Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

Scenario : You are working for a company that uses IBM Cloud to store critical data in a Cloudant database. Your team has developed an API-based application that performs CRUD operations on the Cloudant database, and this application is now ready for deployment in a Kubernetes environment.

As part of the security team, your task is to ensure that the application adheres to security best practices, including limiting network traffic for the pods.

Task :

- **Deploy the existing API-based application on a Kubernetes cluster.**
- **Configure a network policy that blocks all egress traffic from the pod.**

Steps and Screenshots :

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

1. Create nodejs application and dockerfile to connect to cloudant database

Code :

```
const express = require('express');
const { CloudantV1 } = require('@ibm-cloud/cloudant');
const { IamAuthenticator } = require('ibm-cloud-sdk-core');
const bodyParser = require('body-parser');

let PORT = process.env.PORT || 3000;

const url =
'https://60379ad1-b8cb-4933-bb03-b1bfa6df8a7e-bluemix.cloudantno
sqldb.appdomain.cloud';
const apiKey = 'PTf_8CBfnhbiTsz59lJNaLpE93g7kFmxYXdO9Ly4C6DP';

const authenticator = new IamAuthenticator({ apikey: apiKey });
const cloudant = CloudantV1.newInstance({ authenticator });
cloudant.setServiceUrl(url);

const app = express();
app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json());

app.get('/', function (req, res) {
 res.send("Welcome to cloudant database on IBM Cloud");
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

```
});

app.get('/list_of_databases', async function (req, res) {
 try {
   const response = await cloudant.getAllDbs();
   res.send(response.result);
 } catch (err) {
   res.send(err);
 }
});

app.post('/create-database', async (req, res) => {
 const name = req.body.name;
 try {
   await cloudant.putDatabase({ db: name });
   res.send("Database created");
 } catch (err) {
   res.send(err);
 }
});

app.post('/insert-document', async function (req, res) {
 const { db, id, name, address, phone, age } = req.body;
 try {
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

```
    const response = await cloudant.postDocument({
      db,
      document: { _id: id, name, address, phone, age }
    });
    res.send(response.result);
  } catch (err) {
    res.send(err);
  }
});

app.post('/insert-bulk/:database_name', async function (req,
res) {
  const database_name = req.params.database_name;
  const students = req.body.docs.map(doc => ({
    _id: doc.id,
    name: doc.name,
    address: doc.address,
    phone: doc.phone,
    age: doc.age
  }));

  try {
    await cloudant.postBulkDocs({
      db: database_name,
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

```
      bulkDocs: { docs: students }
    });
    res.send('Inserted all documents');
  } catch (err) {
    res.send(err);
  }
});

app.delete('/delete-document', async function (req, res) {
  const { db, id, rev } = req.body;
  try {
    await cloudant.deleteDocument({ db, docId: id, rev });
    res.send('Document deleted');
  } catch (err) {
    res.send(err);
  }
});

app.put('/update-document', async function (req, res) {
  const { db, id, rev, name, address, phone, age } = req.body;
  try {
    const response = await cloudant.postDocument({
      db,
      document: { _id: id, _rev: rev, name, address, phone, age }
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

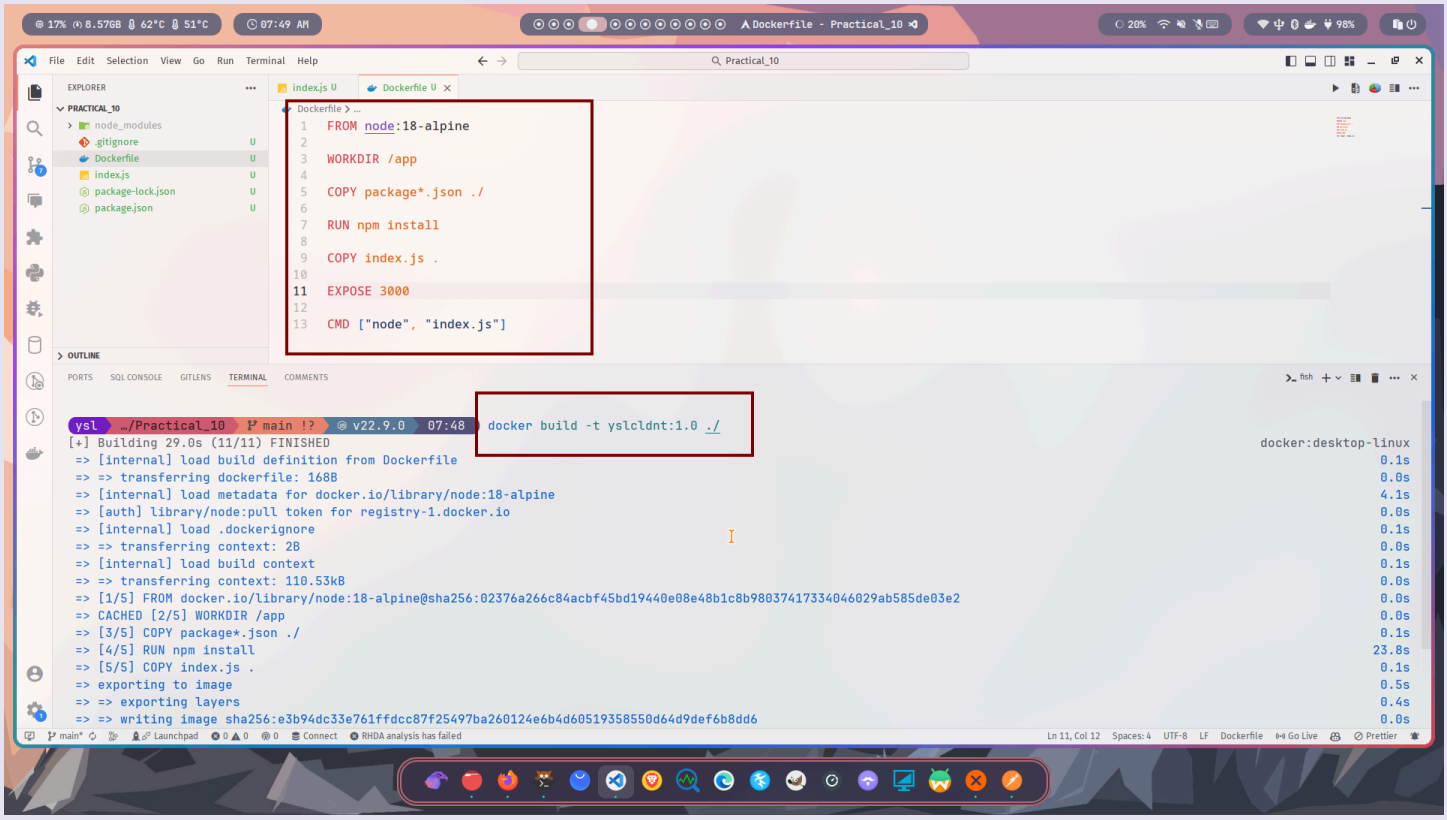
```
    });
    res.send(response.result);
  } catch (err) {
    res.send(err);
  }
});

app.listen(PORT, () => {
  console.log(`Server is running on port ${PORT}`);
});
```

