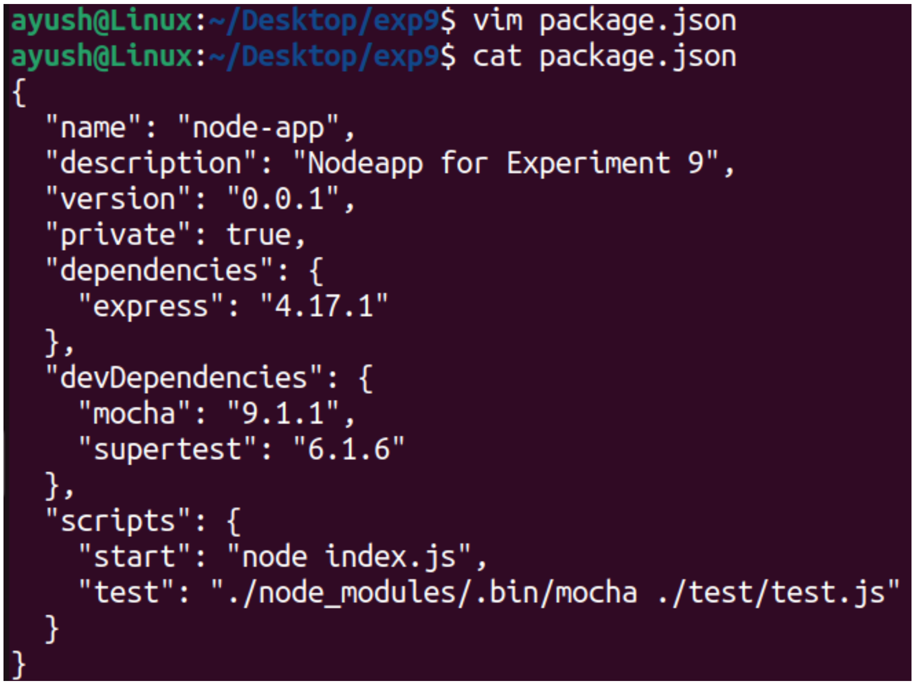
```
ayush@Linux:~/Desktop/exp9$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (exp9)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to /home/ayush/Desktop/exp9/package.json:

{
  "name": "exp9",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
```

2. Create the **index.js** file.



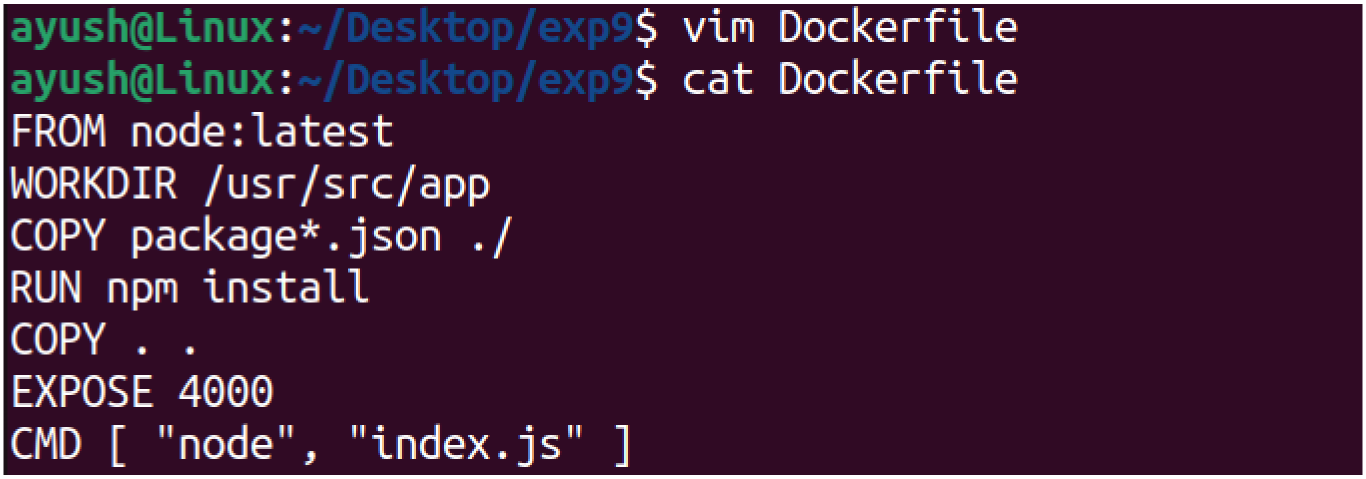
```
ayush@Linux:~/Desktop/exp9$ vim index.js
ayush@Linux:~/Desktop/exp9$ echo index.js
index.js
ayush@Linux:~/Desktop/exp9$ cat index.js
var express = require('express');
var app = express();

app.get('/', function (req, res) {
    res.send('{ "response": "Hey There! This is Shubham" }');
});

app.get('/will', function (req, res) {
    res.send('{ "response": "Hello World! this is /will page" }');
});
app.get('/ready', function (req, res) {
    res.send('{ "response": " Great!, It works!" }');
});
app.listen(process.env.PORT || 3000);
module.exports = app;
```

