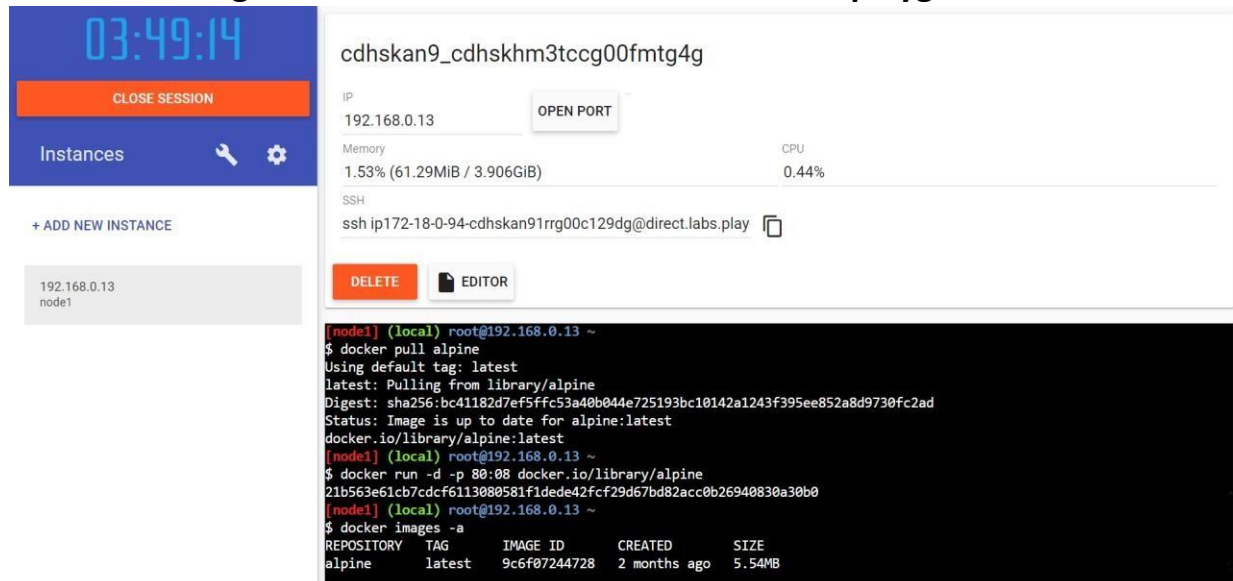


Assignment 4

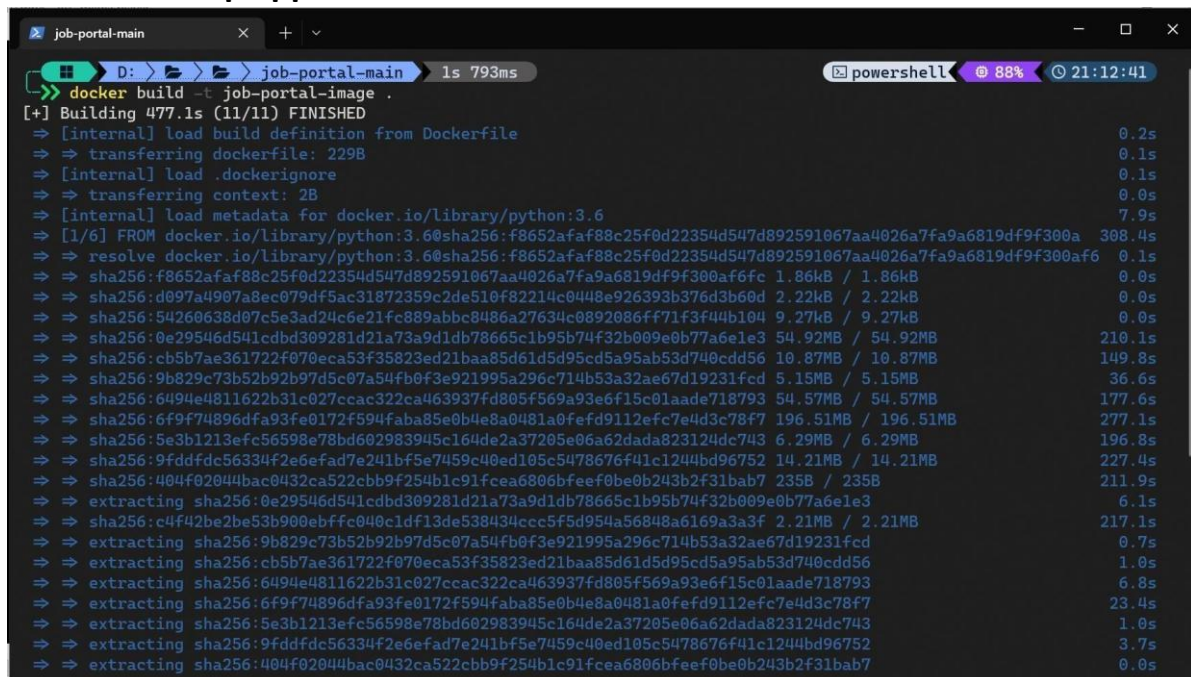
1. Pull an Image from docker hub and run it in docker playground.



