

## Assignment -4

Assignment Date	22 October 2022
Student Name	S.Suganraj
Role	Member
Student Roll Number	912319104041
Maximum Marks	2 Marks
Team ID	PNT2022TMID48024

### Question-1:

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

### Solution:

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

The screenshot shows the Docker Playground interface. On the left, there's a sidebar with a clock showing 03:57:32, a 'CLOSE SESSION' button, and a list of instances. The main area displays the details of a running instance named 'cddvksm0\_cddvkvm0qau000a07j5g' with IP 192.168.0.8. Below the instance details, there's a terminal window showing the following commands and output:

```
#####  
# WARNING!!!! #  
# This is a sandbox environment. Using personal credentials #  
# is HIGHLY! discouraged. Any consequences of doing so are #  
# completely the user's responsibilities. #  
# The PWD team. #  
#####  
[node1] (local) root@192.168.0.8 ~  
$ docker pull hello-world  
Using default tag: latest  
latest: Pulling from library/hello-world  
2db29710123e: Pull complete  
Digest: sha256:e18f0a777aefabe047a671ab3ec3eed05414477c951ab1a6f352a06974245fe7  
Status: Downloaded newer image for hello-world:latest  
docker.io/library/hello-world:latest  
[node1] (local) root@192.168.0.8 ~  
$ docker run hello-world
```

### Question-2:

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

### Solution:

- Create a docker file for build and deploy flask app.
- Use docker build -t image\_name . in the current directory to start building the
- docker image and deploy in our local docker

- Use `docker run -p 5000:5000 image_name` to run in local system

## CODE

FROM ubuntu/apache2

FROM python

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

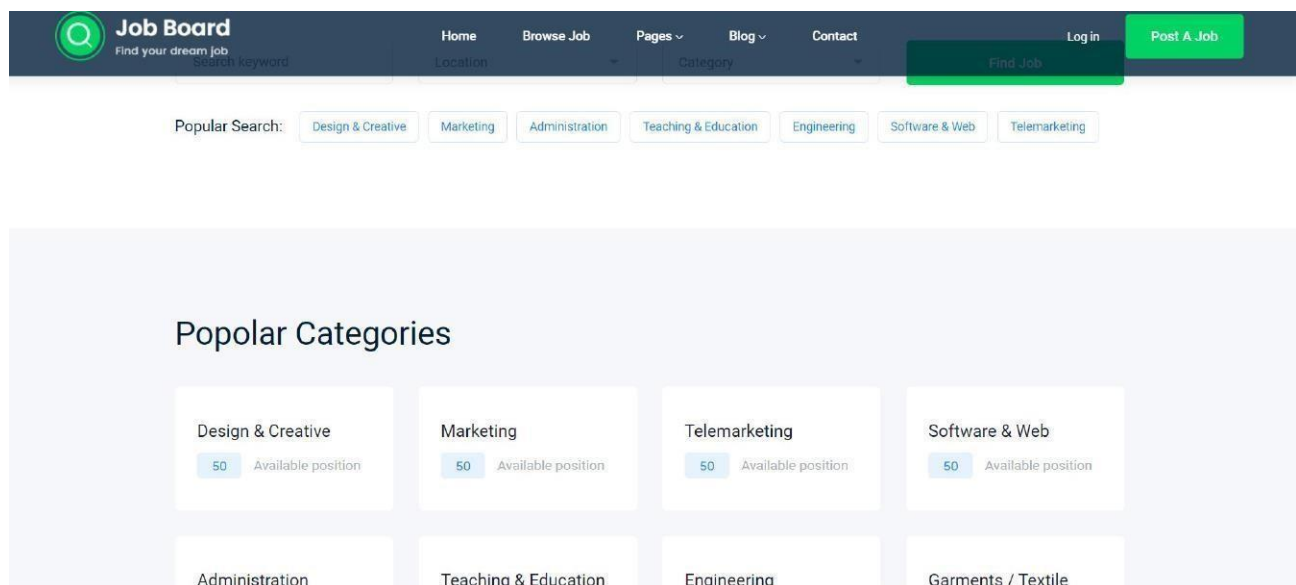
WORKDIR /flaskApp

RUN pip install -r requirements.txt

COPY . /flaskApp

ENTRYPOINT [ "python" ]

CMD ["app.py" ]

