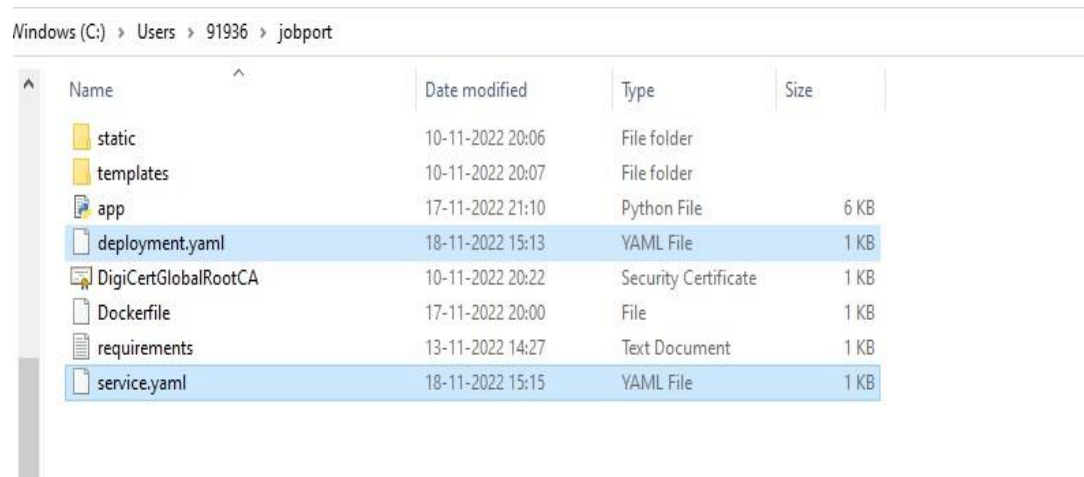


Deploy on Kubernetes Cluster

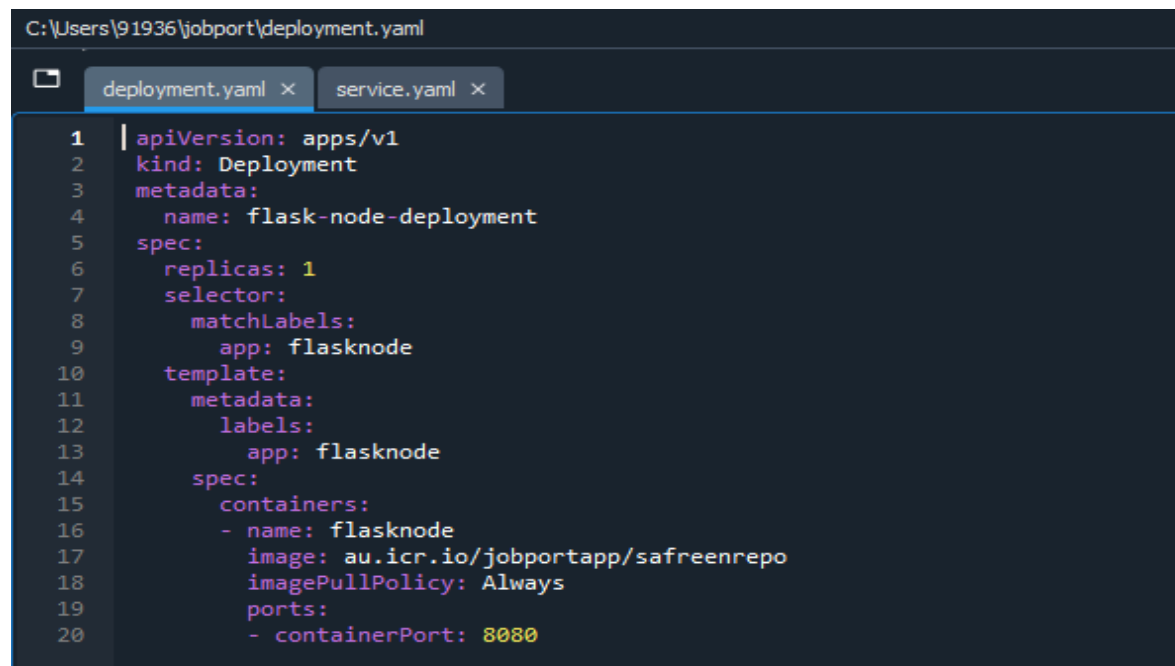
Deploy on Kubernetes

Date	18 November 2022
Team ID	PNT2022TMID23033
Project Name	Skill/Job Recommender Application

