

Assignment – 4

Cloud Application Development

Assignment Date	19 September 2022
Student Name	Harsha vardhan V
Student Roll Number	211719106027
Maximum Mark	2 Marks

1.Pull an Image from docker hub and run it in docker playground.

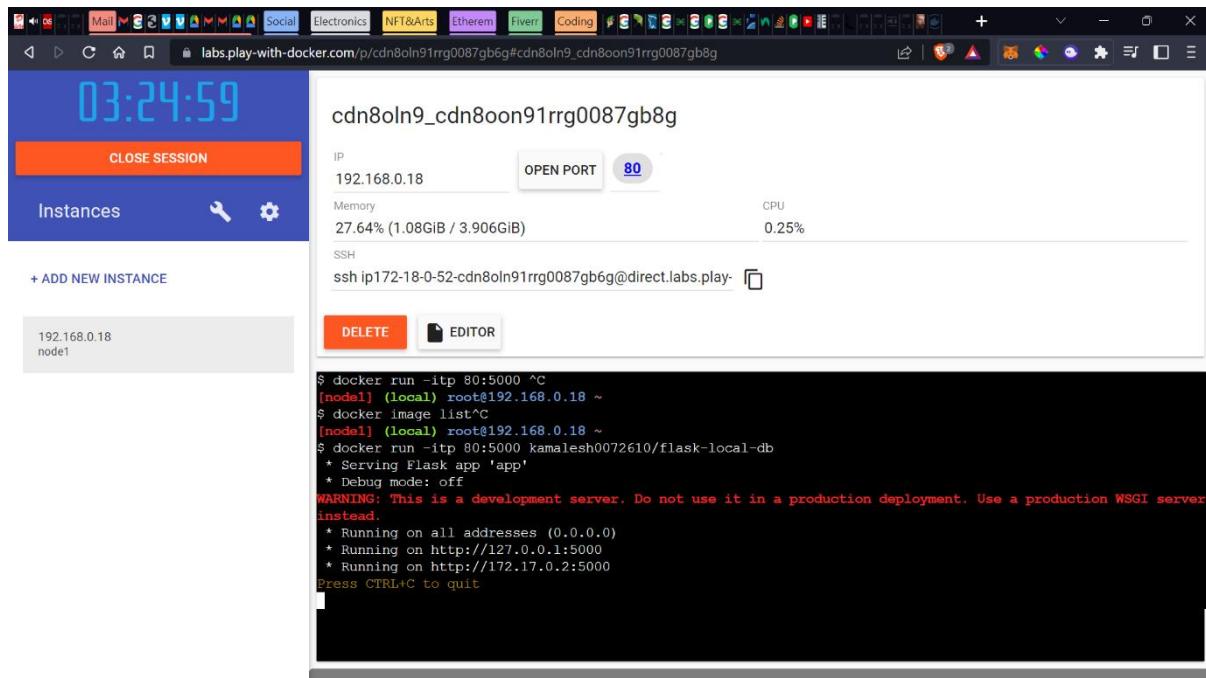
Pushed my own Image to Docker Hub and used that for this assignment.

```
docker pull kamalesh0072610/flask-local-db:latest
docker image list
```

The screenshot shows the Docker Playground interface. On the left, there's a sidebar with a timer at 03:26:56, a 'CLOSE SESSION' button, and an 'Instances' section. Below it, there's a '+ ADD NEW INSTANCE' button and a list of instances showing '192.168.0.18 node1'. The main area displays details for the instance 'cdn8oIn9_cdn8oon91rrg0087gb8g', including its IP address, memory usage (26.79%), CPU usage (0.45%), and an SSH command. Below this, there are 'DELETE' and 'EDITOR' buttons. The bottom part of the screen shows a terminal window with the following output:

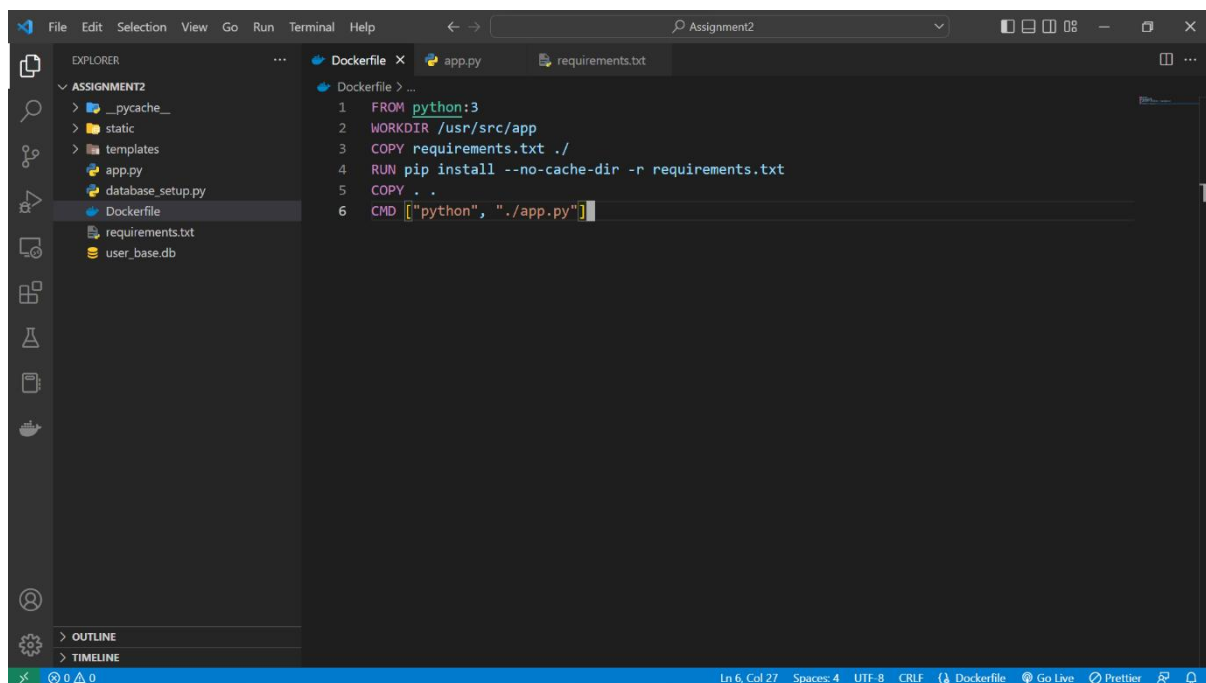
```
94da21e53a5b: Pull complete
95e3473ba5b7: Pull complete
fc16ec4a0f48: Pull complete
47721b0607f9: Pull complete
bd89fa278679: Pull complete
Digest: sha256:e3d6f139775465b9be3f2e7f7c43e27307c6e10f4fb2382d321c80830753e16
Status: Downloaded newer image for kamalesh0072610/flask-local-db:latest
docker.io/kamalesh0072610/flask-local-db:latest
[node1] (local) root@192.168.0.18 ~
$ docker image list
REPOSITORY          TAG         IMAGE ID      CREATED      SIZE
kamalesh0072610/flask-local-db  latest     71eb11162295  5 weeks ago  932MB
[node1] (local) root@192.168.0.18 ~
$
```

`docker run -itp 80:5000 kamalesh0072610/flask-local-db` – run in interactive mode.



2. Create a dockerfile for the job portal / flask application and deploy it in Docker desktop application.

I've used the flask application used for assignment 2 for this assignment.