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



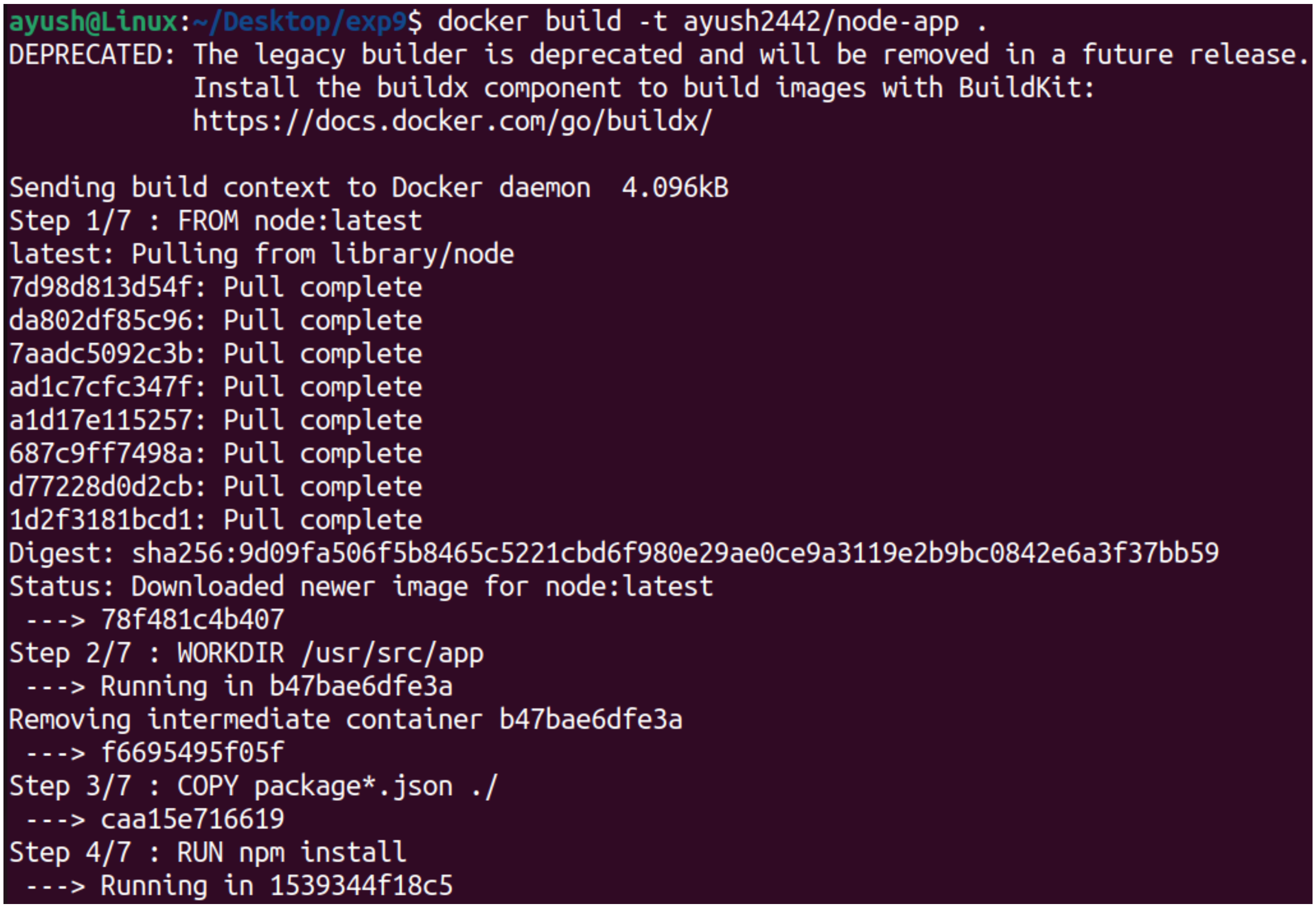
```
ayush@Linux:~/Desktop/exp9$ vim package.json
ayush@Linux:~/Desktop/exp9$ cat package.json
{
  "name": "node-app",
  "description": "Nodeapp for Experiment 9",
  "version": "0.0.1",
  "private": true,
  "dependencies": {
    "express": "4.17.1"
  },
  "devDependencies": {
    "mocha": "9.1.1",
    "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:

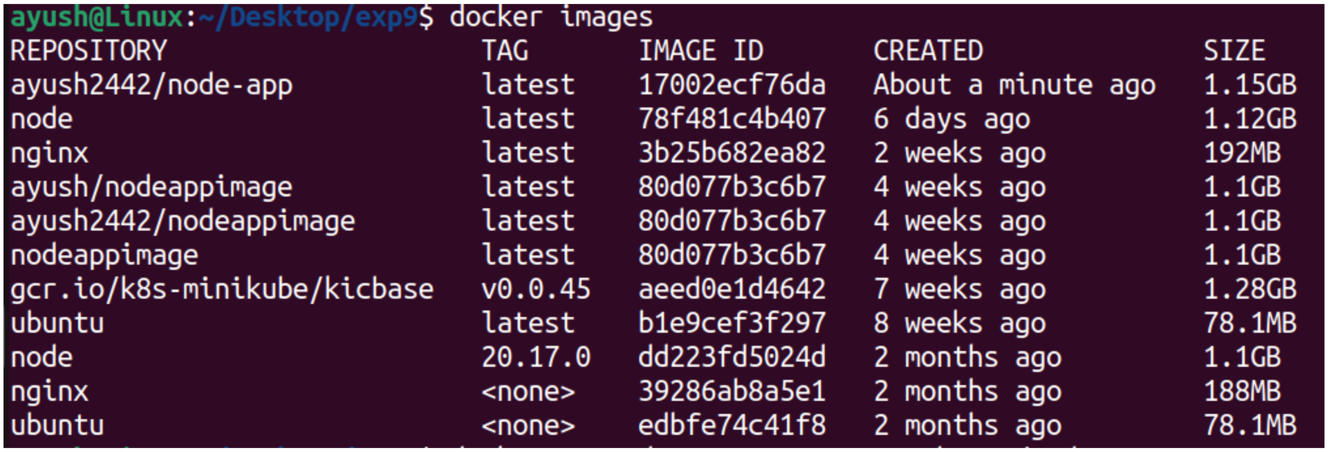


