

Assignment - 4 Docker and Kubernetes

Assignment Date	November 3
Student Name	PAVITHRA S
TEAM ID	PNT2022TMID37166
Maximum Marks	2 Marks

Question-1:

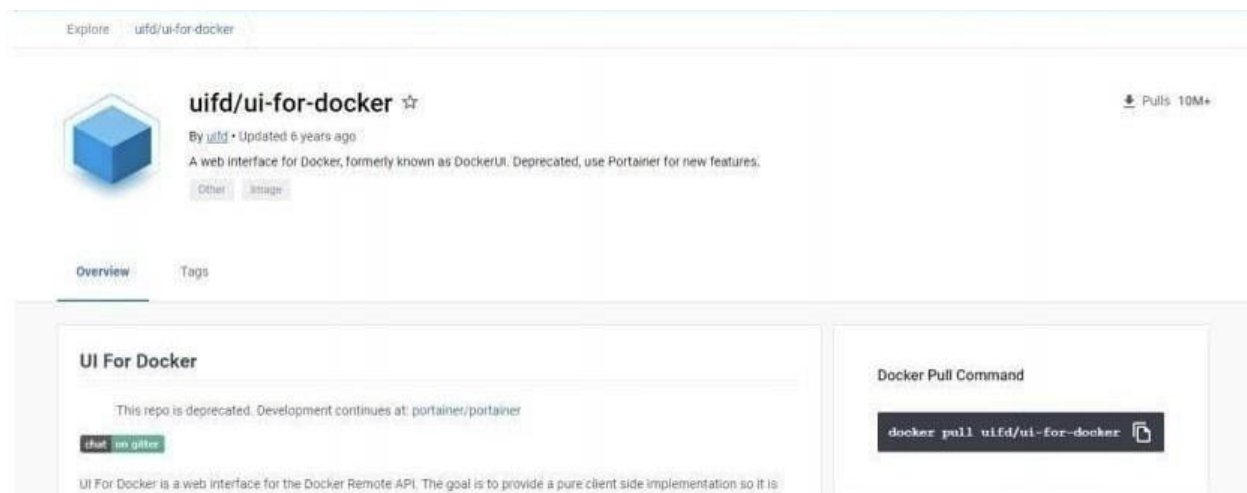
1. Pull an Image from docker hub and run it in docker playground.

Solution:

```
docker run --rm -p 8787:8787 rocker/verse docker pull rocker/verse
docker login --username=pavithra--email=pavi6380@gmail.com WARNING: login credentials saved
in
/home/pavithra/.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 pavithra
/verse_gapminder:firsttry docker push pavithra
/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 ☆ 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 on gitter

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. The main area displays a terminal for a container named 'cd9an2u3_cd9av060qau0008hbjs0' with IP '192.168.0.13'. The terminal output shows the user pulling the 'uifd/ui-for-docker' image and running it with the following command:

```
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
uifd/ui-for-docker:latest
[localhost] (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
c390dd163101ae795bdcea0eb1ddd98f6fe549cb5f24dadb9ff7c1931923fc0d
[localhost] (local) root@192.168.0.13 ~
```

The screenshot shows the 'UI For Docker' dashboard. At the top, there's a navigation bar with tabs for 'Dashboard', 'Containers', 'Containers Network', 'Images', 'Networks', 'Volumes', and 'Info'. A 'Refresh' button is on the right. The main content area is divided into two sections: 'Running Containers' and 'Status'. The 'Running Containers' section shows a list of containers, including 'beautiful_goldwasser' which is 'Up About a minute'. The 'Status' section features a green donut chart indicating that 100% of containers are running. Below the chart, there's a legend for 'Running' (green), 'Stopped' (red), and 'Ghost' (grey). At the bottom, there are two line graphs: 'Containers created' and 'Images created', both showing a count of 1 over time.

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:f852efef88c25f6d22354d547d892591867aa4826a7fabed819d9f388af6fc
-> resolve docker.io/library/python:3.8@sha256:f852efef88c25f6d22354d547d892591867aa4826a7fabed819d9f388af6fc
-> sha256:f852efef88c25f6d22354d547d892591867aa4826a7fabed819d9f388af6fc 1.86kB / 1.86kB
-> sha256:d907a408748ec879d75ac31872359c2de519f82214c0448e92e333b376d3b8ed 2.22kB / 2.22kB
-> sha256:54288b3a087c5a3ad34c6e21fc889abbc8486a27634c8892086ff71f1f4ab18a 9.27kB / 9.27kB
-> sha256:8e29544d541cdd389281d21a73aed1db78865c1b95b74f32b88e6b77a6e1e3 54.92MB / 54.92MB
-> sha256:9b829c73b52b97d5c07a54fbef3e921995a296c714b53a32a67d19231fcd 5.15MB / 5.15MB
-> sha256:cb5b7ae361722f078eca53f35823ed21baa85d61d5d95cd5a95ab53d740cdd56 19.87MB / 19.87MB
-> sha256:6494e4811622b31c027ccac322ca63937f7d892591867aa4826a7fabed819d9f388af6fc 54.57MB / 54.57MB
-> sha256:6f9f74886d7a93fe8172f594fab85eb4e8a821a8fcd0112efc7e4d3c78f7 196.51MB / 196.51MB
-> sha256:5e3b1213efc56598e78bd682983945c164de2a37205e06a62da623124dc743 6.20MB / 6.20MB
-> extracting sha256:8e29544d541cdd389281d21a73aed1db78865c1b95b74f32b88e6b77a6e1e3
-> sha256:9fddfd356334f2e6fad7e241bf5e7459c40ed105c5478676f41c1244bd96752 14.21MB / 14.21MB
-> extracting sha256:9b829c73b52b97d5c07a54fbef3e921995a296c714b53a32a67d19231fcd
-> extracting sha256:cb5b7ae361722f078eca53f35823ed21baa85d61d5d95cd5a95ab53d740cdd56
-> sha256:404f02944bac0432ca522cb9f254b1c91fca0886bfeef0be0b243b2f31bab7 235B / 235B
-> sha256:c4f42be2ba53b900ebffc048c1df13de538434ccc5f5d054a5686a6169a3a3f 2.21MB / 2.21MB
-> extracting sha256:6494e4811622b31c027ccac322ca63937f7d892591867aa4826a7fabed819d9f388af6fc
-> extracting sha256:6f9f74886d7a93fe8172f594fab85eb4e8a821a8fcd0112efc7e4d3c78f7
-> extracting sha256:5e3b1213efc56598e78bd682983945c164de2a37205e06a62da623124dc743
-> extracting sha256:9fddfd356334f2e6fad7e241bf5e7459c40ed105c5478676f41c1244bd96752
-> extracting sha256:404f02944bac0432ca522cb9f254b1c91fca0886bfeef0be0b243b2f31bab7
-> extracting sha256:c4f42be2ba53b900ebffc048c1df13de538434ccc5f5d054a5686a6169a3a3f
[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 libm_db
exporting to image
-> exporting layers
-> writing image sha256:1756719486df0807fad5dae305c5221513f2ff92b1b49add242b22a28af0379f19
-> naming to docker.io/library/job-portal-main

```

or '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. The 'Rolling Deploy' section contains a 'Deploy configuration' table with the following fields:

Field	Value
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 }
```

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 pavithra-add  
<yourpavithra>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: bindingapiVersion:
```

```
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
```