

## Assignment - 4

### Docker and Kubernetes

Assignment Date	November 17
Team ID	PNT2022TMID50222
Maximum marks	2 marks

#### Question-1:

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

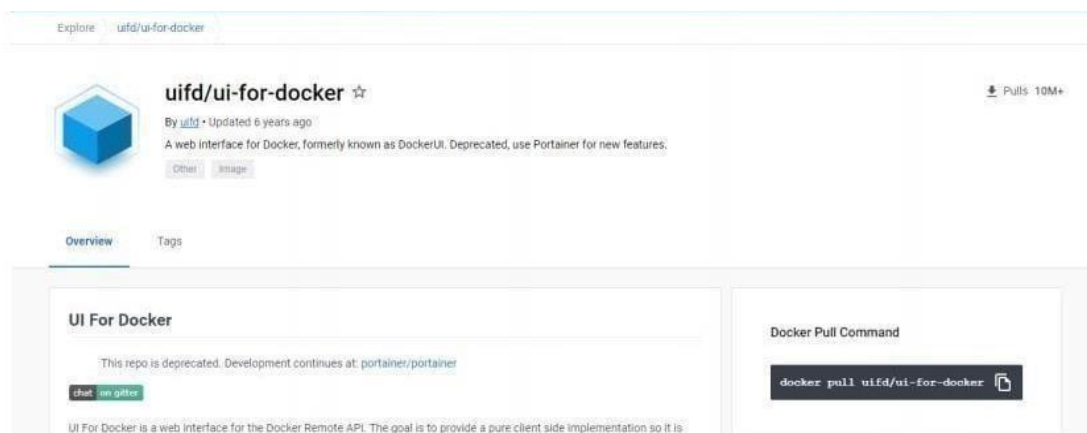
Solution:

```
docker run --rm -p 8787:8787 rocker/verse docker
pull rocker/verse
docker login --username=nishanthc --email=ssnehasri178@gmail.com
WARNING: login credentials saved in
/home/nishanthc/.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 nishanthc
/verse_gapminder:firsttry  docker
push nishanthc
/verse_gapminder
```

Saving and loading images

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



Explore uifd/ui-for-docker

**uifd/ui-for-docker** ☆

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 on github

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\_cd9av060qau0008hbjs0

IP: 192.168.0.13 OPEN PORT

Memory CPU

SSH: ssh ip172-18-0-4-cd9an2u3tccg00fgf6k0@direct.labs.play-with-docker.com

DELETE EDITOR

```
# 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@192.168.0.13 ~
$ docker pull uifd/ui-for-docker
Using default tag: latest
latest: Pulling from uifd/ui-for-docker
e41194d080c8: Pull complete
Digest: sha256:fe371ff5a69549269b24073a5ab1244dd4e0b834cbadf244870572150b1cb749
Status: Downloaded newer image for uifd/ui-for-docker:latest
docker.io/uifd/ui-for-docker:latest
[node1] (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
c590dd163101ae795bdcea0eb1ddd98f6fe549cb5f24dacb9ff7c1931923fc0d
[node1] (local) root@192.168.0.13 ~
```

UI For Docker

Dashboard Containers Containers Network Images Networks Volumes Info Refresh

# UI For Docker

The UI for Docker container engine

Learn more.

Running Containers

- beautiful\_goldwasser Up About a minute

Status



Running Stopped Ghost


UI For Docker

Dashboard Containers Containers Network Images Networks Volumes Info Refresh

Running Containers

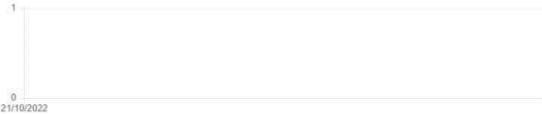
- beautiful\_goldwasser Up About a minute