```
ayush@Linux:~/Desktop/exp9$ vim Dockerfile
ayush@Linux:~/Desktop/exp9$ cat Dockerfile
FROM node:latest
WORKDIR /usr/src/app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 4000
CMD [ "node", "index.js" ]
```

- **Build** the image using docker.

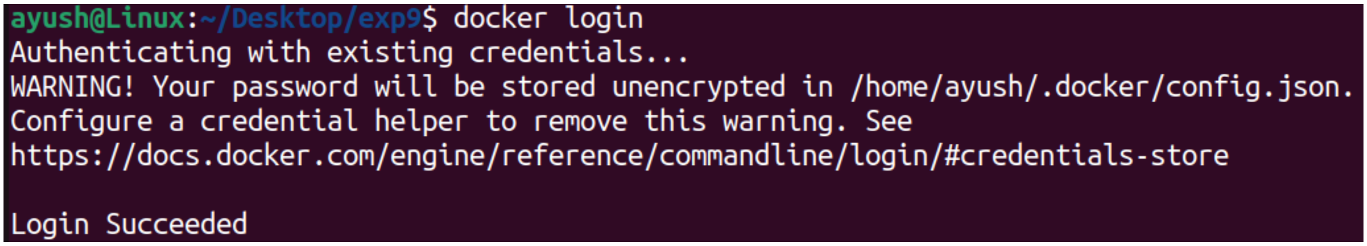


