ASSIGNMENT 4

NAME: K.R.MALAVIKA

REGISTER NUMBER: 311519104033

PROJECT TOPIC: CUSTOMER CARE REGISTRY

COLLEGE NAME: MEENAKSHI SUNDARARAJAN ENGINEERING COLLEGE

TEAM NUMBER: PNT2022TMID27825

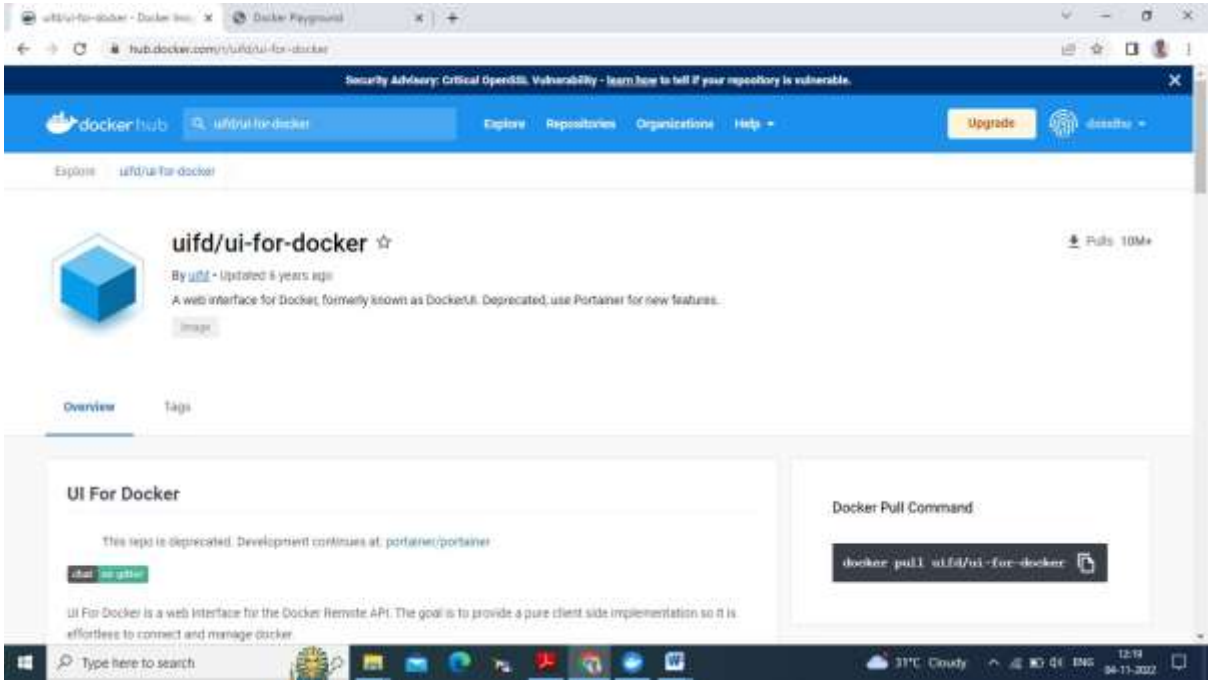
1)PULL AN IMAGE FROM DOCKER HUB AND RUN IT IN DOCKER BACKGROUND

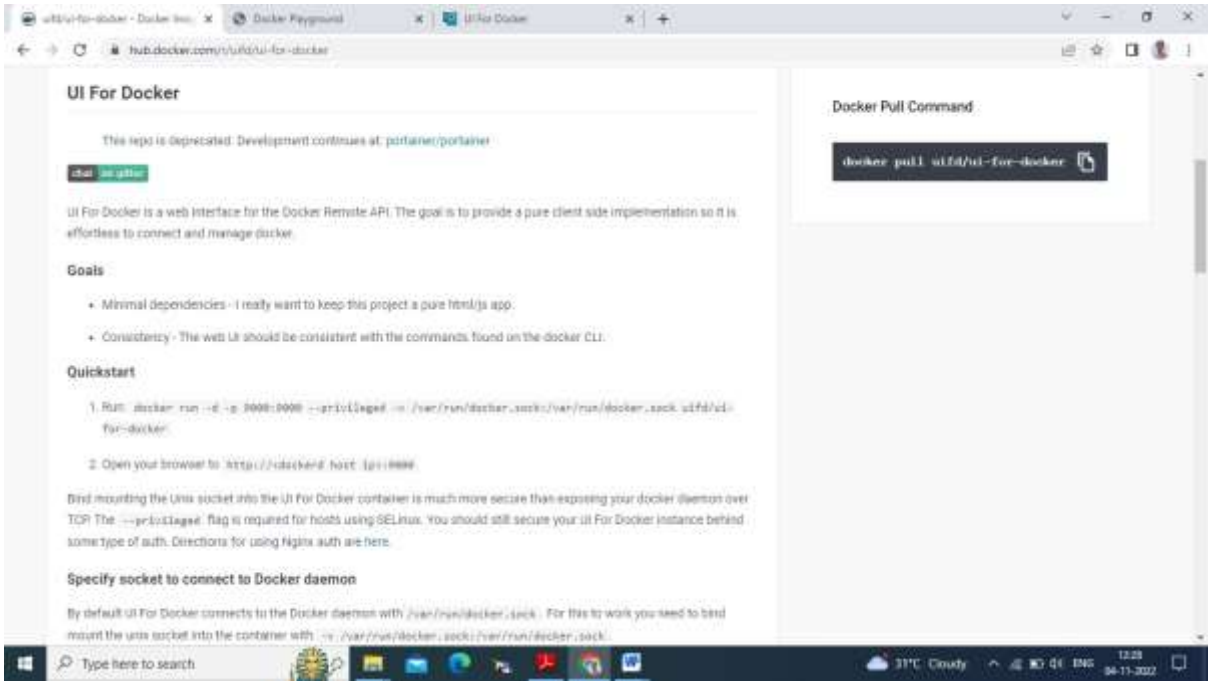
COMMANDS:

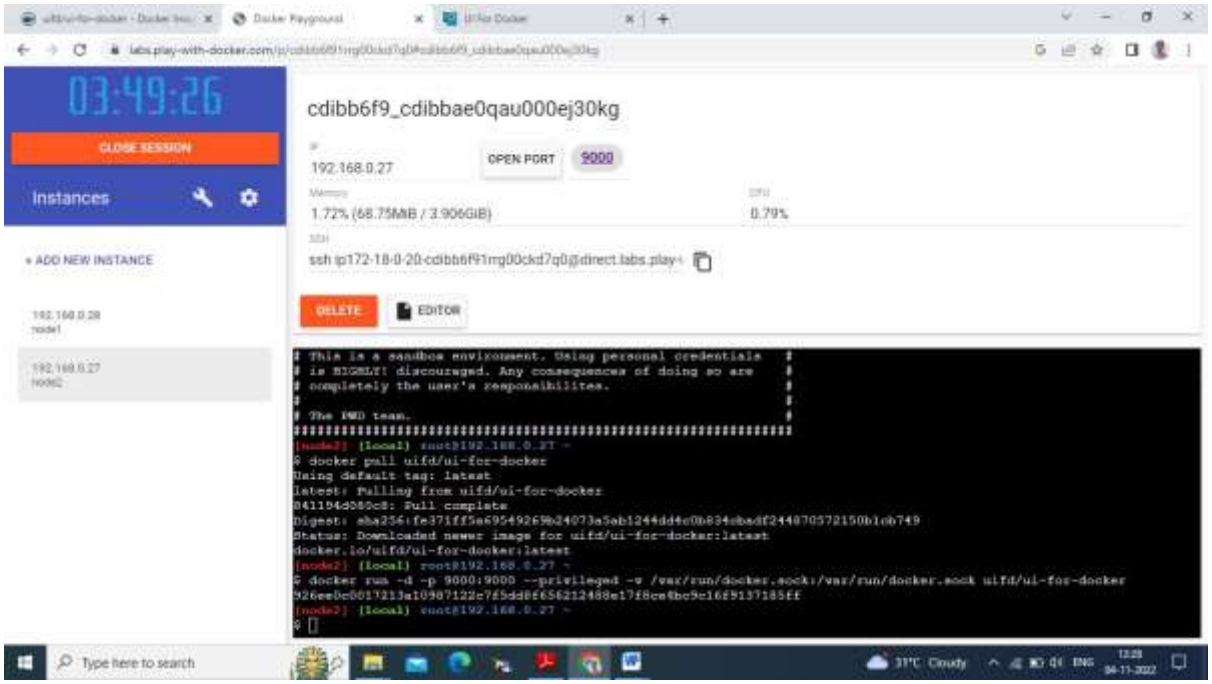
docker pull uifd/ui-for-docker

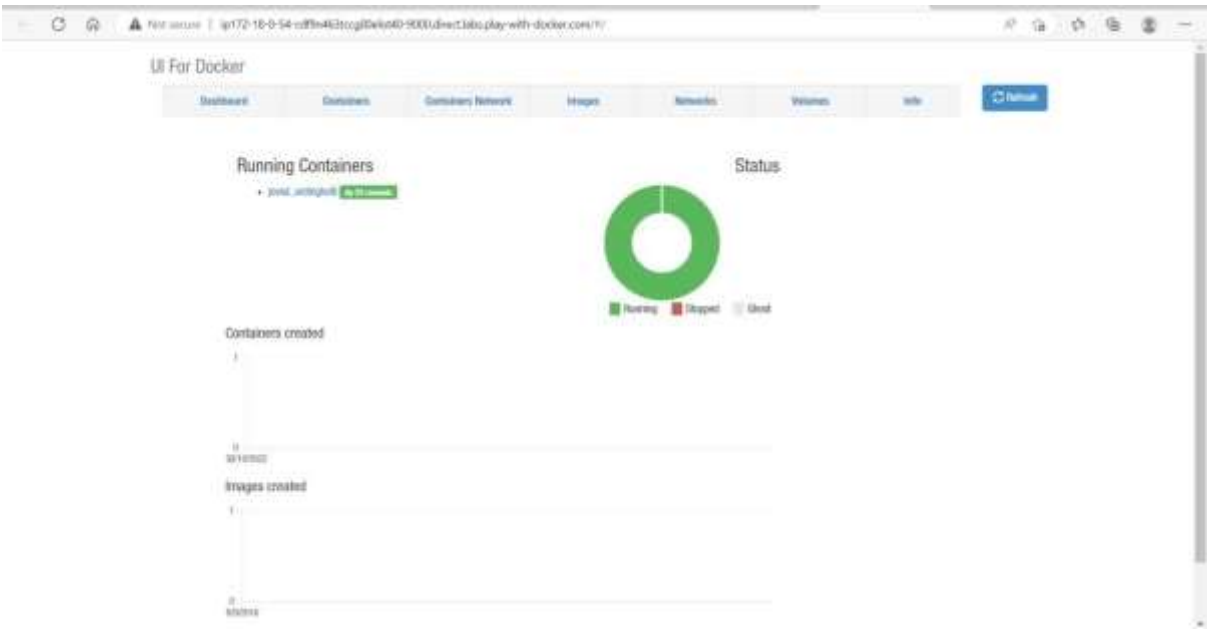
docker run -d -p 9000:9000 --privileged -v /var/run/docker.sock:/var/run/docker.sock uifd/ui-for-docker

OUTPUT:




UI For Docker
This repo is deprecated. Development continues at: portainer/portainer
UI For Docker is a web interface for the Docker Remote API. The goal is to provide a pure client side implementation so it is effortless to connect and manage docker.
Goals
Minimal dependencies - I really want to keep this project a pure html/js app.
Consistency - The web UI should be consistent with the commands found on the docker CLI.
Quickstart
Bind mounting the Unix socket into the UI For Docker container is much more secure than exposing your docker daemon over TCP. The --privileged flag is required for hosts using SELinux. You should still secure your UI For Docker instance behind some type of auth. Directions for using Nginx auth are here.
Specify socket to connect to Docker daemon
Docker Pull Command
docker pull uifd/ui-for-docker
Type here to search


