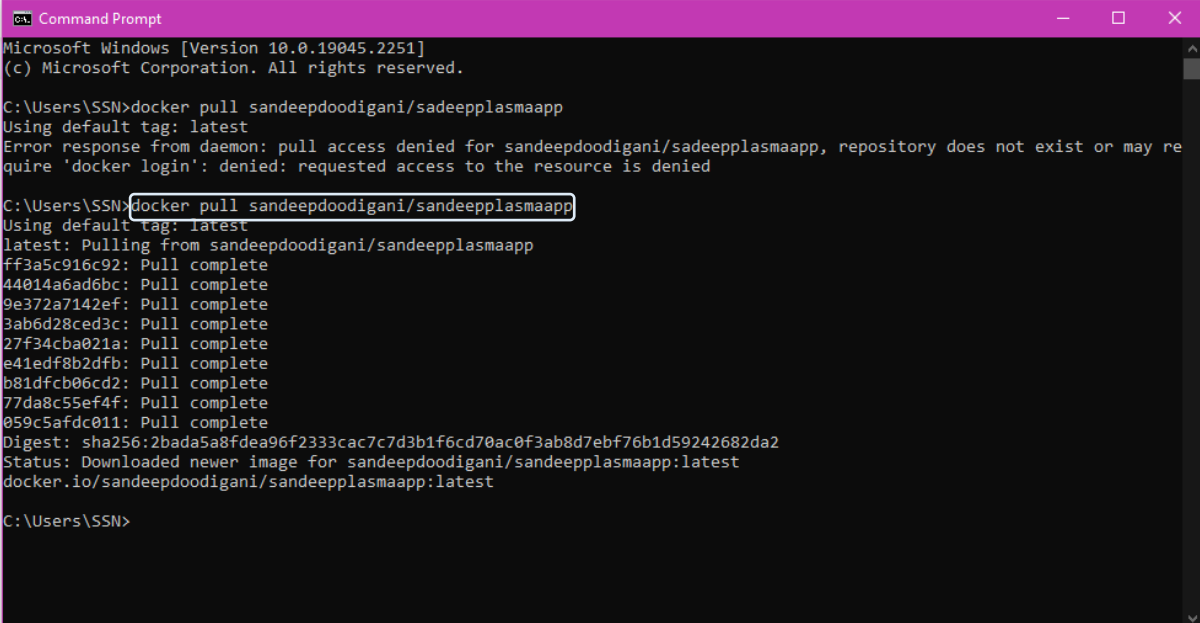


Name	PRASANTH A
Roll No	SSNCE195001080
Date	14 November 2022
Team ID	PNT2022TMID53036
Project Name	Project - Personal Expense Tracker