Dockerfile :

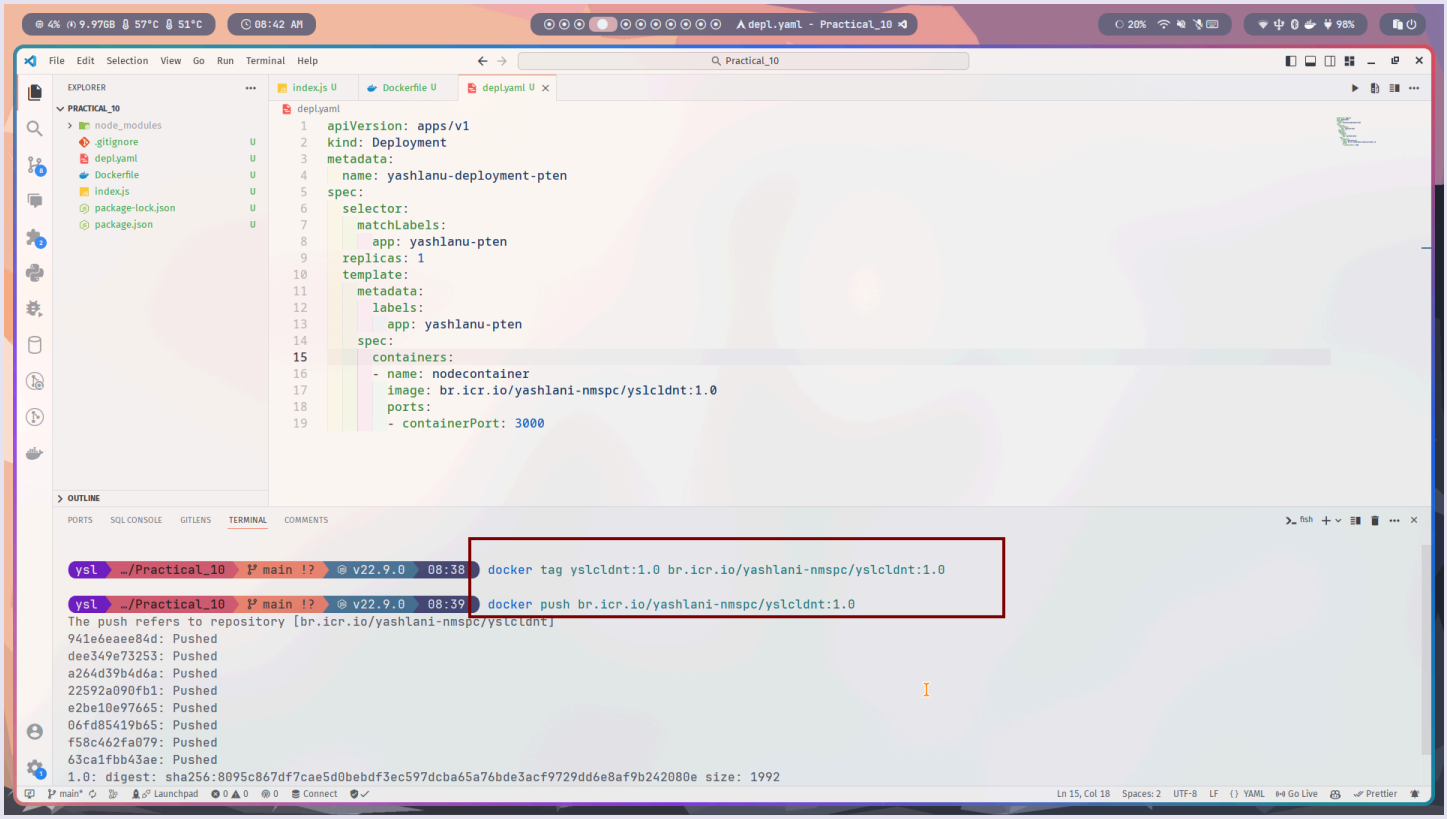
```
FROM node:18-alpine
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY index.js .
EXPOSE 3000
CMD ["node", "index.js"]
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10



Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

2. Login to ibmcloud cli and push docker image to ibm container registry



Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

3. Create deployment and service files

Deployment :

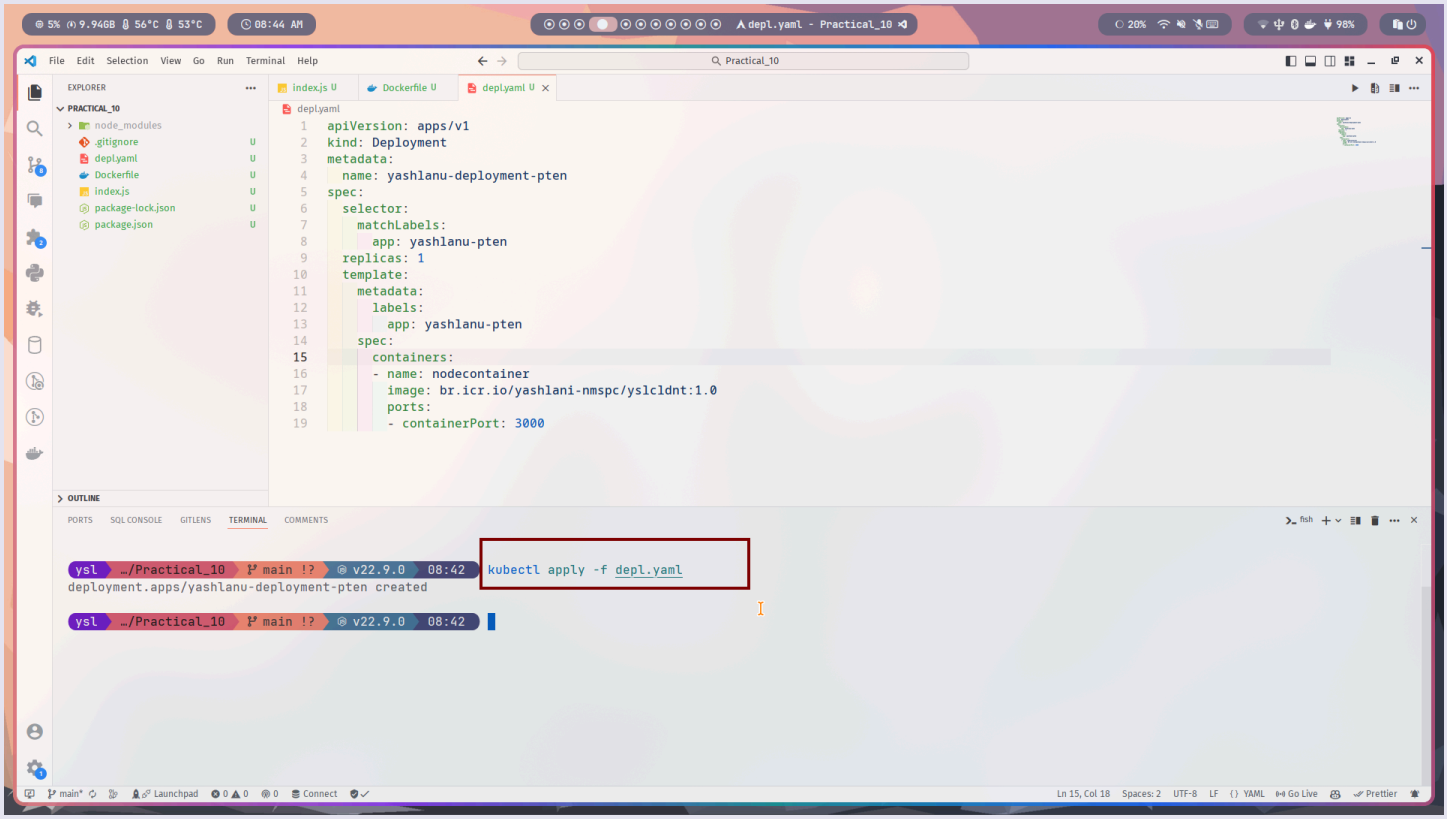
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
 name: yashlanu-deployment-pten
spec:
 selector:
   matchLabels:
     app: yashlanu-pten
 replicas: 1
 template:
   metadata:
     labels:
       app: yashlanu-pten
   spec:
     containers:
     - name: nodecontainer
       image: br.icr.io/yashlani-nmspc/yslcldnt:1.0
       ports:
       - containerPort: 3000
```

