

Assignment	04
Student Name	G.Gayathri
Student Roll Number	952319104011
Maximum Marks	2 Mark

### Question-1

**Pull an image from Dockers hub and run it in Dockers playground.**

#### SOLUTION:

##### STEP: 1

##### STEP: 2

##### STEP: 3

Login to Dockers hub and get an image

- Open Dockers playground
- Login with Dockers
- Create new instance

In the command prompt run the following:

\$	docker pull hello-world	//To pull an image from docker hub
\$	docker run hello-world	//To run the image in docker playground

```

[roch@1] (local) root@192.168.0.8 ~
$ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
Digest: sha256:18a657d0cc1c7d0678a3fbee8b7eb4918bbe25968d3e1b0e5ebfa71caddbc346
Status: Image is up to date for hello-world:latest
docker.io/library/hello-world:latest
[roch@1] (local) root@192.168.0.8 ~
$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (and$4)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

[roch@1] (local) root@192.168.0.8 ~
$

```

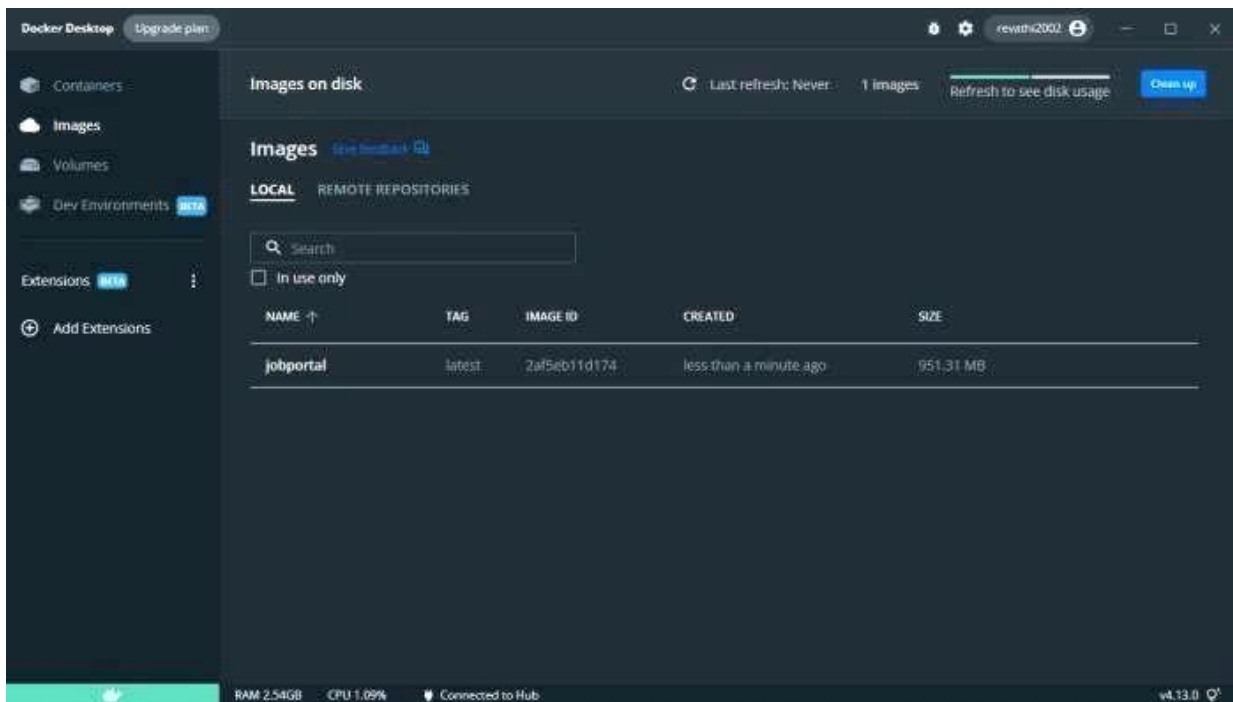
## QUESTION 2:

Create a  
Dockerfile and  
deploy it  
in Docker  
desktop  
application  
SOLUTION:

### STEP: 1

### STEP: 2

- Create a flask application
- Create a Dockerfile in the same folder



Run the following commands to deploy it in docker desktop

\$	<b>docker build -t jobportal</b>	// to deploy all the folders to docker desktop
\$	<b>docker image ls</b>	//to show the list of images in docker desktop