The screenshot shows the Docker Playground interface. On the left, there's a sidebar with a clock showing 03:49:14, a 'CLOSE SESSION' button, and an 'Instances' section with a '+ ADD NEW INSTANCE' button. Below that, a list of instances shows '192.168.0.13 node1'. The main area displays details for the instance 'cdhskan9_cdhskhm3tccg00fmtg4g'. It shows the IP '192.168.0.13', memory usage '1.53% (61.29MiB / 3.906GiB)', and CPU usage '0.44%'. There's an 'SSH' button and a terminal window. The terminal shows the following commands and output:

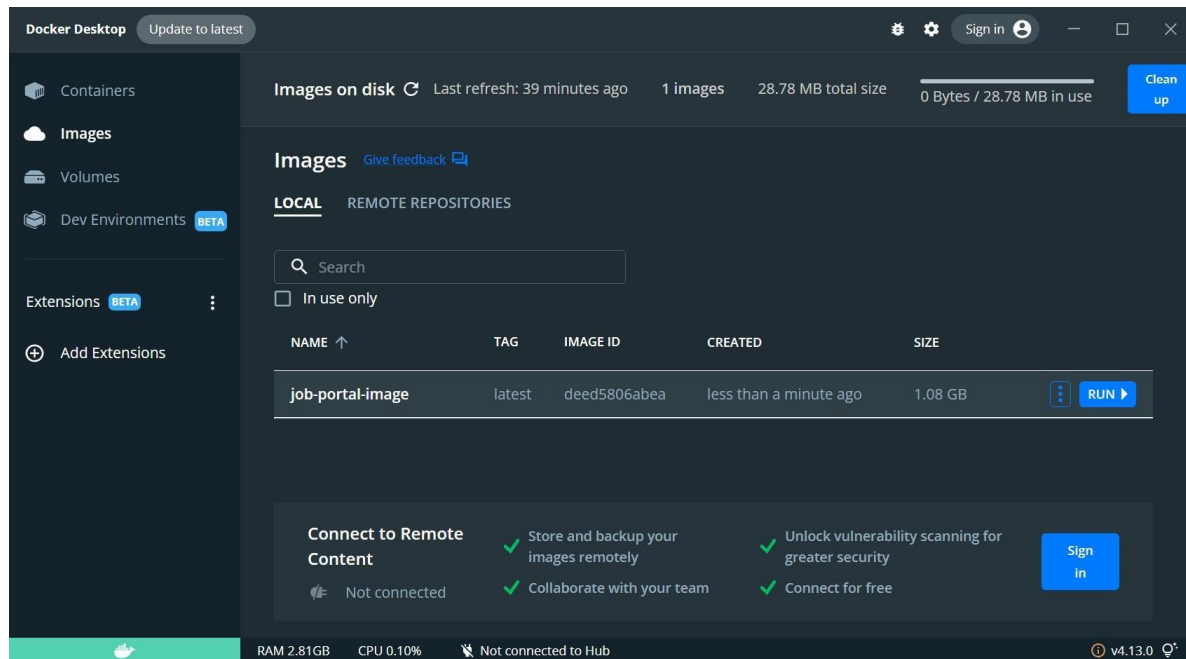
```
[node1] (local) root@192.168.0.13 ~
$ docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
Digest: sha256:bc41182d7ef5ffc53a40b044e725193bc10142a1243f395ee852a8d9730fc2ad
Status: Image is up to date for alpine:latest
docker.io/library/alpine:latest
[node1] (local) root@192.168.0.13 ~
$ docker run -d -p 80:80 docker.io/library/alpine
21b563e61cb7cdcf6113080581f1d9de42fcf29d67bd82acc0b26940830a30b0
[node1] (local) root@192.168.0.13 ~
$ docker images -a
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
alpine        latest    9c6f07244728   2 months ago   5.54MB
```