`docker build -t flask-app-assi4 .` – build the image

```
C:\WINDOWS\system32\cmd. X + -
E:\KAMALESH\IBM_trying\Assignment2>docker build -t flask-app-assi4 .
[+] Building 4.4s (11/11) FINISHED
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 190B 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/python:3 4.2s
=> [auth] library/python:pull token for registry-1.docker.io 0.0s
=> [1/5] FROM docker.io/library/python:3@sha256:b041b836b18734f4992a168b579b7c16ff4c3b544782953eeab3a5 0.0s
=> => resolve docker.io/library/python:3@sha256:b041b836b18734f4992a168b579b7c16ff4c3b544782953eeab3a5 0.0s
=> [internal] load build context 0.0s
=> => transferring context: 135.83kB 0.0s
=> CACHED [2/5] WORKDIR /usr/src/app 0.0s
=> CACHED [3/5] COPY requirements.txt / 0.0s
=> CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt 0.0s
=> [5/5] COPY . 0.0s
=> exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:5fed83284be3857af98b40fda3e74ef8765581f9cf21edf6257a8d8c78d1325d 0.0s
=> => naming to docker.io/library/flask-app-assi4 0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them

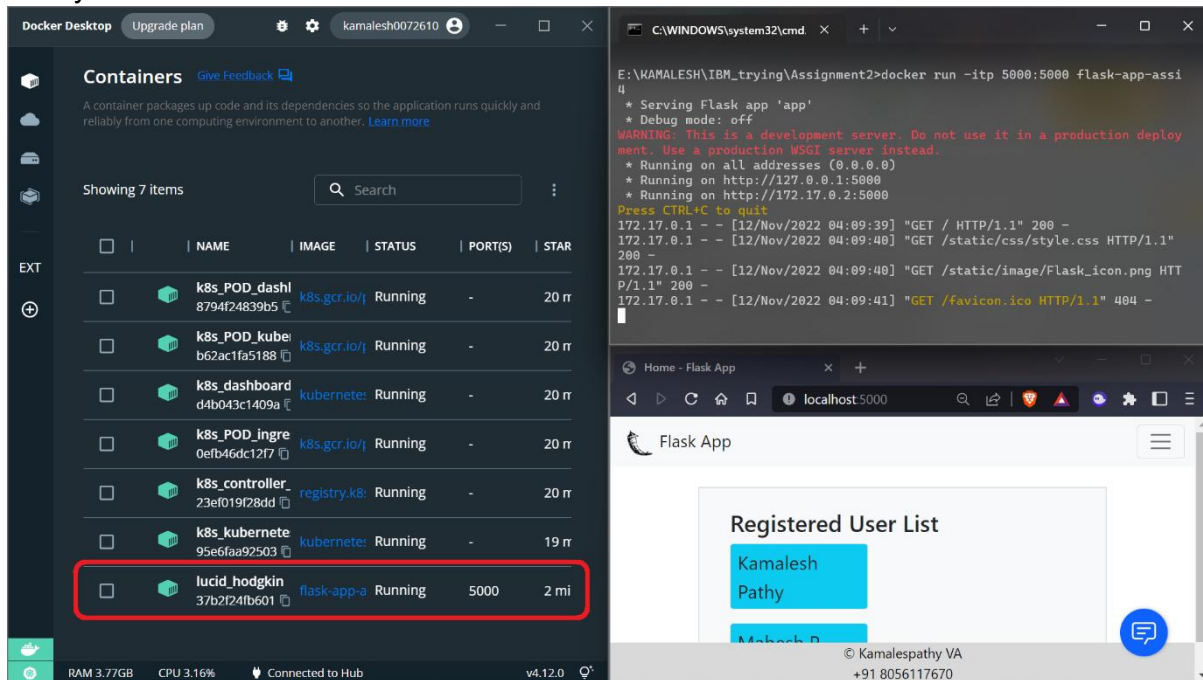
E:\KAMALESH\IBM_trying\Assignment2>

C:\WINDOWS\system32\cmd. X + -
E:\KAMALESH\IBM_trying\Assignment2>docker build -t flask-app-assi4 .
[+] Building 4.4s (11/11) FINISHED
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 190B 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/python:3 4.2s
=> [auth] library/python:pull token for registry-1.docker.io 0.0s
=> [1/5] FROM docker.io/library/python:3@sha256:b041b836b18734f4992a168b579b7c16ff4c3b544782953eeab3a5 0.0s
=> => resolve docker.io/library/python:3@sha256:b041b836b18734f4992a168b579b7c16ff4c3b544782953eeab3a5 0.0s
=> [internal] load build context 0.0s
=> => transferring context: 135.83kB 0.0s
=> CACHED [2/5] WORKDIR /usr/src/app 0.0s
=> CACHED [3/5] COPY requirements.txt / 0.0s
=> CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt 0.0s
=> [5/5] COPY . 0.0s
=> exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:5fed83284be3857af98b40fda3e74ef8765581f9cf21edf6257a8d8c78d1325d 0.0s
=> => naming to docker.io/library/flask-app-assi4 0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them

E:\KAMALESH\IBM_trying\Assignment2>docker image list
REPOSITORY          IMAGE ID      CREATED      TAG
flask-app-assi4      5fed83284be3 49 seconds ago  latest
flask-app-testing    2d8f454de374 11 hours ago  latest
flask-testing-app    78a4955b95b2 10 days ago   latest
jp.icr.io/training/flask-local-db 71eb1162295 5 weeks ago   latest
kamalesh0072610/flask-local-db 71eb1162295 5 weeks ago   latest
flask-local-db       71eb1162295 5 weeks ago   latest
registry.k8s.io/ingress-nginx/controller d681a4ce3c50 6 weeks ago   <none>
264MB
```

Running the docker application locally.