03:49:26
CLOSE SESSION
Instances
+ ADD NEW INSTANCE
192.168.0.28
node1
192.168.0.27
node2
cdibb6f9_cdibbae0qau000ej30kg
192.168.0.27
OPEN PORT
9000
1.72% (68.75MiB / 3.906GiB)
0.79%
ssh ip172-18-0-20-cdibb6f91rrg00ckd7q0@direct.labs.play-
DELETE
EDITOR
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.
[node2] (local) root@192.168.0.27 ~
$ docker pull uifd/ui-for-docker
Using default tag: latest
latest: Pulling from uifd/ui-for-docker
Status: Downloaded newer image for uifd/ui-for-docker:latest
docker.io/uifd/ui-for-docker:latest
[node2] (local) root@192.168.0.27 ~
$ docker run -d -p 9000:9000 --privileged -v /var/run/docker.sock:/var/run/docker.sock uifd/ui-for-docker
[node2] (local) root@192.168.0.27 ~
Type here to search



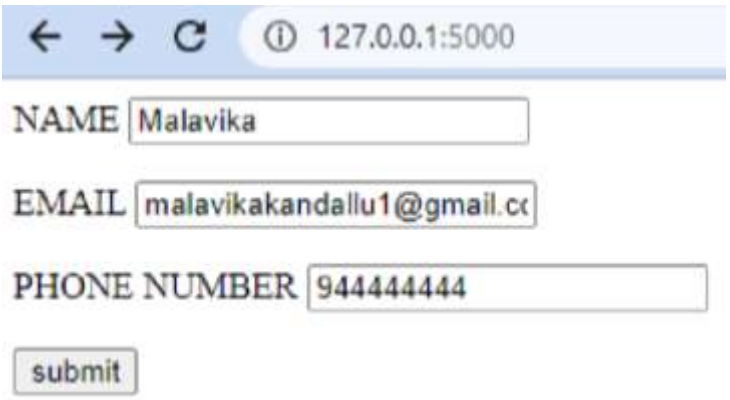
2)CREATE A DOCKER FILE FOR THE JOB PORTAL APPLICATION AND DEPLOY IT IN DOCKER DESKTOP APPLICATION.

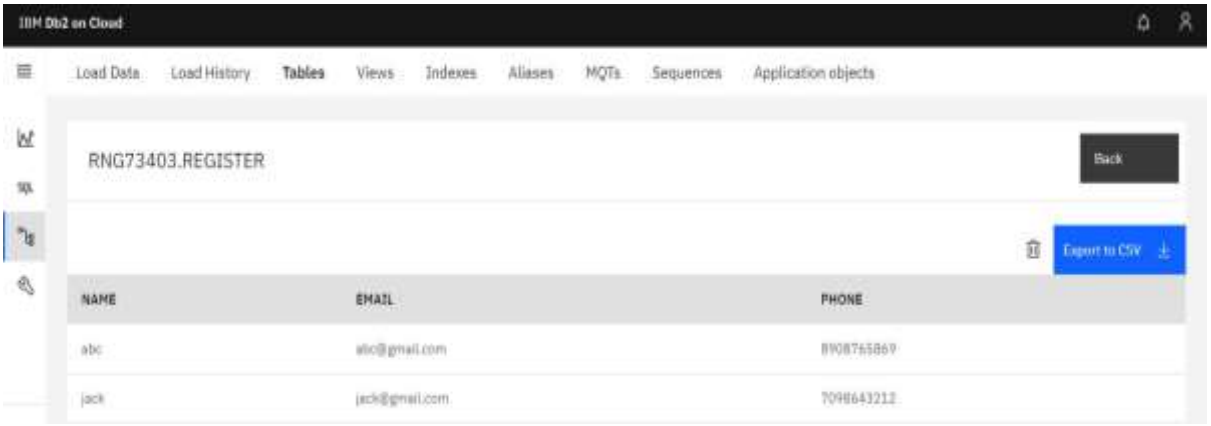
COMMANDS:

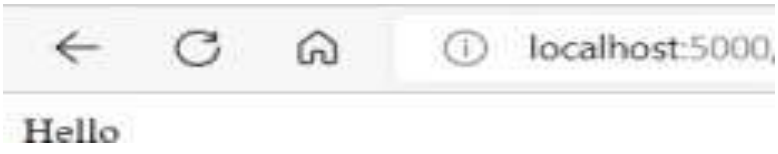
docker build -t jp1 .