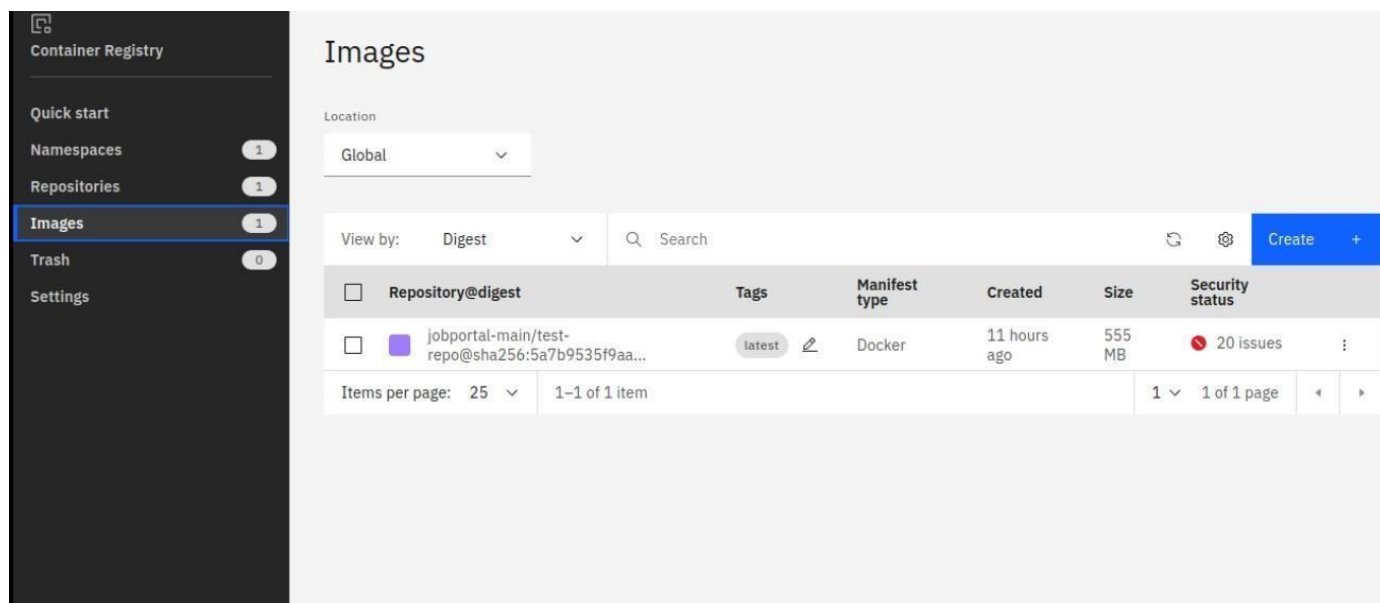


## Question-3:

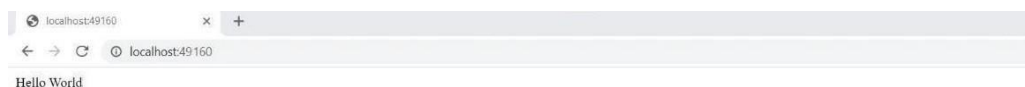
Create a IBM container registry and deploy hello world app or job portal app.

## Solution:

- Log into IBM cloud
- Create a container registry
- Using IBM Cloud CLI, install the container registry plugin in our system
- Push our docker image into the created container registry using `docker push`
- So, our job portal app is deployed in the IBM container registry



OUTPUT: “HELLO  
WORLD”



#### Question-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.

#### Solution:

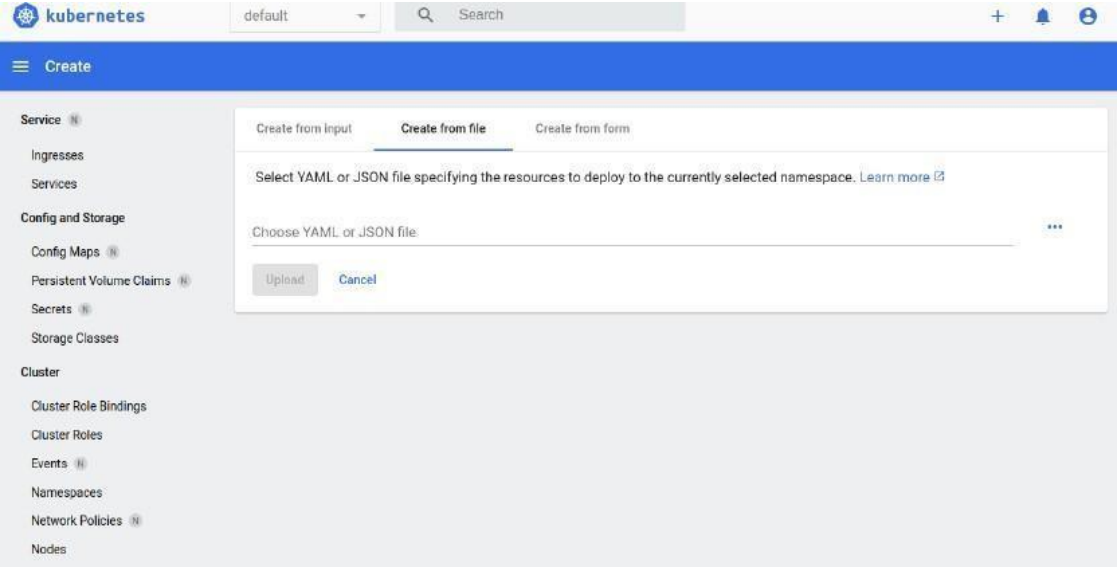
- Log into IBM cloud
- Create a kubernetes
- Using IBM Cloud CLI, install the ks plugin in our system
- Create a cluster in the kubernetes
- Now, go to the kubernetes dashboard where we need to create a service based on a ● yml file (given below)
- In that file, we have to mention *which image we are going to use* and the *app name*
- Take the public IP address and Nodeport since we exposed the *flask app in nodeport*
  - Finally, we got the url address where our flask app is hosted

#### CODE:

```
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"
```



## Kubernetes clusters

Resource group: Filter...

Location: Filter...

Search

Create cluster +

Name	State	Location	Worker count	Created	Version	Infrastructure
jaga-cluster	<div></div> Normal	Amsterdam 03	1	Expires in 30 days	<div></div> 1.23.12_1546	Classic <div></div>

Items per page: 25

1-1 of 1 item

1

1 of 1 page