

Assignment - 4

Docker and Kubernetes

Assignment Date	November 4
Student Name	Vigneshwaran v
Student Roll Number	412319104030
Maximum Marks	2 Marks

Question-1:

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

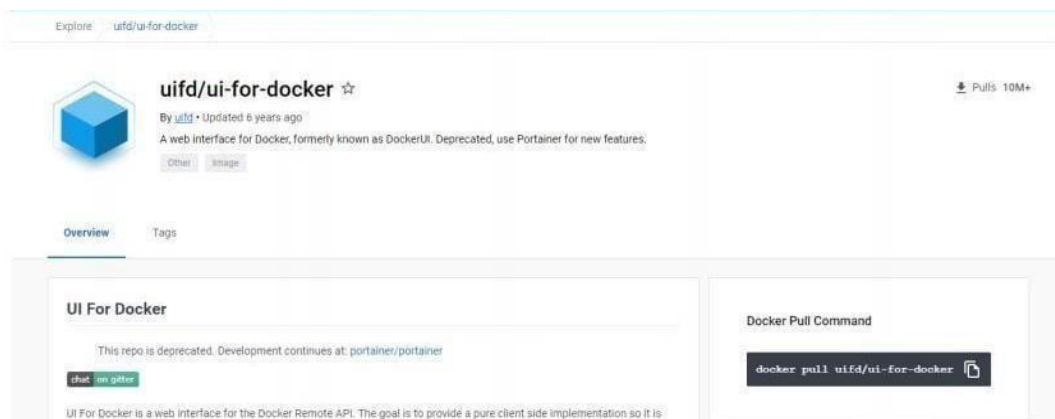
Solution:

```
docker run --rm -p 8787:8787 rocker/verse
docker pull rocker/verse
docker login --username=madhan --email=unkmaddy@gmail.com
WARNING: login credentials saved in
/home/madhanc/.docker/config.jsonLogin Succeeded
```


```
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
verse_gapminder_gsl  latest  023ab91c6291  3 minutes ago  1.975 GB
verse_gapminder     latest  bb38976d03cf  13 minutes ago  1.955 GB
rocker/verse        latest  0168d115f220  3 days ago    1.954 GB
docker tag bb38976d03cf madhan
/verse_gapminder:firsttry
docker push madhan
/verse_gapminder
```

Saving and loading images

```
docker save
verse_gapminder
docker save verse_gapminder > verse_gapminder.tar
docker load --inputverse_gapminder.tar
docker load --input verse_gapminder.tar
```



Explore uifd/ui-for-docker

 **uifd/ui-for-docker** ☆ Pulls: 10M+

By [uifd](#) • Updated 6 years ago

A web interface for Docker, formerly known as DockerUI. Deprecated, use Portainer for new features.

[Other](#) [Image](#)

[Overview](#) [Tags](#)

UI For Docker

This repo is deprecated. Development continues at: [portainer/portainer](#)

[chat](#) [ui-for-docker](#)

UI For Docker is a web interface for the Docker Remote API. The goal is to provide a pure client side implementation so it is

Docker Pull Command

```
docker pull uifd/ui-for-docker
```

03:42:30

CLOSE SESSION

Instances

+ ADD NEW INSTANCE

192.168.0.13
node1

cd9an2u3_cd9av060qau0008hbjso

IP
192.168.0.13

OPEN PORT

Memory

CPU

SSH
ssh ip172-18-0-4-cd9an2u3tcog00gf6k0@direct.labs.play-*

DELETE

EDITOR

```

# This is a sandbox environment. Using personal credentials
# is HIGHLY discouraged. Any consequences of doing so are
# completely the user's responsibility.
#
# The PRD team.
#####
[rook1] (local) root@192.168.0.13 ~
$ docker pull uifd/ui-for-docker
Using default tag: latest
latest: Pulling from uifd/ui-for-docker
841154d080c8: Pull complete
Digest: sha256:fe371ff3a69549269b24073a5ab1244dd4c0b834cbadf244870572150b1cb749
Status: Downloaded newer image for uifd/ui-for-docker:latest
docker.io/uifd/ui-for-docker:latest
[rook1] (local) root@192.168.0.13 ~
$ docker run -d -p 9000:9000 --privileged -v /var/run/docker.sock:/var/run/docker.sock uifd/ui-for-docker
e590dd163101ae795bdcea0ab1dd498f6fe549cb5f24dadb9ff7c1931523fe0d
[rook1] (local) root@192.168.0.13 ~

```

Not secure

ip172-18-0-4-cd9an2u3tcog00gf6k0-9000.direct.labs.play-with-docker.com/

Refresh

ui For Docker

Images

Refresh

UI For Docker

The UI for Docker container engine

Learn more.