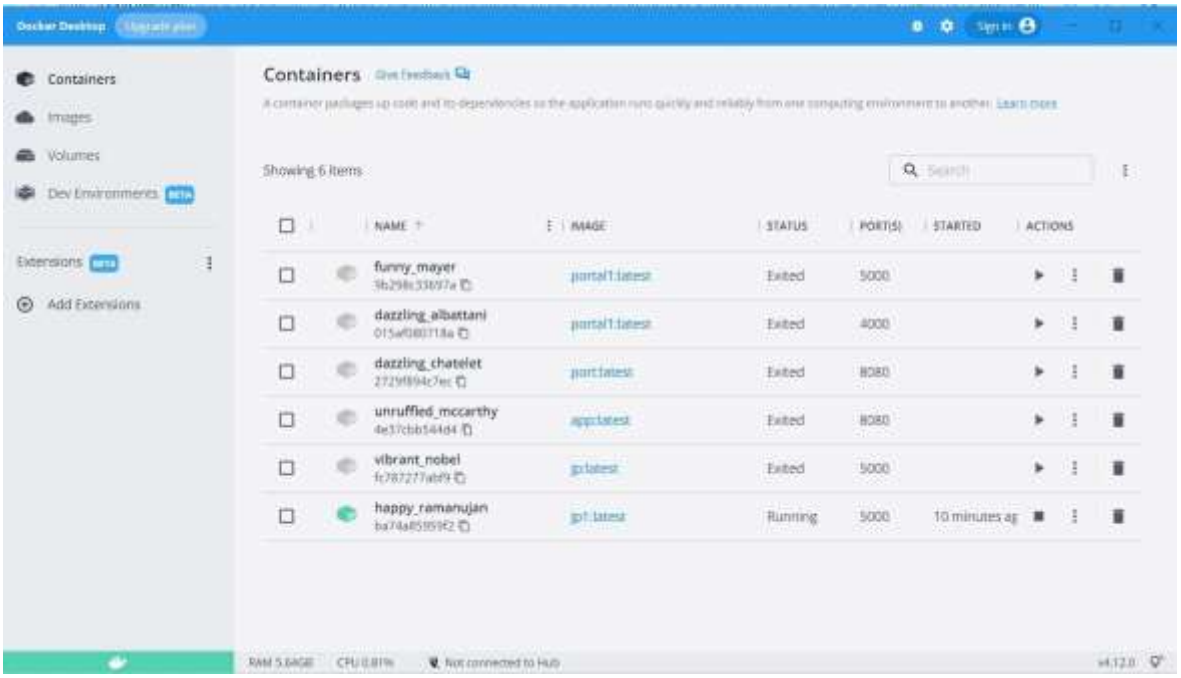
docker run -p 5000:5000 jp1

OUTPUT:

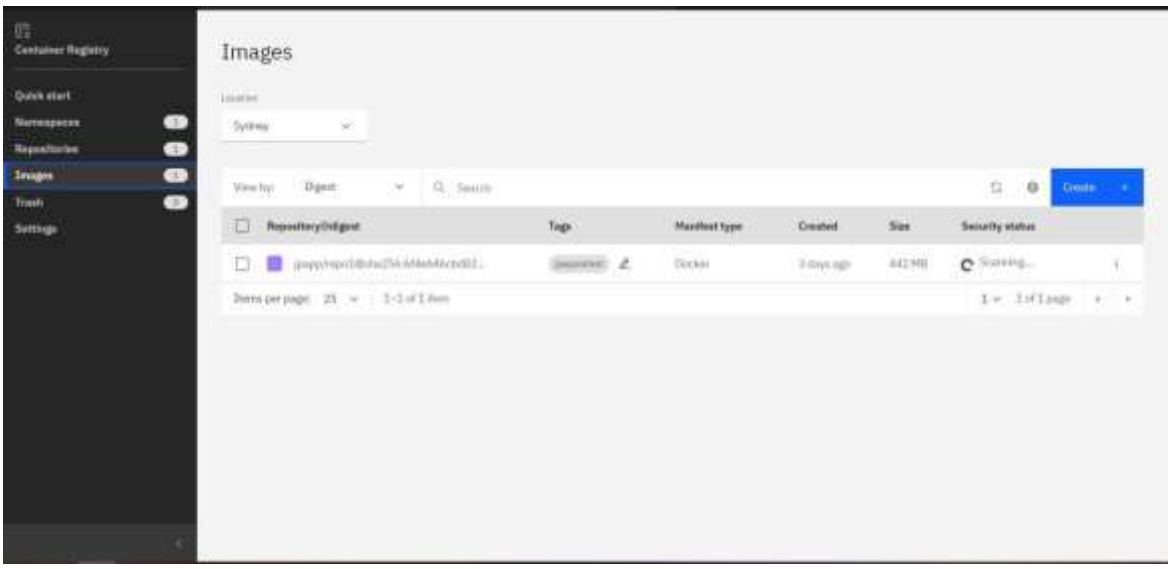

127.0.0.1:5000
NAME
Malavika
EMAIL
malavikakandallu1@gmail.cc
PHONE NUMBER
944444444
submit

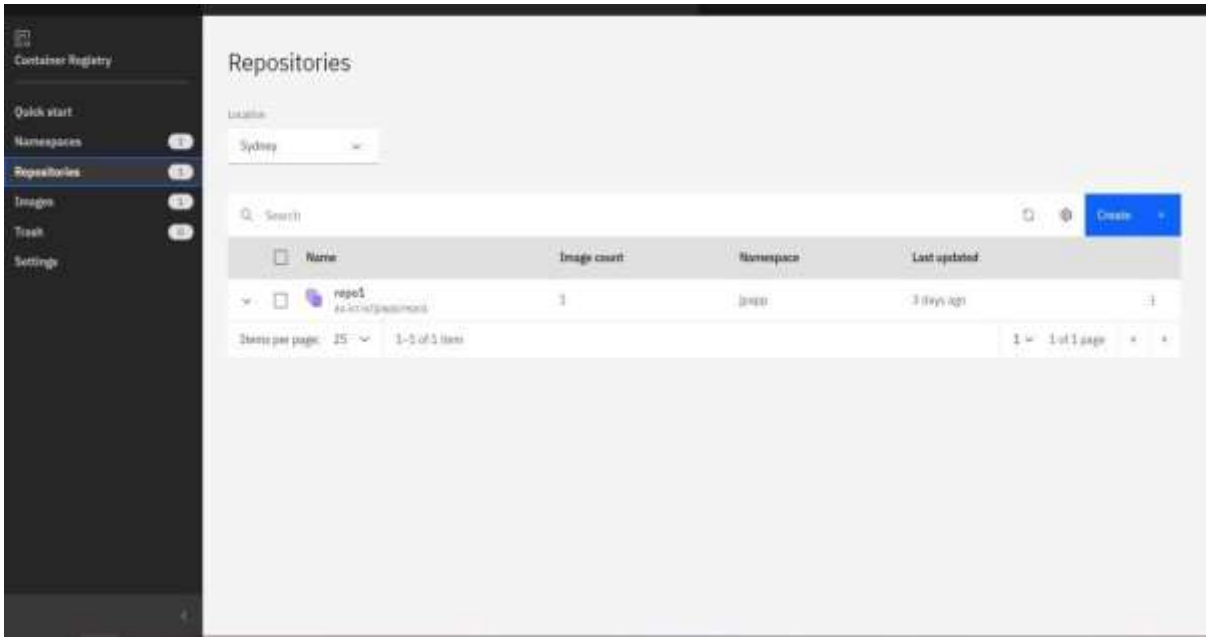

IBM Db2 on Cloud
Load Data
Load History
Tables
Views
Indexes
Aliases
MQTs
Sequences
Application objects
RNG73403.REGISTER
Back
Export to CSV
NAME
EMAIL
PHONE
abc
abc@gmail.com
8908765869
jack
jack@gmail.com
7098643212