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



The screenshot shows a Docker Desktop terminal window with the title 'job-portal-main'. The terminal output shows the build process for 'job-portal-main' using 'docker build -t job-portal-image .'. The build is successful, and the terminal shows the following output:

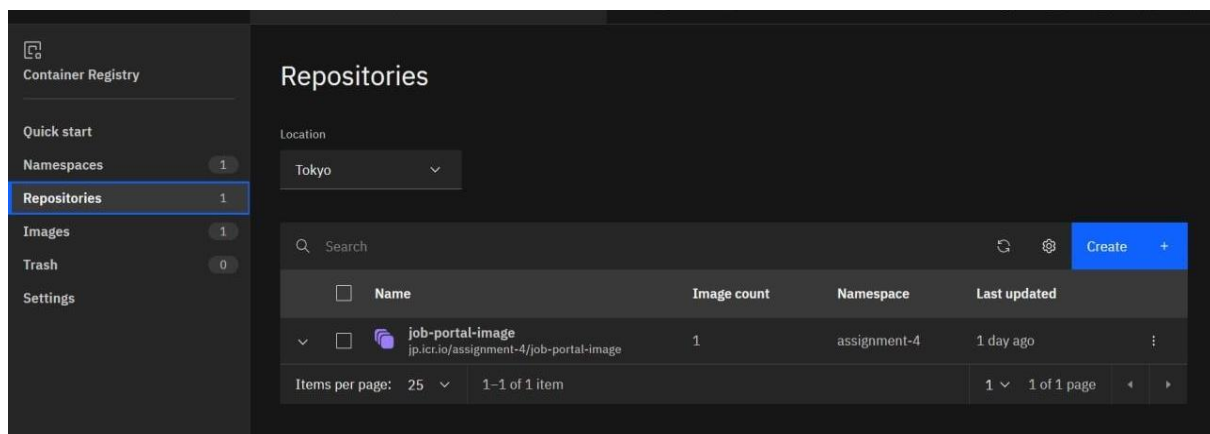
```
[+] Building 477.1s (11/11) FINISHED
=> [internal] load build definition from Dockerfile                                0.2s
=> => transferring dockerfile: 229B                                              0.1s
=> [internal] load .dockerignore                                                  0.1s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for docker.io/library/python:3.6                    7.9s
=> [1/6] FROM docker.io/library/python:3.6@sha256:f8652afaf88c25f0d22354d547d892591067aa4026a7fa9a6819df9f300af6 308.4s
=> => resolve docker.io/library/python:3.6@sha256:f8652afaf88c25f0d22354d547d892591067aa4026a7fa9a6819df9f300af6 0.1s
=> => sha256:f8652afaf88c25f0d22354d547d892591067aa4026a7fa9a6819df9f300af6fc 1.86kB / 1.86kB 0.0s
=> => sha256:d097a4907a8ec079df5ac31872359c2de510f82214c0448e926393b376d3b60d 2.22kB / 2.22kB 0.0s
=> => sha256:54260638d07c5e3ad24c6e21fc889abb8c8486a27634c0892086ff71f3f44b104 9.27kB / 9.27kB 0.0s
=> => sha256:0e29546d541cddb309281d21a73a9d1db78665c1b95b74f32b009e0b77a6e1e3 54.92MB / 54.92MB 210.1s
=> => sha256:cb5b7ae361722f070eca53f35823ed21baa85d61d5d95cd5a95ab53d740cdd56 10.87MB / 10.87MB 149.8s
=> => sha256:9b829c73b52b92b7d5c07a54fb0f3e921995a296c714b53a32ae67d19231fcd 5.15MB / 5.15MB 36.6s
=> => sha256:6494e4811622b31c027ccac322ca463937fd805f569a93e6f15c01aade718793 54.57MB / 54.57MB 177.6s
=> => sha256:6f9f74896dfa93fe0172f594faba85e0b4e8a0481a0fefd9112efc7e4d3c78f7 196.51MB / 196.51MB 277.1s
=> => sha256:5e3b1213efc56598e78bd602983945c164de2a37205e06a62dada823124dc743 6.29MB / 6.29MB 196.8s
=> => sha256:9fddfdc56334f2e6efad7e241bf5e7459c40ed105c5478676f41c1244bd96752 14.21MB / 14.21MB 227.4s
=> => sha256:404f02044bac0432ca522cbb9f254b1c91fcea6806bfeef0be0b243b2f31bab7 235B / 235B 211.9s
=> => sha256:0e29546d541cddb309281d21a73a9d1db78665c1b95b74f32b009e0b77a6e1e3 6.1s
=> => sha256:c4f42be2be53b900ebffcc040c1d13de538434ccc5f5d954a56848a6169a3a3f 2.21MB / 2.21MB 217.1s
=> => extracting sha256:9b829c73b52b92b7d5c07a54fb0f3e921995a296c714b53a32ae67d19231fcd 0.7s
=> => extracting sha256:cb5b7ae361722f070eca53f35823ed21baa85d61d5d95cd5a95ab53d740cdd56 1.0s
=> => extracting sha256:6494e4811622b31c027ccac322ca463937fd805f569a93e6f15c01aade718793 6.8s
=> => extracting sha256:6f9f74896dfa93fe0172f594faba85e0b4e8a0481a0fefd9112efc7e4d3c78f7 23.4s
=> => extracting sha256:5e3b1213efc56598e78bd602983945c164de2a37205e06a62dada823124dc743 1.0s
=> => extracting sha256:9fddfdc56334f2e6efad7e241bf5e7459c40ed105c5478676f41c1244bd96752 3.7s
=> => extracting sha256:404f02044bac0432ca522cbb9f254b1c91fcea6806bfeef0be0b243b2f31bab7 0.0s
```