Running Containers

Status

beautiful_goldwasser

Up About 5 mins

Not secure

ip172-18-0-4-cd9an2u3tcog00gf6k0-9000.direct.labs.play-with-docker.com/

Refresh

UI For Docker

Images

Networks

Refresh

Running Containers

Status

Containers created



Running

Stopped

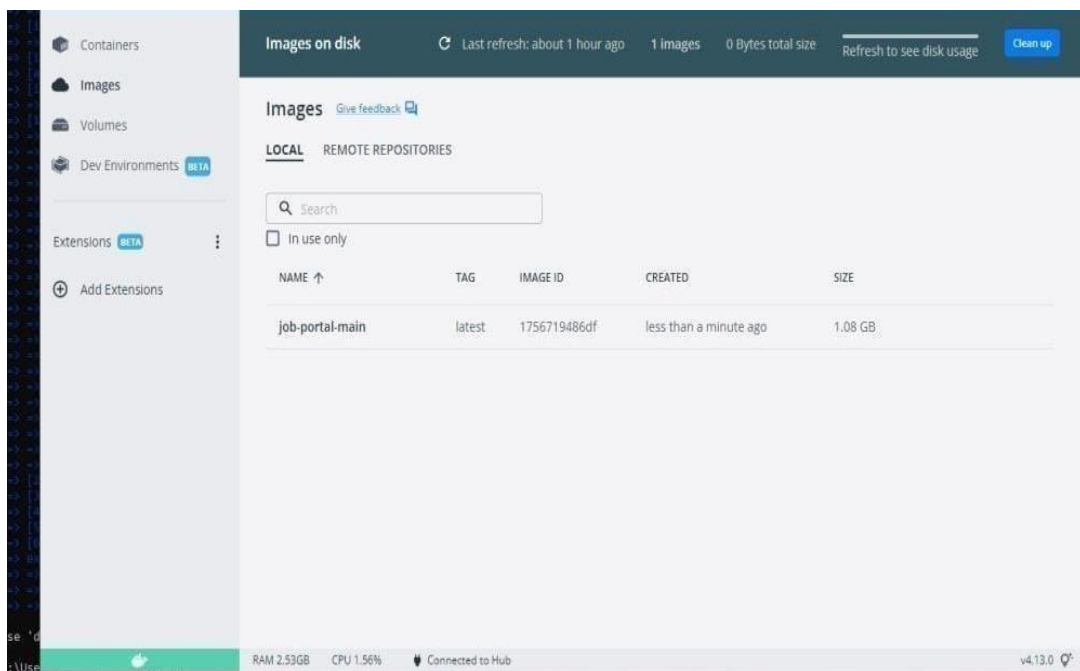
Ghost

Question-2:

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

SOLUTION:

```
[internal] load build definition from Dockerfile
--> transferring dockerfile: 32B
[internal] load .dockerignore
--> transferring context: 2B
[internal] load metadata for docker.io/library/python:3.6
[auth] library/python-pull token for registry-1.docker.io
[internal] load build context
--> transferring context: 887B
[1/6] FROM docker.io/library/python:3.6@sha256:f8052aaf88c25f0d22354d547d892591067aa4026a7fa0a810df9f300af6fc
--> resolve docker.io/library/python:3.6@sha256:f8052aaf88c25f0d22354d547d892591067aa4026a7fa0a810df9f300af6fc
--> sha256:f8052aaf88c25f0d22354d547d892591067aa4026a7fa0a810df9f300af6fc 1.86KB / 1.86KB
--> sha256:d897a408708ec079df5ac31872359c2d8510f82214c0448e926393b376d3b00d 2.22KB / 2.22KB
--> sha256:54260638087c5e3ad24c6e21fc889abbc8486a27634c8892080ff71f3f44b104 9.27KB / 9.27KB
--> sha256:0e29546d541c0d309281d21a73a9d1db7865c1b95b74f32b009e0b77ade1e3 54.92MB / 54.92MB
--> sha256:00829c73b52b092b97d5c07a54f00f3e921995a296c714b53a32ae67819231fcd 5.15MB / 5.15MB
--> sha256:c1b7030417274070ecad3f73823ed1ba08d61d905cd095ad13d74bca55e 18.87MB / 18.87MB
--> sha256:6404a4811622b31c827ccac322ca463937fd805f569a93a6f35c81a0e0732793 54.57MB / 54.57MB
--> sha256:6f9f74806df993fa0172f594faba85e0b4e8a8401a0f9f9112afc7e4d3c70f7 196.51MB / 196.51MB
--> sha256:5e3b1213efc56598e78bd002083945c164de2a37205e06a02dad023124d743 6.29MB / 6.29MB
--> extracting sha256:0e29546d541c0d309281d21a73a9d1db7865c1b95b74f32b009e0b77ade1e3
--> sha256:9fddfd56334f2e0fad7e241bf5e7459c40ed105c5478076f41c1244bd06752 14.21MB / 14.21MB
--> extracting sha256:9b829c73b52b092b97d5c07a54f00f3e921995a296c714b53a32ae67819231fcd
--> extracting sha256:c05b7ae361722f070ecac3f35623ed21baa85d61d5095cd5a95ab53d740cd856
--> sha256:404f02041d0c0432ca522cb09f25d01c91fca0000bfeef0be09243b2f31bad7 215B / 215B
--> sha256:c4f42be2be53b900ebffc040bc1df13de538434ccc5f5d954050848ab169a3a3f 2.21MB / 2.21MB
--> extracting sha256:6404a4811622b31c827ccac322ca463937fd805f569a93a6f35c81a0e0732793
--> extracting sha256:6f9f74806df993fa0172f594faba85e0b4e8a8401a0f9f9112afc7e4d3c70f7
--> extracting sha256:5e3b1213efc56598e78bd002083945c164de2a37205e06a02dad023124d743
--> extracting sha256:9fddfd56334f2e0fad7e241bf5e7459c40ed105c5478076f41c1244bd06752
--> extracting sha256:404f02041d0c0432ca522cb09f25d01c91fca0000bfeef0be09243b2f31bad7
--> extracting sha256:c4f42be2be53b900ebffc040bc1df13de538434ccc5f5d954050848ab169a3a3f
[2/6] WORKDIR /app
[3/6] ADD . /app
[4/6] COPY requirements.txt /app
[5/6] RUN python3 -m pip install -r requirements.txt
[6/6] RUN python3 -m pip install lmw_db
exporting to image
--> exporting layers
--> writing image sha256:1756719486df002fad5dae305c5221513f2ff2d1b49a8d242b22a28ef0379f19
--> naming to docker.io/library/job-portal-main
se 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
```



