|  |  |  |
| --- | --- | --- |
| Name |  | Vignesh M |
| Roll No |  | SSNCE195001128 |
| Team ID |  | PNT2022TMID53102 |
| Project Name |  | Skill and Job Recommander |
|  | **Assignment - 4** | |
|  | **Kubernetes and Docker** | |

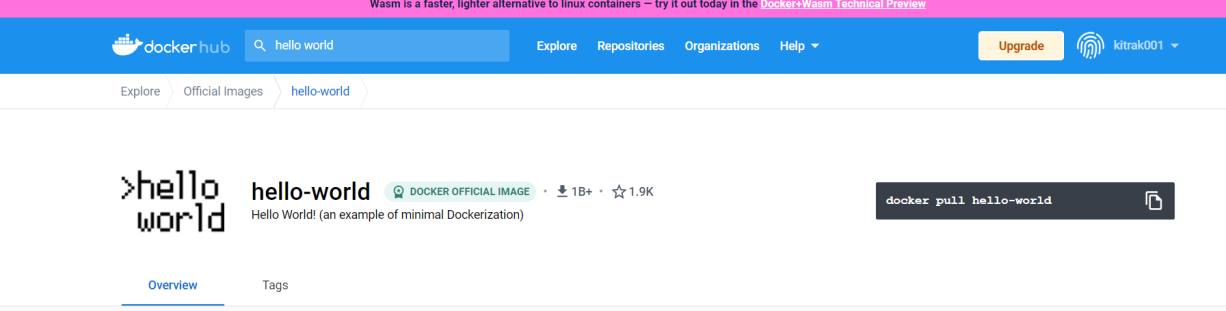
Question

1. Pull an Image from docker hub and run it in Docker Playground
2. Create a docker file for the jobportal application and deploy it in Docker desktop application
3. Create a IBM container registry and deploy helloworld app or jobportal app
4. Create a Kubernetes cluster in IBM cloud and deploy helloworld image or jobportal image and also expose the same app to run in nodeport

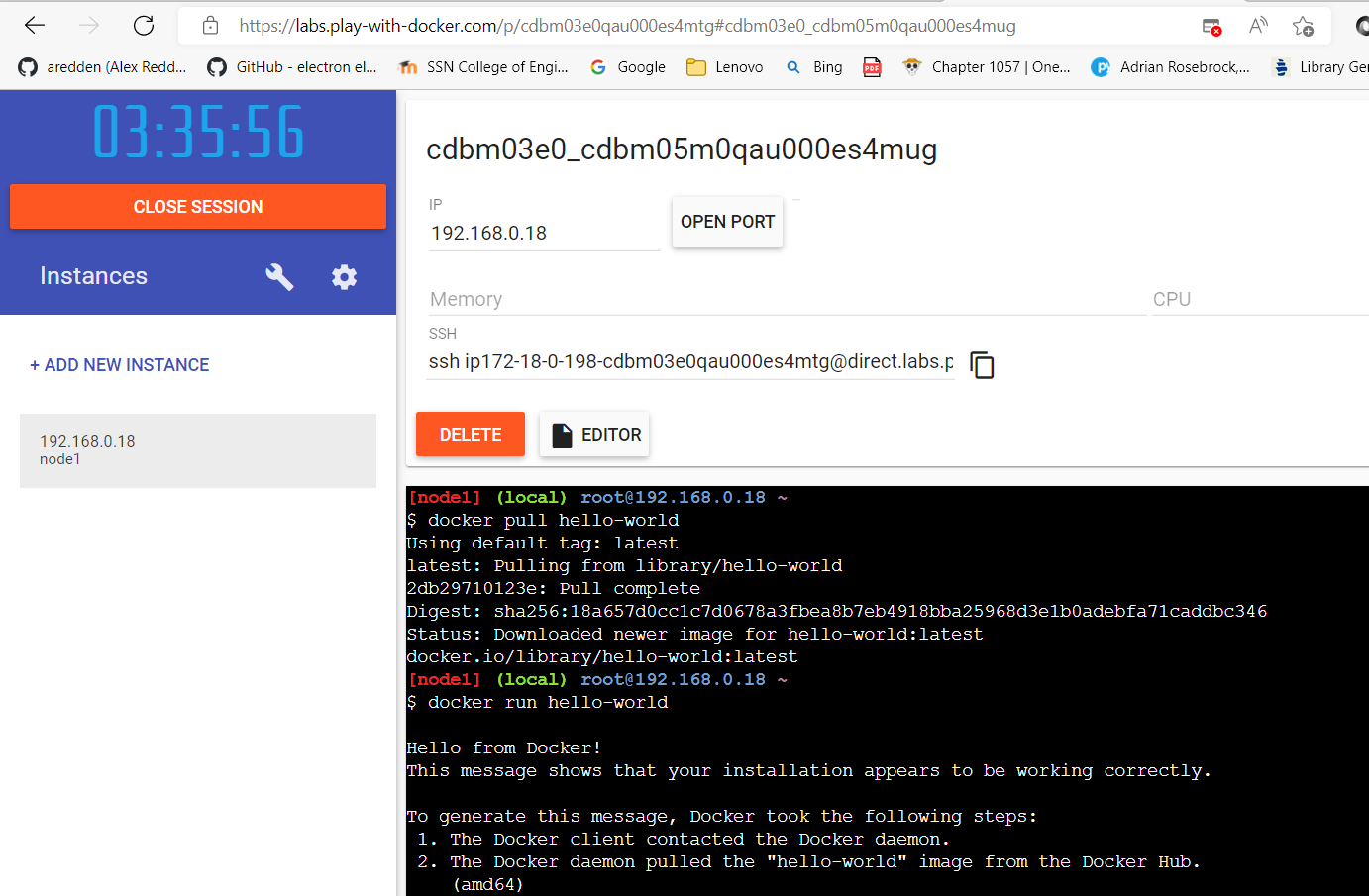
**Solutions**

1. **Pull an Image from docker hub and run it in Docker Playground**
   1. Pull an image *uifd/ui-for-docker* from the docker hub
   2. This image is used for viewing and managing the docker engine
   3. Use **docker pull image\_name** and **docker run -it image\_name** commands to run the above image in the Docker Playground