## Assignment - 4

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



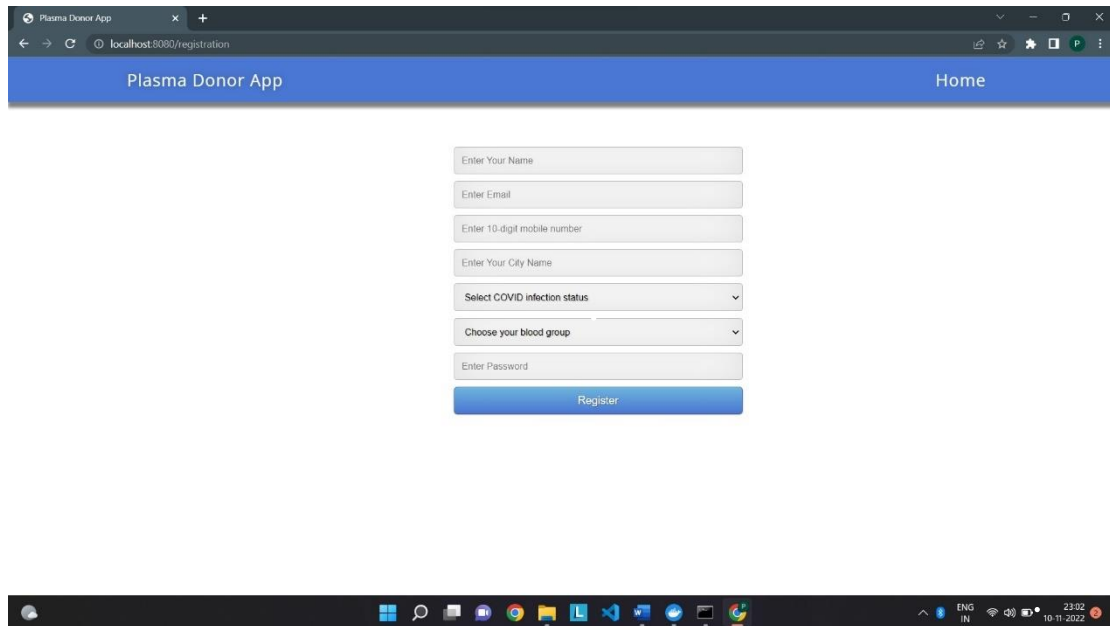
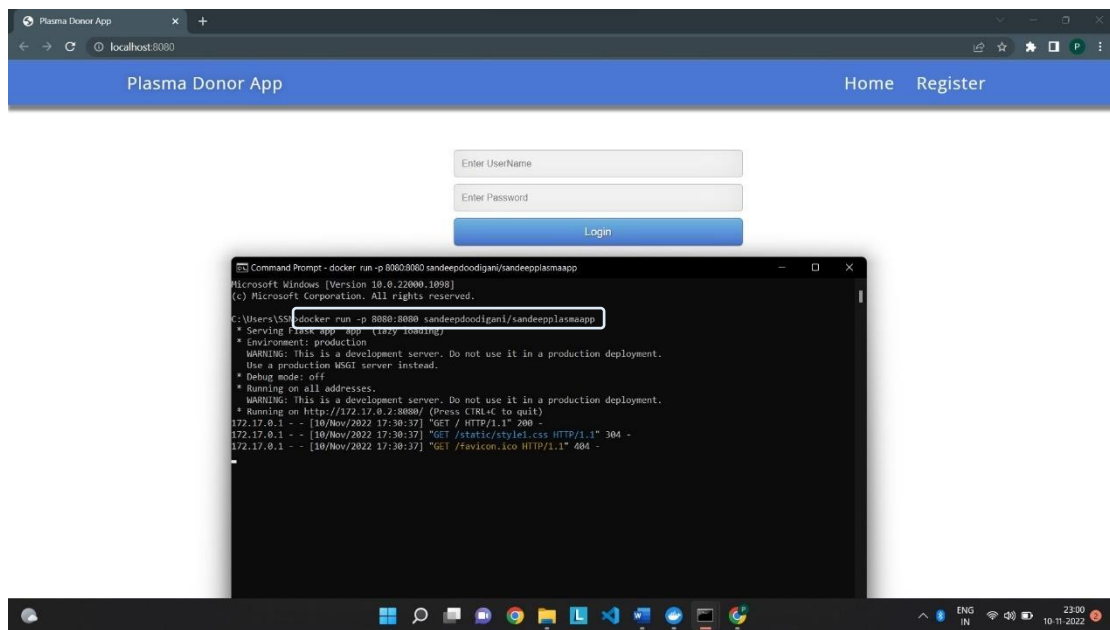
```

C:\Users\SSN>docker pull sandeepdoodigani/sadeepplasmaapp
Using default tag: latest
Error response from daemon: pull access denied for sandeepdoodigani/sadeepplasmaapp, repository does not exist or may require 'docker login': denied: requested access to the resource is denied

C:\Users\SSN>docker pull sandeepdoodigani/sadeepplasmaapp
Using default tag: latest
latest: Pulling from sandeepdoodigani/sadeepplasmaapp
ff3a5c916c92: Pull complete
44014a6ad6bc: Pull complete
9e372a7142ef: Pull complete
3ab6d28ced3c: Pull complete
27f34cba021a: Pull complete
e41edf8b2dfb: Pull complete
b81dfcb06cd2: Pull complete
77da8c55ef4f: Pull complete
059c5afdc011: Pull complete
Digest: sha256:2bada5a8fdea96f2333cac7c7d3b1f6cd70ac0f3ab8d7ebf76b1d59242682da2
Status: Downloaded newer image for sandeepdoodigani/sadeepplasmaapp:latest
docker.io/sandeepdoodigani/sadeepplasmaapp:latest

C:\Users\SSN>

```



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

**Program :**

Dockerfile:

```
FROM python:3.6

WORKDIR /app

ADD . /app

COPY requirements.txt /app

RUN python3 -m pip install -r requirements.txt

RUN python3 -m pip install ibm_db

EXPOSE 5000

CMD ["python", "app.py"]
```

Requirements.txt

```
Flask

ibm_db

sendgrid
```

App.py

```
from flask import Flask, render_template, request, redirect, url_for, session

import ibm_db

import re
```

```
app = Flask(__name__)
```

```
app.secret_key = 'a'
```

```
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=b70af05b-76e4-4bca-a1f5-
```

```
23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32716;SECURITY=SSL;SSLServerCertificate=DigiCe
```

```
rtGlobalRootCA.crt;UID=jzc43091;PWD=PI8VtGRvZISVT65A";,")
```

```
@app.route("/")
```

```
def homer():
```

```
    return render_template("home.html")
```

```
@app.route('/login',methods=['GET', 'POST'])
```

```
def login():
```

```
    global userid
```

```
    msg = "
```

```
    if request.method == 'POST' :
```

```
        username = request.form['username']
```

```
        password = request.form['password']
```

```
        sql = "SELECT * FROM users WHERE username=? AND password=?"
```

```
        stmt = ibm_db.prepare(conn, sql)
```

```
        ibm_db.bind_param(stmt,1,username)
```

```

    ibm_db.bind_param(stmt,2,password)

    ibm_db.execute(stmt)

    account = ibm_db.fetch_assoc(stmt)

    print (account)

    if account:

        session['loggedin'] = True

        session['id'] = account['USERNAME']

        userid= account['USERNAME']

        session['username'] = account['USERNAME']

        msg = 'Logged in successfully !'

        msg = 'Logged in successfully !'

        return render_template('dashboard.html', msg = msg)

    else:

        msg = 'Incorrect username / password !'

    return render_template('login.html', msg = msg)