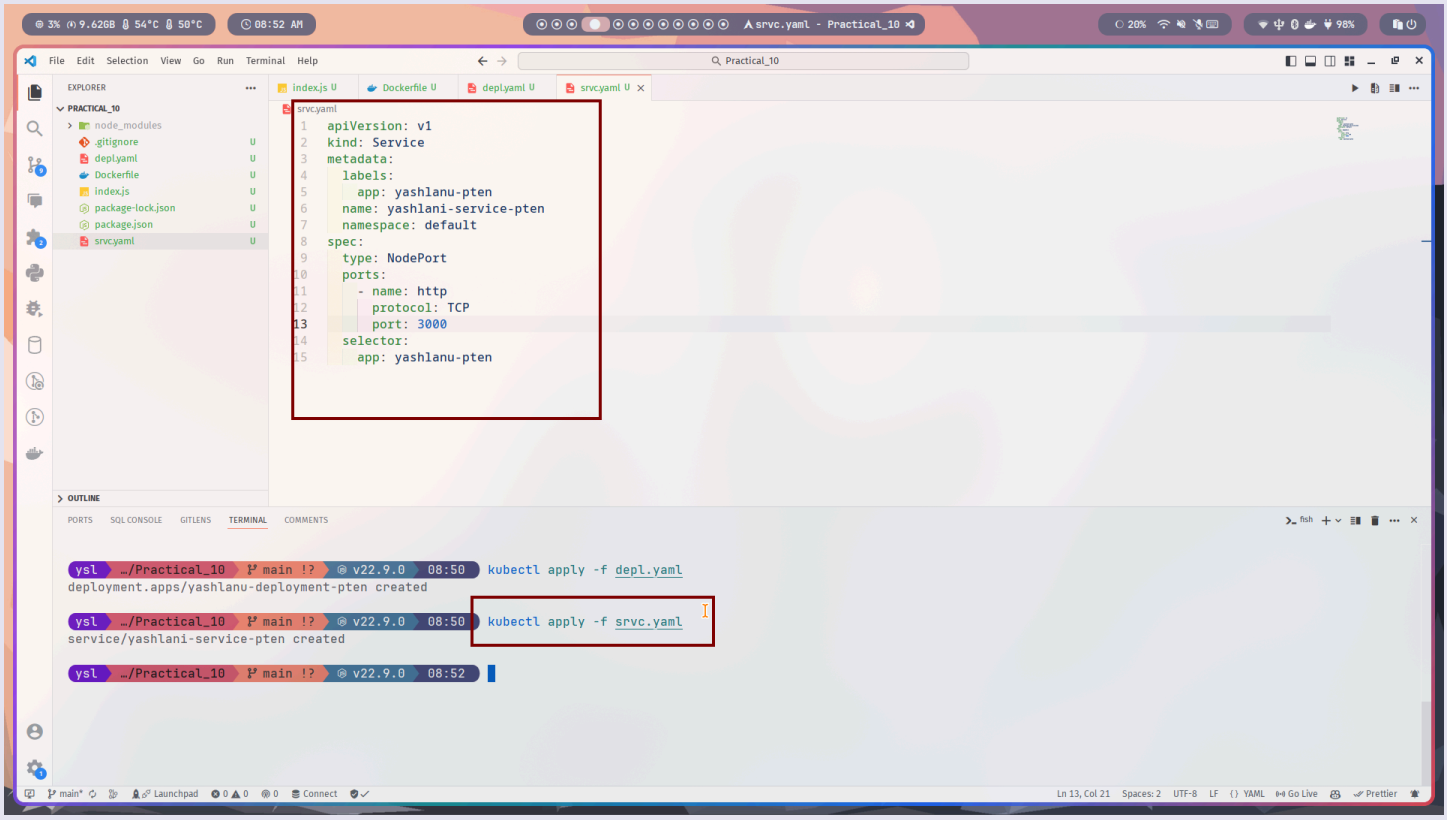
Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