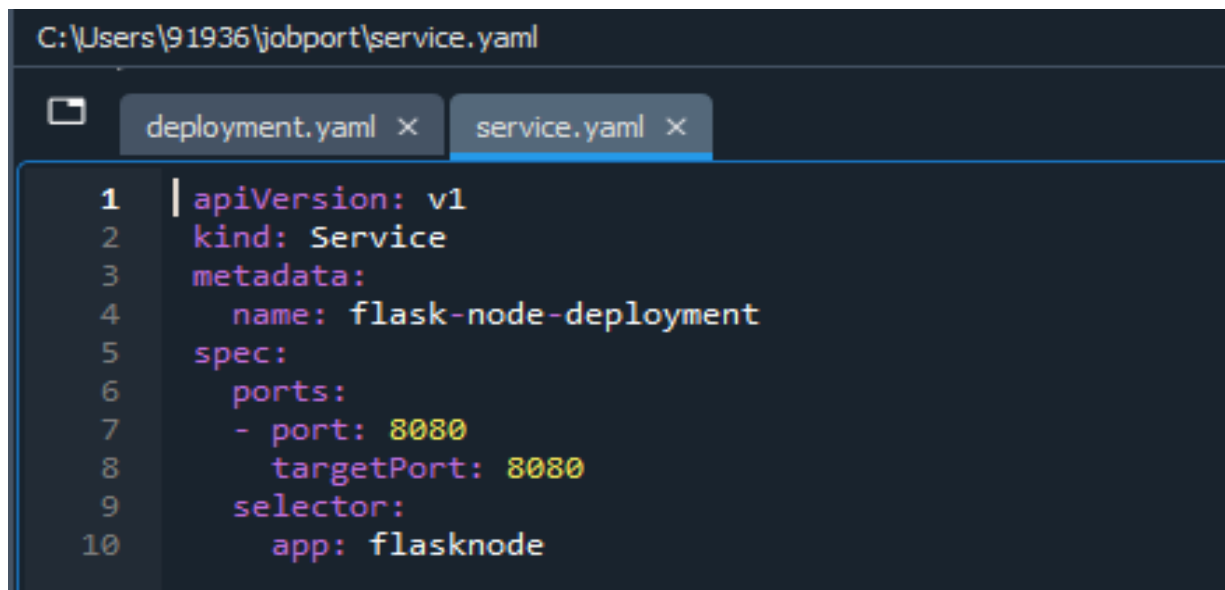
Step 1. Create configuration files for Kubernetes



Step 2: In the deployment.yaml file , add the following information



Step 3: In the service.yaml file , add the following details

A screenshot of a code editor window. The title bar shows the file path: C:\Users\91936\jobport\service.yaml. There are two tabs: deployment.yaml and service.yaml, with service.yaml being the active tab. The code is as follows:

```
1 | apiVersion: v1
2   kind: Service
3   metadata:
4     name: flask-node-deployment
5   spec:
6     ports:
7     - port: 8080
8       targetPort: 8080
9     selector:
10      app: flasknode
```

