

Assignment- 4

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

03:57:32

CLOSE SESSION

Instances

+ ADD NEW INSTANCE

192.168.0.8
node1

cddvksm0_cddvkvm0qau000a07j5g

IP
192.168.0.8

OPEN PORT

Memory
1.24% (49.52MiB / 3.906GiB)

CPU
0.31%

SSH
ssh ip172-18-0-22-cddvksm0qau000a07j50@direct.labs.pla

DELETE

EDITOR

```
#####
#                               #
#   WARNING!!!!                #
#   This is a sandbox environment. Using personal credentials   #
#   is HIGHLY! discouraged. Any consequences of doing so are   #
#   completely the user's responsibilities.                      #
#   The PwD team.                                               #
#####
[node1] (local) root@192.168.0.8 ~
$ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:e18f8a777aefabe047a671ab3ec3eed05414477c951ab1a6f352a06974245fe7
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
[node1] (local) root@192.168.0.8 ~
$ docker run hello-world
```

Activate Windows
Go to Settings to activate Windows.

03:57:05

CLOSE SESSION

Instances

+ ADD NEW INSTANCE

192.168.0.8
node1

cddvksm0_cddvkvm0qau000a07j5g

IP
192.168.0.8
OPEN PORT

Memory
1.26% (50.45MiB / 3.906GiB)

CPU
0.39%

SSH
ssh ip172-18-0-22-cddvksm0qau000a07j50@direct.labs.pla

DELETE

EDITOR

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/>

[node1] (local) root@192.168.0.8 ~
\$

Activate Windows
Go to Settings to activate Windows.

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

DOCKER FILE:

```
1 FROM python:3.8-buster
2
3 WORKDIR /app
4
5 COPY requirements.txt /app/
6
7 RUN pip install -r requirements.txt
8
9 COPY . /app/
10
11 RUN cp .env.dev.sample .env
12
13 EXPOSE 8000
14
15 RUN chmod +x entrypoint.sh
16
17 CMD ["sh", "entrypoint.sh"]
```

DEPLOYMENT OF JOBPORTAL APPLICATION:

Containers

Images

Volumes

Dev Environments BETA

Extensions BETA

Add Extensions

Containers Give feedback

A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. [Learn more](#)

Only show running containers

Search

	NAME	IMAGE	STATUS	PORT(S)	STARTED	ACTIONS
<input type="checkbox"/>	<div>agitated_neumann</div> <div>918d20882039</div>	icr.io/helloapp/ibm:latest	Exited (137)	49160:8080		<div><div></div><div></div><div></div></div>
<input type="checkbox"/>	<div>jolly_turing</div> <div>b62c0712bdd3</div>	jobportalapplication:latest	Running	1234:8000	4 minutes ago	<div><div></div><div></div><div></div></div>

