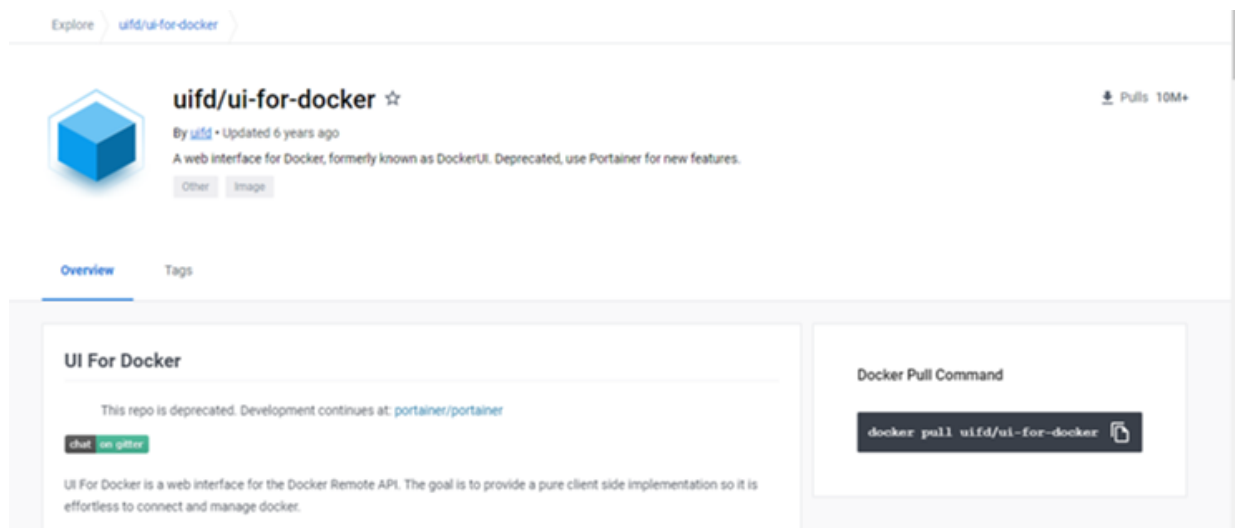



## Assignment - 4 Docker and Kubernetes

|                     |                 |
|---------------------|-----------------|
| Assignment Date     | 5 November 2022 |
| Student Name        | MSR.Koshikha    |
| Student Roll Number | 910619106026    |
| Maximum Marks       | 2 Marks         |

### Pull an image from docker hub and run it in docker Playground



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**

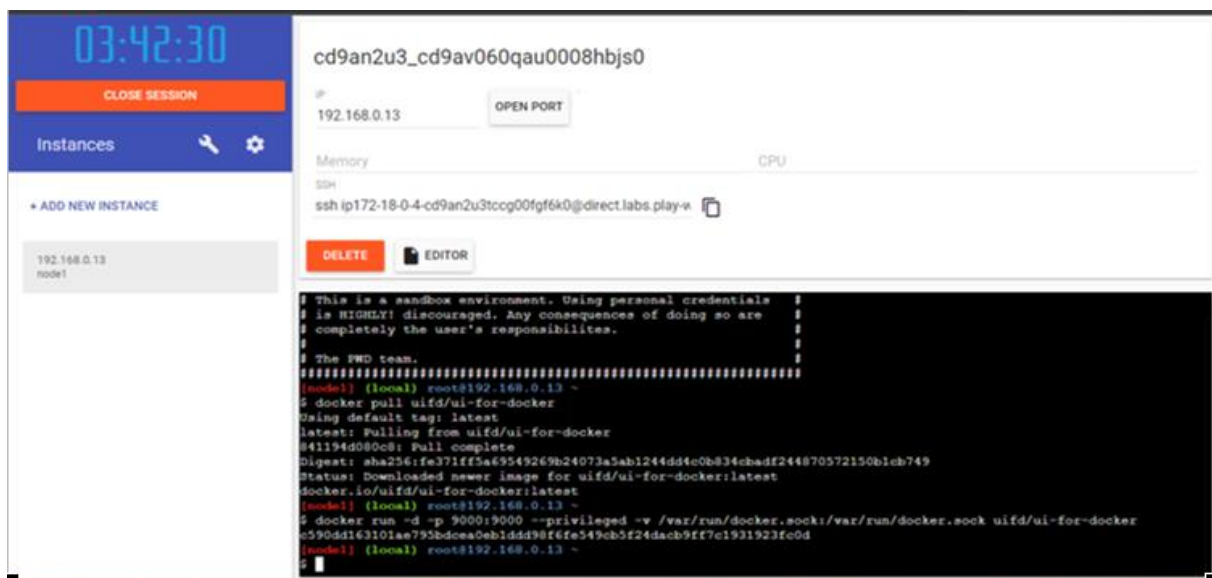
This repo is deprecated. Development continues at: [portainer/portainer](#)

[chat](#) [on 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 effortless to connect and manage docker.