```
ayush@Linux:~/Desktop/exp9$ docker build -t ayush2442/node-app .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  4.096kB
Step 1/7 : FROM node:latest
latest: Pulling from library/node
7d98d813d54f: Pull complete
da802df85c96: Pull complete
7aadc5092c3b: Pull complete
ad1c7cfc347f: Pull complete
a1d17e115257: Pull complete
687c9ff7498a: Pull complete
d77228d0d2cb: Pull complete
1d2f3181bcd1: Pull complete
Digest: sha256:9d09fa506f5b8465c5221cbd6f980e29ae0ce9a3119e2b9bc0842e6a3f37bb59
Status: Downloaded newer image for node:latest
 ---> 78f481c4b407
Step 2/7 : WORKDIR /usr/src/app
 ---> Running in b47bae6dfe3a
Removing intermediate container b47bae6dfe3a
 ---> f6695495f05f
Step 3/7 : COPY package*.json ./
 ---> caa15e716619
Step 4/7 : RUN npm install
 ---> Running in 1539344f18c5
```

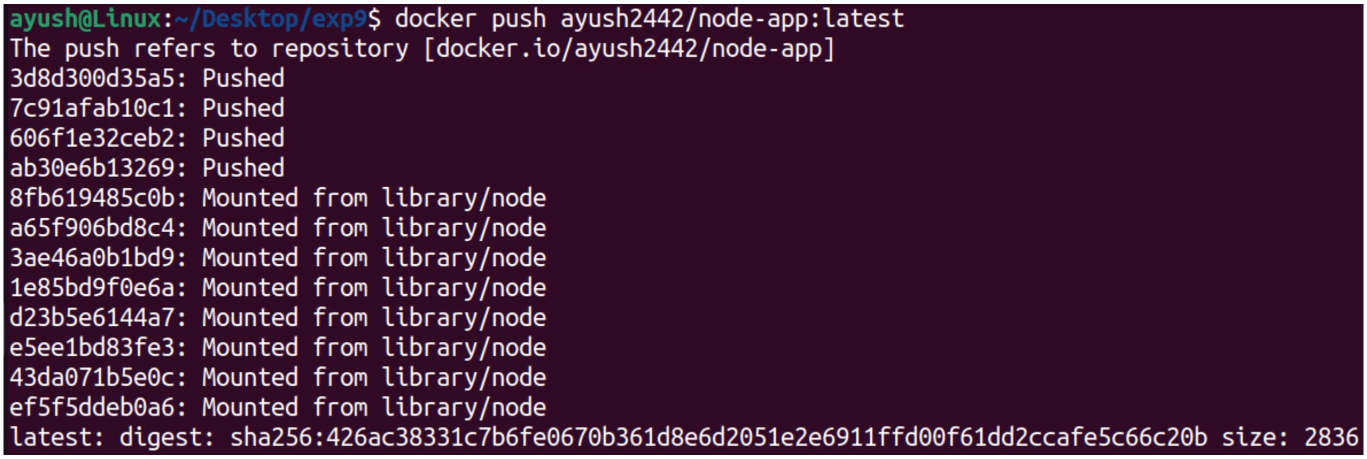