Service :

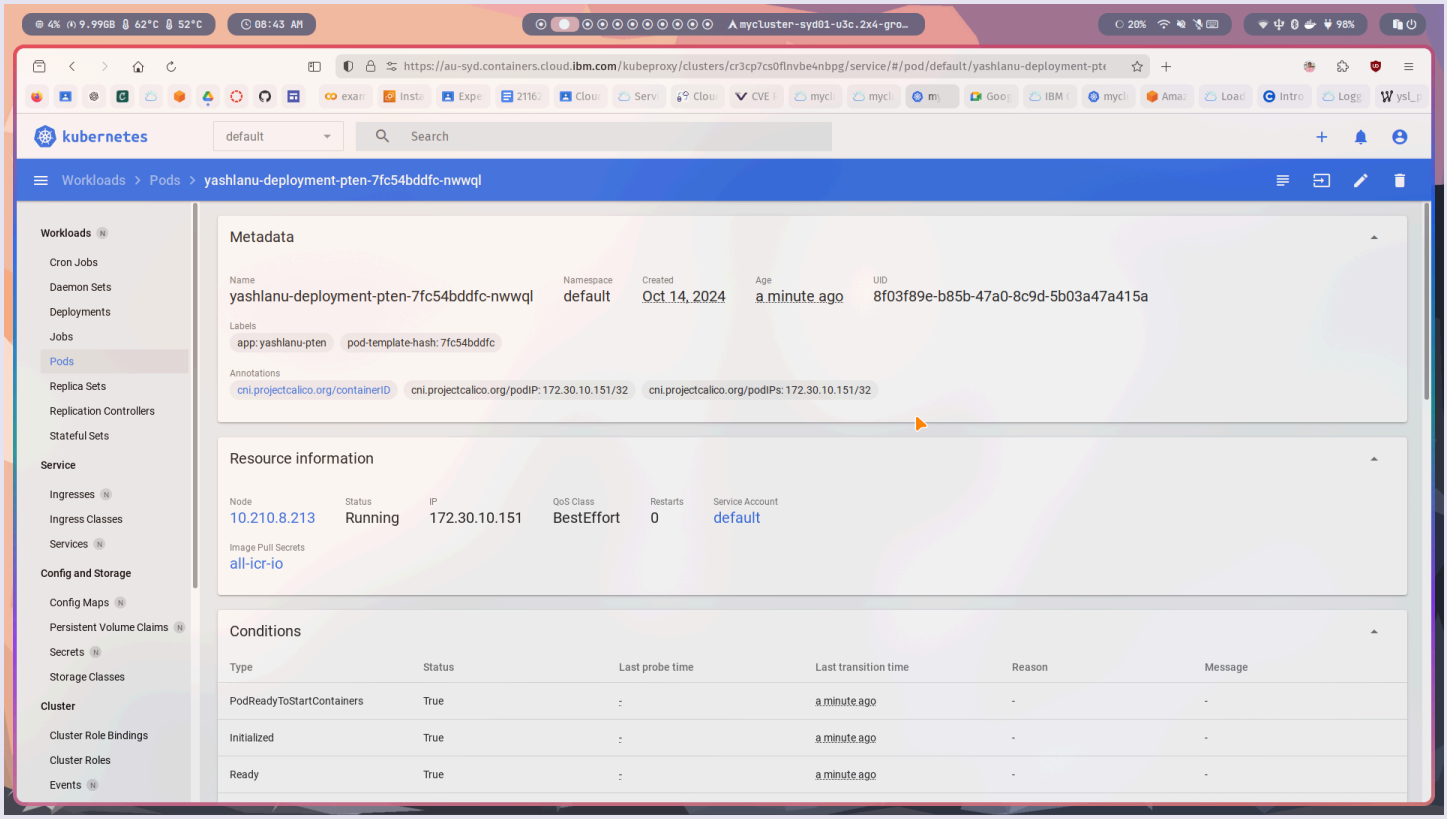
```yaml
apiVersion: v1
kind: Service
metadata:
 labels:
   app: yashlanu-pten
 name: yashlani-service-pten
 namespace: default
spec:
 type: NodePort
 ports:
   - name: http
     protocol: TCP
     port: 3000
