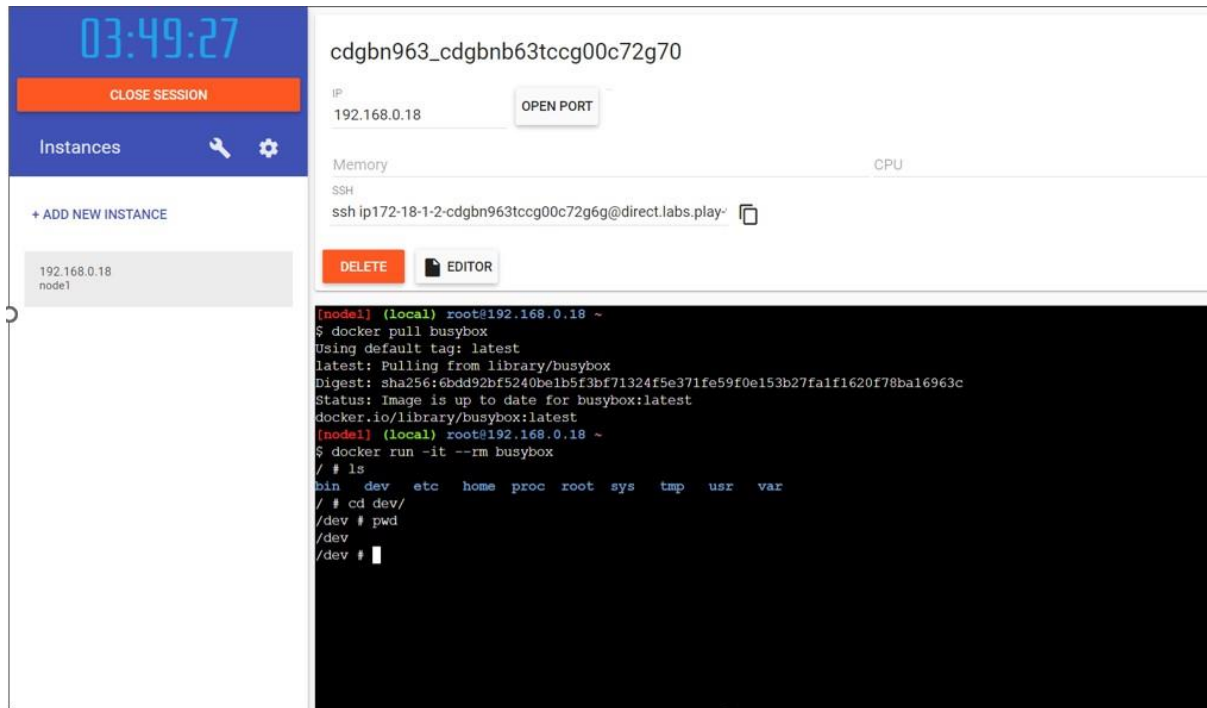


## ASSIGNMENT 4

|                |                      |
|----------------|----------------------|
| PROJECT TITLE  | SKILL JOB RECOMENDER |
| STUDENT NAME   | SHRI VARDHINI.M      |
| STUDENT ROLLNO | 813819104094         |
| MARKS          | 2 MARKS              |

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



The screenshot displays the Docker Playground interface. On the left, a sidebar shows a list of instances with one instance named 'node1' at IP 192.168.0.18. The main panel shows the details of this instance, including its IP, memory, CPU, and SSH access. Below this, a terminal window is open, showing the following commands and output:

```
[node1] (local) root@192.168.0.18 ~
$ docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
Digest: sha256:6bdd92bf5240be1b5f3bf71324f5e371fe59f0e153b27fa1f1620f78ba16963c
Status: Image is up to date for busybox:latest
docker.io/library/busybox:latest
[node1] (local) root@192.168.0.18 ~
$ docker run -it --rm busybox
/ # ls
bin  dev  etc  home  proc  root  sys  tmp  usr  var
/ # cd dev/
/dev # pwd
/dev
/dev #
```

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

```
FROM python:3-alpine3.15
WORKDIR /app
COPY . /app
RUN pip install -r requirements.txt
EXPOSE 5000
CMD python ./app.py
```



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

The screenshot displays the IBM Cloud Container Registry console interface. The left sidebar shows the navigation menu with 'Namespaces' selected. The main area shows the 'Namespaces' page for the 'Global' location. A table lists the namespace 'ibm-sahana' with 0 repositories and 0 images. Below the table, a terminal window shows the execution of Docker commands to push a 'hello-flask' image to the registry.

**Namespaces**

Location: Global

Resource group: Filter... Search

Create +

| Name       | Resource group | Repository count | Image count | Retention policy  |
|------------|----------------|------------------|-------------|-------------------|
| ibm-sahana | Default        | 0                | 0           | Retain all images |