Showing 2 items

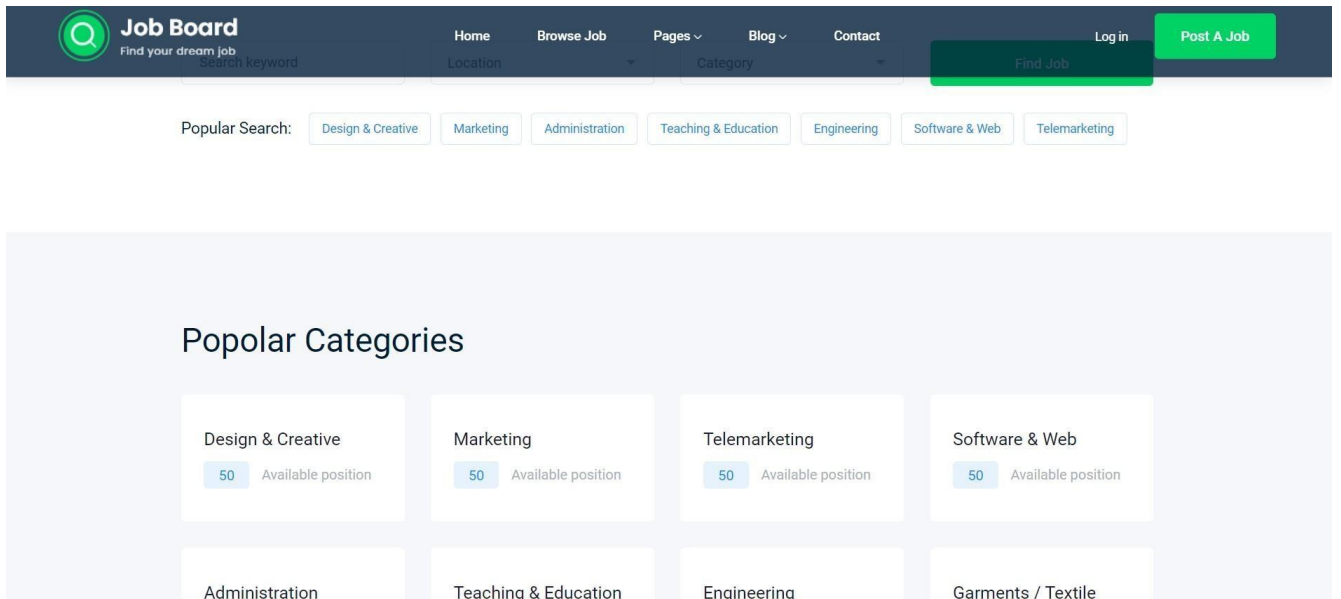
RAM 3.06GB

CPU 0.57%

Connected to Hub

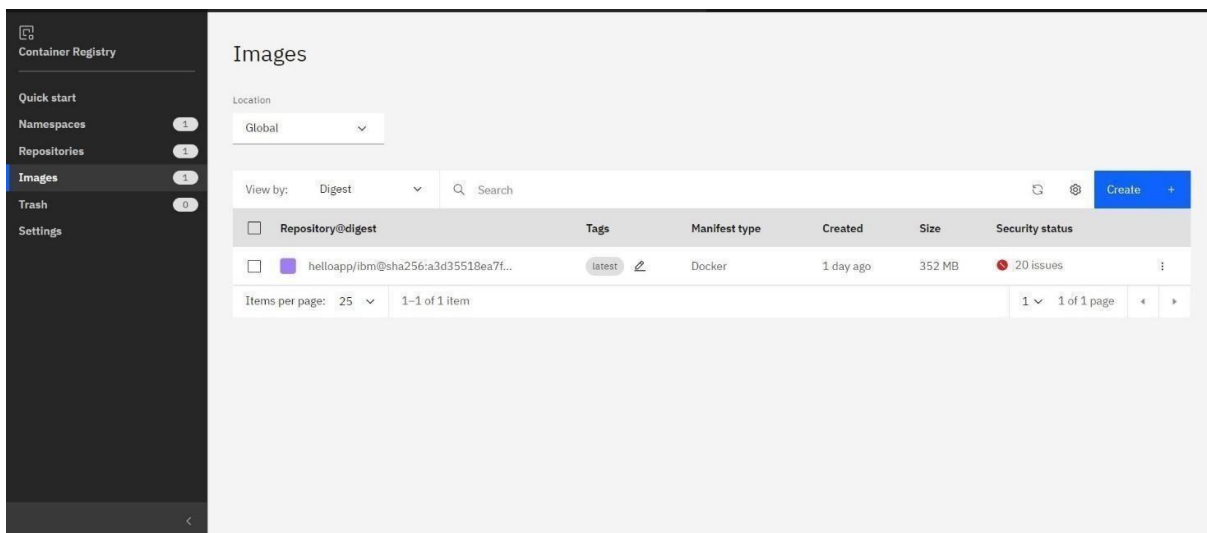
v4.13.0

OUTPUT:

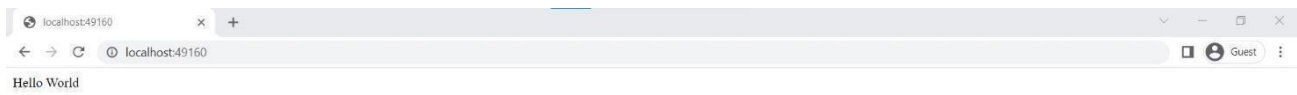


3.Create a IBM container registry and deploy hello-world app or job port app.IBM

CONTAINER REGISTRY DEPLOYMENT:

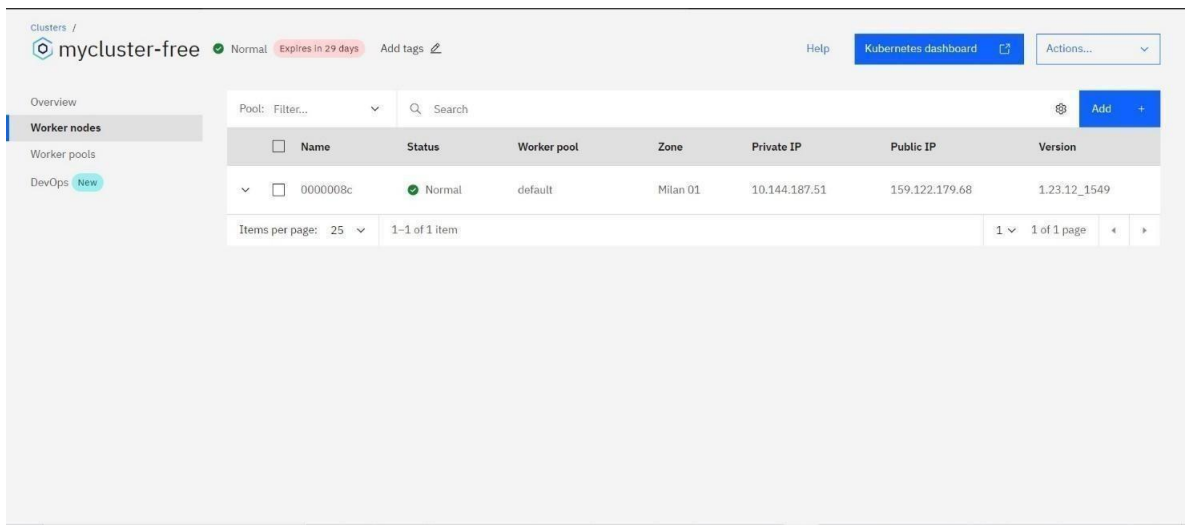


OUTPUT:



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.

Creating Kubernetes cluster in IBM cloud and exposing node port:



Output:

The image displays a side-by-side comparison of an HTML file and its rendered output. On the left, the VS Code editor shows the source code for 'jobportal.html'. The code includes a navigation bar with a 'Login Page' link, a main content area with a centered message 'Find your dream job by using this Job portal.', and a footer with a list of job roles. Each role is presented in a card format with a title, description, and a 'Submit' button. The roles listed are Software Engineer, Front-End Developer, Special title treatment, Java Developer, Business Analyst, Python Developer, Back-End Developer, and Product Manager. The code uses Bootstrap classes for styling and layout.

```
30 <div class="nav-bar">
31   <a href="#" class="nav-link">Home</a>
32   <a href="#" class="nav-link">About</a>
33   <a href="#" class="nav-link">Contact</a>
34   <a href="#" class="nav-link">Login Page</a>
35 </div>
36 <div class="main-content">
37   <div class="text-center">
38     <p>Find your dream job by using this Job portal.</p>
39   </div>
40   <div class="text-center">
41     <p>Below listed jobs are suited for your career.</p>
42   </div>
43   <div class="row">
44     <div class="col-sm-12">
45       <div class="card">
46         <div class="card-body">
47           <h5 class="card-title">Software Engineer</h5>
48           <p class="card-text">Execute full software development life cycle (SDLC).
49             Develop flowcharts, layouts and documentation to identify requirements and solutions.
50           </p>
51           <a href="{{url_for('register')}}" class="btn btn-primary">Submit</a>
52         </div>
53       </div>
54     </div>
55     <div class="col-sm-12">
56       <div class="card">
57         <div class="card-body">
58           <h5 class="card-title">Front-End Developer</h5>
59           <p class="card-text">Optimizing the user experience.
60             Using HTML, JavaScript and CSS to bring concepts to life.
61           </p>
62           <a href="{{url_for('register')}}" class="btn btn-primary">Submit</a>
63         </div>
64       </div>
65     </div>
66   </div>
67 </div>
```

On the right, the rendered output is shown in a web browser. It features a blue header with a 'Login Page' link. The main content area has a light blue background with a centered message. Below this, a list of job roles is displayed, each in a white card with a blue border. The roles and their descriptions are: Software Engineer (Execute full software development life cycle (SDLC). Develop flowcharts, layouts and documentation to identify requirements and solutions.), Front-End Developer (Optimizing the user experience. Using HTML, JavaScript and CSS to bring concepts to life.), Special title treatment (A data analyst collects and stores data on sales, marketing, research, logistics, linguistics, or other behaviors.), Java Developer (A Java Developer is responsible for planning, designing, developing, and managing Java-based applications and software.), Business Analyst (Business analysts assess how organizations are performing and help them improve their processes and systems.), Python Developer (Python Developer is responsible for coding, designing, deploying, and debugging development projects.), Back-End Developer (Compile and analyze data, processes, and codes to troubleshoot problems and identify areas for improvement.), and Product Manager (A product manager is the person who identifies the customer need and the larger business objectives that a product or feature will fulfill.). Each card has a blue 'Submit' button.