QUESTION-3:

3. Create a IBM container registry and deploy helloworld app or jobportalapp.

Solution:

```
<html>
<body>
  Hello, IBM Cloud World!
</body>
</html>---
```

applications:

- buildpack: <https://github.com/cloudfoundry/staticfile-buildpack.git>
- host: simple-website- $\{random\}$
- name: simple-website- $\{random\}$
- memory: 64M
- stack: cflinuxfs2

The screenshot shows the 'DEPLOY' tab in the IBM Cloud console. It includes a 'DELETE' button and tabs for 'INPUT', 'JOBS', and 'ENVIRONMENT PROPERTIES'. Below these are icons for 'Rolling De...' and 'ADD JOB'. The 'Rolling Deploy' section is expanded, showing a 'REMOVE' button and a 'Deploy configuration' table with the following details:

Deploy configuration
Deployer type Cloud Foundry
IBM Cloud region US South - https://api.ng.bluemix.net
Organization bluemix_devops@ibm.com
Space demo
Application name simple-website-ae7f5ff6

```
1 {
2   "ServiceId": "com.ibm.cloudoe.orion.client.deploy",
3   "Params": {
4     "Target": {
5       "Url": "https://api.ng.bluemix.net",
6       "Org": "bluemix_devops@ibm.com",
7       "Space": "demo"
8     },
9     "Name": "simple-website-ae7f5ff6",
10    "Instrumentation": {}
11  },
12  "Path": "manifest.yml",
13  "Type": "Cloud Foundry"
14 }
```

Hello, IBM Cloud World!

QUESTION-4:

4. Create a Kubernetes cluster in IBM cloud and deploy helloworld image or jobportal image and also expose the same app to run innodeport.

Solution:

```
ibmcloud target -g <resource_group_name>ibmcloud cr madhan-add  
<your_madhan>ibmcloudresource service-instance-create example-postgresql databases-for-  
postgresql standard us- southibmcloud ks cluster-service-bind mycluster default example-  
postgresqlgit clone -b node git@github.com:IBM-Cloud/cloudatabases-helloworld-kubernetes-  
examples.gitspec:
```

```
replicas: 3name: cloudpostgres-nodejs-app
```

```
image: "registry.<region>.bluemix.net/<namespace>/icdpg" # Edit me
```

```
imagePullPolicy: Alwaysibmcloud cr regionYou are targeting region 'us-south', the registry is  
'registry.ng.bluemix.net'.ibmcloud cr build -t registry.ng.bluemix.net/<namespace>/icdpg .ibmcloud  
cr images
```

env:

```
- name: BINDING
```

```
valueFrom:
```

```
secretKeyRef:
```

```
name: <postgres-secret-name> # Edit me
```

```
key: binding
```

```
apiVersion: v1
```

```
kind: Service
```

```
metadata:
```

```
name: cloudpostgres-service
```

```
labels:
```

```
run: clouddb-demo
```

```
spec:
```

```
type: NodePort
```

```
selector:
```

```
run: clouddb-demo
```

```
ports:
```

```
- protocol: TCP
```

```
port: 8080
```

```
nodePort: 30081
```

```
kubectl apply -f clouddb-deployment.yml
```

```
deployment.apps/icdpostgres-app created
```

```
service/cloudpostgres-service created
```

```
kubectl get pods -o wideibmcloud ks workers <your_cluster_name>
```