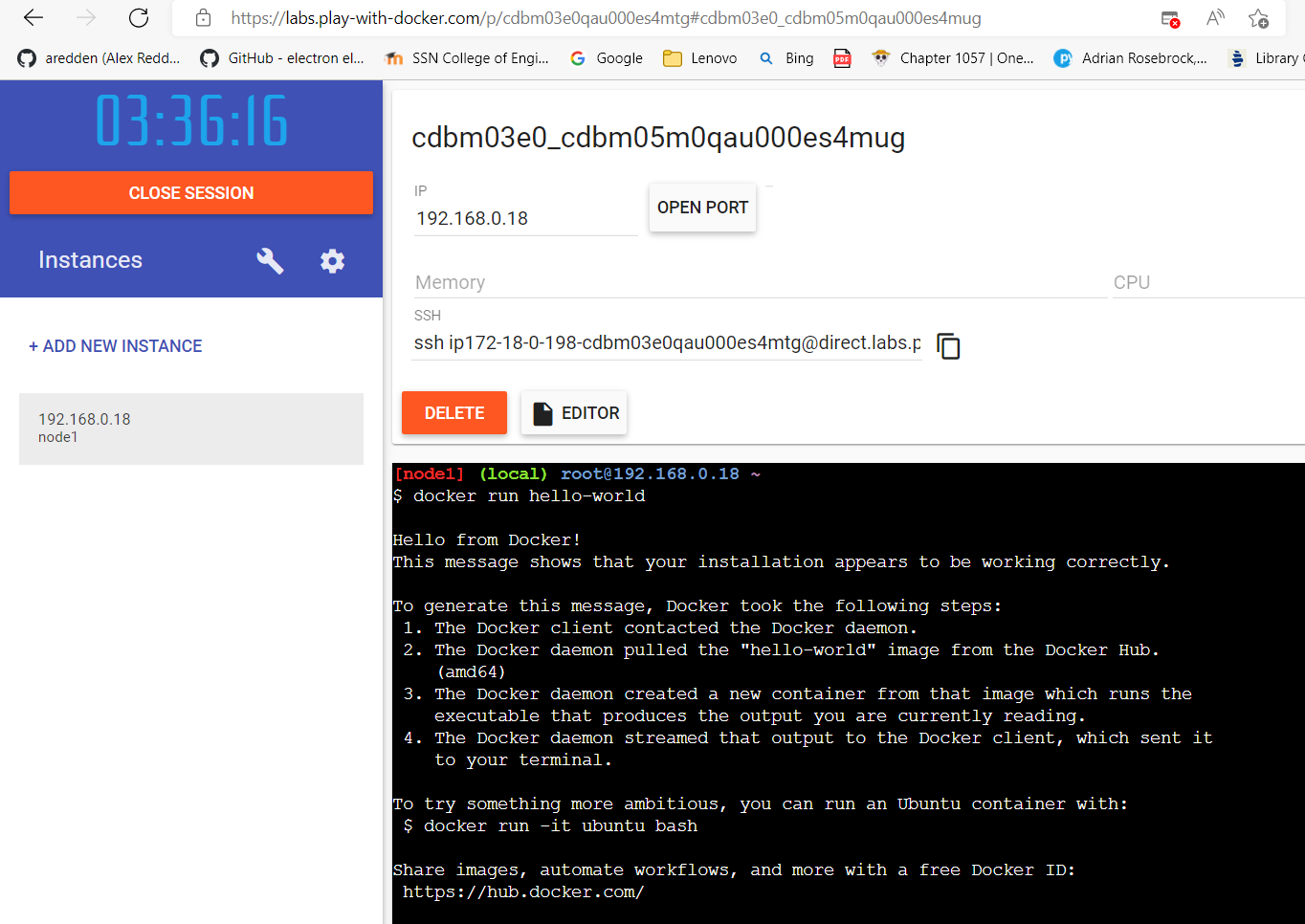
**[hello-world - Official Image | Docker Hub](https://hub.docker.com/_/hello-world)**(<https://hub.docker.com/_/hello-world>)



Pull



Run



**2. Create a docker file for the jobportal application and deploy it in Docker**

**desktop application**

1. Create a docker file for build and deploy flask app.
2. Use **docker build -t image\_name .** in the current directory to start building the docker image and deploy in our local docker
3. Use **docker run -p 5000:5000 image\_name** to run in local system

