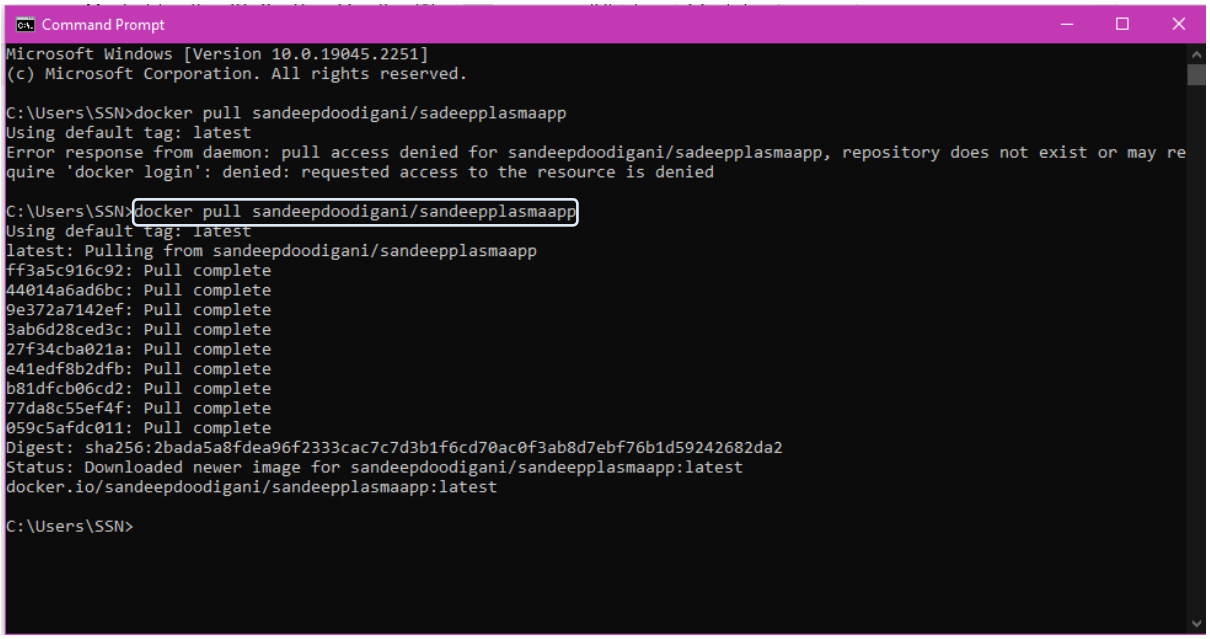
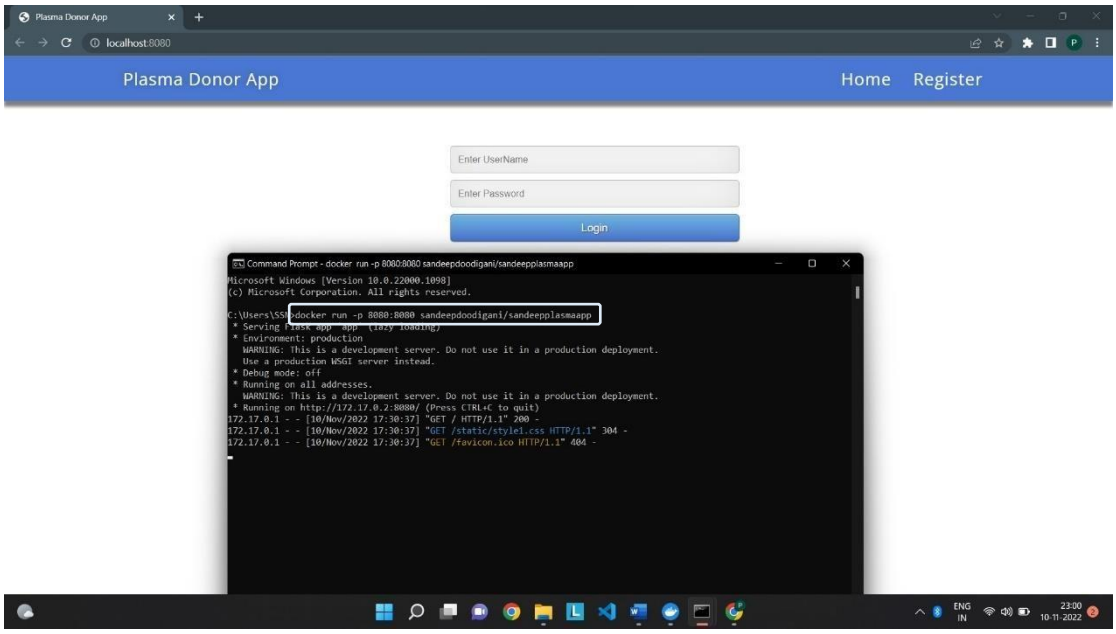