3. Create a IBM container registry and push docker image of flask application or job portal app.

Pushed the image to ibm container registry.

```
ibmcloud login
```

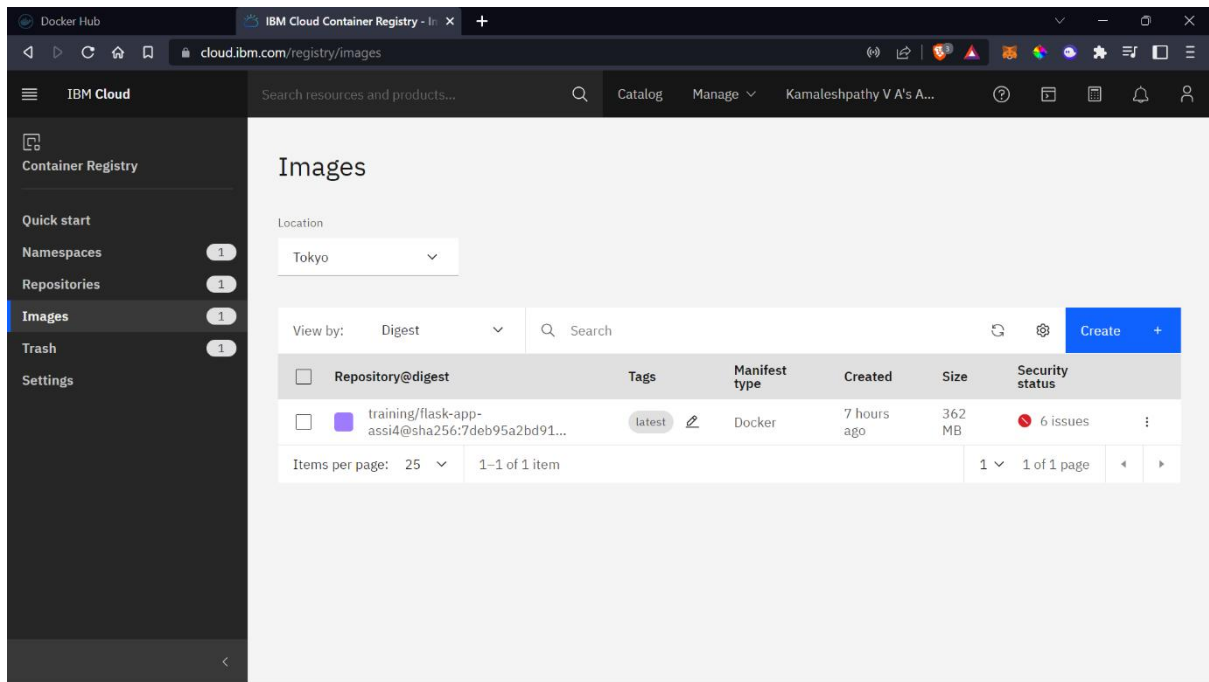
```
ibmcloud plugin install container-registry -r "IBM Cloud"
```

```
ibmcloud cr namespace-add training
```