**Dockerfile**

FROM python

COPY ./requirements.txt /flaskApp/requirements.txt

WORKDIR /flaskApp

RUN pip install scipy

RUN pip install -r requirements.txt

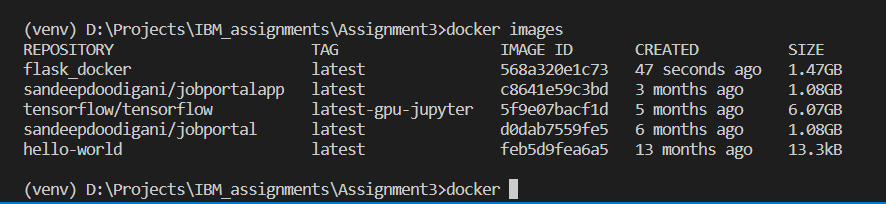
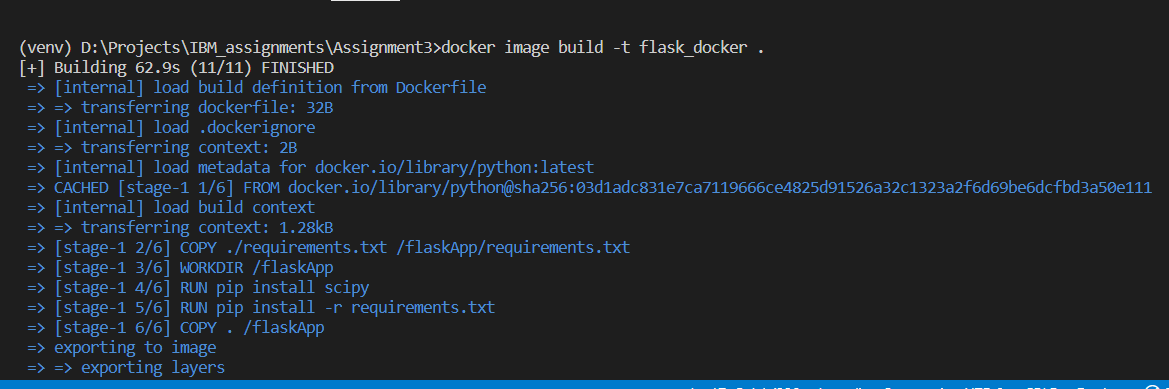
COPY . /flaskApp

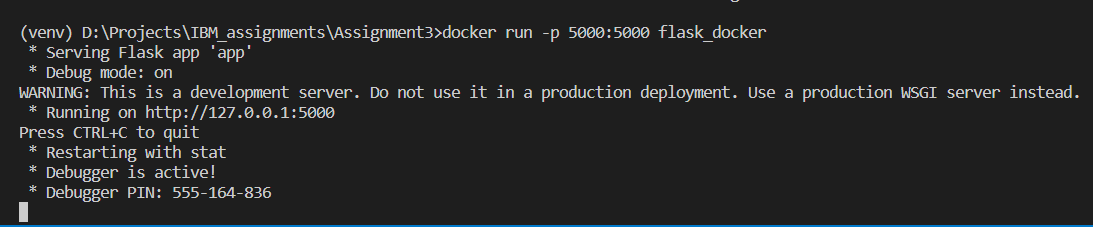
ENTRYPOINT [ "python" ]

CMD ["app.py" ]

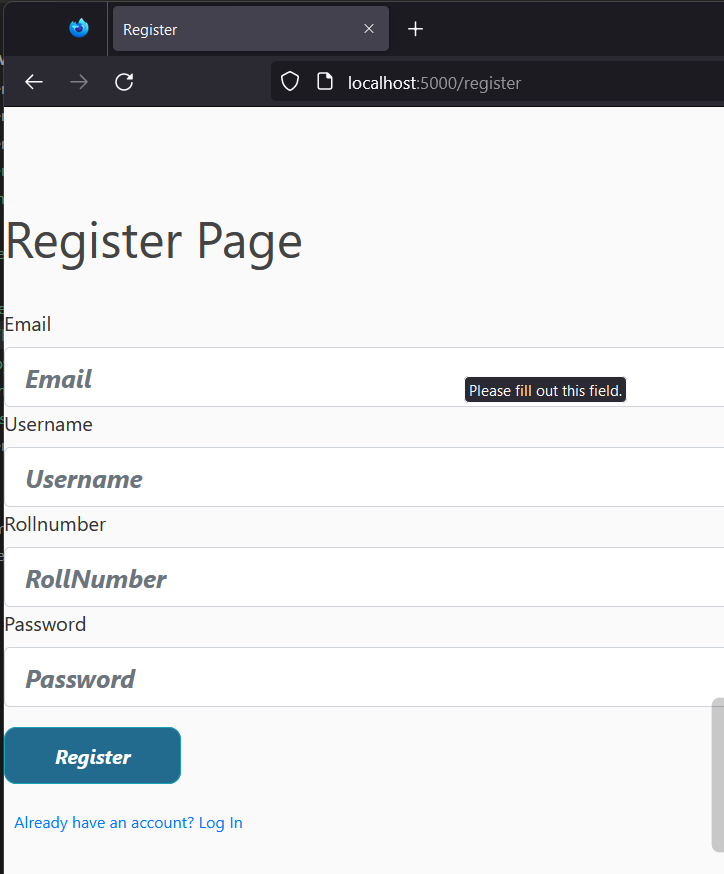
EXPOSE 5000

Steps:

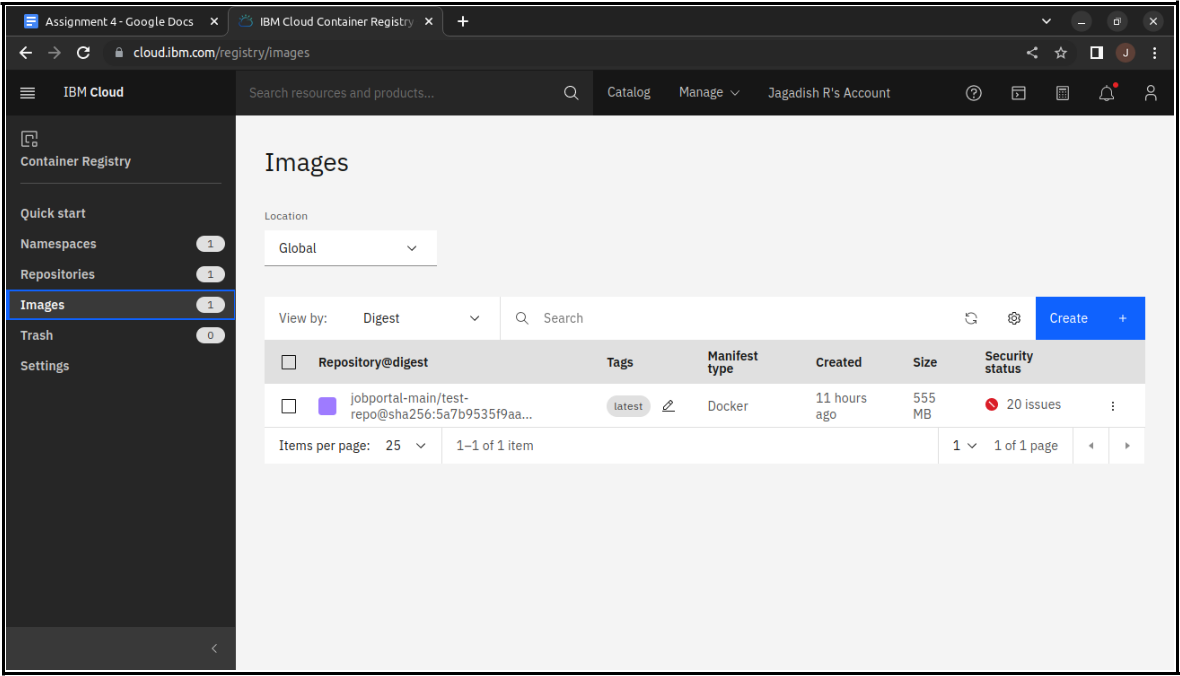




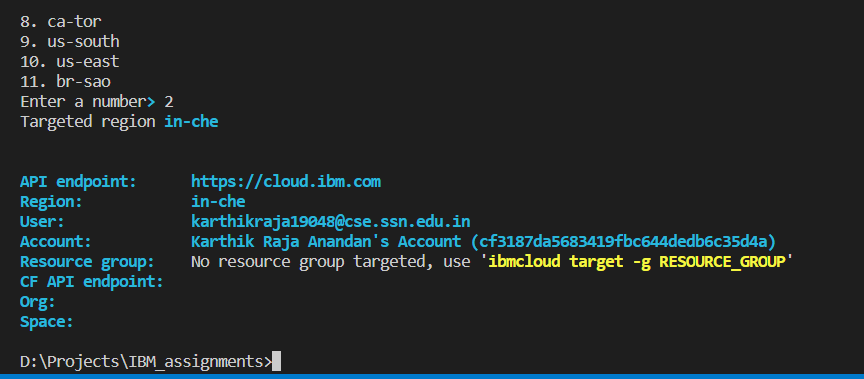
**Run locally using docker**

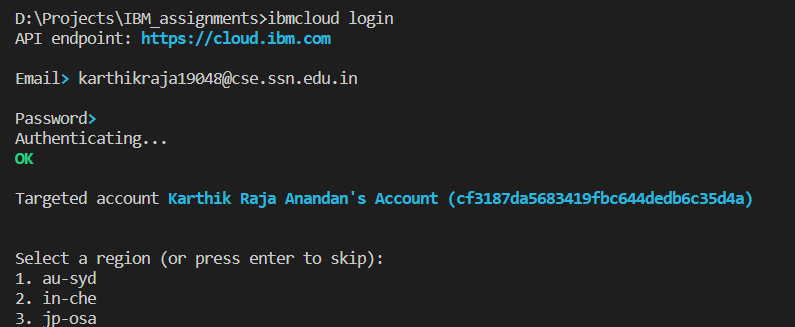


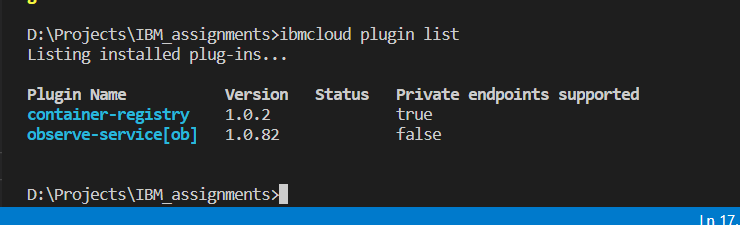
1. **Create a IBM container registry and deploy helloworld app or jobportal app**
   1. Log into IBM cloud
   2. Create a **container registry**
   3. Using IBM Cloud CLI, install the **container registry plugin** in our system
   4. Push our docker image into the created container registry using **docker push**
   5. So, our job portal app is deployed in the IBM container registry

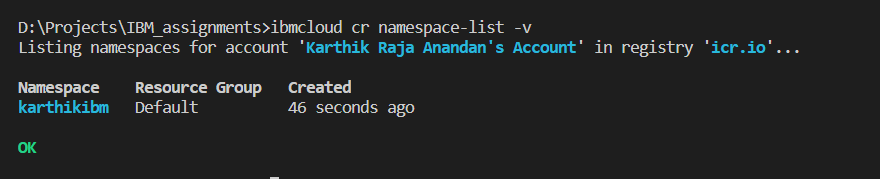
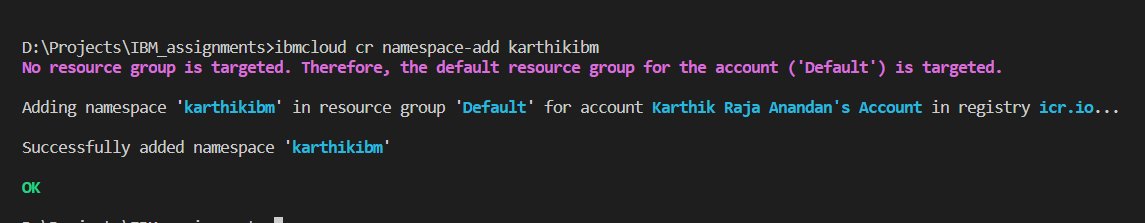
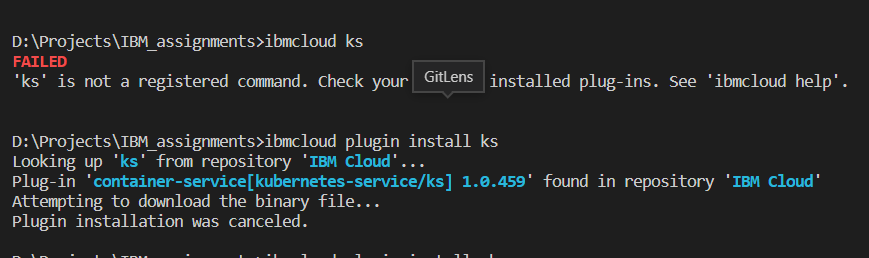