**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**

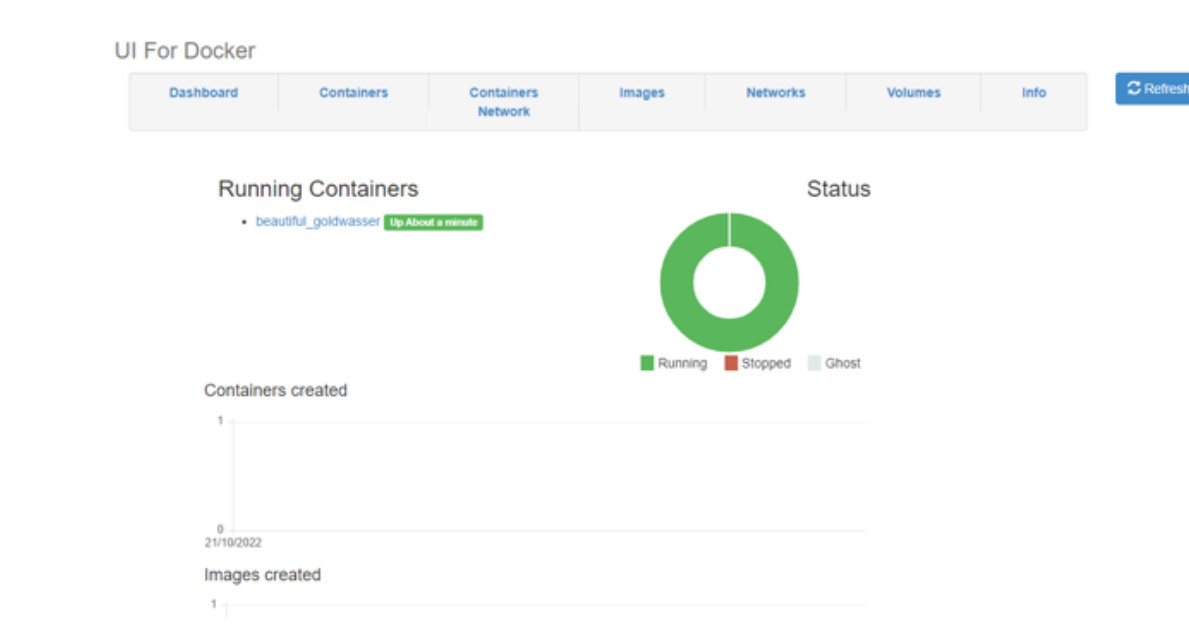
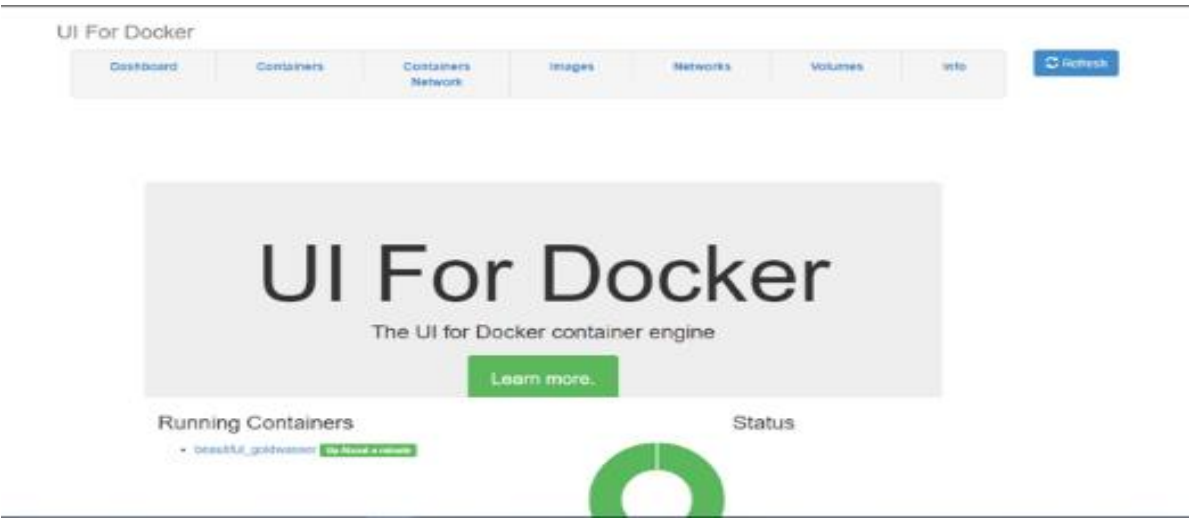
Memory CPU