```

---

```

@app.route('/register', methods=['GET', 'POST'])

def registet():

    msg = ""

    if request.method == 'POST' :

        username = request.form['username']

```

```
email = request.form['email']

password = request.form['password']

sql = "SELECT * FROM users WHERE username =?"

stmt = ibm_db.prepare(conn, sql)

ibm_db.bind_param(stmt, 1, username)

ibm_db.execute(stmt)

account = ibm_db.fetch_assoc(stmt)

print(account)

if account:

    msg = 'Account already exists !'

elif not re.match(r'[^@ ]+@[^@ ]+\.[^@ ]+', email):

    msg = 'Invalid email address !'

elif not re.match(r'[A-Za-z0-9]+', username):

    msg = 'name must contain only characters and numbers !'

else:

    insert_sql = "INSERT INTO users VALUES (?, ?, ?)"

    prep_stmt = ibm_db.prepare(conn, insert_sql)

    ibm_db.bind_param(prepare_stmt, 1, username)

    ibm_db.bind_param(prepare_stmt, 2, email)

    ibm_db.bind_param(prepare_stmt, 3, password)

    ibm_db.execute(prepare_stmt)

    msg = 'You have successfully registered !'

elif request.method == 'POST':

    msg = 'Please fill out the form !'
```

```
return render_template('register.html', msg = msg)
```

```
@app.route('/dashboard')
```

```
def dash():
```

```
return render_template('dashboard.html')
```

```
@app.route('/apply',methods =['GET', 'POST'])
```

```
def apply():
```

```
    msg = "
```

```
    if request.method == 'POST' :
```

```
        username = request.form['username']
```

```
        email = request.form['email']
```

```
        qualification= request.form['qualification']
```

```
        skills = request.form['skills']
```

```
        jobs = request.form['s']
```

```
        sql = "SELECT * FROM users WHERE username =?"
```

```
        stmt = ibm_db.prepare(conn, sql)
```

```
        ibm_db.bind_param(stmt,1,username)
```

```
        ibm_db.execute(stmt)
```

```
        account = ibm_db.fetch_assoc(stmt)
```

```
        print(account)
```

```
        if account:
```

```
            msg = 'there is only 1 job position! for you'
```

```
return render_template('apply.html', msg = msg)
```

```
insert_sql = "INSERT INTO job VALUES (?, ?, ?, ?, ?)"

prep_stmt = ibm_db.prepare(conn, insert_sql)

ibm_db.bind_param(prepare_stmt, 1, username)

ibm_db.bind_param(prepare_stmt, 2, email)

ibm_db.bind_param(prepare_stmt, 3, qualification)

ibm_db.bind_param(prepare_stmt, 4, skills)

ibm_db.bind_param(prepare_stmt, 5, jobs)

ibm_db.execute(prepare_stmt)

msg = 'You have successfully applied for job !'

session['loggedin'] = True

TEXT = "Hello,a new application for job position" +jobs+"is requested"

elif request.method == 'POST':

    msg = 'Please fill out the form !'

return render_template('apply.html', msg = msg)
```

```
@app.route('/display')
```



```
def display():

    print(session["username"],session['id'])


    cursor = mysql.connection.cursor()

    cursor.execute('SELECT * FROM job WHERE userid = % s', (session['id'],))

    account = cursor.fetchone()

    print("accountdisplay",account)
```

```
return render_template('display.html',account = account)
```

```
@app.route('/logout')
```

```
def logout():

    session.pop('loggedin', None)

    session.pop('id', None)

    session.pop('username', None)

    return render_template('home.html')
```

```
if __name__ == '__main__':

    app.run(host='0.0.0.0')
```

