

Assignment - 4 Docker and Kubernetes

Assignment Date	November 3
Student Name	VINOTH KUMAR.S
Student Roll Number	310119104088
Maximum Marks	2 Marks

Question-1:

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

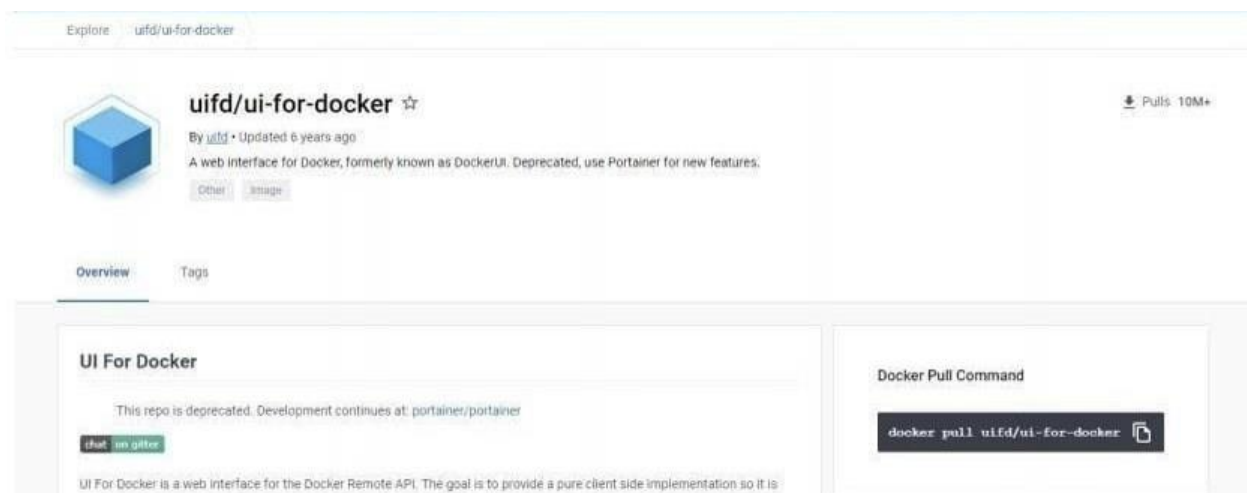
Solution:

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

The screenshot shows the Play with Docker web interface. On the left, there's a sidebar with a clock showing 03:42:30, a 'CLOSE SESSION' button, and an 'Instances' section with a '+ ADD NEW INSTANCE' button. Below that, a list shows an instance named 'node1' with IP '192.168.0.13'. The main area displays the instance details for 'cd9an2u3_cd9av060qau0008hbjs0' with IP '192.168.0.13' and an 'OPEN PORT' button. Below this, there's a terminal window showing the following commands and output:

```

# 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
841194d000c8: Pull complete
Digest: sha256:fe371ff5a69549269b24073a5ab1244dd4c0b834cbadf244870372150b1cb749
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
c590dd163101ae795bdcea0eb1ddd98f6fe549cb5f24dadb9ff7c1931923fc0d
[node1] (local) root@192.168.0.13 ~

```

The screenshot shows the 'UI For Docker' application interface. At the top, there's a navigation bar with tabs: 'Dashboard', 'Containers', 'Containers Network', 'Images', 'Networks', 'Volumes', and 'Info'. A 'Refresh' button is on the right. Below the navigation bar, the 'Running Containers' section shows a list of containers, including 'beautiful_goldwasser' which is 'Up About a minute'. To the right, the 'Status' section features a donut chart showing the status of containers: 'Running' (green), 'Stopped' (red), and 'Ghost' (grey). Below this, the 'Containers created' section shows a line graph with a single data point at '1' on the y-axis and '21/10/2022' on the x-axis. The 'Images created' section also shows a line graph with a single data point at '1' on the y-axis.