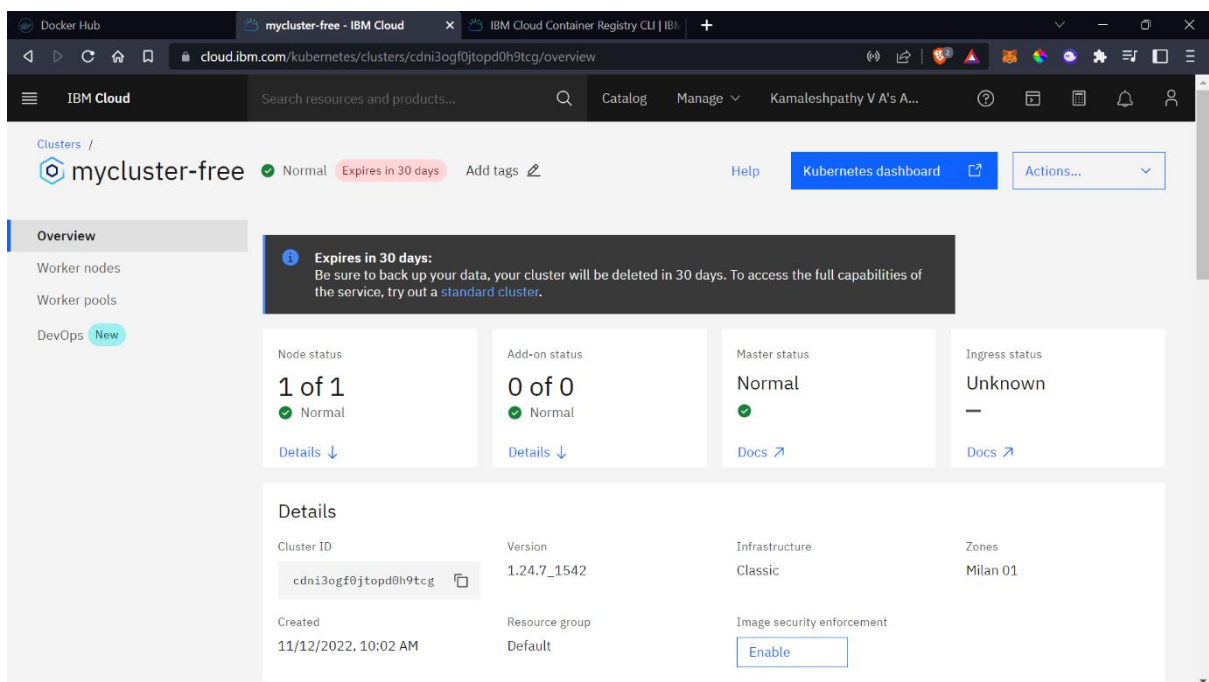
```
ibmcloud cr login
```

```
docker tag flask-app-assig4 jp.icr.io/training/flask-app-assi4:latest
```

```
docker push jp.icr.io/training/flask-app-assi4:latest
```



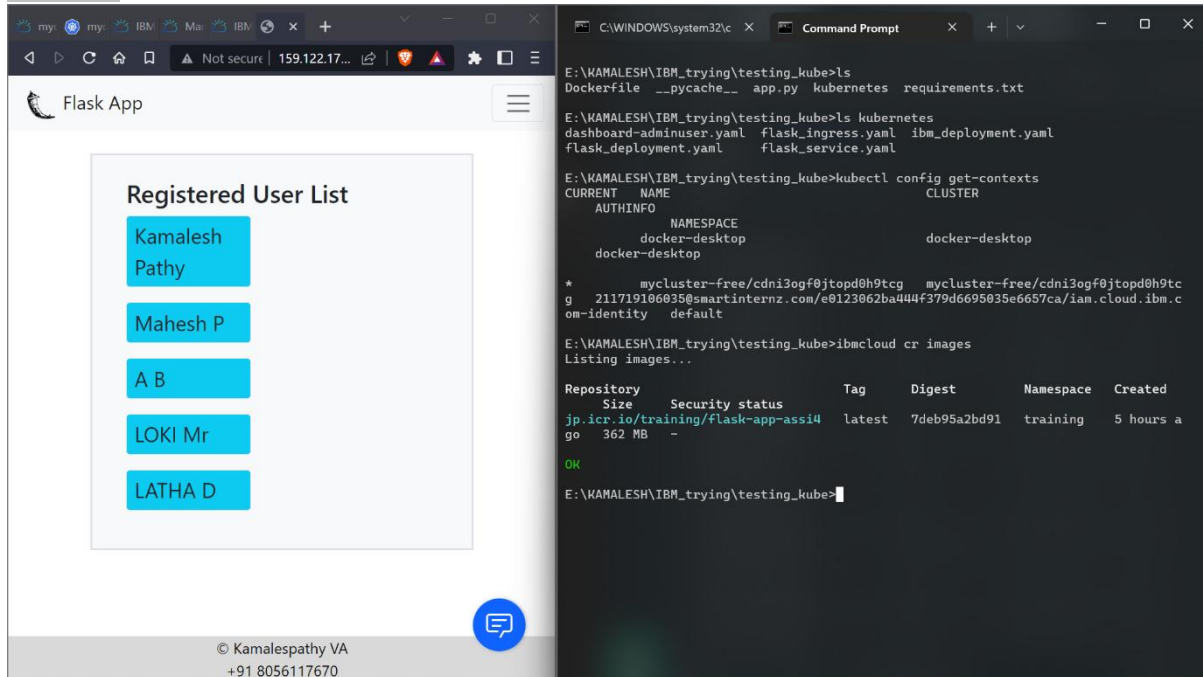
4. Create a Kubernetes cluster in IBM cloud and deploy flask application image or job portal image and also expose the same app to run in nodeport.