```
ayush@Linux:~/Desktop/exp9$ docker images
REPOSITORY                     TAG       IMAGE ID       CREATED              SIZE
ayush2442/node-app             latest    17002ecf76da   About a minute ago   1.15GB
node                           latest    78f481c4b407   6 days ago           1.12GB
nginx                          latest    3b25b682ea82   2 weeks ago          192MB
ayush/nodeappimage             latest    80d077b3c6b7   4 weeks ago          1.1GB
ayush2442/nodeappimage         latest    80d077b3c6b7   4 weeks ago          1.1GB
nodeappimage                   latest    80d077b3c6b7   4 weeks ago          1.1GB
gcr.io/k8s-minikube/kicbase    v0.0.45   aeed0e1d4642   7 weeks ago          1.28GB
ubuntu                         latest    b1e9cef3f297   8 weeks ago          78.1MB
node                           20.17.0   dd223fd5024d   2 months ago         1.1GB
nginx                          <none>    39286ab8a5e1   2 months ago         188MB
ubuntu                         <none>    edbfe74c41f8   2 months ago         78.1MB
```

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

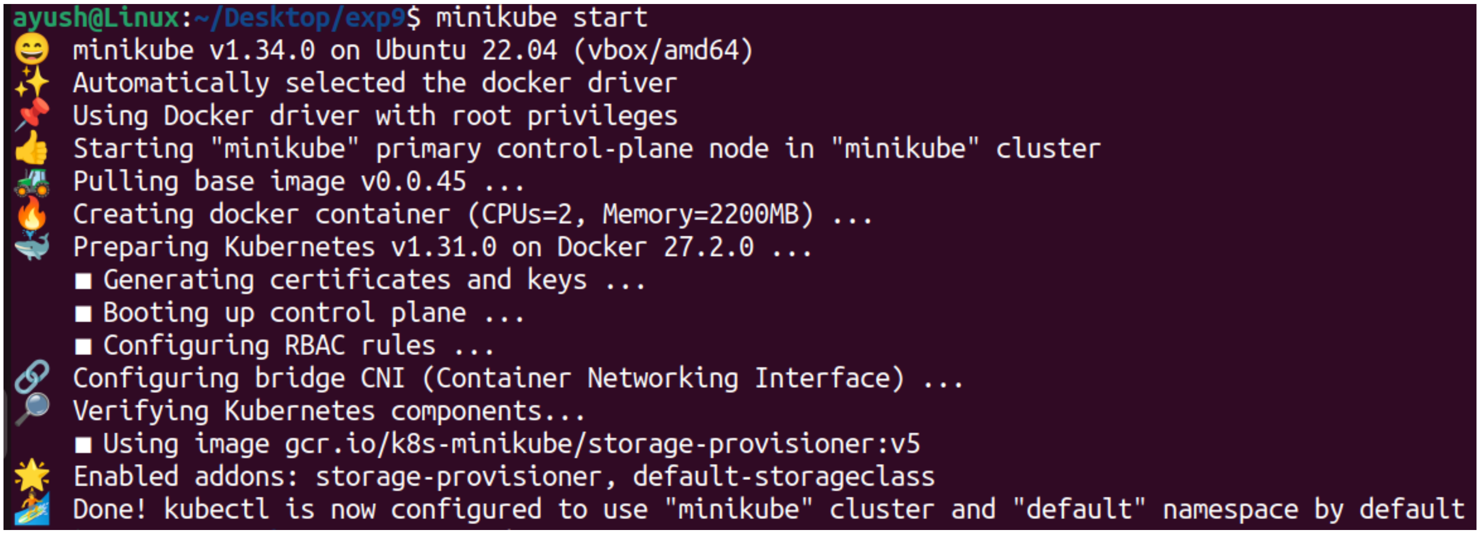


```
ayush@Linux:~/Desktop/exp9$ docker login
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /home/ayush/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