 selector:
   app: yashlanu-pten
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10



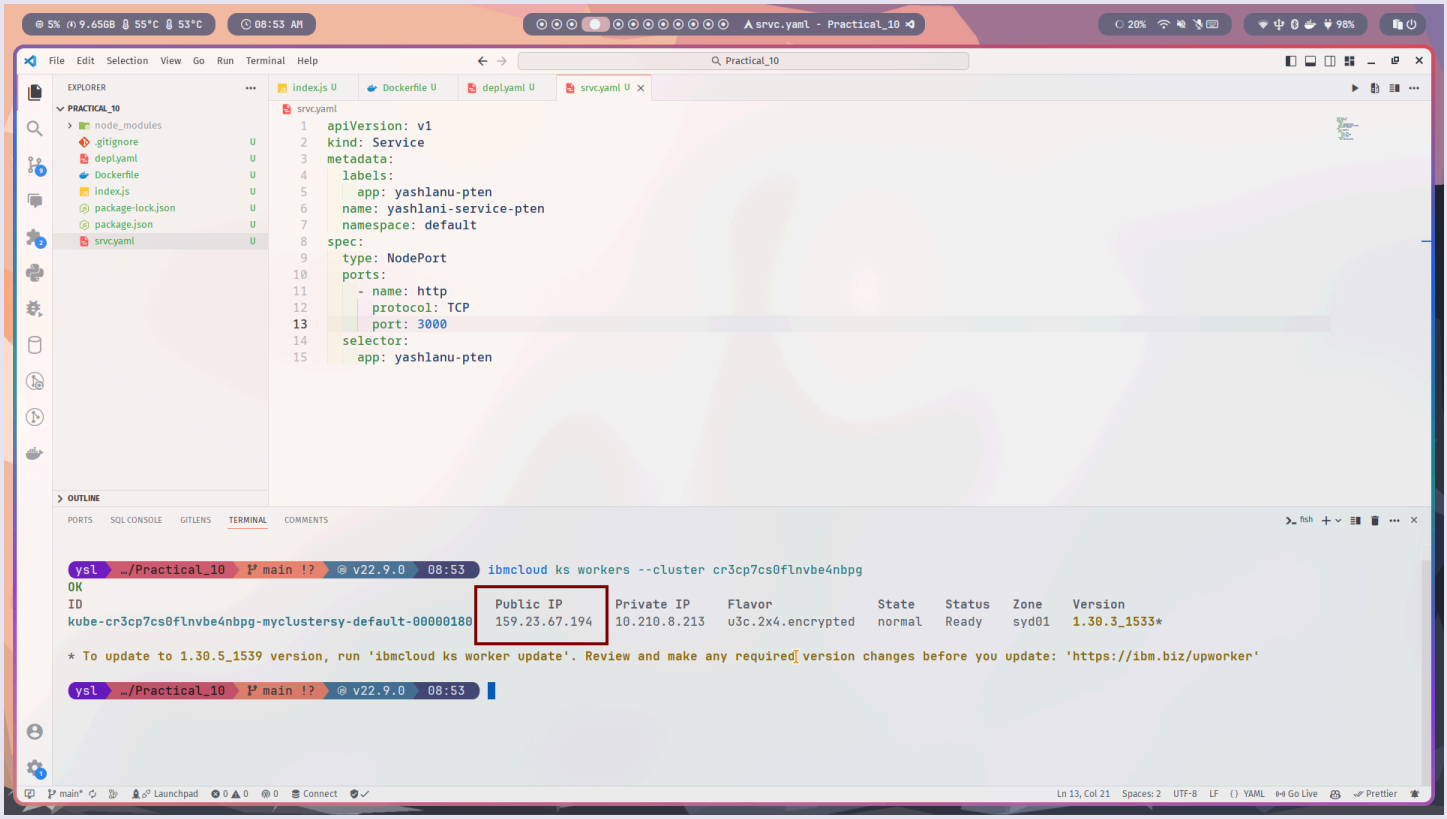


Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

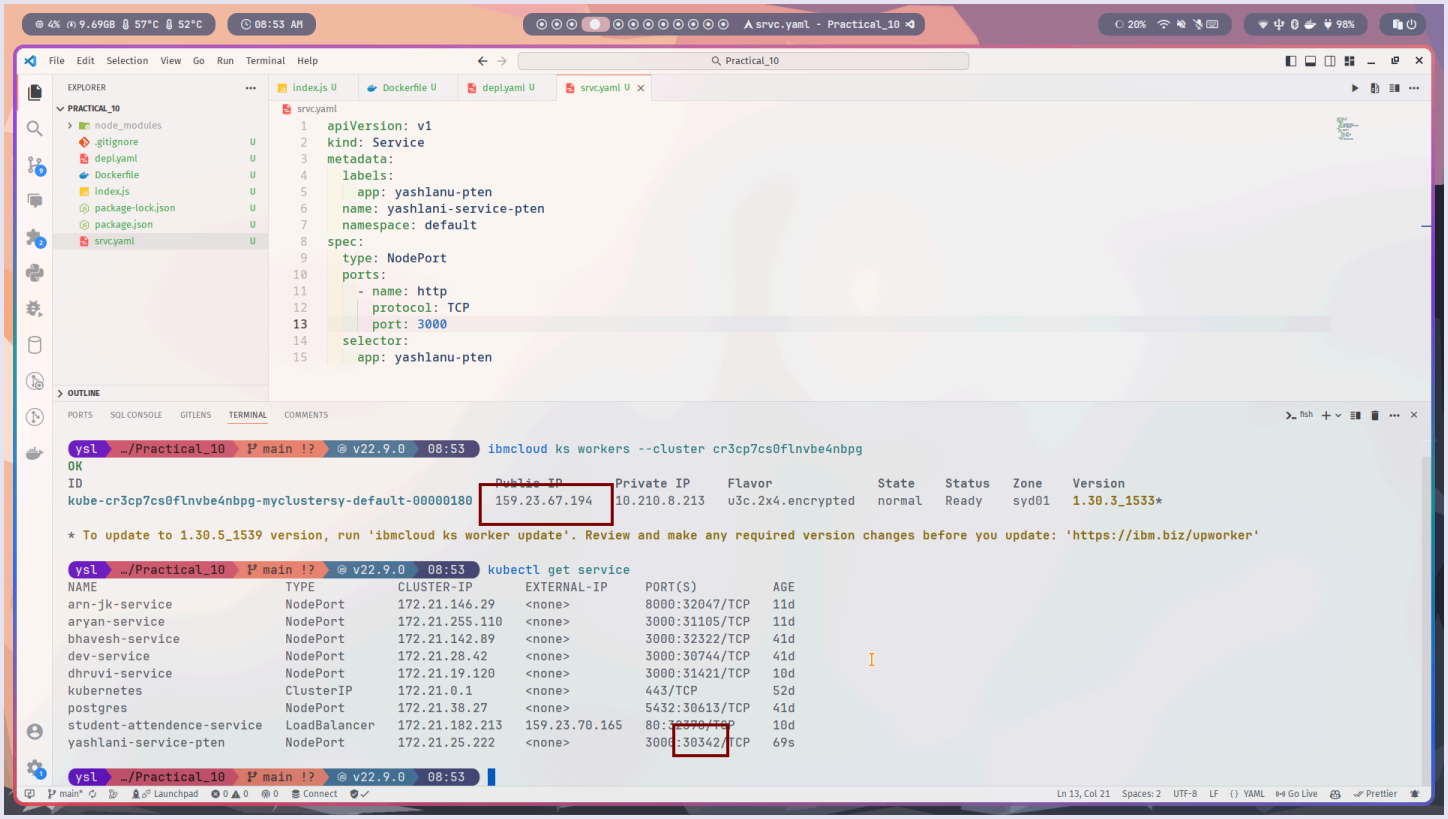


