

DEPLOYMENT OF APP IN IBM CLOUD

DEPLOY IN KUBERNETES

Team ID	PNT2022TMID53529
Project Name	Customer Care Registry

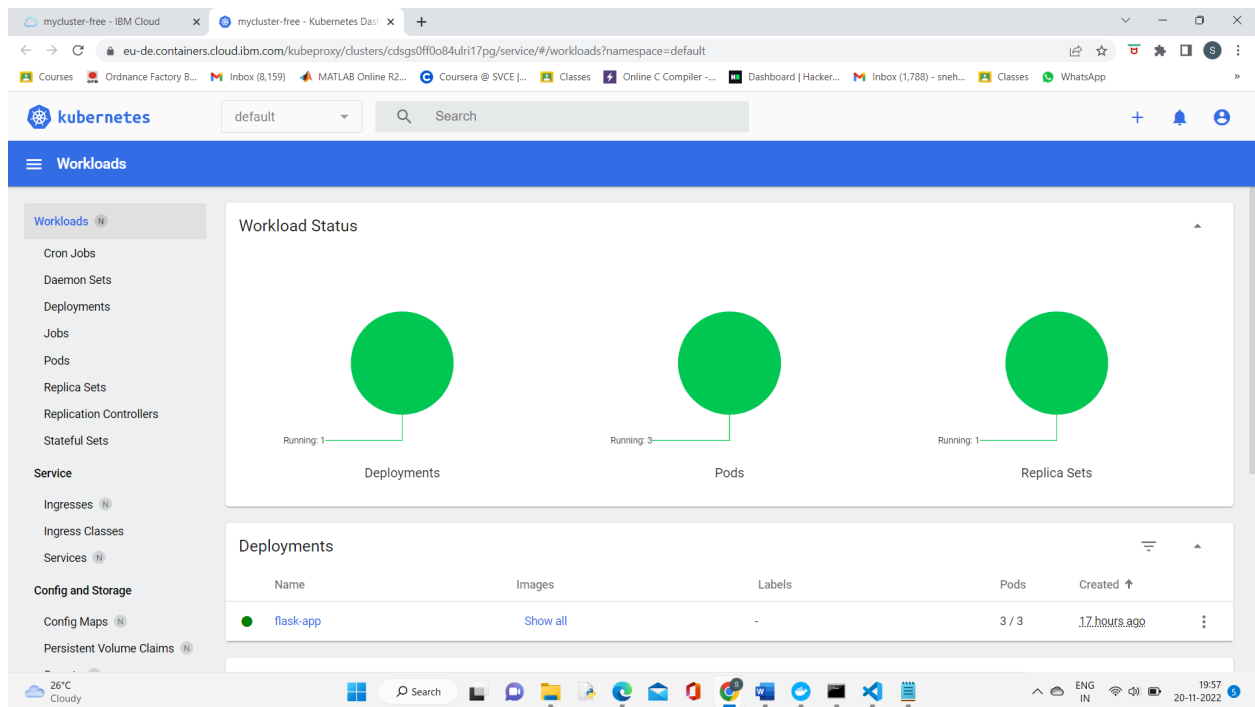
Step 1: Create a Kubernetes cluster in

<https://cloud.ibm.com/kubernetes/catalog/create>

The screenshot displays the IBM Cloud Kubernetes clusters management interface. The main content area shows a table of clusters. The table has columns for Name, State, Location, Worker count, Created, Version, and Infrastructure. A single cluster, 'mycluster-free', is listed with a 'Normal' state, located in 'Paris 01', and has 1 worker. The 'Created' column indicates it expires in 29 days. The 'Version' is 1.24.8_1544, and the 'Infrastructure' is Classic. The interface includes a sidebar with navigation options like 'Kubernetes', 'Clusters', 'Reservations', 'Helm catalog', and 'Container Registry'. The top navigation bar shows 'Catalog', 'Manage', and 'Sneha M's Account'.