3. Create a IBM container registry and deploy hello world app or job portal app

```

D:\> cd job-portal-main
D:\job-portal-main> docker push jp.icr.io/assignment-4/job-portal-image:latest
The push refers to repository [jp.icr.io/assignment-4/job-portal-image]
1123cd971779: Pushed
371132460927: Pushed
31ad0315bd33: Pushed
4b2875ca3326: Pushed
2b3f5aac8b57: Pushed
aa4c808c19f6: Pushed
8ba9f690e8ba: Pushed
3e607d59ef9f: Pushed
1e18e7e1fcc2: Pushed
c3a0d593ed24: Pushed
26a504e63be4: Pushed
8bf42db0de72: Pushed
31892cc314cb: Pushed
11936051f93b: Pushed
latest: digest: sha256:07e869893f7f8bc1e3ba9df888ae907f29d28ba7bda3b3f3e39b6a7aa1106cec size: 3260
  
```



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 nodeport.

The screenshot displays the IBM Cloud Kubernetes Service console for a cluster named 'mycluster-free'. The cluster is in the 'Preparing master, workers...' state and will expire in 30 days. The left sidebar shows navigation options: Overview, Worker nodes (selected), Worker pools, and DevOps (marked as New). The main area shows a table of worker nodes. One node is listed with the following details:

Name	Status	Worker pool	Zone	Private IP	Public IP	Version
000000df	Normal	default	Milan 01	10.144.217.27	159.122.181.104	1.24.6_1541

Below the table, the node's ID is 'kube-cdik2hgf0pkvlp5jdlg-myclusterfr-default-000000df'. The status is 'Normal'. The flavor is 'Free - 2 vCPUs 4GB RAM'. The private VLAN is '2218181' and the public VLAN is '2218179'. The bottom of the console shows 'Items per page: 25' and '1-1 of 1 item'.