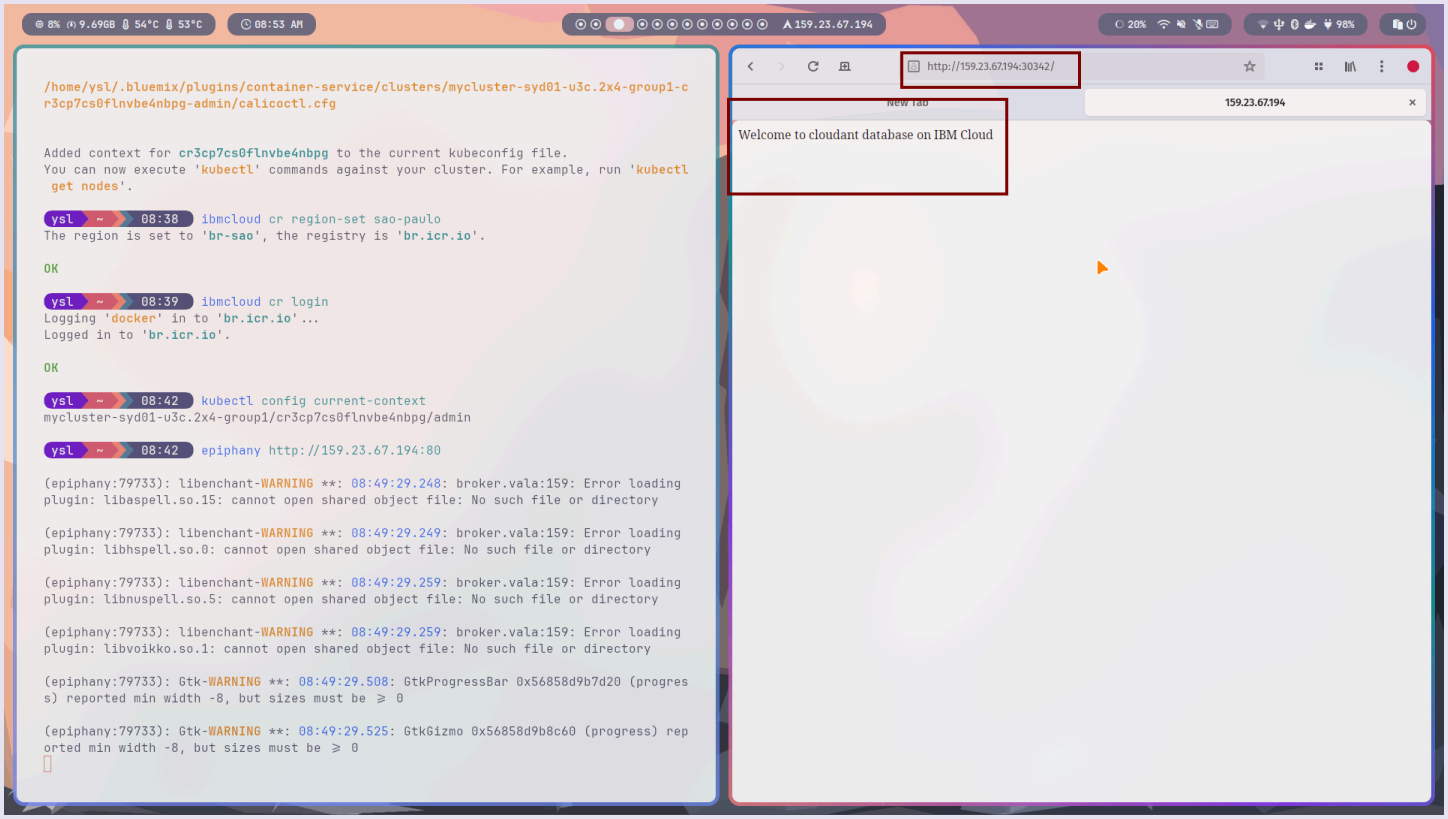
Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

4. Get IP and port and check connectivity



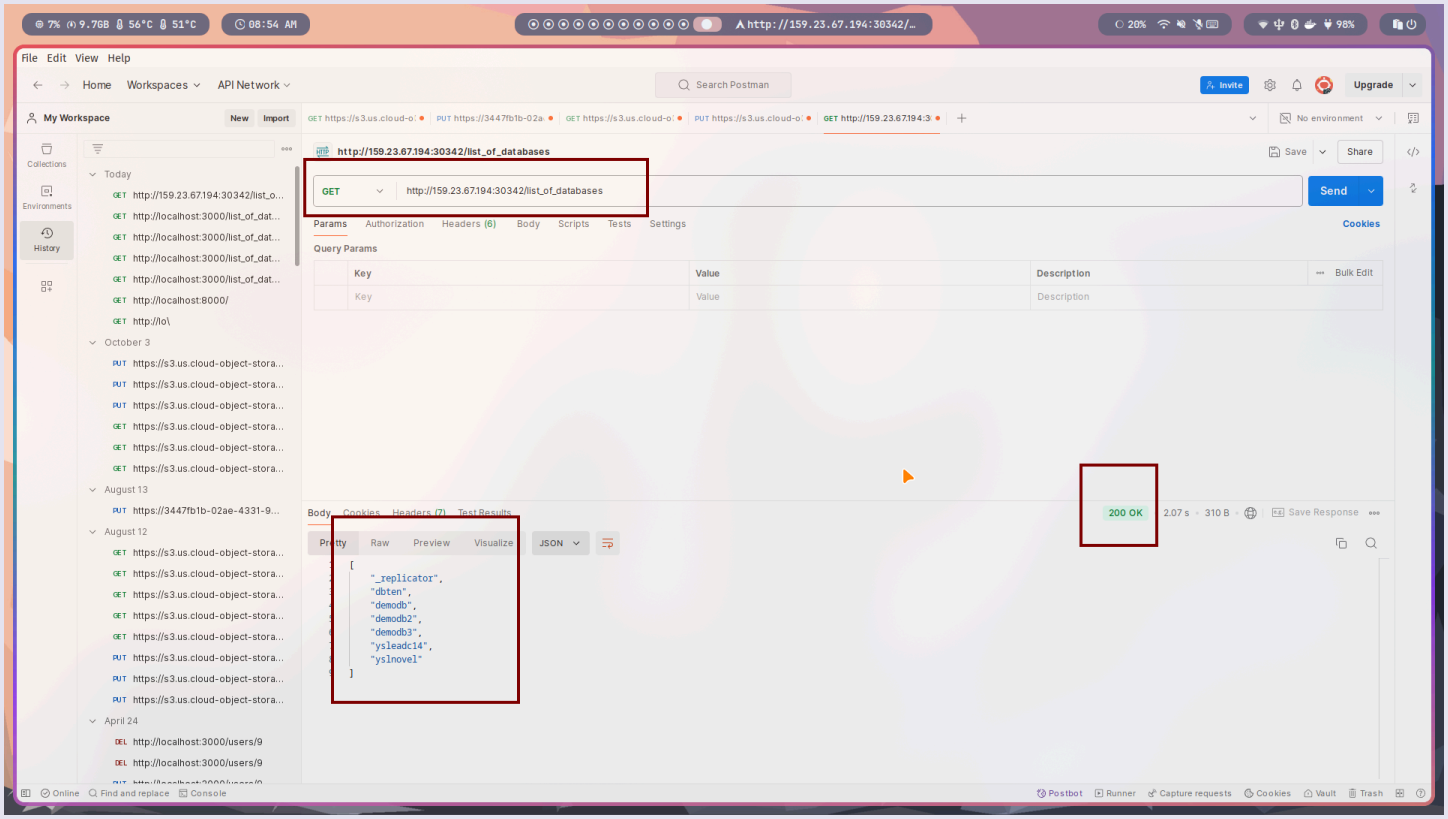
Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10



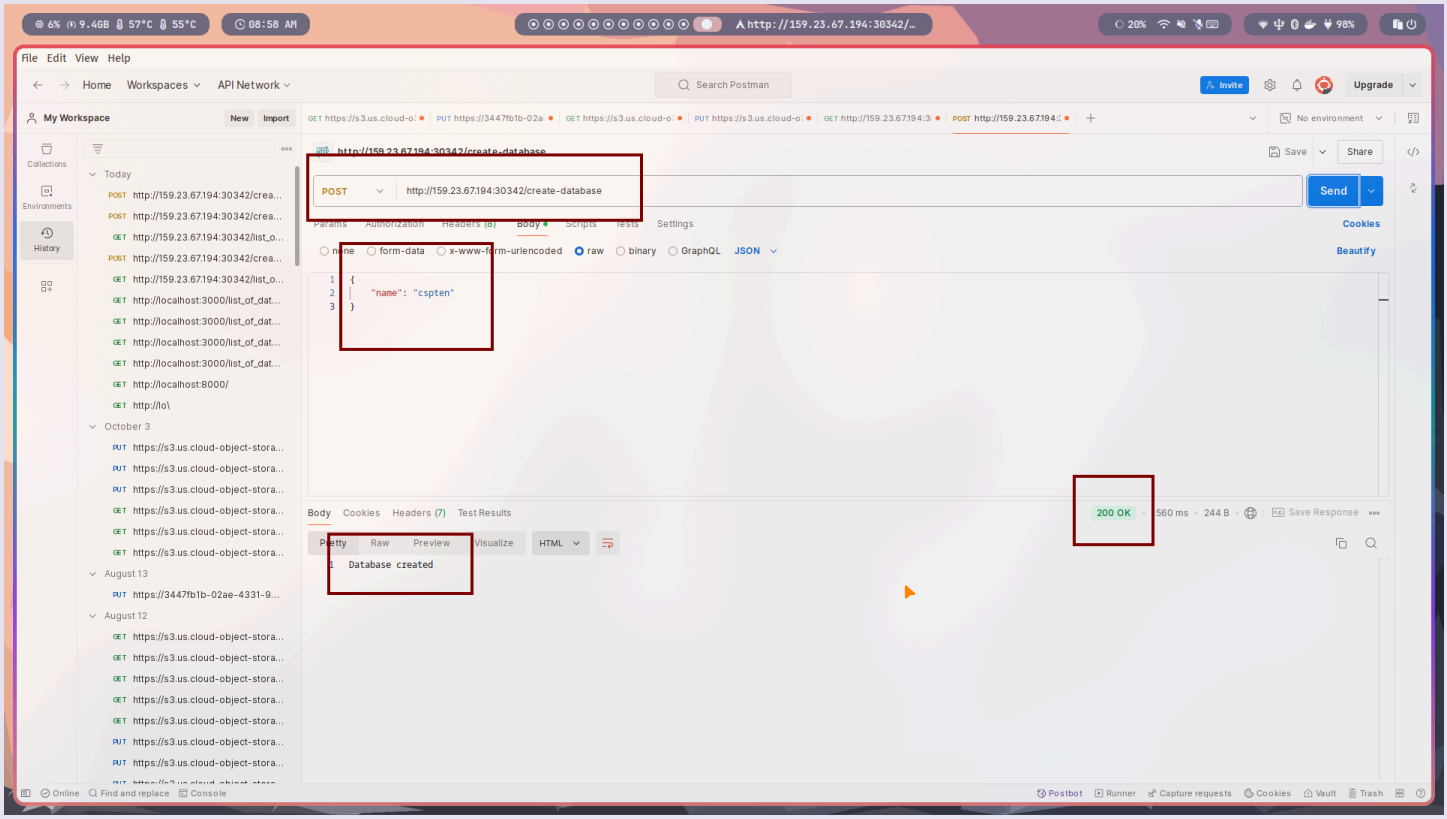