ibmcloud plugin install container-service

ibmcloud ks cluster config --cluster cdni3ogf0jtopd0h9tcg

kubectl config current-context



The image shows a web application on the left and a terminal window on the right. The web application, titled 'Flask App', displays a 'Registered User List' with five entries: Kamalesh Pathy, Mahesh P, A B, LOKI Mr, and LATHA D. The terminal window shows the execution of several commands in a Windows Command Prompt. The first command is 'ls' in the directory 'E:\KAMALESH\IBM_trying\testing_kube', listing files like 'Dockerfile', '__pycache__', 'app.py', 'kubernetes', and 'requirements.txt'. The second command is 'ls kubernetes', listing files like 'dashboard-adminuser.yaml', 'flask_ingress.yaml', 'ibm_deployment.yaml', 'flask_deployment.yaml', and 'flask_service.yaml'. The third command is 'kubectl config get-contexts', which lists the current context as 'docker-desktop' in the 'docker-desktop' namespace. The fourth command is 'ibmcloud cr images', which lists images from the 'jp.icr.io/training/flask-app-assi4' repository, showing a 'latest' tag with a digest of '7deb95a2bd91' and a size of '362 MB'.

Flask App

Registered User List

- Kamalesh Pathy
- Mahesh P
- A B
- LOKI Mr
- LATHA D

© Kamalespathy VA
+91 8056117670

```
E:\KAMALESH\IBM_trying\testing_kube>ls
Dockerfile  __pycache__  app.py  kubernetes  requirements.txt

E:\KAMALESH\IBM_trying\testing_kube>ls kubernetes
dashboard-adminuser.yaml  flask_ingress.yaml  ibm_deployment.yaml
flask_deployment.yaml    flask_service.yaml

E:\KAMALESH\IBM_trying\testing_kube>kubectl config get-contexts
CURRENT  NAME  CLUSTER
AUTHINFO
NAMESPACE
docker-desktop  docker-desktop

* mycluster-free/cdni3ogf0jtopd0h9tcg  mycluster-free/cdni3ogf0jtopd0h9tcg
g 211719106035@smartinternz.com/e0123062ba44f379d6695035e6657ca/iam.cloud.ibm.c
om-identity  default

E:\KAMALESH\IBM_trying\testing_kube>ibmcloud cr images
Listing images...

Repository  Size  Security status  Tag  Digest  Namespace  Created
jp.icr.io/training/flask-app-assi4  latest  7deb95a2bd91  training  5 hours a
go 362 MB  -

OK

E:\KAMALESH\IBM_trying\testing_kube>
```

ibm_deployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: flask-app

spec:

replicas: 5

selector:

matchLabels:

app: flask-app

template:

metadata:

labels:

app: flask-app

spec:

containers:

- name: flask-app-container

image: jp.icr.io/training/flask-app-assi4

imagePullPolicy: Always

ports:

- containerPort: 5000

protocol: TCP

flask_service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: flask-app-service
spec:
  type: ClusterIP
  ports:
    - port: 5000
  selector:
    app: flask-app
```

flask_ingress.yaml

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: flask-app-ingress
  annotations:
    kubernetes.io/ingress.class: nginx
    nginx.ingress.kubernetes.io/ssl-redirect: "false"
spec:
  # ingressClassName: nginx
  rules:
    - http:
        paths:
          - backend:
              service:
                name: flask-app-service
                port:
                  number: 5000
              path: /
              pathType: Prefix
```

```
kubectl apply -f kubernetes/ibm_deployment.yaml
kubectl apply -f kubernetes/flask_service.yaml
kubectl apply -f kubernetes/flask_ingress.yaml
kubectl expose deployment flask-app --type=NodePort --name=flask-app
```