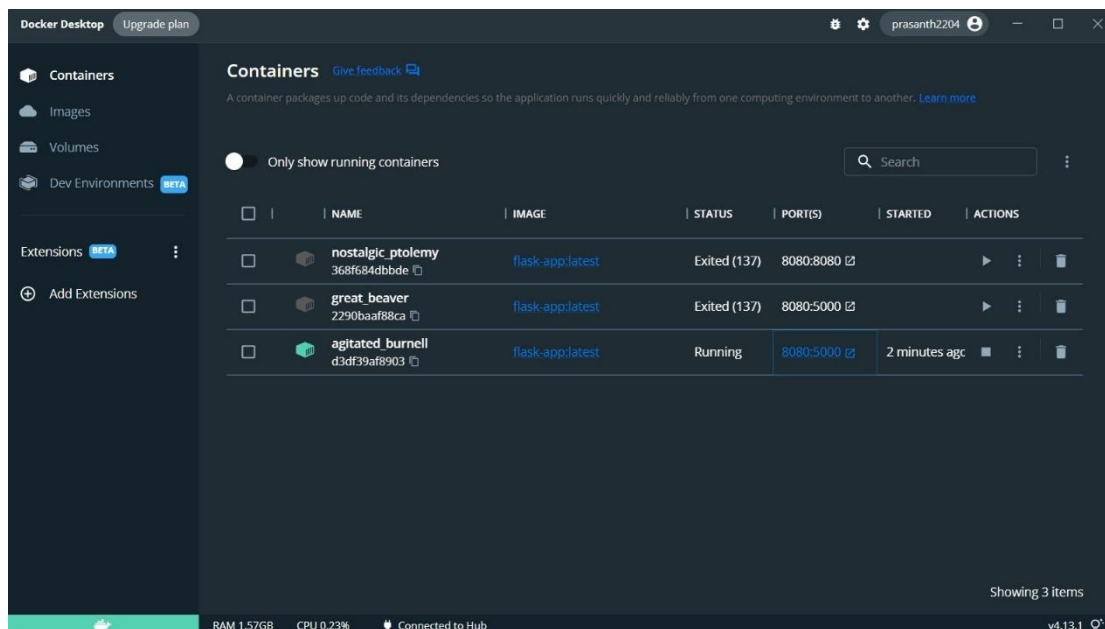
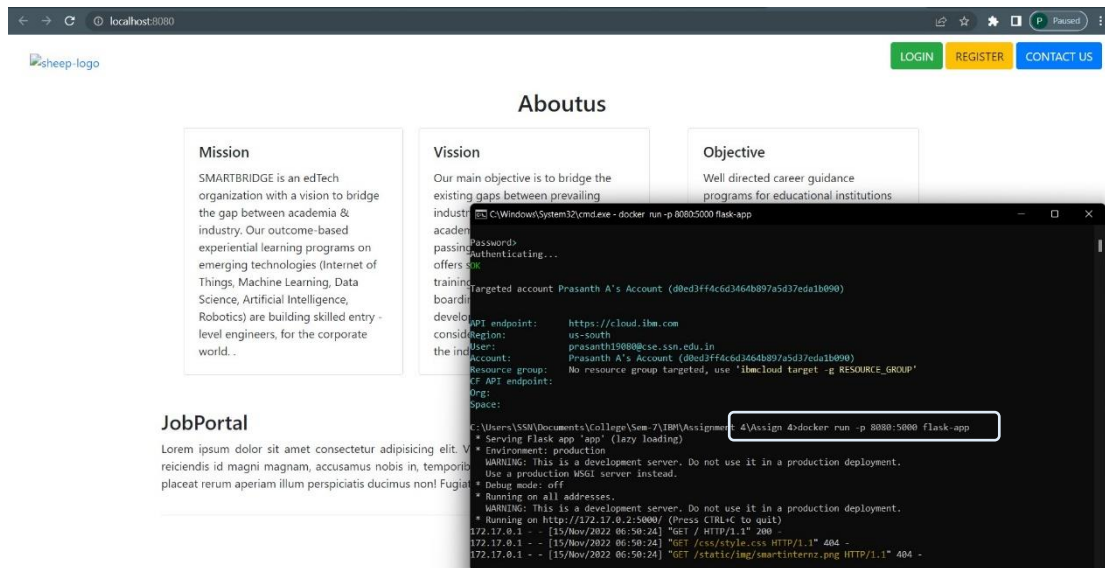
```
C:\Windows\System32\cmd.exe

C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign 4>docker build -t flask-app .
[+] Building 284.9s (12/12) FINISHED
=> [internal] load build definition from Dockerfile 0.1s
=> => transferring dockerfile: 229B 0.1s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/python:3.6 3.7s
=> [auth] library/python:pull token for registry-1.docker.io 0.0s
=> [1/6] FROM docker.io/library/python:3.6@sha256:f8652afaf88c25f0d22354d547d892591067aa4026a7fa9a6819df9f300af6 68.2s
=> => resolve docker.io/library/python:3.6@sha256:f8652afaf88c25f0d22354d547d892591067aa4026a7fa9a6819df9f300af6 0.0s
=> => sha256:f8652afaf88c25f0d22354d547d892591067aa4026a7fa9a6819df9f300af6f6c 1.86kB / 1.86kB 0.0s
=> => sha256:d0974d907a8ec079df5ac31872359c2de510f82214c0448e926393b376d3b60d 2.22kB / 2.22kB 0.0s
=> => sha256:54260638d07c5e3ad24c6e21fc889abb8486a27634c0892086ff71f3f44b104 9.27kB / 9.27kB 0.0s
=> => sha256:0e29546d541cddb309281d21a73a9d1db78665c1b95b74f32b009e0b77a6e1e3 54.92MB / 54.92MB 35.1s
=> => sha256:9b829c73b52b92b97d5c07a54fb0f3e921995a296c714b53a32ae67d19231fcd 5.15MB / 5.15MB 1.7s
=> => sha256:cb5b7ae361722f070eca53f35823ed21baa85d61d5d95cd5a95ab53d740cdd56 10.87MB / 10.87MB 2.3s
=> => sha256:6494e4811622b31c027ccac322ca463937fd805f569a93e6f15c01aade718793 54.57MB / 54.57MB 20.2s
=> => sha256:6f9f74896dfa93fe0172f594faba85e0b4e8a0481a0fef9d112efc7e4d3c78f7 196.51MB / 196.51MB 31.3s
=> => sha256:5e3b1213efc56598e78bd602983945c164de2a37205e06a62dada823124dc743 6.29MB / 6.29MB 23.4s
=> => sha256:9fddfdc56334f2e6efad7e241bf5e7459c40ed105c5478676f41c1244bd96752 14.21MB / 14.21MB 29.1s
=> => sha256:404f02044bac0432ca522cbb9f254b1c91fcea6806bfeef0be0b243b2f31bab7 235B / 235B 31.1s
=> => sha256:c4f42be2be53b900ebffc040c1df13de538434ccc5f5d954a56848a6169a3a3f 2.21MB / 2.21MB 31.8s
=> => extracting sha256:0e29546d541cddb309281d21a73a9d1db78665c1b95b74f32b009e0b77a6e1e3 4.2s
=> => extracting sha256:9b829c73b52b92b97d5c07a54fb0f3e921995a296c714b53a32ae67d19231fcd 0.5s
=> => extracting sha256:cb5b7ae361722f070eca53f35823ed21baa85d61d5d95cd5a95ab53d740cdd56 0.5s
=> => extracting sha256:6494e4811622b31c027ccac322ca463937fd805f569a93e6f15c01aade718793 4.9s
=> => extracting sha256:6f9f74896dfa93fe0172f594faba85e0b4e8a0481a0fef9d112efc7e4d3c78f7 17.3s
=> => extracting sha256:5e3b1213efc56598e78bd602983945c164de2a37205e06a62dada823124dc743 0.5s
=> => extracting sha256:9fddfdc56334f2e6efad7e241bf5e7459c40ed105c5478676f41c1244bd96752 1.2s
```