Name	State	Location	Worker count	Created	Version	Infrastructure
mycluster-free	Normal	Paris 01	1	Expires in 29 days	1.24.8_1544	Classic

Step 2: Open the Kubernetes Dashboard



Step 3: Install IBM Cloud CLI (ibmcloud)

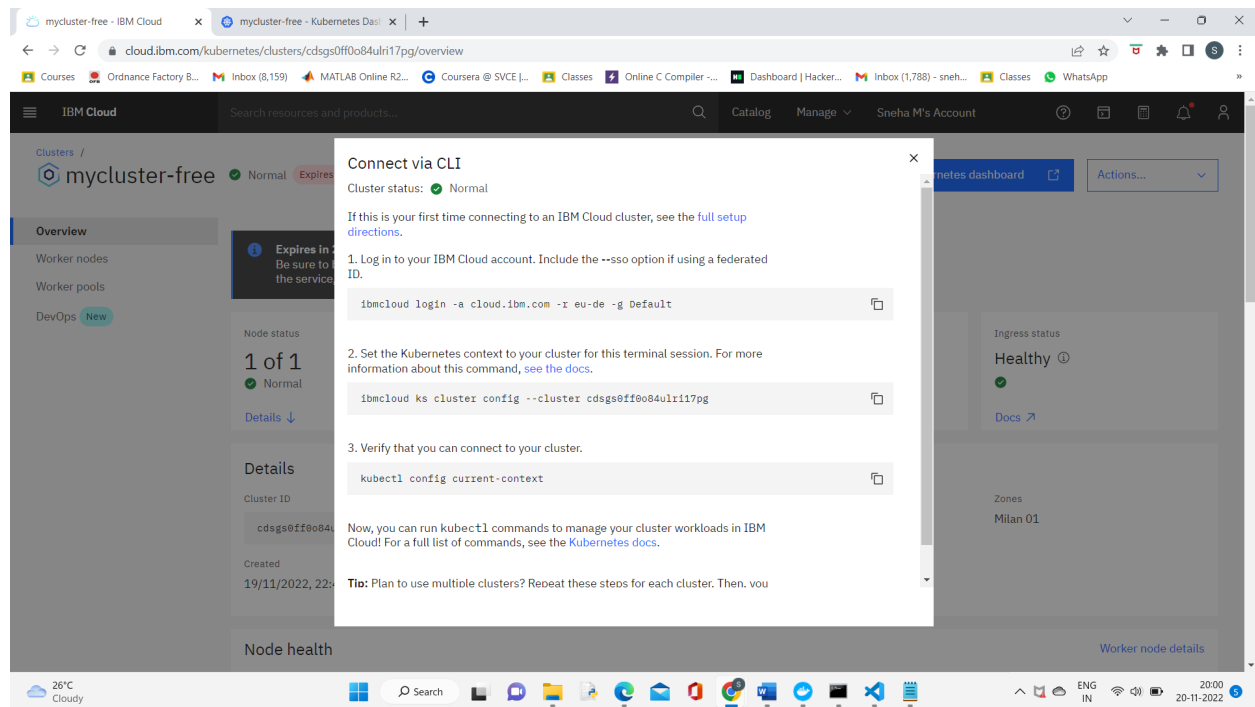
IBM Cloud Kubernetes Service plug-in (ibmcloud ks)

IBM Cloud Container Registry plug-in (ibmcloud cr)

IBM Cloud Kubernetes Service observability plug-in (ibmcloud ob)

https://cloud.ibm.com/docs/containers?topic=containers-cs_cli_install

Step 4: Follow the steps to connect via CLI



Step 5: Create files `ibm_deployment.yaml`, `flask_service.yaml`, `flask_ingress.yaml` files.

Step 6: Mention the name of the image from IBM container registry in `ibm_deployment.yaml`

Step 7: Execute the commands

```
kubectl config get-contexts
```

```
kubectl config use-context docker-desktop
```

```
kubectl apply -f kubernetes/ibm_deployment.yaml
```

```
kubectl apply -f kubernetes/flask_service.yaml
```

```
kubectl apply -f kubernetes/flask_ingress.yaml
```

```
kubectl get ing  
kubectl get svc  
kubectl get nodes -o wide
```

```
kubectl expose deployment flask-app --type=NodePort  
--name=flask-app
```

```
kubectl expose deployment flask_app --type=NodePort  
--name=flask-app-service
```

```
kubectl describe svc flask-app-service
```

```
kubectl describe svc flask-app-
```

Step 8: Get the cluster IP and the nodeport.

Step 9: Flask app is deployed in the below address.

http://<cluster-ip>:<port>