SSH: [ssh ip172-18-0-4-cd9an2u3tccg00fgf6k0@direct.labs.play-w](#)

**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 FWD team.  
#####  
[root@192.168.0.13 ~]  
$ docker pull uifd/ui-for-docker  
Using default tag: latest  
latest: Pulling from uifd/ui-for-docker  
841194d080c8: Pull complete  
Digest: sha256:fe371ff5a69549269b24073a5ab1244dd4c0b834cbad244870572150b1cb749  
Status: Downloaded newer image for uifd/ui-for-docker:latest  
docker.io/uifd/ui-for-docker:latest  
[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  
c590dd163101ae795bdc0eb1ddd98f6fe549cb5f24dab9ff7c1931923fc0d  
[root@192.168.0.13 ~]  
$
```

# UI For Docker



```

-> [internal] load build definition from Dockerfile
-> transferring Dockerfile: 32B
-> [internal] load .dockerignore
-> transferring context: 0B
-> [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: 607B
-> [1/8] FROM docker.io/library/python:3.6@sha256:f8852a6a788c25f6d2154654788c2591867aa402ba7f3e6810d9f9188a76fc
-> resolve docker.io/library/python:3.6@sha256:f8852a6a788c25f6d2154654788c2591867aa402ba7f3e6810d9f9188a76fc
-> sha256:f8852a6a788c25f6d2154654788c2591867aa402ba7f3e6810d9f9188a76fc 1.05kB / 1.05kB
-> sha256:d007a907a8c070d5ac11072159c2de510f82214c0440e92b103b375d0b6d0 2.22kB / 2.22kB
-> sha256:542600380075e3a74c8e21fc088abbc0486a27634c0007000ff71f3f440104 9.27kB / 9.27kB
-> sha256:0e29546d841c0d309281d21a73a9d1db70605c1091b74f32b080e0077ade1e3 54.92MB / 54.92MB
-> sha256:0b825c71052b07d5c07a54f0f3e921095a290c710b53a32ae67d18211fcd 5.15MB / 5.15MB
-> sha256:c0507ae103722f070eac53f30821ed21baa8581d5d95cd5a95ae533748cd56 10.67MB / 10.67MB
-> sha256:6434e081102031c027cc322ca483937f0886f568a3b6f15c01aade718793 54.57MB / 54.57MB
-> sha256:0f9f74805d7e93fe0172594fana05004e6a0401a9fef0d112efc7e4d3c70f7 180.51MB / 180.51MB
-> sha256:5e361211efr36508e70b00070b1045c164de2a37385e06a03daa21174dc743 6.20MB / 6.20MB
-> extracting sha256:0e29546d841c0d309281d21a73a9d1db70605c1091b74f32b080e0077ade1e3
-> sha256:0f0d0f58134f0de6ad7e242b75e7459c40a0195c5478076741c1240d000732 14.11MB / 14.11MB
-> extracting sha256:0b825c71052b07d5c07a54f0f3e921095a290c710b53a32ae67d18211fcd
-> extracting sha256:cd5b7ae361722f070eac53f30821ed21baa85d01d5d95cd5a95ae533748cd56
-> sha256:444f02044bac0432ca521c0e7154b1c91fca0800f0e700c005430f31ba07 210B / 210B
-> sha256:c4f42bc2053000000f9c000c10f13de38234cccc5f5d95435084ba18a1a1f 2.21MB / 2.21MB
-> extracting sha256:8484a811622031a027cc322ca483937f0886f568a3b6f15c01aade718793
-> extracting sha256:0f9f74805d7e93fe0172594fana05004e6a0401a9fef0d112efc7e4d3c70f7
-> extracting sha256:5e361211efr36508e70b00070b1045c164de2a37385e06a03daa21174dc743
-> extracting sha256:0f0d0f58134f0de6ad7e242b75e7459c40a0195c5478076741c1240d000732
-> extracting sha256:0f0d0f58134f0de6ad7e242b75e7459c40a0195c5478076741c1240d000732
-> extracting sha256:444f02044bac0432ca521c0e7154b1c91fca0800f0e700c005430f31ba07
-> extracting sha256:c4f42bc2053000000f9c000c10f13de38234cccc5f5d95435084ba18a1a1f
-> [2/8] WORKDIR /app
-> [3/8] ADD . /app
-> [4/8] COPY requirements.txt /app
-> [5/8] RUN python3 -m pip install -r requirements.txt
-> [6/8] RUN python3 -m pip install lib_db
-> exporting to image
-> exporting layers
-> writing image sha256:175b7194800f902fa55da585c3211373ff2d1b008d382022a28a90379f10
-> naming to docker.io/library/job-portal-main

```

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

C:\Users\WK-PC\Desktop\job-portal-main>

create a docker file for the job portal application and deploy it in Docker desktop application