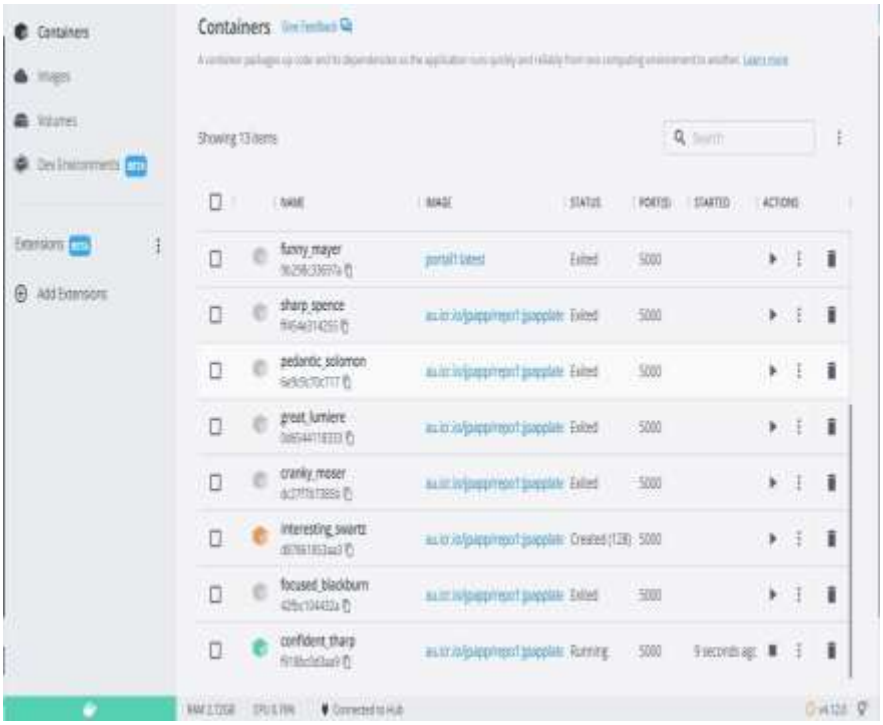

localhost:5000,
Hello

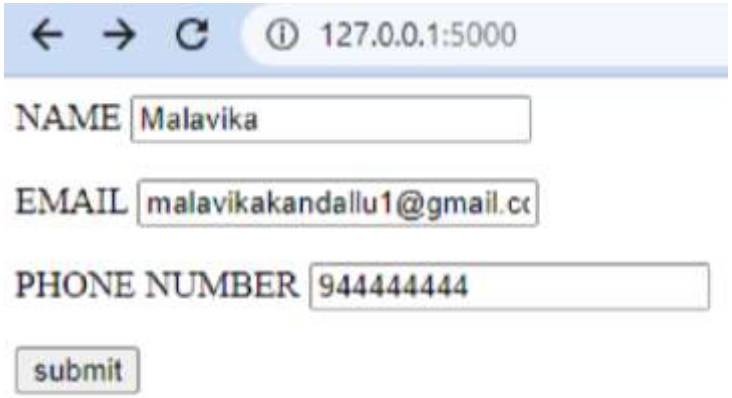

Containers
Images
Volumes
Dev Environments BETA
Extensions BETA
Add Extensions
Containers
Showing 6 items
NAME
IMAGE
STATUS
PORT(S)
STARTED
ACTIONS
funny_mayer
dazzling_albattani
dazzling_chatelet
unruffled_mccarthy
vibrant_nobel
happy_ramanujan
Exited
Exited
Exited
Exited
Exited
Running
5000
4000
8080
8080
5000
5000
10 minutes ag