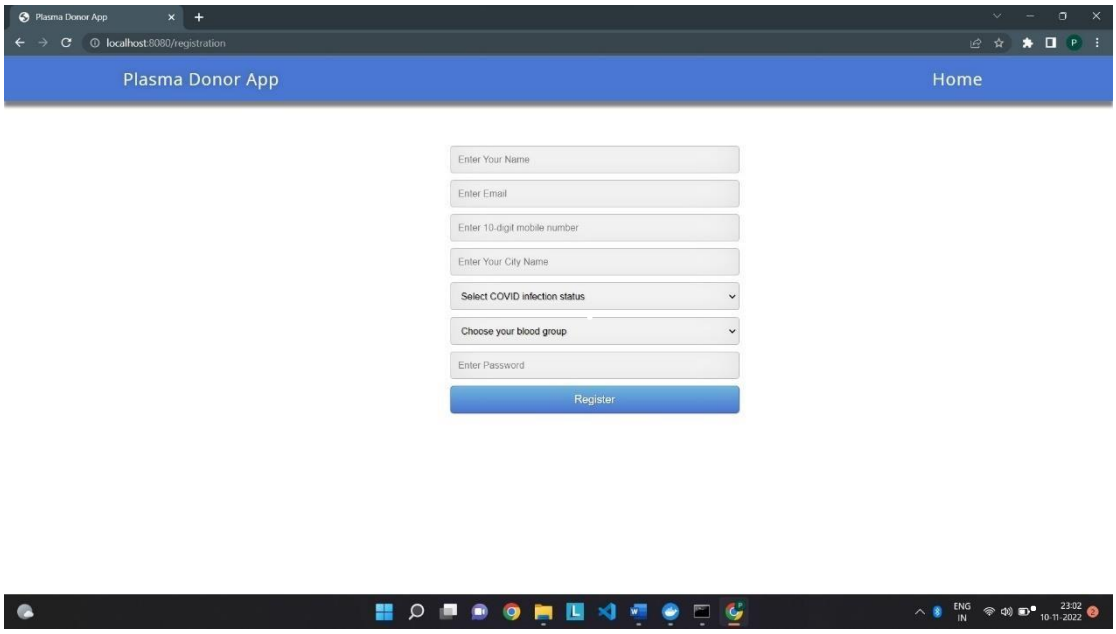
Assignment-4