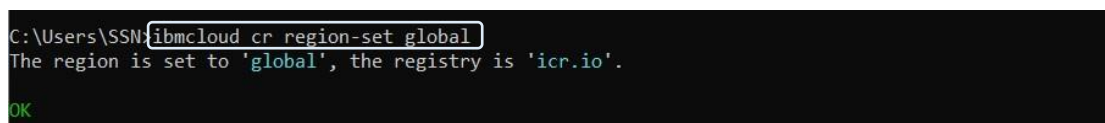
```
C:\Windows\System32\cmd.exe

=> => sha256:9b829c73b52b92b97d5c07a54fb0f3e921995a296c714b53a32ae67d19231fcd 5.15MB / 5.15MB 1.7s
=> => sha256:cb5b7ae361722f070eca53f35823ed21baa85d61d5d95cd5a95ab53d740cdd56 10.87MB / 10.87MB 2.3s
=> => sha256:6494e4811622b31c027ccac322ca463937fd805f569a93e6f15c01aade718793 54.57MB / 54.57MB 20.2s
=> => sha256:6f9f74896dfa93fe0172f594faba85e0b4e8a0481a0fef9d112efc7e4d3c78f7 196.51MB / 196.51MB 31.3s
=> => sha256:5e3b1213efc56598e78bd602983945c164de2a37205e06a62dada823124dc743 6.29MB / 6.29MB 23.4s
=> => sha256:9fddfdc56334f2e6efad7e241bf5e7459c40ed105c5478676f41c1244bd96752 14.21MB / 14.21MB 29.1s
=> => sha256:404f02044bac0432ca522cbb9f254b1c91fcea6806bfeef0be0b243b2f31bab7 235B / 235B 31.1s
=> => sha256:c4f42be2be53b900ebffc040c1df13de538434ccc5f5d954a56848a6169a3a3f 2.21MB / 2.21MB 31.8s
=> => extracting sha256:0e29546d541cddb309281d21a73a9d1db78665c1b95b74f32b009e0b77a6e1e3 4.2s
=> => extracting sha256:9b829c73b52b92b97d5c07a54fb0f3e921995a296c714b53a32ae67d19231fcd 0.5s
=> => extracting sha256:cb5b7ae361722f070eca53f35823ed21baa85d61d5d95cd5a95ab53d740cdd56 0.5s
=> => extracting sha256:6494e4811622b31c027ccac322ca463937fd805f569a93e6f15c01aade718793 4.9s
=> => extracting sha256:6f9f74896dfa93fe0172f594faba85e0b4e8a0481a0fef9d112efc7e4d3c78f7 17.3s
=> => extracting sha256:5e3b1213efc56598e78bd602983945c164de2a37205e06a62dada823124dc743 0.5s
=> => extracting sha256:9fddfdc56334f2e6efad7e241bf5e7459c40ed105c5478676f41c1244bd96752 1.2s
=> => extracting sha256:404f02044bac0432ca522cbb9f254b1c91fcea6806bfeef0be0b243b2f31bab7 0.0s
=> => extracting sha256:c4f42be2be53b900ebffc040c1df13de538434ccc5f5d954a56848a6169a3a3f 0.2s
=> [internal] load build context 0.1s
=> => transferring context: 30.82kB 0.1s
=> [2/6] WORKDIR /app 0.5s
=> [3/6] ADD . /app 0.0s
=> [4/6] COPY requirements.txt /app 0.0s
=> [5/6] RUN python3 -m pip install -r requirements.txt 208.9s
=> [6/6] RUN python3 -m pip install ibm_db 2.2s
=> exporting to image 1.1s
=> => exporting layers 1.0s
=> => writing image sha256:7d417fd6570f7414499cde4681586932c4bd84662177857e226160828b6e1061 0.0s
=> => naming to docker.io/library/flask-app 0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
```