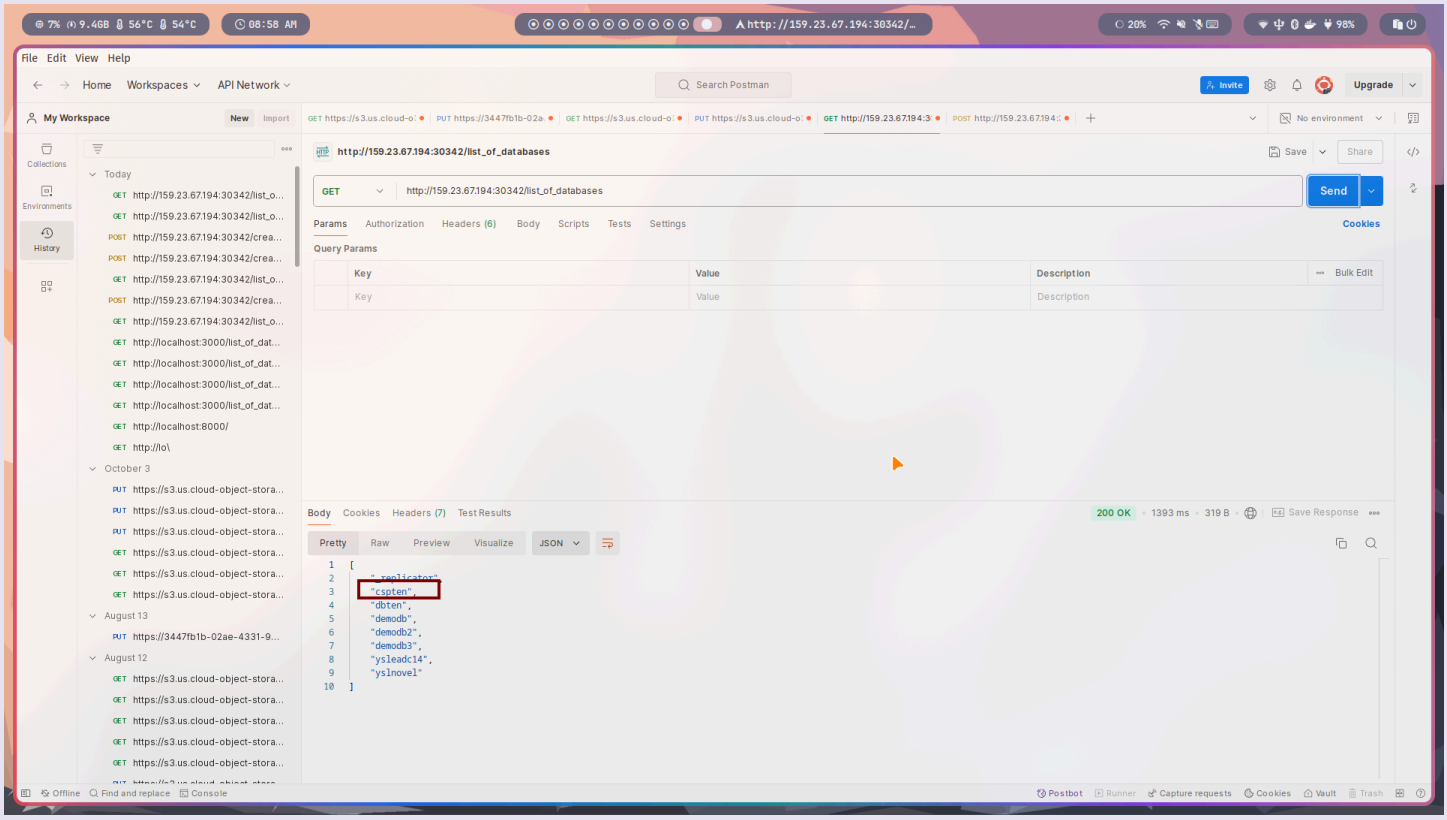
Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

5. Check requests via Postman if the application accepts so



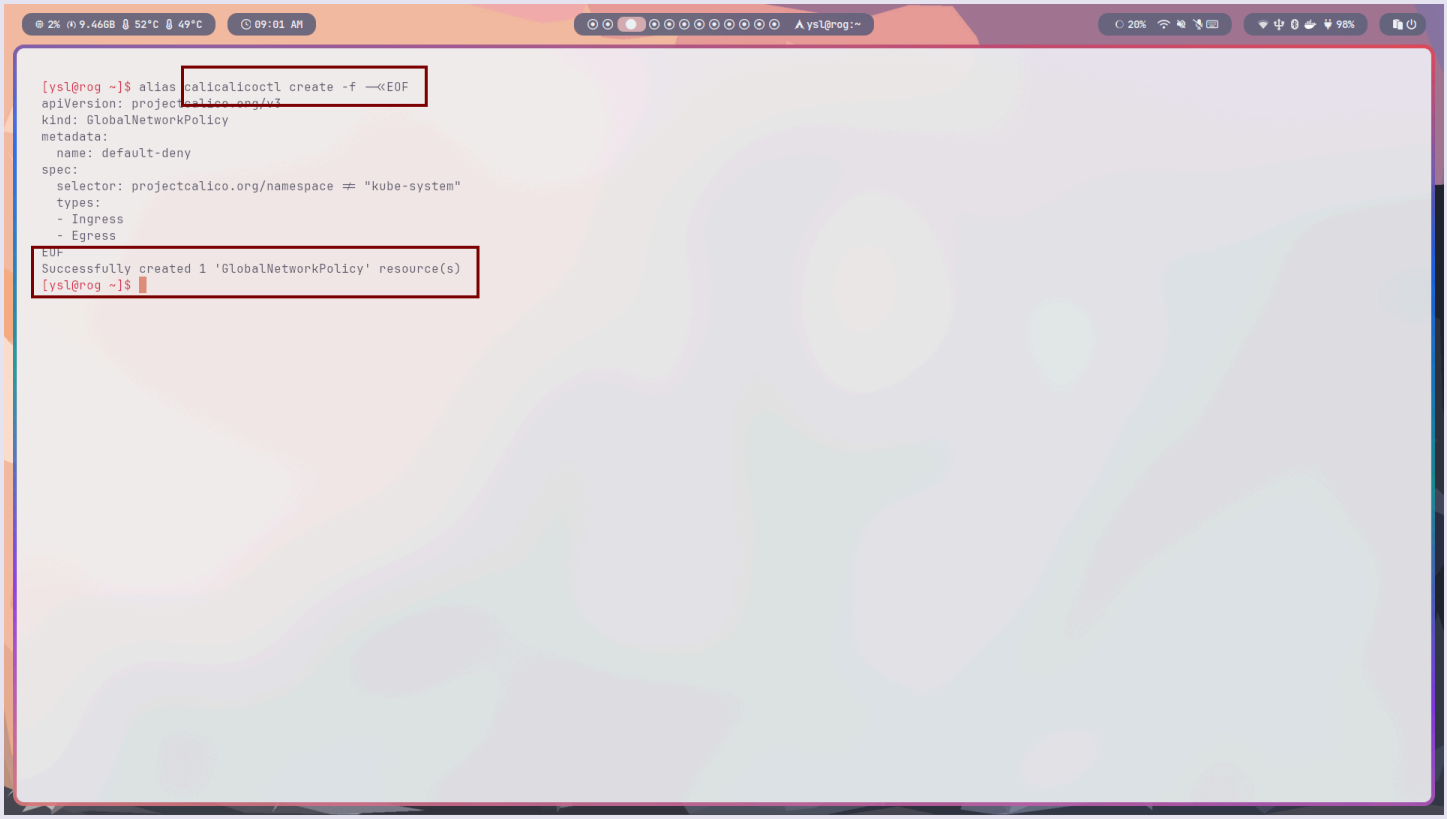
Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10





Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

6. Create global network policy to deny both egress and ingress traffic



Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CS Practical 10

7. Now the requests won't be accepted