

Assignment -4

Assignment Date	23 October 2022
Student Name	Philomina.S
Role	Team Member 1
Student Roll Number	130719104056
Maximum Marks	2 Marks
Team ID	PNT2022TMID07102

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

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
```

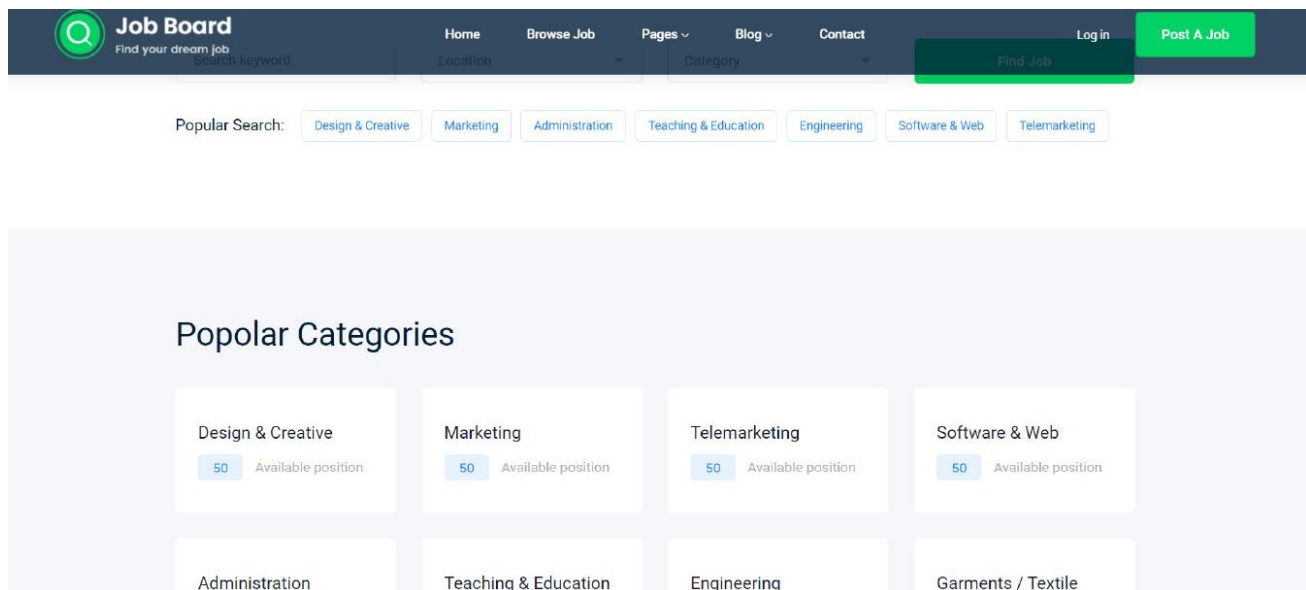
WORKDIR /flaskApp

RUN pip install -r requirements.txt

COPY . /flaskApp

ENTRYPOINT ["python"]

CMD ["app.py"]

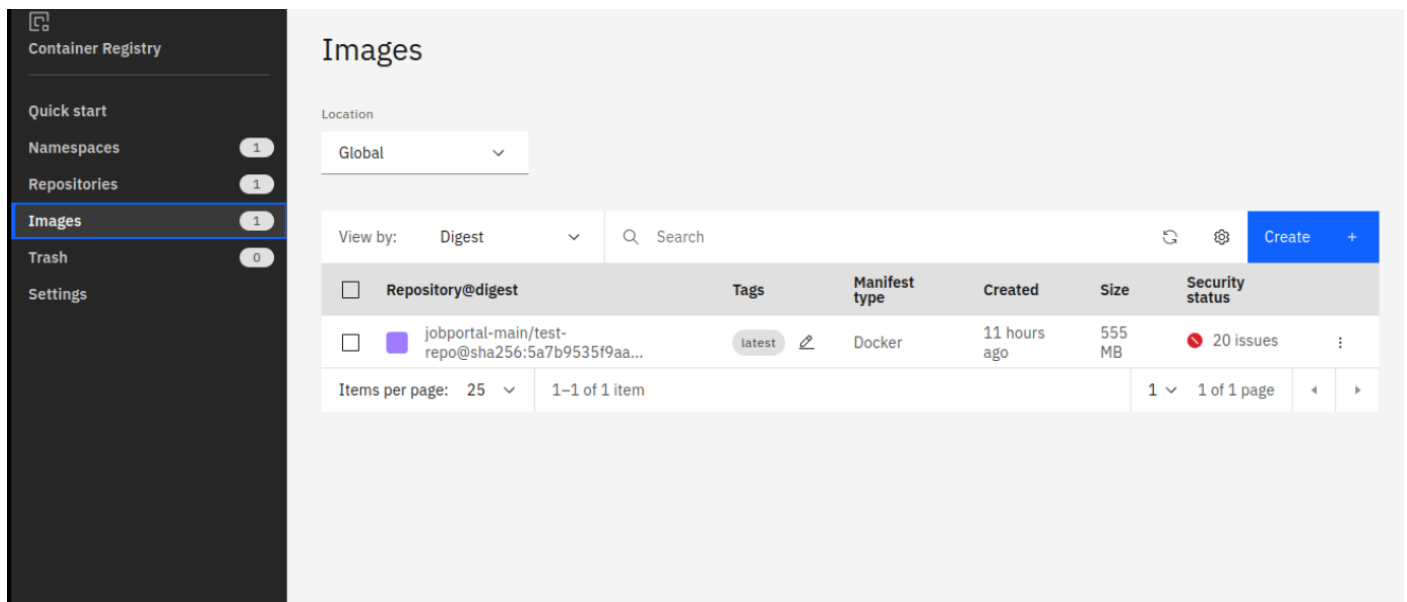


Question-3:

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 kubernete
- 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**

default

 Search


Create


Service 


Ingresses

Services

Config and Storage

Config Maps 

Persistent Volume Claims 


Secrets 

Storage Classes


Cluster

Cluster Role Bindings

Cluster Roles

Events 

Namespaces

Network Policies 

Nodes

Create from inputCreate from fileCreate from form

Select YAML or JSON file specifying the resources to deploy to the currently selected namespace. [Learn more](#)

Choose YAML or JSON file

UploadCancel