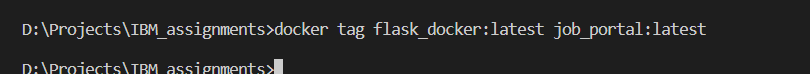
1. **Create a Kubernetes cluster in IBM cloud and deploy helloworld image or jobportal image and also expose the same app to run in nodeport**
   1. Log into IBM cloud
   2. Create a **kubernete**
   3. Using IBM Cloud CLI, install the **ks plugin** in our system
   4. Create **a cluster** in the kubernetes
   5. Now, go to the **kubernetes dashboard** where we need to create a service based on a yml file (given below)
   6. In that file, we have to mention *which image we are going to use* and the *app name*
   7. Take the **public IP address** and **Nodeport** since we exposed the *flask app in nodeport*
   8. Finally, we got the **url address** where our flask app is hosted

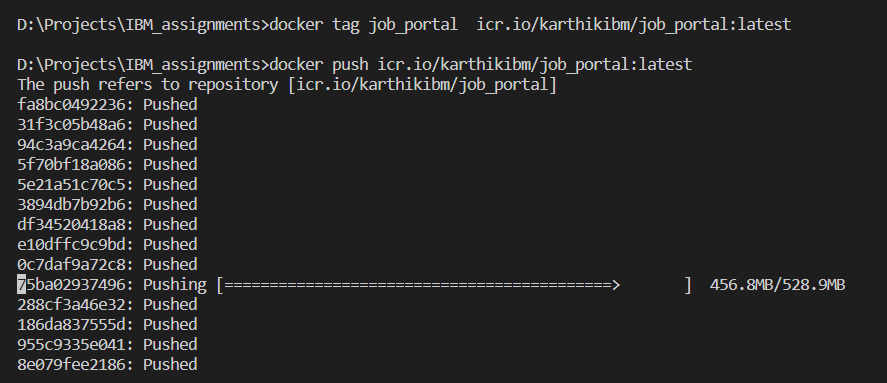


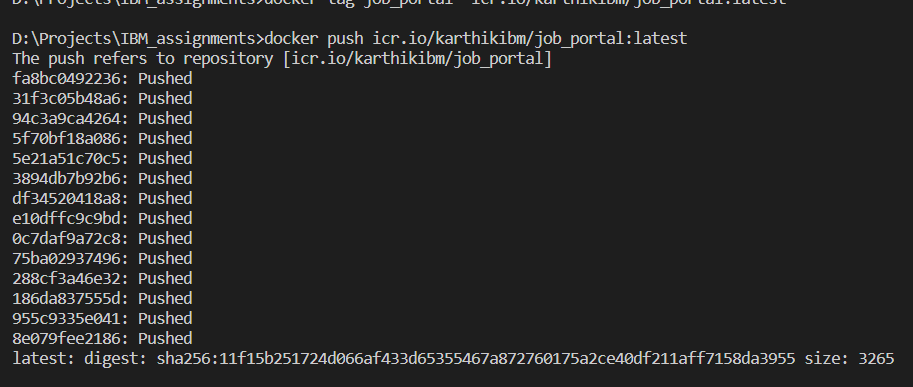


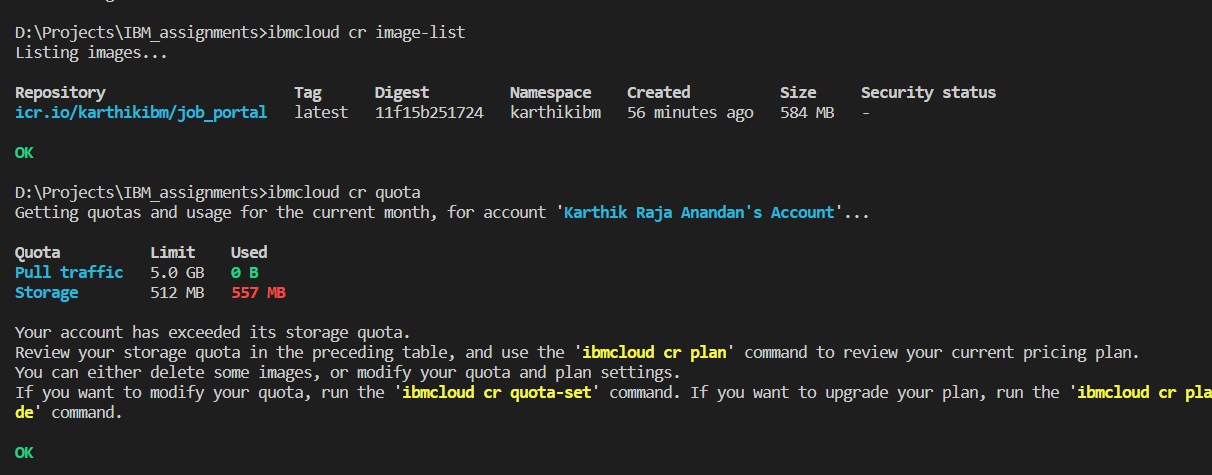












**job-portal-app.yml**

apiVersion: v1

kind: Service

metadata:

name: job-portal-app

spec:

selector:

app: job-portal-app

ports:

* port: 5000

type: NodePort

---

apiVersion: apps/v1

kind: Deployment

metadata:

name: job-portal-app

labels:

app: job-portal-app

spec:

selector:

matchLabels:

app: job-portal-app

replicas: 1

template:

metadata:

labels:

app: job-portal-app

spec:

containers:

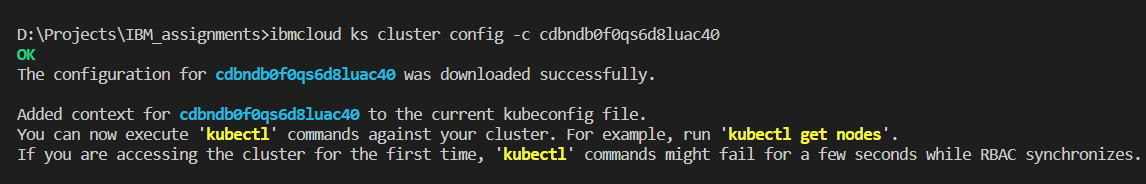
* name: job-portal-app

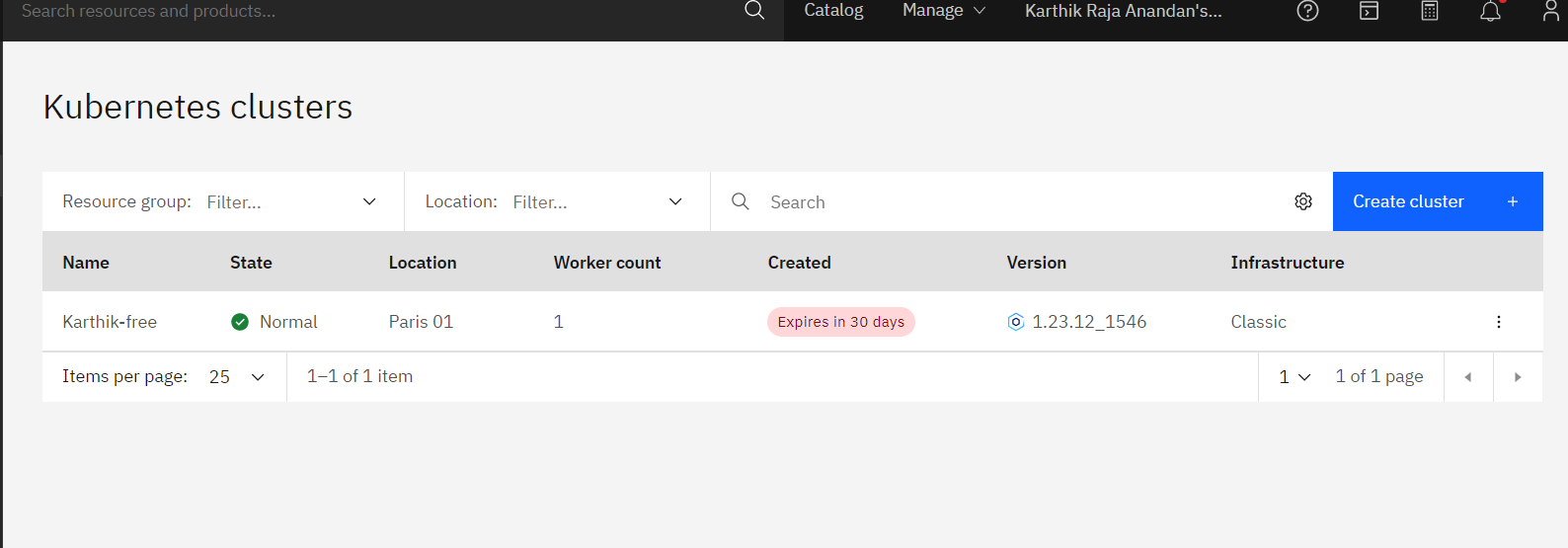
image: image\_name ports:

* + containerPort: 5000 env:
  + name: DISABLE\_WEB\_APP

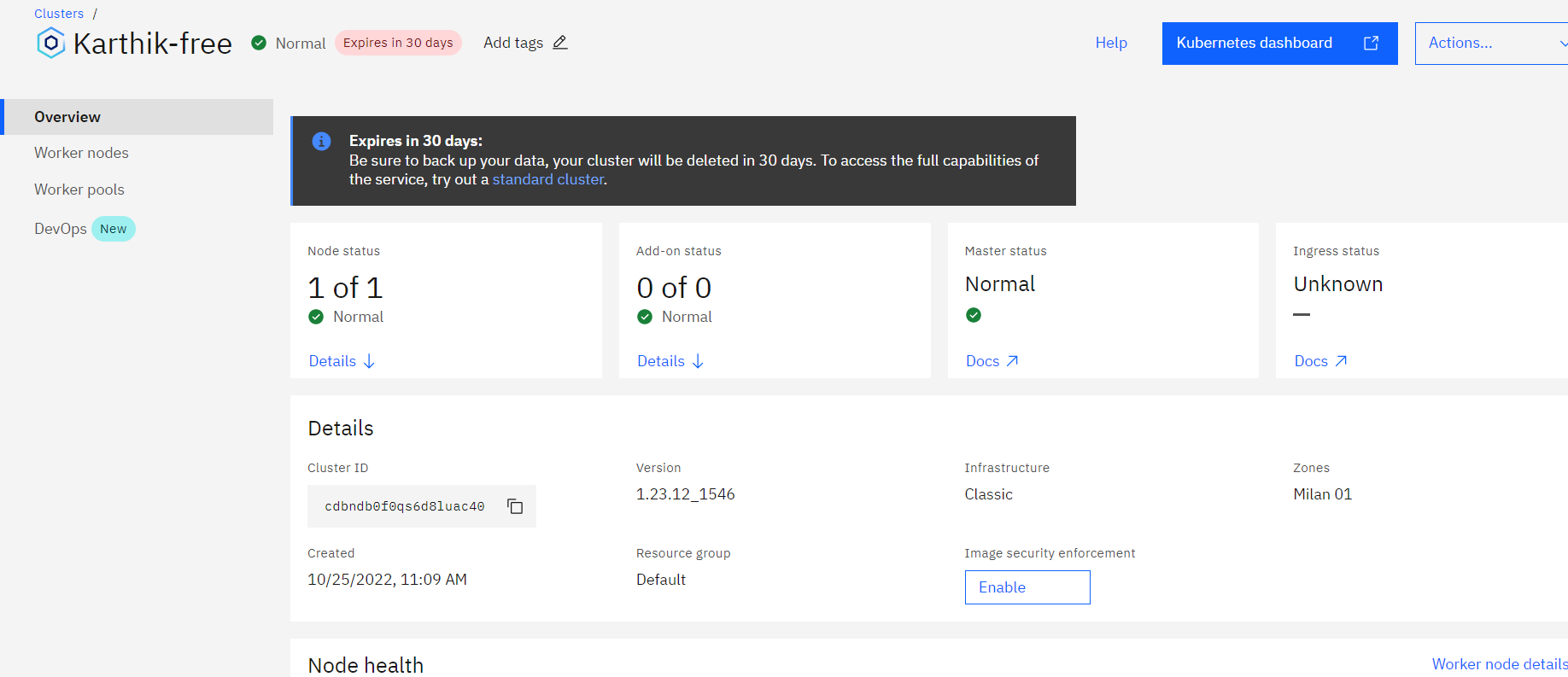
value: "false"