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

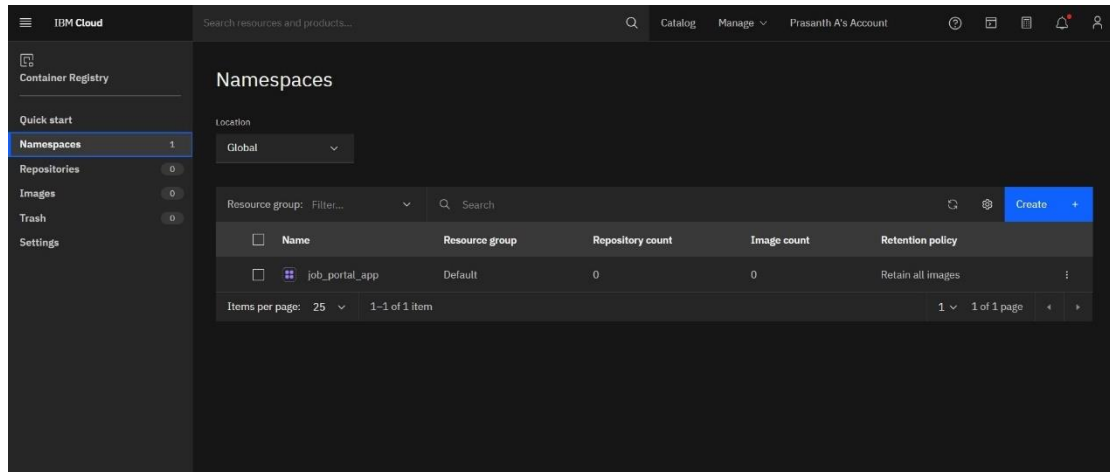


```
C:\Users\SSN>ibmcloud cr namespace-add job_portal_app
No resource group is targeted. Therefore, the default resource group for the account ('Default') is targeted.

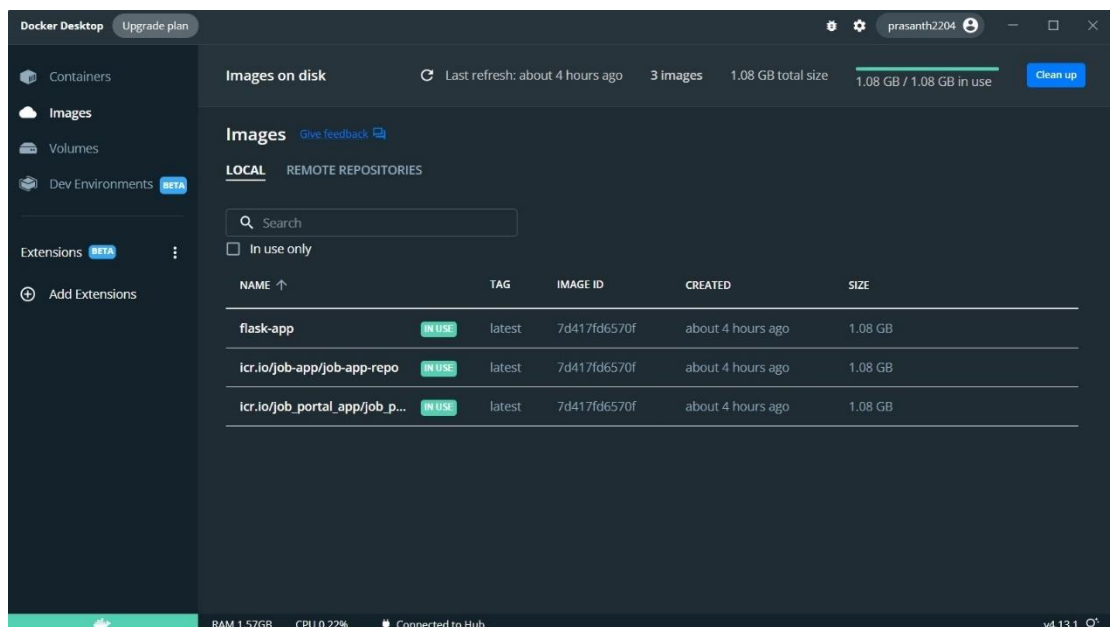
Adding namespace 'job_portal_app' in resource group 'Default' for account Prasanth A's Account in registry icr.io...

Successfully added namespace 'job_portal_app'

OK
```



```
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign 4>docker tag flask-app icr.io/job-app/job-app-repo
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign 4>docker push icr.io/job-app/job-app-repo
```