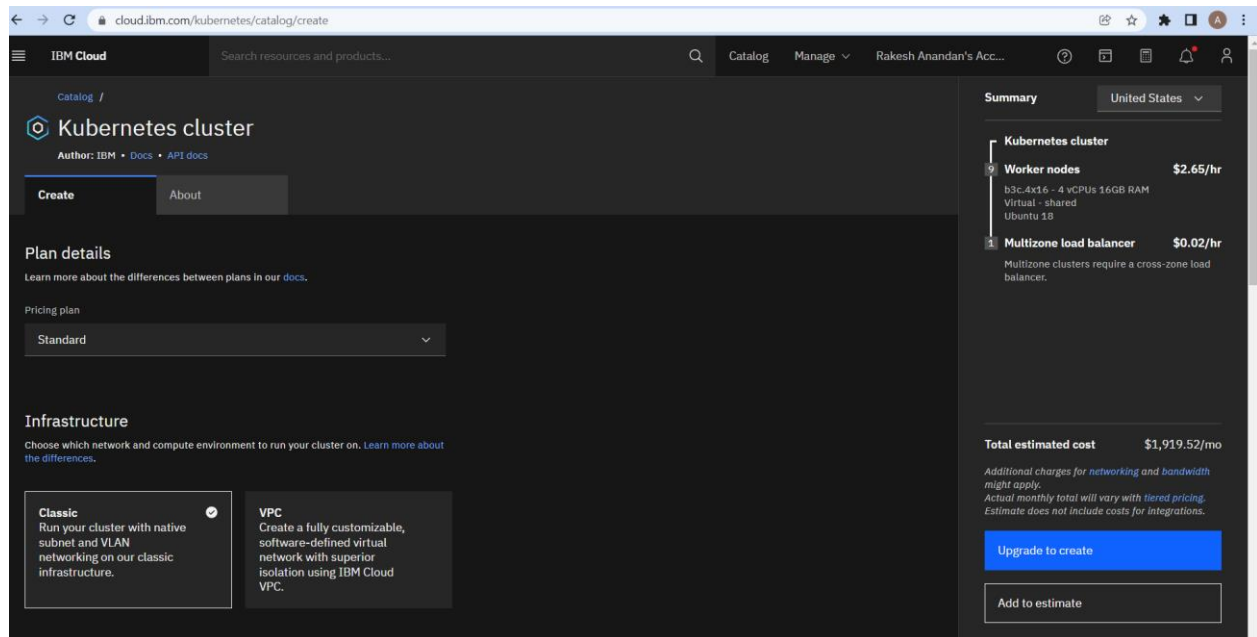
Explanation and breakdown of the deployment.yaml code

1. A deployment named flask-node-deployment is created, indicated by the .metadata.name field.
2. The deployment creates one replicated pod, indicated by the replicas field.
3. The selector field defines how the Deployment finds which Pods to manage. In this case, we simply select on one label defined in the Pod template (app: flasknode). However, more sophisticated selection rules are possible, as long as the Pod template itself satisfies the rule.
4. The pod template's specification, .template.spec, indicates that the pods run one container, flasknode, which runs the app private registry image.
5. The deployment opens port 8080 for use by the Pods.

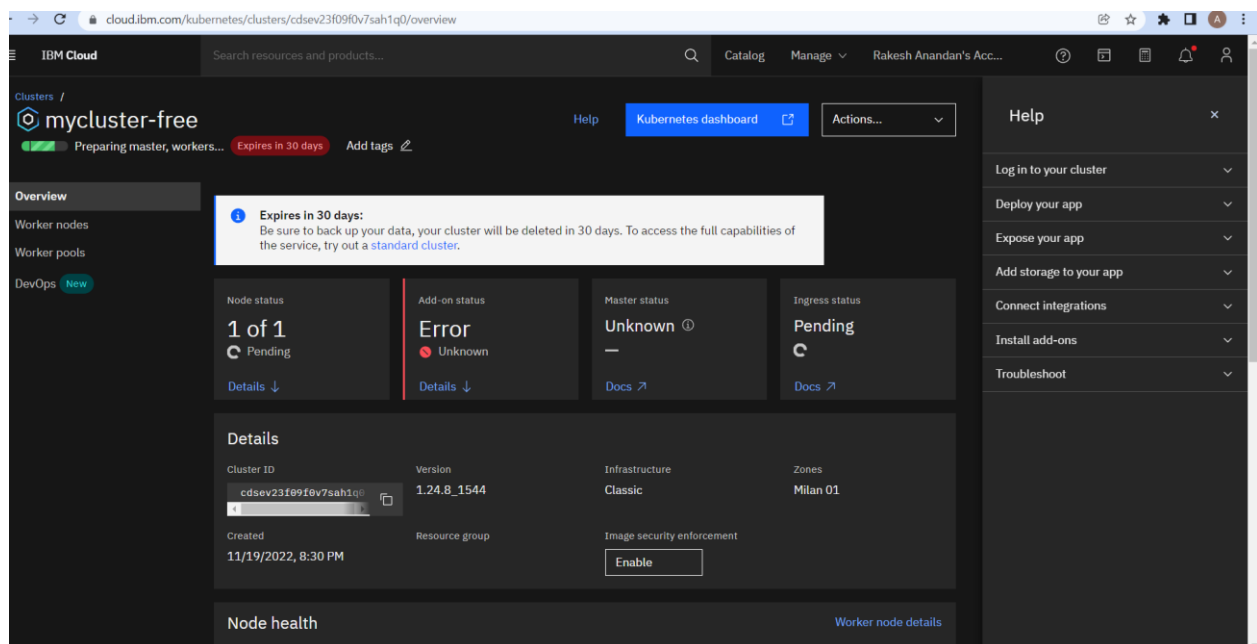
Explanation and breakdown of the service.yaml code