Items per page: 25 1-1 of 1 item 1 1 of 1 page

```
C:\Windows\System32\cmd.exe
Logged in to 'icr.io'.

C:\Users\moham\Downloads\hello-flask-main\hello-flask-main>docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
sahanaparveen/hello-flask  0.0.1.RELEASE      3eae3a73470        39 minutes ago     70.7MB

C:\Users\moham\Downloads\hello-flask-main\hello-flask-main>docker tag sahanaparveen/hello-flask:0.0.1.RELEASE icr.io/ibm-sahana/hello-flask:hello-flask

C:\Users\moham\Downloads\hello-flask-main\hello-flask-main>docker push icr.io/ibm-sahana/hello-flask:hello-flask
The push refers to repository [icr.io/ibm-sahana/hello-flask]
7c79766297f5: Pushing [=====] 20.45MB
775984e51ce4: Pushed
0f6f29d7d129: Pushed
f9a01ea63d59: Pushed
eb71c8b7b3b7: Pushed
76d682e14461: Pushing [=====] 31.35MB/33MB
d59c8eb8f9a4: Pushed
34d5ebaa5410: Pushed
 dialling icr.io:443 no HTTPS proxy: connecting to 169.60.98.86:443: dial tcp 169.60.98.86:443: i/o timeout

C:\Users\moham\Downloads\hello-flask-main\hello-flask-main>docker push sahanaparveen/hello-flask:0.0.1.RELEASE
The push refers to repository [docker.io/sahanaparveen/hello-flask]
7c79766297f5: Pushed
775984e51ce4: Pushed
0f6f29d7d129: Pushed
f9a01ea63d59: Mounted from library/python
eb71c8b7b3b7: Mounted from library/python
76d682e14461: Mounted from library/python
d59c8eb8f9a4: Mounted from library/python
34d5ebaa5410: Mounted from library/python
0.0.1.RELEASE: digest: sha256:4150943baa4076d49dcfd4038739424f957a6b915528da581fec8313e4d44c2d size: 1993

C:\Users\moham\Downloads\hello-flask-main\hello-flask-main>docker push sahanaparveen/hello-flask:0.0.1.RELEASE
The push refers to repository [docker.io/sahanaparveen/hello-flask]
7c79766297f5: Layer already exists
775984e51ce4: Layer already exists
0f6f29d7d129: Layer already exists
f9a01ea63d59: Layer already exists
eb71c8b7b3b7: Layer already exists
76d682e14461: Layer already exists
d59c8eb8f9a4: Layer already exists
34d5ebaa5410: Layer already exists
0.0.1.RELEASE: digest: sha256:4150943baa4076d49dcfd4038739424f957a6b915528da581fec8313e4d44c2d size: 1993

C:\Users\moham\Downloads\hello-flask-main\hello-flask-main>
```

### 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.

WhatsApp mycluster-free - IBM Cloud IBM IBM-Project-17016-1659626790

cloud.ibm.com/kubernetes/clusters/cdj0p7tf0ndbf15jdn0/overview

IBM Cloud Search resources and products... Catalog Manage Sahana parveen M's Ac...

Clusters / mycluster-free

Preparing master, workers... Expires in 30 days Add tags

Overview

Worker nodes

Worker pools

DevOps New

Expires in 30 days: Be sure to back up your data, your cluster will be deleted in 30 days. To access the full capabilities of the service, try out a standard cluster.

Node status 1 of 1 Normal Details

Add-on status 0 of 0 Normal Details

Master status Normal Docs

Ingress status Pending Docs

Details

Cluster ID cdj0p7tf0ndbf15jdn0

Version 1.24.7\_1542

Infrastructure Classic

Zones Milan 01

Created 05/11/2022, 12:41

Resource group Default

Image security enforcement Enable

Node health Worker node details

29°C Cloudy

Help

Log in to your cluster

Deploy your app

1. Set up your image registry Store your images in a registry that the cluster can access to run containers.
2. Deploy your app Kubernetes dashboard
3. Manage your app lifecycle Keep your apps up to date with Kubernetes container orchestration tools.

Expose your app

Add storage to your app

Connect integrations

Install add-ons

Troubleshoot

WhatsApp mycluster-free - IBM Cloud mycluster-free - Kubernetes Dashboard IBM IBM-Project-17016-1659626790

eu-de.containers.cloud.ibm.com/kubeproxy/clusters/cdj0p7tf0ndbf15jdn0/service/#/create?namespace=default

kubernetes default Search

Create

Workloads

Cron Jobs

Daemon Sets

Deployments 11 / 24

Jobs

Pods

Replica Sets

Replication Controllers

Stateful Sets

Service

Ingresses

Ingress Classes

Services

Config and Storage

Config Maps

Persistent Volume Claims

Secrets

Create from input Create from file Create from form

App name \* hello-flask

Container image \* sahanaparveen/hello-flask:0.0.1.RELEASE

Number of pods \* 1

Service \* External

Port \* 5000 Target port \* 5000 Protocol \* TCP

Namespace \* default

Deploy Preview Cancel Show advanced options

An 'app' label with this value will be added to the Deployment and Service that get deployed. Learn more

Enter the URL of a public image on any registry, or a private image hosted on Docker Hub or Google Container Registry. Learn more

A Deployment will be created to maintain the desired number of pods across your cluster. Learn more

Optionally, an internal or external Service can be defined to map an incoming Port to a target Port seen by the container. Learn more

Namespaces let you partition resources into logically named groups. Learn more

29°C Cloudy

The screenshot shows the Kubernetes Dashboard interface. On the left, a sidebar lists various workload types under 'Workloads' and 'Service' sections. The 'Deployments' link is selected. The main content area, titled 'Deployments', shows a table with one deployment named 'hello-flask'. The table has columns for Name, Images, Labels, Pods, and Created. The 'hello-flask' deployment is in a ready state (green dot) and has 1 pod running. The bottom of the image shows a Windows taskbar with the date and time as 05-11-2022, 13:10.

| Name        | Images   | Labels   | Pods  | Created        |
|-------------|----------|----------|-------|----------------|
| hello-flask | Show all | Show all | 1 / 1 | 48 seconds ago |