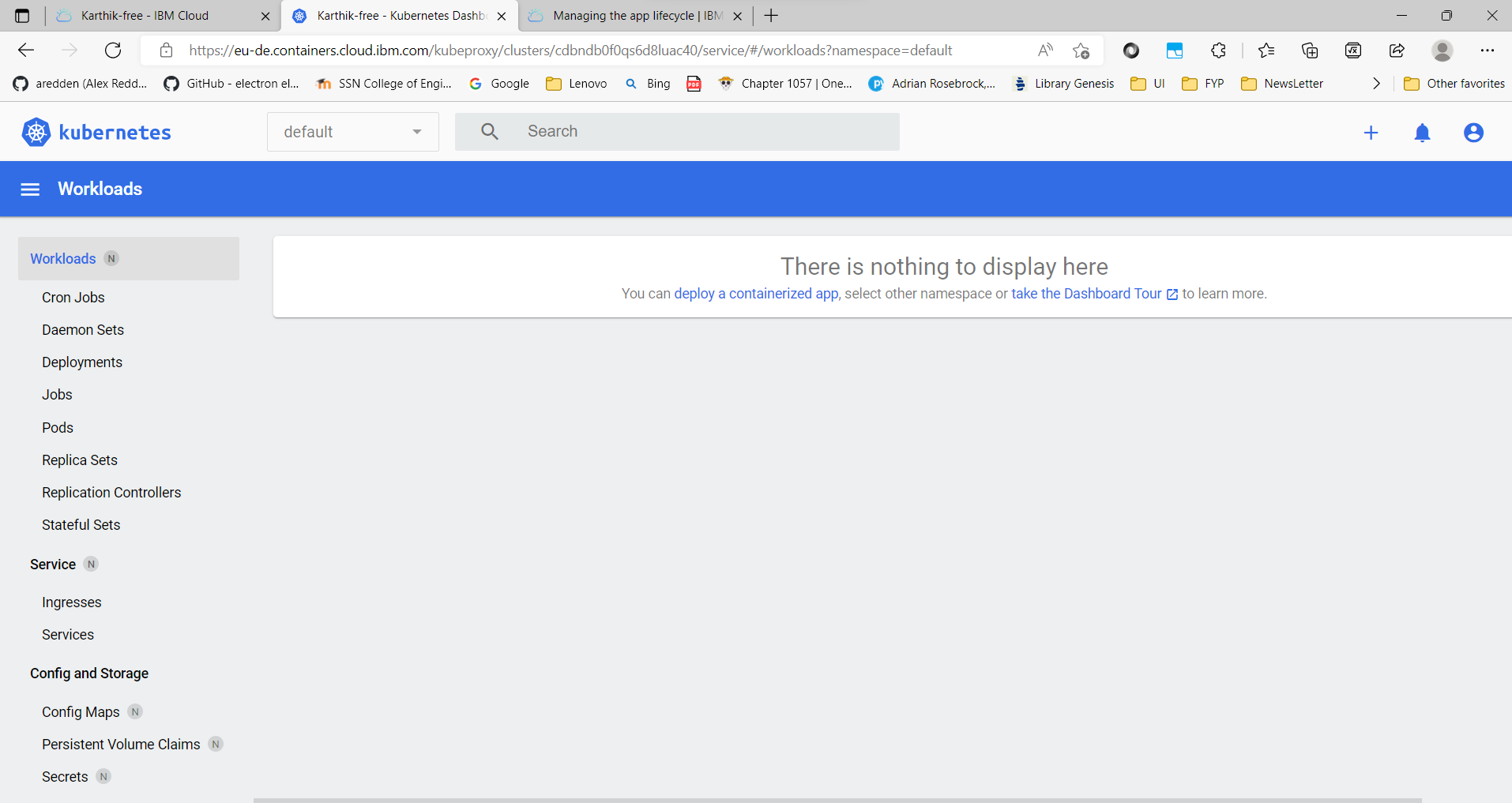
**Cluster creation**

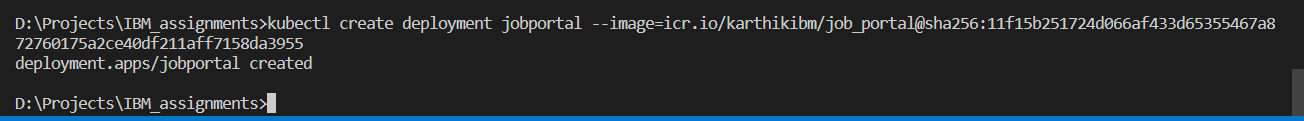
****

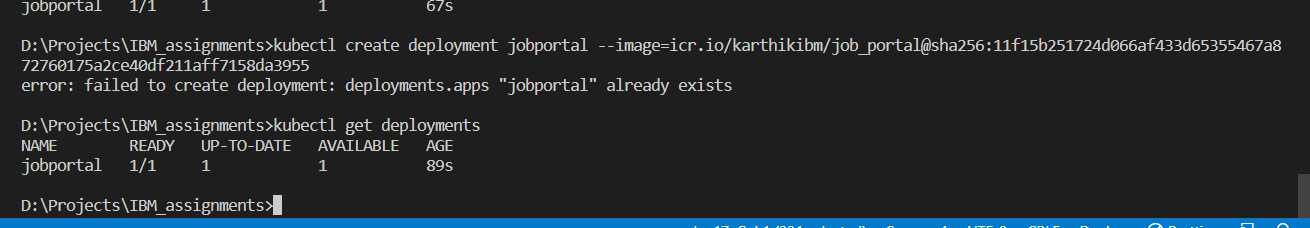


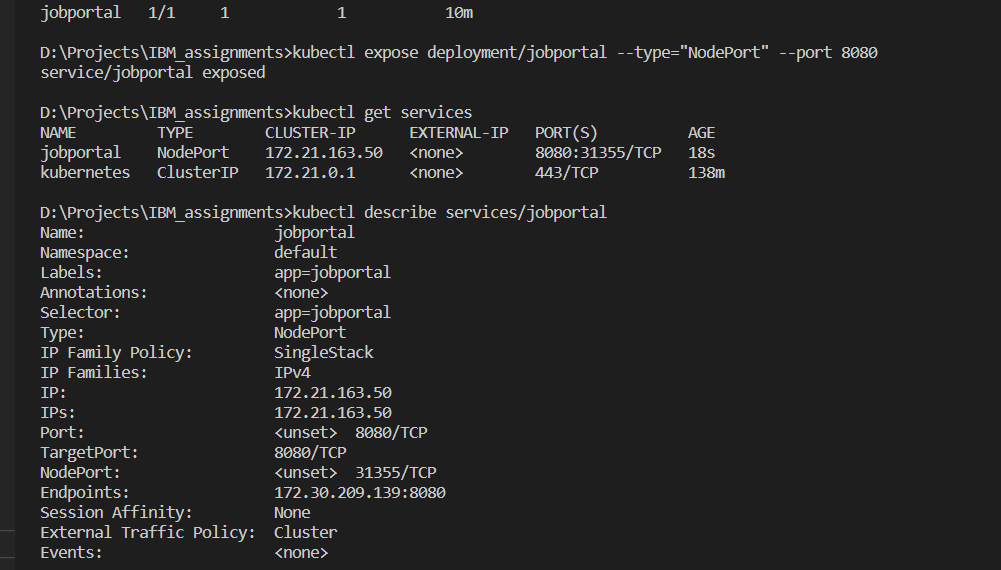