1. The service.yaml's specification will create a new service object named flask-node-deployment which targets TCP port 8080 on any Pod with the "app=flasknode" label. This Service will also be assigned an IP address (sometimes called the cluster IP), which is used by the service proxies (see below). The Service's selector will be evaluated continuously and the results will be POSTed to an Endpoints object also named flask-node-deployment.
2. Note that a service can map an incoming port to any targetPort. By default the targetPort will be set to the same value as the port field. Perhaps more interesting is that targetPort can be a string, referring to the name of a port in the backend Pods. The actual port number assigned to that name can be different in each backend Pod. This offers a lot of flexibility for deploying and evolving your Services. For example, you can change the port number that pods expose in the next version of your backend software, without breaking clients.

Step 4: Login to ibmcloud and in the Catalog , search for the Kubernetes Service and create one cluster



Step 5: When the Kubernetes cluster is opened , you see the page which contains Workerernodes, click the workernodes , Note the public Ip



Step 6: Click the Actions and click Connect via CLI

Step 7: Deploy your application to Kubernetes

Target the IBM Cloud Kubernetes Service region where you want to work.

```
ibmcloud cs region-set us-south
```

Set the context for the cluster in your CLI.

```
ibmcloud cs cluster-config cluster_kunal
```

Verify that you can connect to your cluster by listing your worker nodes.

```
kubectl get nodes
```

Create the deployment.

```
kubectl create -f deployment.yaml
```

Create the service.

```
kubectl create -f service.yaml
```

Step 8: Look at the Kubernetes dashboard from the IBM Kubernetes Service overview page.

The screenshot displays the Kubernetes Dashboard 'Overview' page. The left sidebar contains navigation links for Cluster, Namespaces, Nodes, Persistent Volumes, Roles, Storage Classes, Namespaces (Default), Overview (selected), Workloads (Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets), Discovery and Load Balancing (Ingresses, Services), and Config and Storage (Config Maps). The main content area shows several resource lists:

- Deployments:** A table with one entry: 'flask-node-deployment' with 1/1 pods, 3 minutes age, and image 'registry.ng.blazend.net/flask-node/app'.
- Pods:** A table with one entry: 'flask-node-deployment-5d8f6fc-ct6s' with Node '18.47.79.201', Status 'Running', 0 replicas, 3 minutes age, 0% CPU usage, and 16.332 MB memory usage.
- Replica Sets:** A table with one entry: 'flask-node-deployment-5d8f6fc' with 1/1 pods, 5 minutes age, and image 'registry.ng.blazend.net/flask-node/app'.
- Services:** A table with two entries:
 - 'kubernetes' with labels 'component: apiserver' and 'provider: kubernetes', Cluster IP '172.17.0.1', and internal endpoints 'kubernetes:443 TCP' and 'kubernetes:80 TCP'.
 - 'flask-node-deployment' with labels 'component: flask-node-deployment', Cluster IP '172.21.104.14', and internal endpoints 'flask-node-deployment:5000 TCP' and 'flask-node-deployment:80 TCP'.

Step 9 : Finally, go to your browser and ping the Public IP of your worker node.