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







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=PI8VtGRvZlSVT65A",",")
```

```
@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(prep_stmt, 1, username)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, password)
            ibm_db.execute(prep_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(prep_stmt, 1, username)

    ibm_db.bind_param(prep_stmt, 2, email)

    ibm_db.bind_param(prep_stmt, 3, qualification)
```

```
        ibm_db.bind_param(prep_stmt, 4, skills)

        ibm_db.bind_param(prep_stmt, 5, jobs)

        ibm_db.execute(prep_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)
```

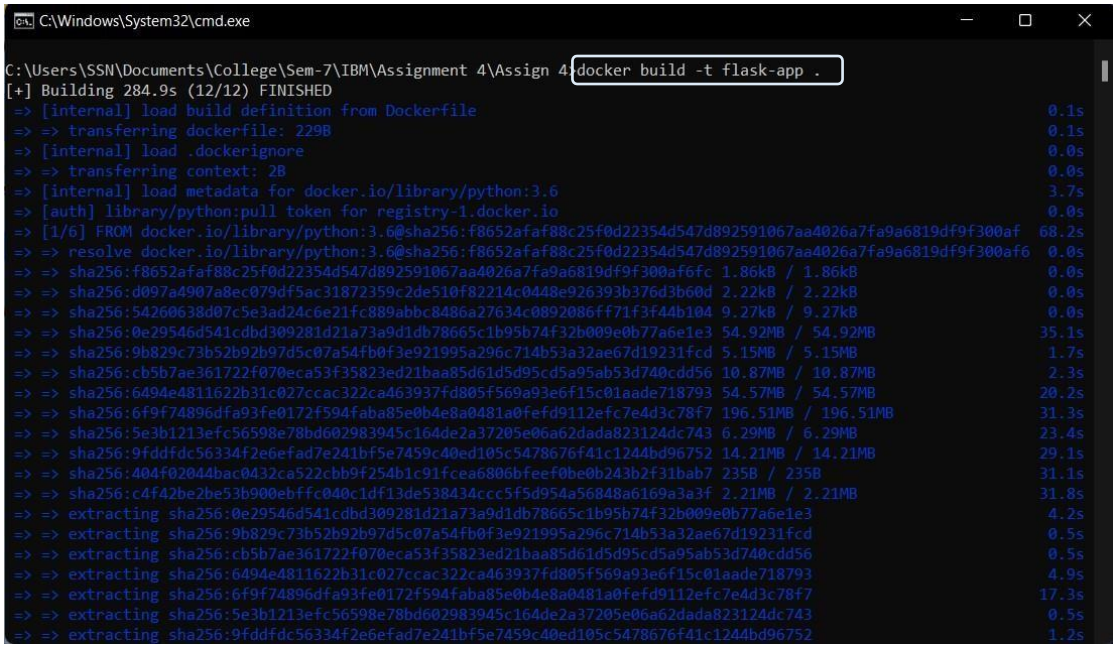
```
@app.route('/display')

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("accountdislay",account)
```

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

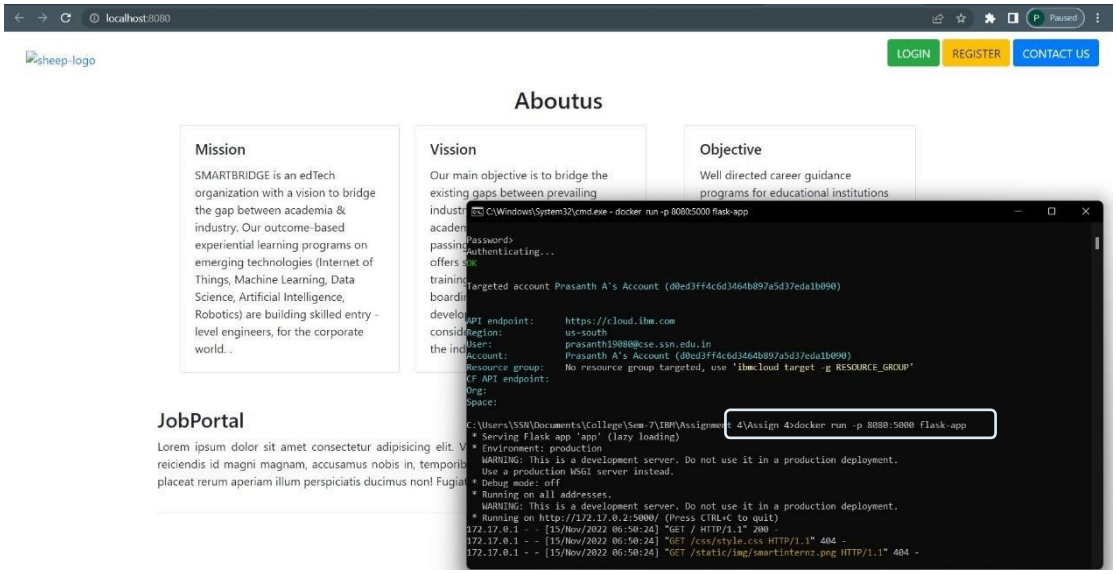
```
@app.route('/logout')
```