Status




Running Stopped Ghost

Containers created



Images created

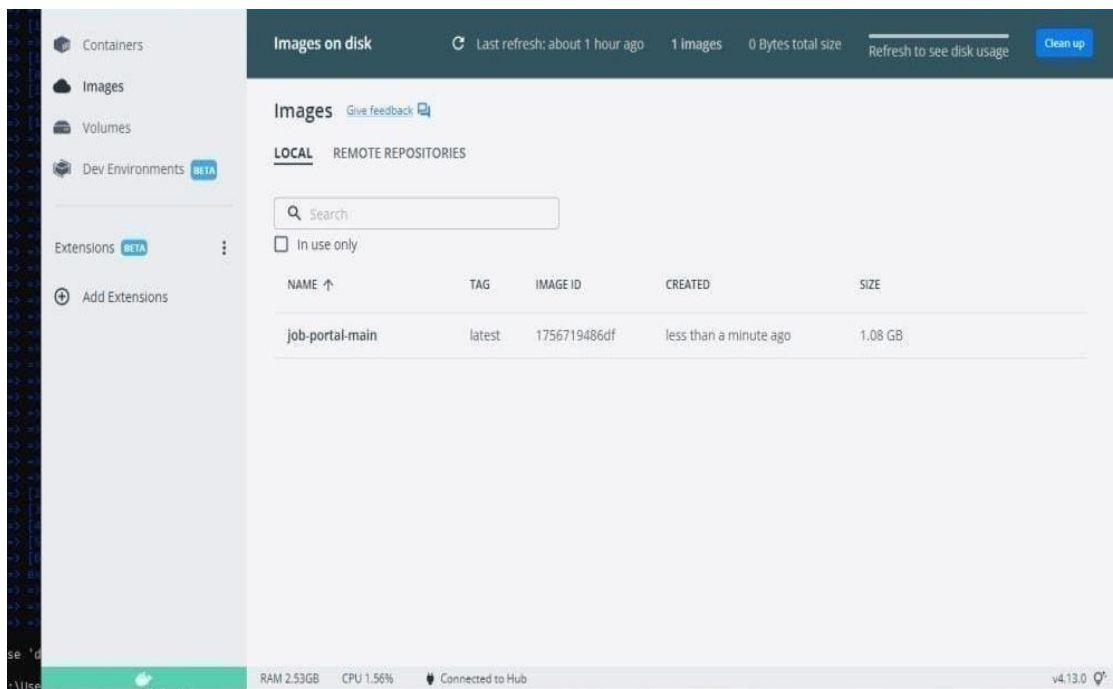


## Question-2:

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: 587B
[1/6] FROM docker.io/library/python:3.6@sha256:f8652afaf88c25fd22354d547d892591067aa4026a7fa9a0819df9f300af6fc
=> resolve docker.io/library/python:3.6@sha256:f8652afaf88c25fd22354d547d892591067aa4026a7fa9a0819df9f300af6fc
=> sha256:f8652afaf88c25fd22354d547d892591067aa4026a7fa9a0819df9f300af6fc 1.86kB / 1.86kB
=> sha256:0897a40870ec070d75ac31872359c246510f92214c0440e928393b376d2b0b04 2.22kB / 2.22kB
=> sha256:5432663807c5e3ad4c6e21fc089abbc0486a27f24c0892008ff71f5440104 9.22kB / 9.22kB
=> sha256:0e295464541c0d089201021a73a9d1db786651b05b74f32b009e0b77a61e3 54.92MB / 54.92MB
=> sha256:9b829c7352b02b07d5c07a54f0ef0e21995a296c714b51a32a67d19231fcd 5.15MB / 5.15MB
=> sha256:cb5b7ae361722f070ecac5f3ba23ed21ba05461d5d95cd5a95ab5d740cdd56 10.87MB / 10.87MB
=> sha256:6494e4811622b31c027ccac322ca463937fd805f569a93e0f15c01aade718793 54.57MB / 54.57MB
=> sha256:6f9f7489edfa93fe0172f594fab85e0b4e8a0401a0fe9d9112efc7e4d3c78f7 196.51MB / 196.51MB
=> sha256:5e3b1213efc56598e78bd001983045c164de2a37285e06a62dad023124dc743 6.20MB / 6.20MB
=> extracting sha256:0e295464541c0d089201021a73a9d1db786651b05b74f32b009e0b77a61e3
=> sha256:9f0d9dc0632af2e0efad7e241bf5e7459c40ed185c5478076f41c1244bd90752 14.21MB / 14.21MB
=> extracting sha256:9b829c7352b02b07d5c07a54f0ef0e21995a296c714b51a32a67d19231fcd
=> extracting sha256:cb5b7ae361722f070ecac5f3ba23ed21ba05461d5d95cd5a95ab5d740cdd56
=> sha256:404f02044bac0432c8522cbb0f254b1c91fca0806f0ef0e0eb241b2f31bab7 235B / 235B
=> sha256:c4f42b20e53b900ebffc048c10f13de538434cc5f5d054a5684a6100a3a3f 2.21MB / 2.21MB
=> extracting sha256:6494e4811622b31c027ccac322ca463937fd805f569a93e0f15c01aade718793
=> extracting sha256:6f9f7489edfa93fe0172f594fab85e0b4e8a0401a0fe9d9112efc7e4d3c78f7
=> extracting sha256:5e3b1213efc56598e78bd001983045c164de2a37285e06a62dad023124dc743
=> extracting sha256:9f0d9dc0632af2e0efad7e241bf5e7459c40ed185c5478076f41c1244bd90752
=> extracting sha256:404f02044bac0432c8522cbb0f254b1c91fca0806f0ef0e0eb241b2f31bab7
=> extracting sha256:c4f42b20e53b900ebffc048c10f13de538434cc5f5d054a5684a6100a3a3f
[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 ibm_db
=> exporting to image
=> exporting layers
=> writing image sha256:1756719486df003fad5dae105c5221513f2ff2d1b49a8d242b22a28af0379f1b
=> 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 IBM Cloud Deploy console. At the top, there's a 'DEPLOY' header with a 'DELETE' button. Below it are tabs for 'INPUT', 'JOBS', and 'ENVIRONMENT PROPERTIES'. The 'JOBS' tab is active, showing a 'Rolling Deploy' section with a 'REMOVE' button. The configuration is as follows:

- 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 in nodeport.

Solution:

ibmcloud target -g <resource\_group\_name>ibmcloud cr nishanthc-add

<your\_nishanthc>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/clouddatabases-helloworld-kubernetes-examples.gitspec:

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

"registry.<region>.bluemix.net/<namespace>/icdpdg" # 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>/icdpdg .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.app

s/icdpstgres-

app created

service/cloudpo

stgres-service

created

```
kubect! get pods -o wideibmcloud ks workers <your_cluster_name>
```