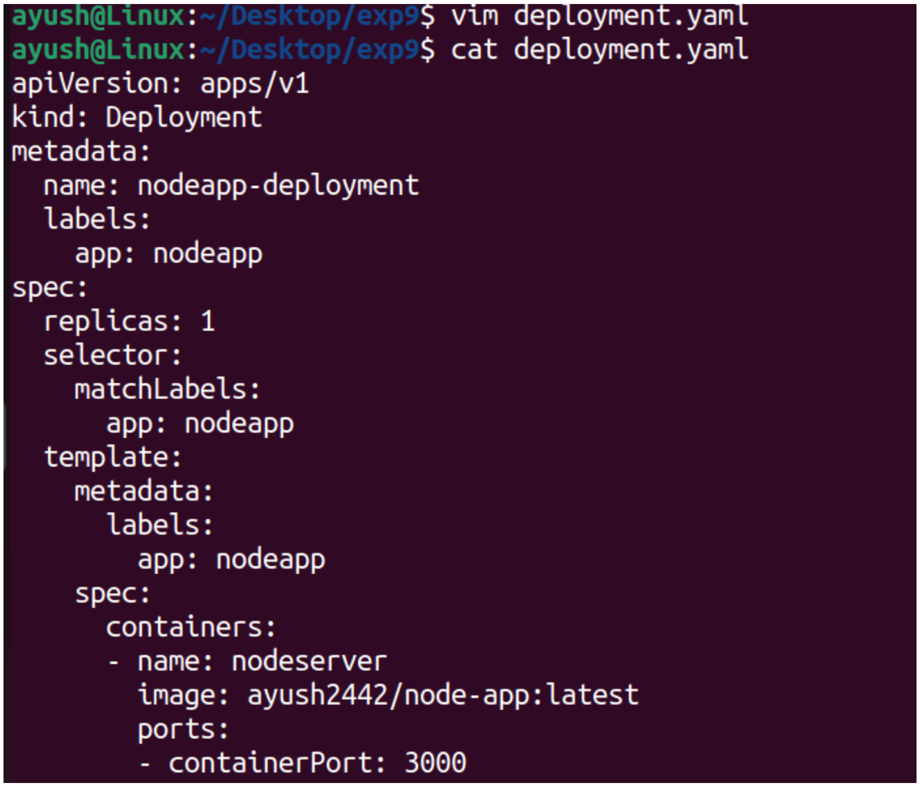


```
ayush@Linux:~/Desktop/exp9$ docker push ayush2442/node-app:latest
The push refers to repository [docker.io/ayush2442/node-app]
3d8d300d35a5: Pushed
7c91afab10c1: Pushed
606f1e32ceb2: Pushed
ab30e6b13269: Pushed
8fb619485c0b: Mounted from library/node
a65f906bd8c4: Mounted from library/node
3ae46a0b1bd9: Mounted from library/node
1e85bd9f0e6a: Mounted from library/node
d23b5e6144a7: Mounted from library/node
e5ee1bd83fe3: Mounted from library/node
43da071b5e0c: Mounted from library/node
ef5f5ddeb0a6: Mounted from library/node
latest: digest: sha256:426ac38331c7b6fe0670b361d8e6d2051e2e6911ffd00f61dd2ccafe5c66c20b size: 2836
```

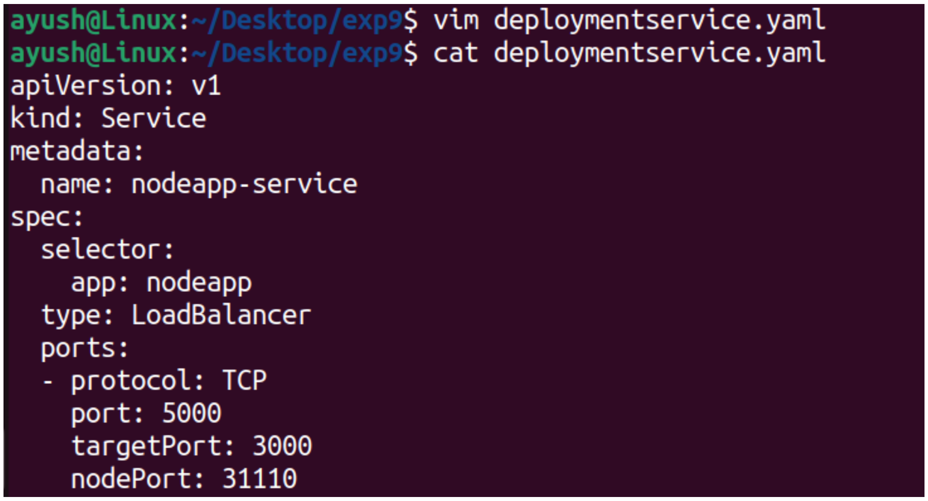
- Create the Kubernetes cluster and configure the **deployment** and **service YAML** files.