3)CREATE AN IBM CONTAINER REGISTRY AND DEPLOY HELLOWORLD APP OR JOBPORTAL APP



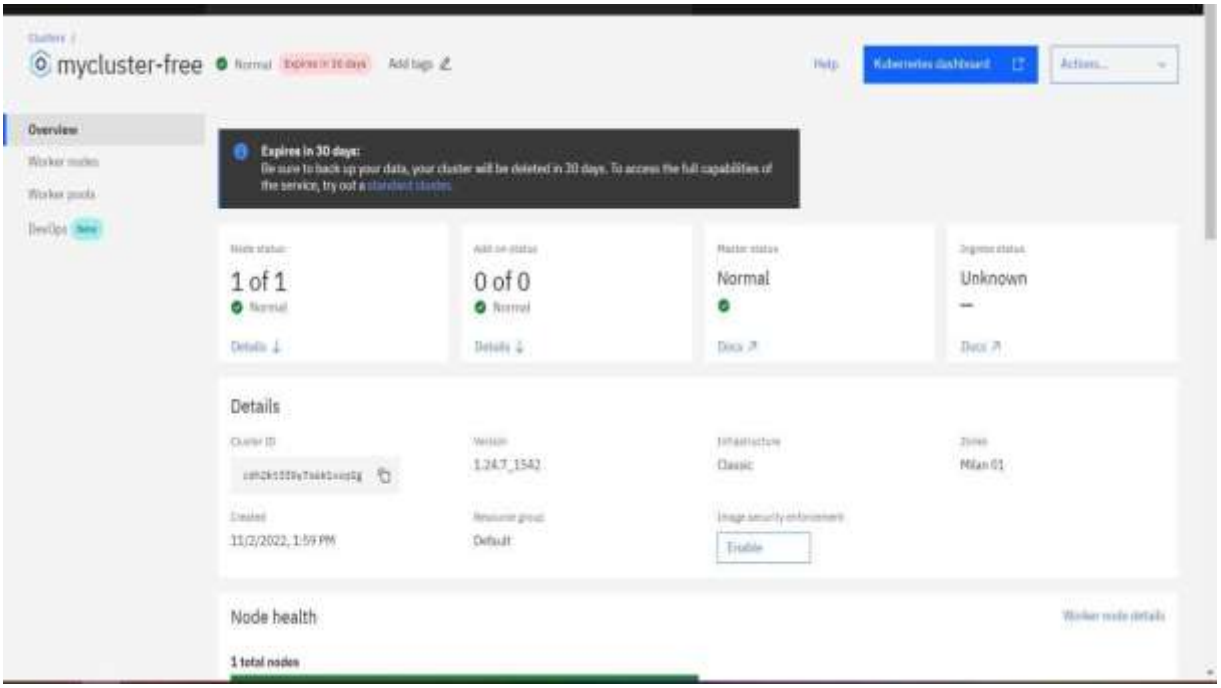




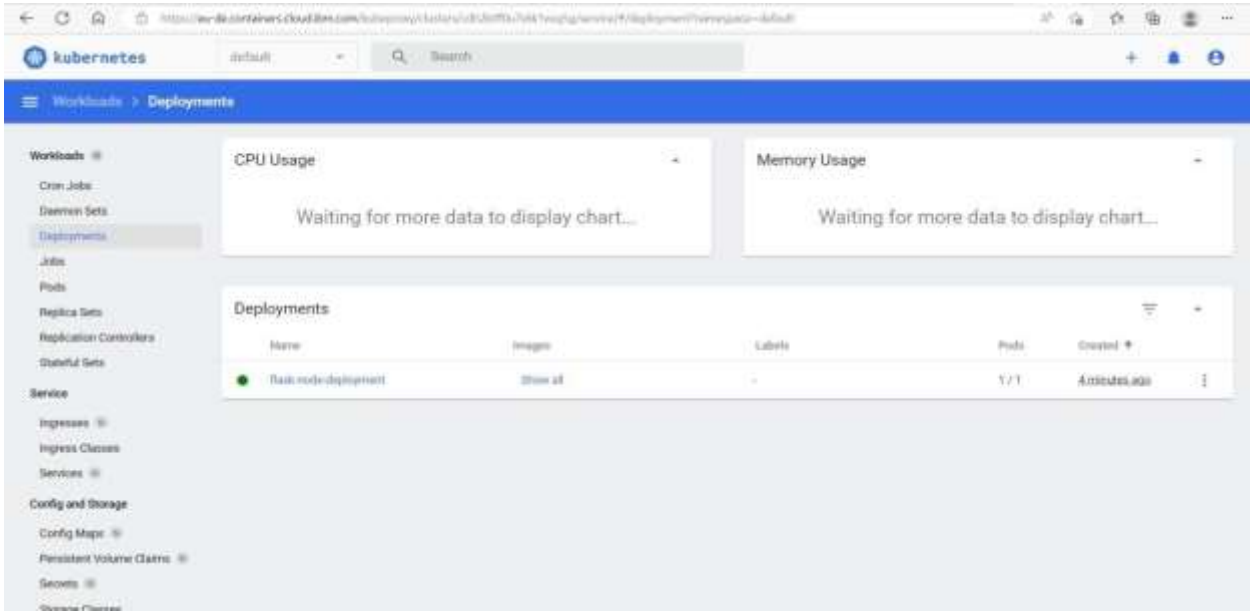


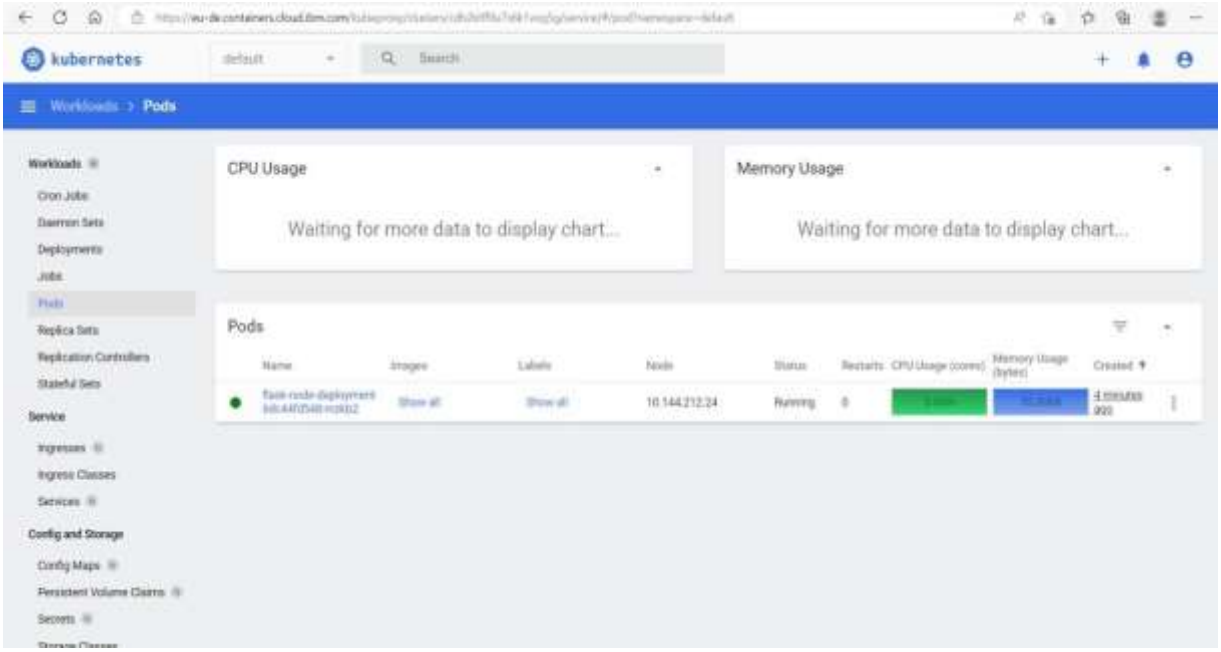
4) CREATE A KUBERNETES CLUSTER IN IBM CLOUD AND DEPLOY HELLO WORLD IMAGE OR JOB PORTAL IMAGE AND ALSO EXPOSE THE SAME APP TO RUN IN A NODEPORT.