\$ **docker container run -p 5000:5000 jobportal** //to run

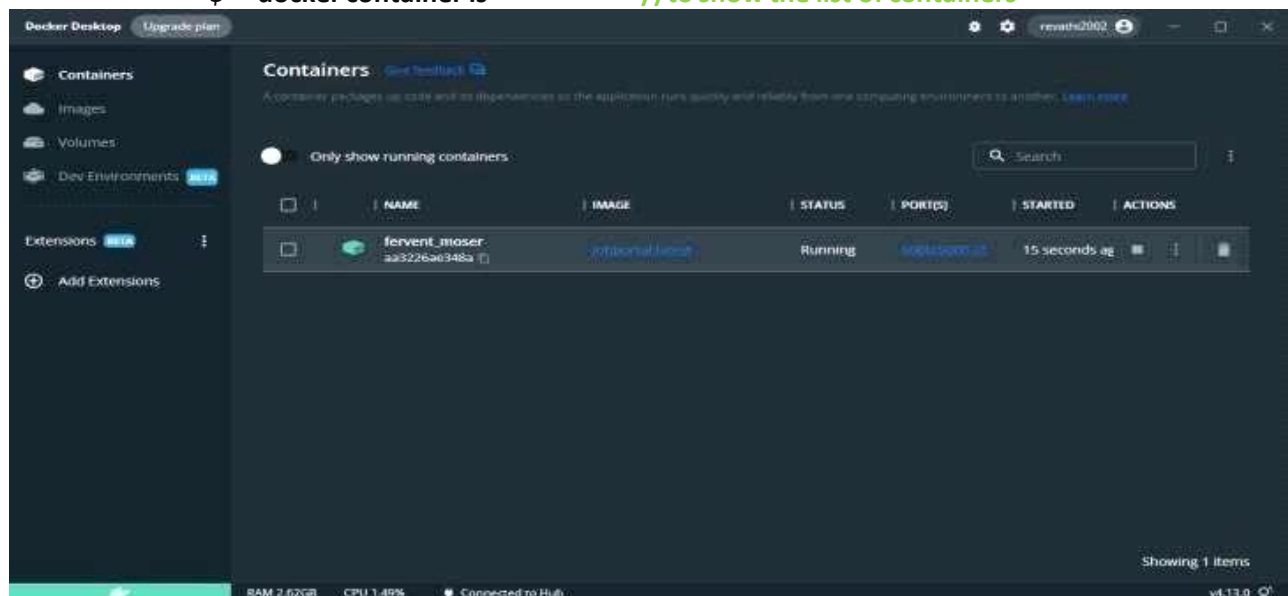
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS D:\Ibse project\Assignments\Team lead\Assignment 4\Docker Desktop> docker container run -p 5000:5000 jobportal
* Serving Flask app "app"
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://172.17.0.1:5000
* Running on http://172.17.0.2:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 221-899-599
172.17.0.1 - - [25/Oct/2022 11:30:28] "GET / HTTP/1.1" 200 -
172.17.0.1 - - [25/Oct/2022 11:30:29] "GET /static/css/style.css HTTP/1.1" 200 -
172.17.0.1 - - [25/Oct/2022 11:30:29] "GET /static/images/1.png HTTP/1.1" 200 -
172.17.0.1 - - [25/Oct/2022 11:30:30] "GET /favicon.ico HTTP/1.1" 404 -
PS D:\Ibse project\Assignments\Team lead\Assignment 4\Docker Desktop> docker image ls
REPOSITORY TAG IMAGE ID CREATED SIZE
jobportal latest 2af5eb11d174 7 minutes ago 951MB
PS D:\Ibse project\Assignments\Team lead\Assignment 4\Docker Desktop> docker container ls
CONTAINER ID IMAGE COMMAND STATUS PORTS NAMES
PS D:\Ibse project\Assignments\Team lead\Assignment 4\Docker Desktop>

```



\$ docker container ls //to show the list of containers



### QUESTION 3:

Create an IBM container  
registry and deploy hello-world-  
app or job-portal- app

SOLUTION:

#### QUESTION 4:

Create a Kubernetes cluster in IBM cloud and deploy hello- world-image or job-portal-image and also expose the same app to run in node-port.

#### SOLUTION:

- Select your cluster from the cluster list to open the details for your cluster.
- Click Kubernetes dashboard.
- From the menu bar, click the Create new resource icon (+).
- Select the Create from form tab.
  - Enter a name for your app, i.e [hello-world](#).
  - Enter [websphere-liberty](#) for your container image.
  - Enter the number of pods for your app deployment, such as 1.
  - Leave the Service drop-down menu set to None.
- Click Deploy. During the deployment, the cluster downloads the websphere-liberty container image from Docker Hub and deploys the app in your cluster.
- Create a node port so that your app can be accessed by other users internally or externally. Because your cluster is a free cluster, you can only expose an app with a node port, not a load balancer or Ingress.
  - Click the Create new resource icon (+).
  - Copy the node port YAML from GitHub.
    - In the Create from input box, paste the node port YAML that you copied in the previous step.
    - Click Upload. The node port service is created.
- From the menu, click Services, and note the TCP endpoint port of your liberty service in the node port range 30000 - 32767, i.e [liberty:30357 TCP](#).
- From the menu, click Pods, and note the Node that your pod runs on, such as 10.xxx.xx.xxx.
- Return to the IBM Cloud clusters console, select your cluster, and click the Worker Nodes tab. Find the Public IP of the worker node that matches the private IP of the node that the pod runs on.
- In a tab in your browser, form the URL of your app by combining [http://](#) with the public IP and TCP port that you previously retrieved i.e. [http:// 159.122.178.57: 30357](#). The

Welcome to Liberty page is displayed. Great job! You just deployed your first app in your Kubernetes cluster.