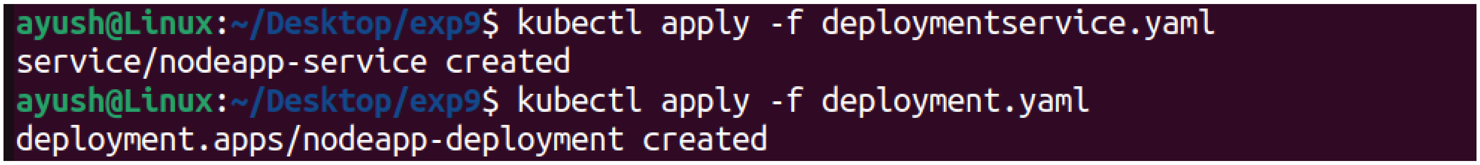
```
ayush@Linux:~/Desktop/exp9$ minikube start
😄  minikube v1.34.0 on Ubuntu 22.04 (vbox/amd64)
✨  Automatically selected the docker driver
📌  Using Docker driver with root privileges
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.45 ...
🔥  Creating docker container (CPUs=2, Memory=2200MB) ...
🐳  Preparing Kubernetes v1.31.0 on Docker 27.2.0 ...
    ■ Generating certificates and keys ...
    ■ Booting up control plane ...
    ■ Configuring RBAC rules ...
🔗  Configuring bridge CNI (Container Networking Interface) ...
🔎  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
```



```
ayush@Linux:~/Desktop/exp9$ vim deployment.yaml
ayush@Linux:~/Desktop/exp9$ cat deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nodeapp-deployment
  labels:
    app: nodeapp
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nodeapp
  template:
    metadata:
      labels:
        app: nodeapp
    spec:
      containers:
      - name: nodeserver
        image: ayush2442/node-app:latest
        ports:
        - containerPort: 3000
```

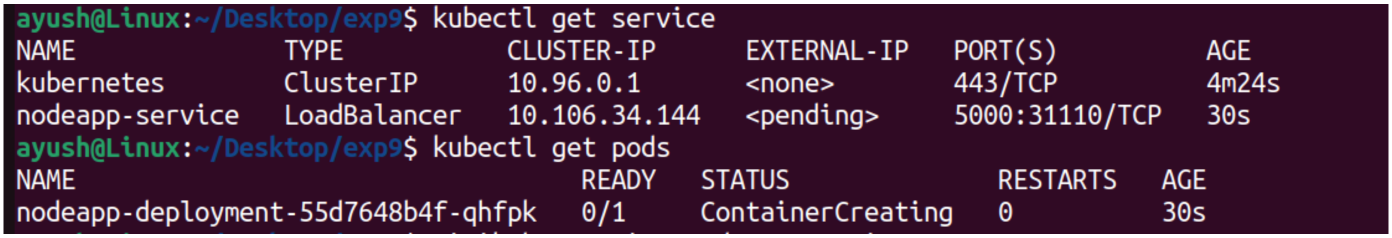


```
ayush@Linux:~/Desktop/exp9$ vim deploymentservice.yaml
ayush@Linux:~/Desktop/exp9$ cat deploymentservice.yaml
apiVersion: v1
kind: Service
metadata:
  name: nodeapp-service
spec:
  selector:
    app: nodeapp
  type: LoadBalancer
  ports:
  - protocol: TCP
    port: 5000
    targetPort: 3000
    nodePort: 31110
```



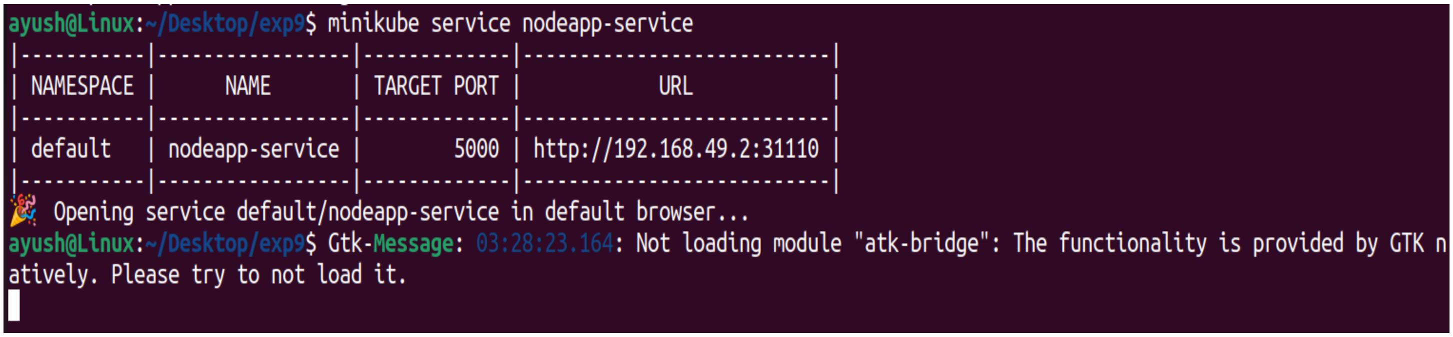
```
ayush@Linux:~/Desktop/exp9$ kubectl apply -f deploymentservice.yaml
service/nodeapp-service created
ayush@Linux:~/Desktop/exp9$ kubectl apply -f deployment.yaml
deployment.apps/nodeapp-deployment created
```

