

ASSIGNMENT 4

Student Name	P.JAYASHREE
Student Roll Number	951919CS031
Team ID	PNT2022TMID13229

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

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

```
docker run -d -p 9000:9000 --privileged -v
```

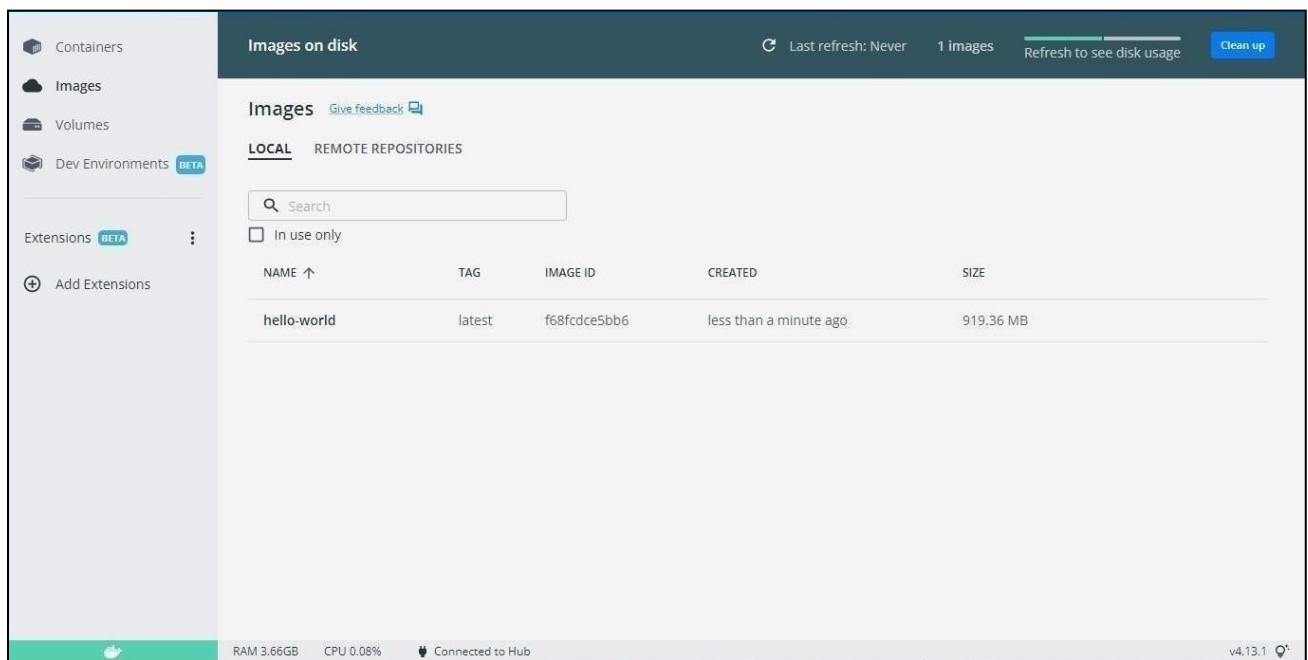
```
/var/run/docker.sock:/var/run/docker.sock uifd/ui-for- docker
```

The screenshot displays the Docker Playground interface. At the top, a blue header shows the time 03:57:05 and a 'CLOSE SESSION' button. Below this, the 'Instances' section lists a single instance named 'node1' with IP 192.168.0.13. The main panel shows the details of the selected instance, including its IP (192.168.0.13), memory usage (1.59%), CPU usage (0.45%), and an SSH command to connect. Below the instance details is a terminal window showing the command 'docker pull uifd/ui-for-docker' and its output, which includes a warning about the sandbox environment and the successful pull of the 'latest' tag.

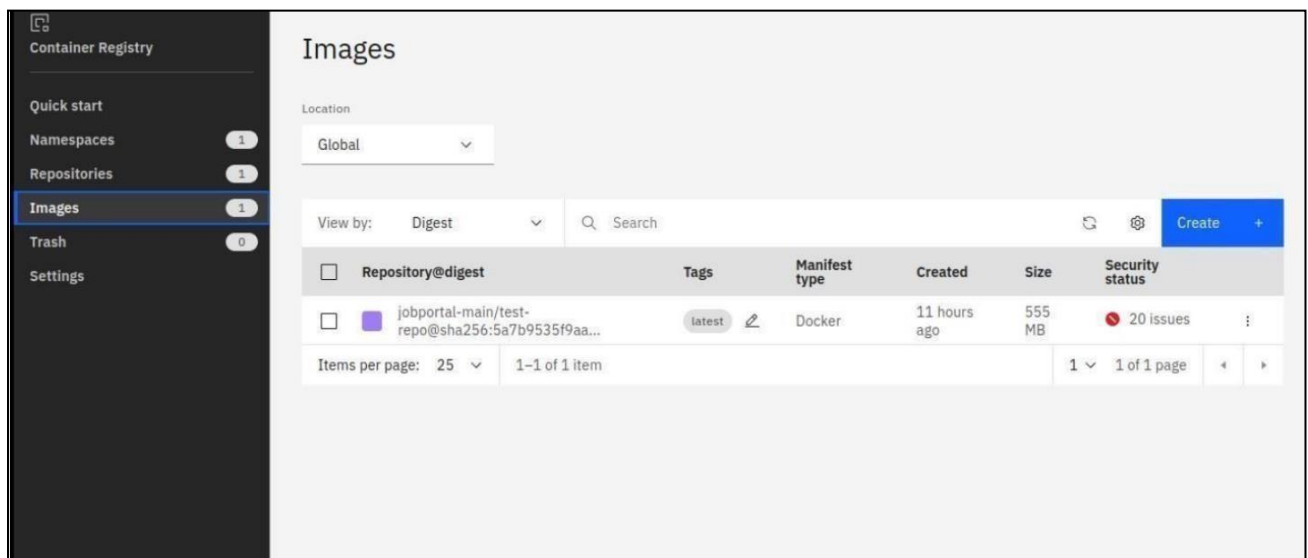
Below the terminal window is the 'UI For Docker' dashboard. It features a navigation bar with tabs for Dashboard, Containers, Containers Network, Images, Networks, Volumes, and Info. A 'Refresh' button is located on the right. The main content area is divided into two sections: 'Running Containers' and 'Status'. The 'Running Containers' section shows a single container named 'serene_keller' with a status of 'Up 47 seconds'. The 'Status' section features a donut chart showing the status of containers: Running (green), Stopped (red), and Ghost (grey). Below the chart, there are two line graphs: 'Containers created' and 'Images created', both showing a count of 1 over time.

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

```
FROM helloworld:latest
WORKDIR ~/Desktop/
ADD . helloworld/
WORKDIR ~/Desktop/htmlfile
RUN pip install -r requirements
RUN chmod +x app.sh
CMD
```



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



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.

apiVersion: v1 kind:

Service metadata:

name: hello-world- deployment spec:

ports: - port:

5000 targetPort:

5000 selector: app:

hello-world --- apiVersion: apps/v1 kind: Deployment metadata: name:

hello-world- deployment spec:

replic as: 1 select or:

matchLabels: app: helloworld

template: meta da ta :

la be ls :

app: hello- world spec:

containers:

- name: hello-world image: au.icr.io/hello-worldapp/helloworld imagePullPolicy:

Always ports:

- containerPort: 5000