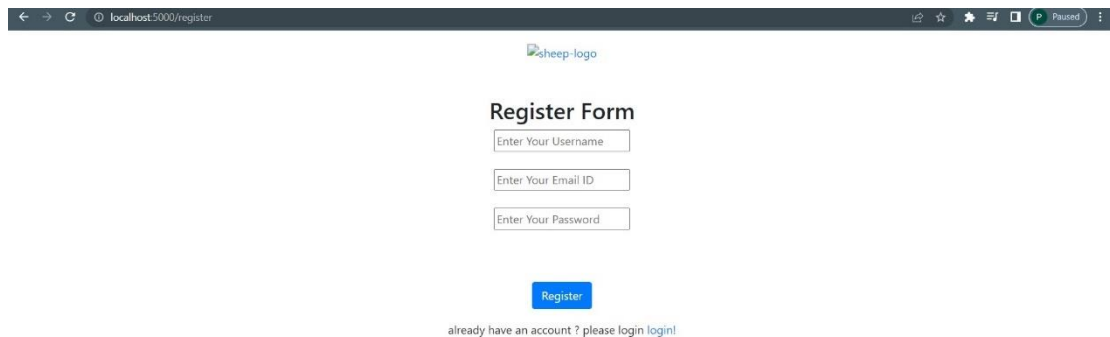
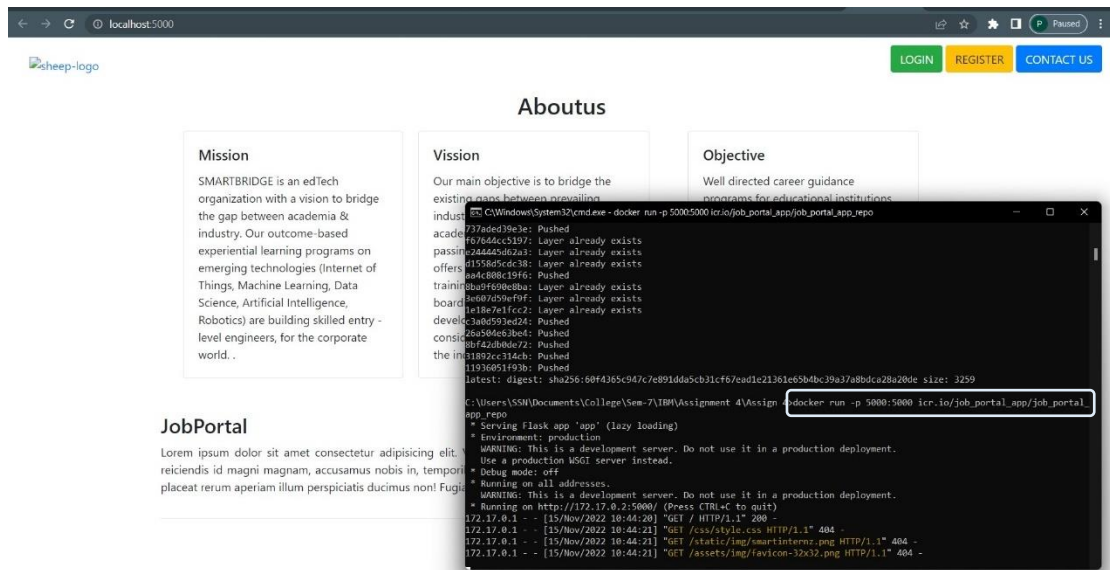
```
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign 4>ibmcloud cr login
Logging 'docker' in to 'icr.io'...
Logged in to 'icr.io'.

OK
```

```
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign 4>docker push icr.io/job_portal_app/job_portal_app_repo
Using default tag: latest
The push refers to repository [icr.io/job_portal_app/job_portal_app_repo]
8fd68227c2d6: Layer already exists
737aded39e3e: Pushed
f67644cc5197: Layer already exists
e244445d62a3: Layer already exists
d1558d5cdc38: Layer already exists
aa4c808c19f6: Pushed
8ba9f690e8ba: Layer already exists
3e607d59ef9f: Layer already exists
1e18e7e1fcc2: Layer already exists
c3a0d593ed24: Pushed
26a504e63be4: Pushed
8bf42db0de72: Pushed
31892cc314cb: Pushed
11936051f93b: Pushed
latest: digest: sha256:60f4365c947c7e891dda5cb31cf67ead1e21361e65b4bc39a37a8bdca28a20de size: 3259
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign 4>
```