**Configuring the cluster**



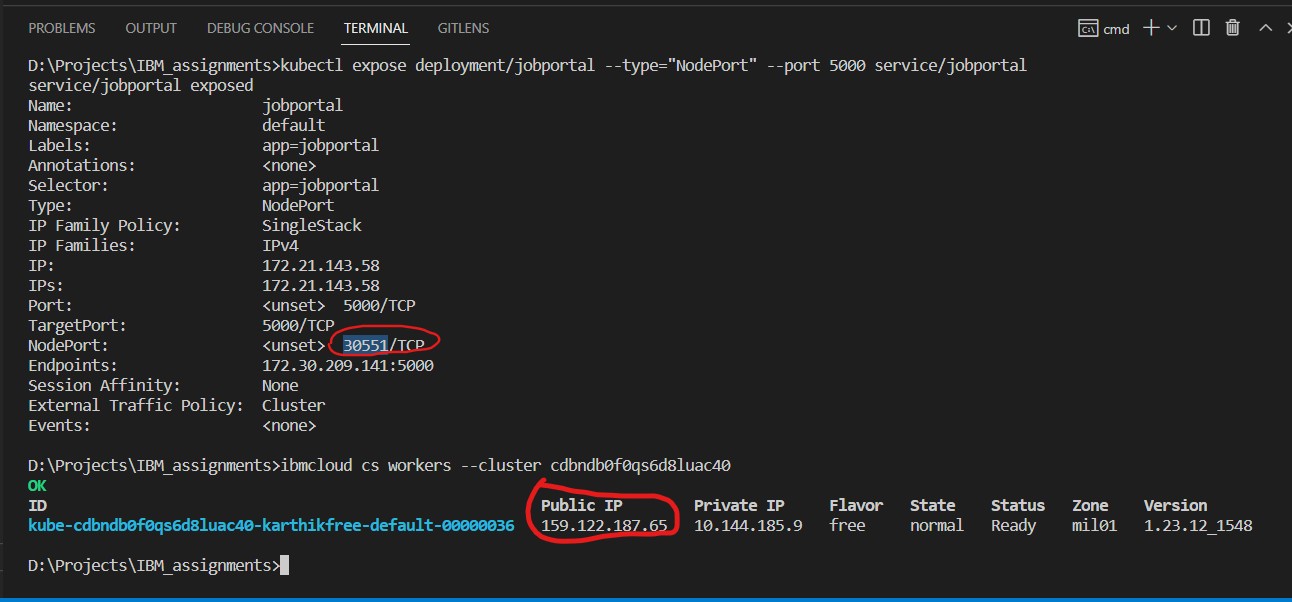








**Procedure to find the exposed url**



]

**Run our flask app in the IBM kubernetes**