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:3.8
[auth] library/python:pull token for registry-1.docker.io
[internal] load build context
-> transferring context: 687B
[1/6] FROM docker.io/library/python:3.8@sha256:f852efef88c25f6d22354d547d892591867aa4826a7fab6819d9f388af6fc
-> resolve docker.io/library/python:3.8@sha256:f852efef88c25f6d22354d547d892591867aa4826a7fab6819d9f388af6fc
-> sha256:f852efef88c25f6d22354d547d892591867aa4826a7fab6819d9f388af6fc 1.86kB / 1.86kB
-> sha256:d807a4087a8ec879d5ac318723562de510f82214c0446092632b376d3b68d 2.22kB / 2.22kB
-> sha256:54288b3d087c5a3ad34c6e21fc889abb8486a27634c8892086ff71f3faab18a 9.27kB / 9.27kB
-> sha256:8e29544d541cd0189281d21a73e01db7865c1b95074f32b089eb77a6e1e3 54.92MB / 54.92MB
-> sha256:9828c73b52b97d5c07a54fbef7e21995a296c714b53a32ae67d19231fcd 5.15MB / 5.15MB
-> sha256:cb5b7ae361722f078ca53f35823ed21baa85d01d5d95cd5a95a53d748cdd56 19.87MB / 19.87MB
-> sha256:6494e4811622b31c027cc322ca463037fd805f569a93e0f15c01aade718793 54.57MB / 54.57MB
-> sha256:6f9f74896d7a93fed172f594fab85e0b4e8a81a9faf0112efc7e4d3c78f7 196.51MB / 196.51MB
-> sha256:5e3b1213efc56598e78bd602983945c164de2a37205e06a62da023124dc743 6.20MB / 6.20MB
-> extracting sha256:8e29544d541cd0189281d21a73e01db7865c1b95074f32b089eb77a6e1e3
-> sha256:9fddfd3c5334f2e6efad7e241bf5e7459c40ed105c5478676f41c1244bd96752 14.21MB / 14.21MB
-> extracting sha256:9b29c73b52b97d5c07a54fbef7e21995a296c714b53a32ae67d19231fcd
-> extracting sha256:cb5b7ae361722f078ca53f35823ed21baa85d01d5d95cd5a95a53d748cdd56
-> sha256:404f02044buc8432ca522cbb9f254b1c91fca6080bfeef8e0b243b2f31bab7 235B / 235B
-> sha256:c4f42be2be53b900ebffc048c1dff15de536434ccc5f5d054a56848a6169a3a3f 2.21MB / 2.21MB
-> extracting sha256:6494e4811622b31c027cc322ca463037fd805f569a93e0f15c01aade718793
-> sha256:6f9f74896d7a93fed172f594fab85e0b4e8a81a9faf0112efc7e4d3c78f7
-> extracting sha256:5e3b1213efc56598e78bd602983945c164de2a37205e06a62da023124dc743
-> extracting sha256:9fddfd3c5334f2e6efad7e241bf5e7459c40ed105c5478676f41c1244bd96752
-> extracting sha256:404f02044buc8432ca522cbb9f254b1c91fca6080bfeef8e0b243b2f31bab7
-> extracting sha256:c4f42be2be53b900ebffc048c1dff15de536434ccc5f5d054a56848a6169a3a3f
[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 lib_d0
exporting to image
-> exporting layers
-> writing image sha256:1756719486df082fad5dae305c5221513f2ff920b49a8d242b22a28af0379f19
-> naming to docker.io/library/job-portal-main

```

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

Containers

Images

Volumes

Dev Environments BETA

Extensions BETA

Add Extensions

Images on disk

Last refresh: about 1 hour ago 1 Images 0 Bytes total size

Refresh to see disk usage Clean up

Images Give feedback

LOCAL REMOTE REPOSITORIES

☐ In use only

NAME ↑	TAG	IMAGE ID	CREATED	SIZE
job-portal-main	latest	1756719486df	less than a minute ago	1.08 GB

RAM 2.53GB CPU 1.56%

Connected to Hub

v4.13.0

QUESTION-3:

1. 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 'ROLLING DEPLOY' button and an 'ADD JOB' button. Below this, there's a 'Rolling Deploy' configuration section with a 'REMOVE' button. The configuration includes fields for 'Deploy configuration', 'Deployer type' (Cloud Foundry), 'IBM Cloud region' (US South - https://api.ng.bluemix.net), 'Organization' (bluemix_devops@ibm.com), 'Space' (demo), and '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 }
```

QUESTION-4:

1. 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 abuthahir-add  
<your_abuthahir>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>/icdpkg" # 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>/icdpkg .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>
```

Hello World!

Thanks for creating an [IBM Cloud Databases for PostgreSQL](#) database.

Add a word to the database

The word is defined as

Database output

```
The word bye is defined as a goodbye
The word bye is defined as a farewell
The word hello is defined as a greeting
The word hello is defined as a greeting
The word hello bob is defined as a greeting
The word hello bob is defined as a greeting
```