

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

The screenshot shows the Docker Playground interface. At the top, there's a blue header with a clock showing 03:56:44 and a 'CLOSE SESSION' button. Below it, there's a section for 'Instances' with a '+ ADD NEW INSTANCE' button. A list of instances shows one instance named 'node1' with IP '192.168.0.13'. The main area displays the details of the selected instance, including its IP, memory usage (1.20% of 48.18MB / 3.906GB), CPU usage (0.24%), and an SSH command: 'ssh ip172-18-0-15-cdrikj60qau000891v40@direct.labs.pla'. Below this, there's a 'DELETE' button and an 'EDITOR' button. The terminal window shows the following commands and output:

```
[root@192.168.0.13 ~]# docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:fae8b744c97107ef34423fcccceec2398ec8a5759259f94d99078f264e9d7af
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
[root@192.168.0.13 ~]# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world    latest    fa5d9fa6a5    14 months ago  13.3kB
[root@192.168.0.13 ~]# 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.
   (amd64)
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/

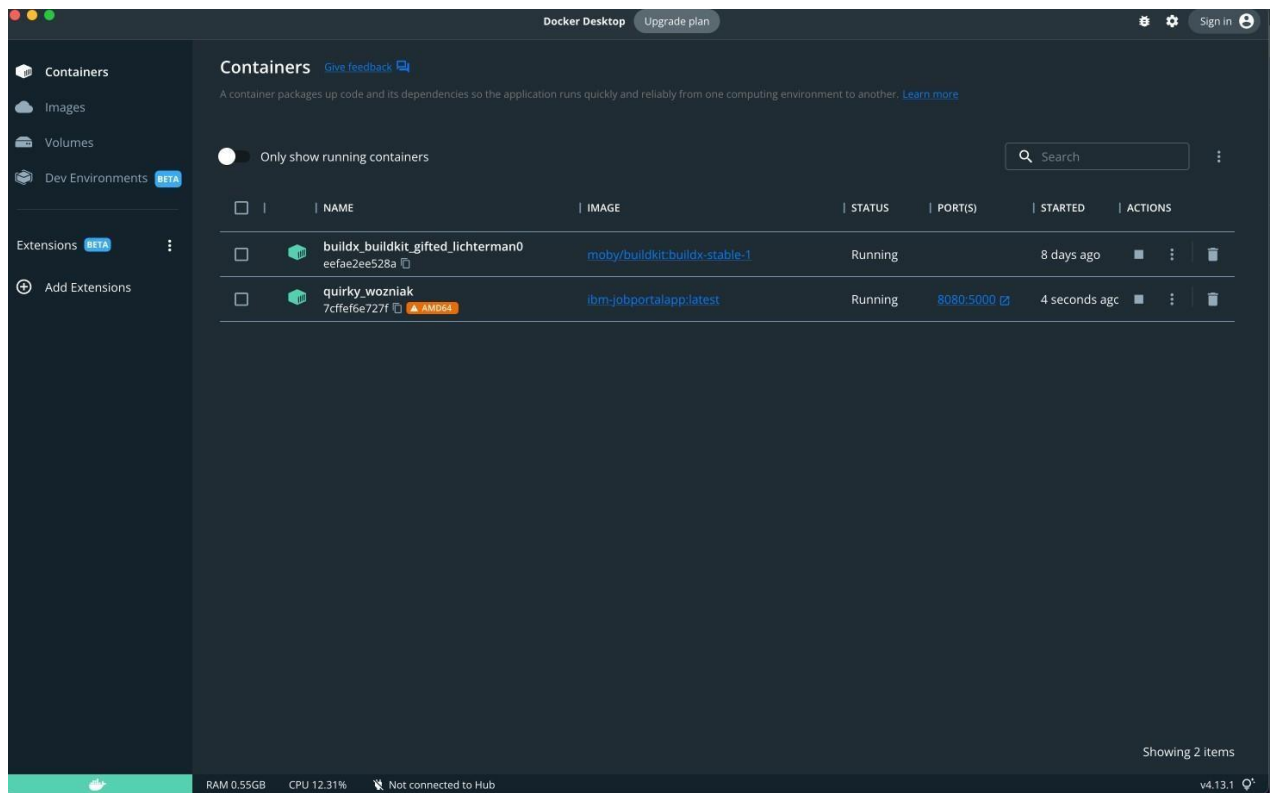
For more examples and ideas, visit:
https://docs.docker.com/get-started/
[root@192.168.0.13 ~]#
```