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

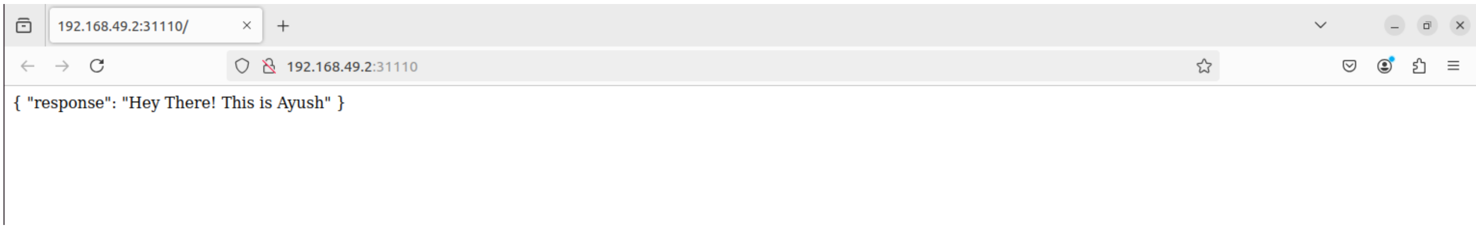


```
ayush@Linux:~/Desktop/exp9$ kubectl get service
NAME              TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes        ClusterIP      10.96.0.1       <none>        443/TCP          4m24s
nodeapp-service   LoadBalancer   10.106.34.144   <pending>     5000:31110/TCP   30s
ayush@Linux:~/Desktop/exp9$ kubectl get pods
NAME                               READY   STATUS              RESTARTS   AGE
nodeapp-deployment-55d7648b4f-qhfpk   0/1     ContainerCreating   0          30s
```

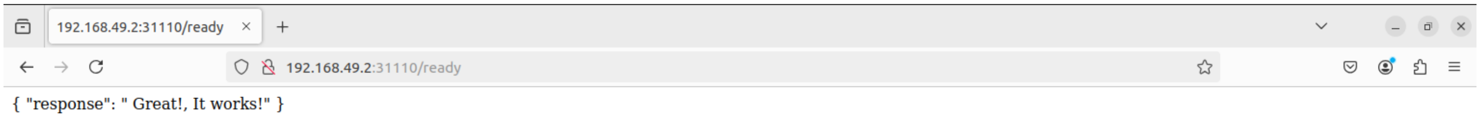
- In order to access the application on the web browser directly, you can use the following command which exposes our service to our local machine:
  1. minikube service <service-name
     a. minikube service nodeapp-service



```
ayush@Linux:~/Desktop/exp9$ minikube service nodeapp-service
|-----------|-----------------|-------------|---------------------------|
| NAMESPACE |      NAME       | TARGET PORT |            URL            |
|-----------|-----------------|-------------|---------------------------|
| default   | nodeapp-service |        5000 | http://192.168.49.2:31110 |
|-----------|-----------------|-------------|---------------------------|
🎉  Opening service default/nodeapp-service in default browser...
ayush@Linux:~/Desktop/exp9$ Gtk-Message: 03:28:23.164: Not loading module "atk-bridge": The functionality is provided by GTK natively. Please try to not load it.
```



{ "response": "Hey There! This is Ayush" }



{ "response": " Great!, It works!" }

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.			