The screenshot displays the IBM Cloud Container Registry interface. On the left, a sidebar contains navigation links: Container Registry, Quick start, Namespaces (1), Repositories (1), Images (1), Trash (0), and Settings. The main content area is titled 'Images' and shows a table of image manifests. The table has columns for Repository@digest, Tags, Manifest type, Created, Size, and Security status. A single item is listed: 'job\_portal\_app/job\_portal\_app\_repo@sha256:60f4365c947c...'. The 'Tags' column for this item shows 'latest'. The 'Created' column indicates it was created '4 hours ago'. The 'Size' is '435 MB'. The 'Security status' column shows '102 Issues'. At the bottom of the table, it says 'Items per page: 25' and '1-1 of 1 item'.

Repository@digest	Tags	Manifest type	Created	Size	Security status
job_portal_app/job_portal_app_repo@sha256:60f4365c947c...	latest	Docker	4 hours ago	435 MB	102 Issues



← → ↻ localhost:5000/login

sheep-logo

### Login Form

Enter Your Username

Enter Your Password

Login

Don't have an account yet? Click here to [register!](#)

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

cloud.ibm.com/kubernetes/clusters

IBM Cloud

Search resources and products...

Kubernetes

Clusters

Reservations

Helm catalog

Container Registry

### Kubernetes clusters

Resource group: Filter... Location: Filter... Search

Create cluster

Name	State	Location	Worker count	Created	Version	Infrastructure
mycluster-free	Normal	Paris	1	Expires in 27 days	@ 1.24.7_1542	Classic

Items per page: 25 1-1 of 1 item



```
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assignment 4>ibmcloud plugin install container-service
Looking up 'container-service' from repository 'IBM Cloud'...
Plug-in 'container-service[kubernetes-service/ks] 1.0.459' found in repository 'IBM Cloud'
Attempting to download the binary file...
26.86 MiB / 26.86 MiB [=====] 100.00% 7s
28168192 bytes downloaded
Installing binary...
OK
Plug-in 'container-service 1.0.459' was successfully installed into C:\Users\SSN\bluemix\plugins\container-service. Use
'ibmcloud plugin show container-service' to show its details.
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assignment 4>
```

```
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign 4>ibmcloud plugin show container-service

Plugin Name           container-service[kubernetes-service/ks]
Plugin Version        1.0.459
Plugin SDK Version     0.3.0
Minimal IBM Cloud CLI version required 0.18.2
Private endpoints supported false

Commands:
  sat                Manage IBM Cloud Satellite clusters.
  sat keys           List all Satellite Config keys in your IBM Cloud account.
  sat messages       View the current user messages.
  sat subscriptions  List all Satellite subscriptions in your IBM Cloud
```

```
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assignment 4>ibmcloud ks cluster ls
OK
Name      ID              State    Created      Workers  Location  Version  Resource Group Name
mycluster-free classic      cdntg4tf02end88h9t10  deploying  37 minutes ago  1        par      1.24.7_1542  Default

C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assignment 4>ibmcloud ks cluster config --cluster cdntg4tf02end88h9t10
OK
The configuration for cdntg4tf02end88h9t10 was downloaded successfully.

Added context for cdntg4tf02end88h9t10 to the current kubeconfig file.
You can now execute 'kubectl' commands against your cluster. For example, run 'kubectl get nodes'.
If you are accessing the cluster for the first time, 'kubectl' commands might fail for a few seconds while RBAC synchronizes.
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assignment 4>
```

```
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign 4>kubectl create deploy webserver --image=icr.io/job_portal_app/job_portal_app_repo
deployment.apps/webserver created
```