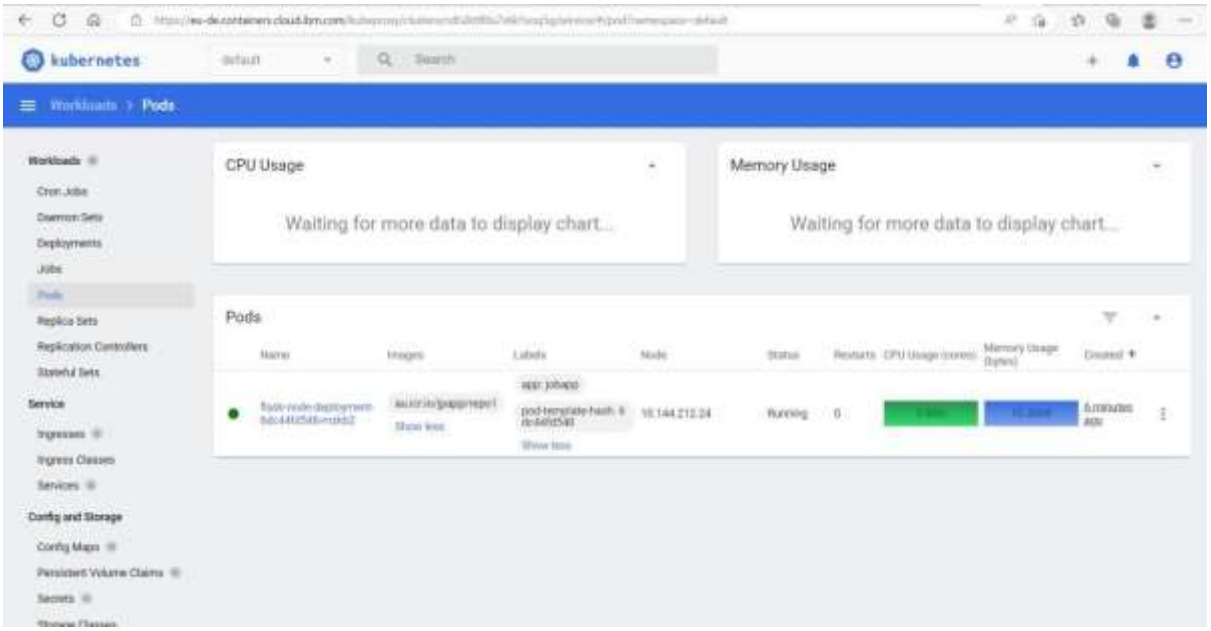
KUBERNETES CLUSTER CREATION:

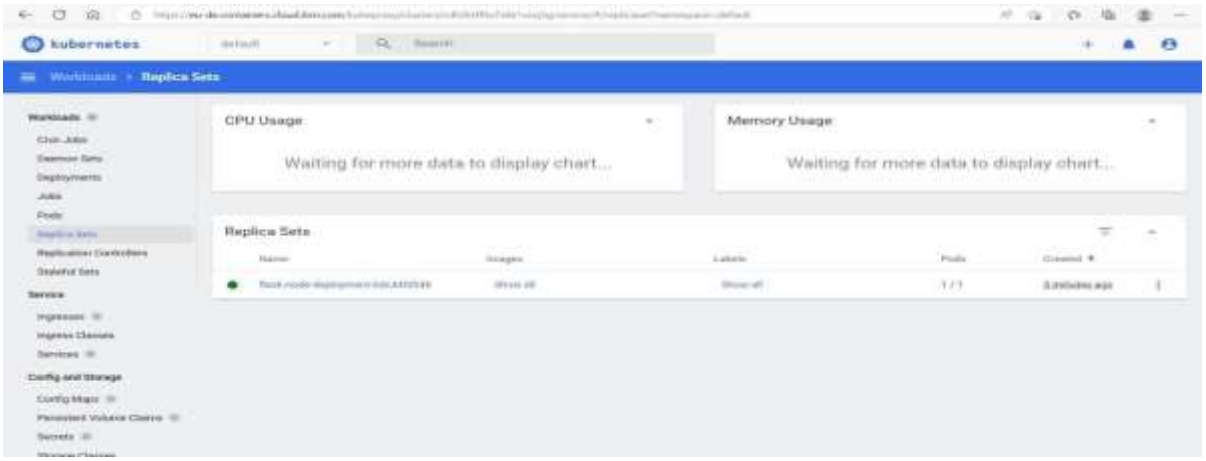


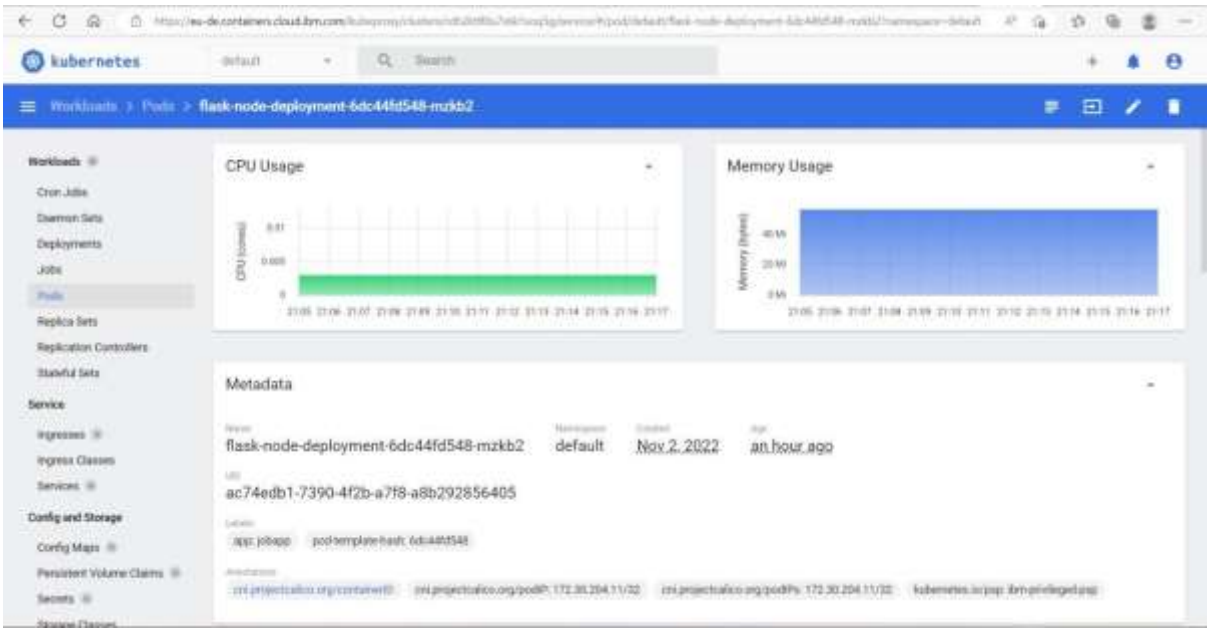
DEPLOYMENT OF JOB PORTAL APP IN CLUSTER




kubernetes
default
Search
Workloads > Pods
Workloads
Cron Jobs
Daemon Sets
Deployments
Jobs
Pods
Replica Sets
Replication Controllers
Stateful Sets
Service
Ingresses
Ingress Classes
Services
Config and Storage
Config Maps
Persistent Volume Claims
Secrets
CPU Usage
Waiting for more data to display chart...
Memory Usage
Waiting for more data to display chart...
Pods
Name
Images
Labels
Node
Status
Restarts
CPU Usage (cores)
Memory Usage (bytes)
Created
Running
10.144.212.24

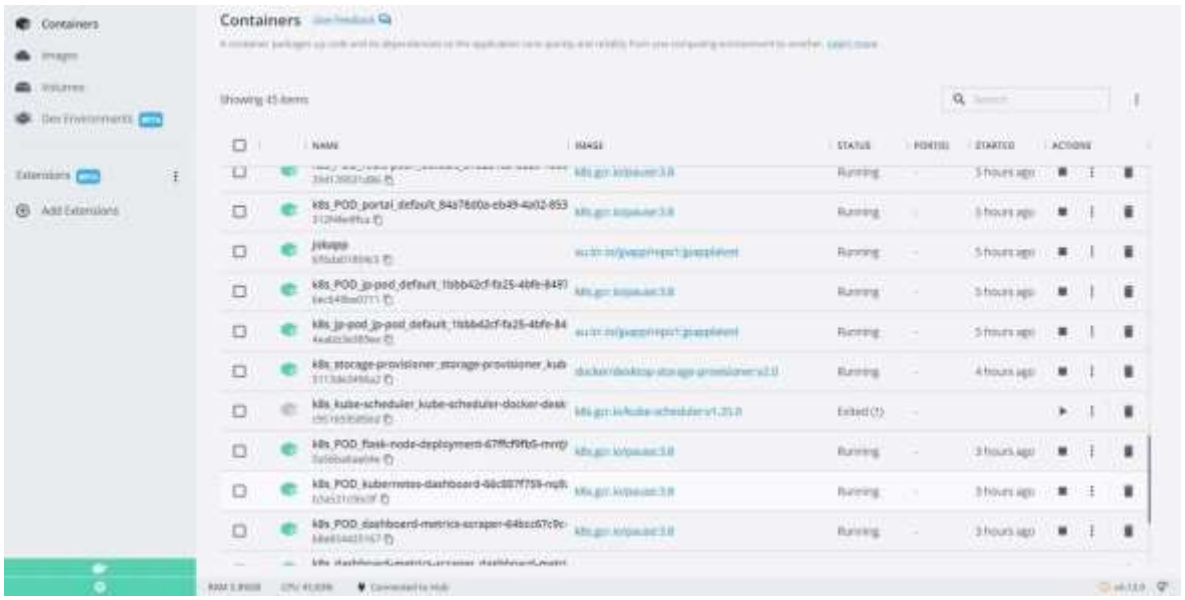

kubernetes
default
Search
Workloads > Pods
Workloads
Cron Jobs
Daemon Sets
Deployments
Jobs
Pods
Replica Sets
Replication Controllers
Stateful Sets
Service
Ingresses
Ingress Classes
Services
Config and Storage
Config Maps
Persistent Volume Claims
Secrets
CPU Usage
Waiting for more data to display chart...
Memory Usage
Waiting for more data to display chart...
Pods
Name
Images
Labels
Node
Status
Restarts
CPU Usage (cores)
Memory Usage (bytes)
Created
app: jobapp
Running
10.144.212.24


kubernetes
Workloads > Replica Sets
Workloads
Cron Jobs
Daemon Sets
Deployments
Jobs
Pods
Replica Sets
Replication Controllers
Stateful Sets
Service
Ingresses
Ingress Classes
Services
Config and Storage
Config Maps
Persistent Volume Claims
Secrets
CPU Usage
Waiting for more data to display chart...
Memory Usage
Waiting for more data to display chart...
Replica Sets




```
Name:              flask-node-deployment
Namespace:         default
Labels:            <none>
Annotations:       <none>
Selector:          app=flasknode
Type:              ClusterIP
IP Family Policy:  SingleStack
IP Families:       IPv4
IP:                172.21.233.153
IPs:               172.21.233.153
Port:              <unset>  5000/TCP
TargetPort:        5000/TCP
Endpoints:         <none>
Session Affinity:  None
Events:            <none>
```

```
NAME                 TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
flask-node-deployment   ClusterIP   172.21.233.153   <none>        5000/TCP   68m
kubernetes           ClusterIP   172.21.0.1       <none>        443/TCP    6h58m
Name:              flask-node
Namespace:         default
Labels:            <none>
Annotations:       <none>
Selector:          app=jobapp
Type:              ClusterIP
IP Family Policy:  SingleStack
IP Families:       IPv4
IP:                172.21.30.184
IPs:               172.21.30.184
Port:              <unset>  80/TCP
TargetPort:        8000/TCP
Endpoints:         172.30.204.11:8000
Session Affinity:  None
Events:            <none>
```

EXPOSE APP TO NODEPORT:

```
Name:                     flask
Namespace:                default
Labels:                   <none>
Annotations:              <none>
Selector:                 app=jobapp
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       172.21.15.156
IPs:                      172.21.15.156
Port:                     <unset>  80/TCP
TargetPort:               8000/TCP
NodePort:                 <unset>  32153/TCP
Endpoints:                172.30.204.11:8000
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
```