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

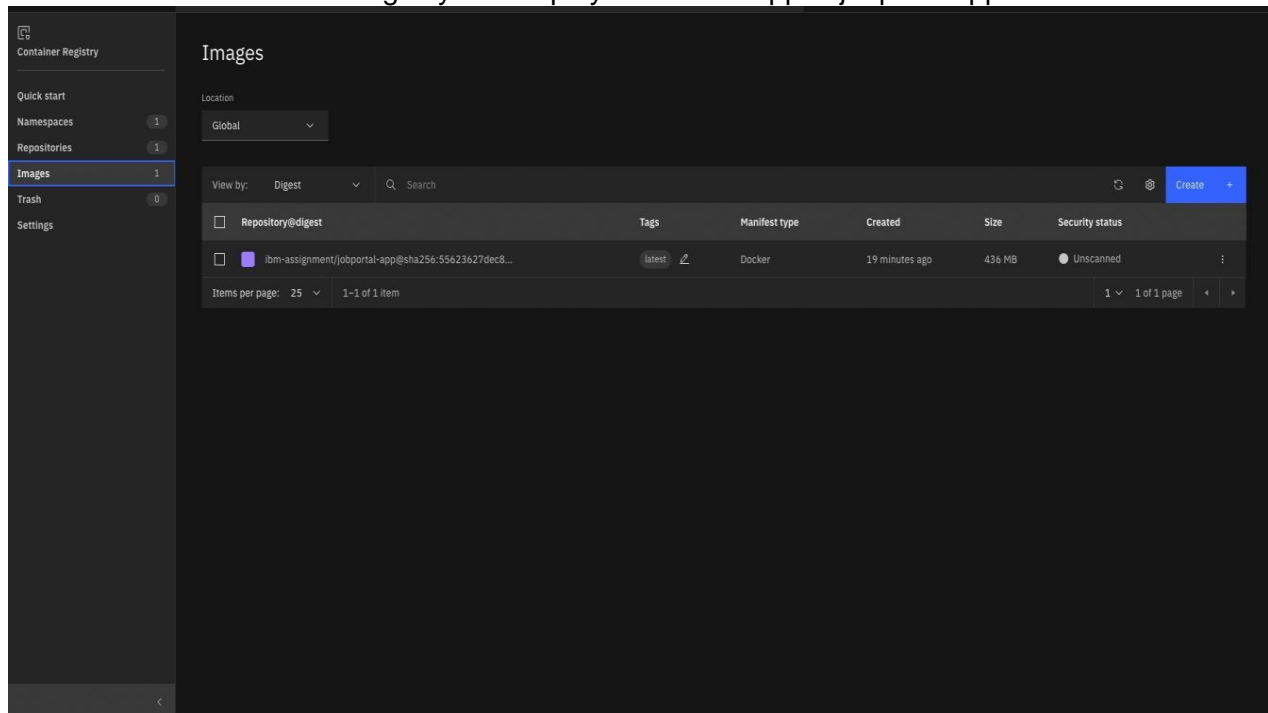
The screenshot shows the Docker Desktop interface. The left sidebar has a menu with 'Containers', 'Images', 'Volumes', 'Dev Environments', 'Extensions', and 'Add Extensions'. The main area is titled 'Images on disk' and shows a list of images. The 'LOCAL' tab is selected, and the 'In use only' checkbox is checked. The list of images is as follows:

NAME	TAG	IMAGE ID	CREATED	SIZE
ibm-jobportalapp	latest	0ec838117a9d	3 minutes ago	1.08 GB
moby/buildkit	buildx-stable-1	71ac63309b0f	about 1 month ago	133.56 MB

At the bottom, there's a section for 'Connect to Remote Content' with three options: 'Store and backup your images remotely', 'Unlock vulnerability scanning for greater security', and 'Collaborate with your team'. A 'Sign in' button is also present.



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



3. Create a Kubernetes cluster in IBM cloud and deploy helloworld image or jobportal image and also expose the same app to run in nodeport.