```
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')
```



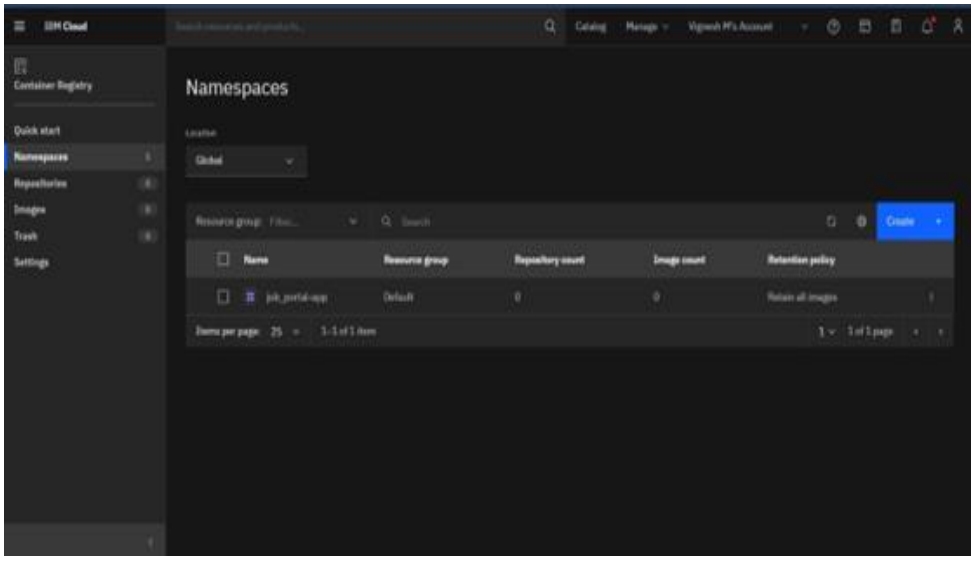
```
C:\Windows\System32\cmd.exe
=> => sha256:9b829c73b52b92b97d5c07a54fb0f3e921995a296c714b53a32ae67d19231fcd 5.15MB / 5.15MB                    1.7s
=> => sha256:cb5b7ae361722f070eca53f35823ed21baa85d61d5d95cd5a95ab53d740cdd56 10.87MB / 10.87MB                  2.3s
=> => sha256:6494e4811622b31c027ccac322ca463937fd805f569a93e6f15c01aade718793 54.57MB / 54.57MB                 20.2s
=> => sha256:6f9f74896dfa93fe0172f594faba85e0b4e8a0481a0fefd9112efc7e4d3c78f7 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 sha256:0e29546d541cdbd309281d21a73a9d1db78665c1b95b74f32b009e0b77a6e1e3                         4.2s
=> => extracting sha256:9b829c73b52b92b97d5c07a54fb0f3e921995a296c714b53a32ae67d19231fcd                         0.5s
=> => extracting sha256:cb5b7ae361722f070eca53f35823ed21baa85d61d5d95cd5a95ab53d740cdd56                         0.5s
=> => extracting sha256:6494e4811622b31c027ccac322ca463937fd805f569a93e6f15c01aade718793                         4.9s
=> => extracting sha256:6f9f74896dfa93fe0172f594faba85e0b4e8a0481a0fefd9112efc7e4d3c78f7                        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
```



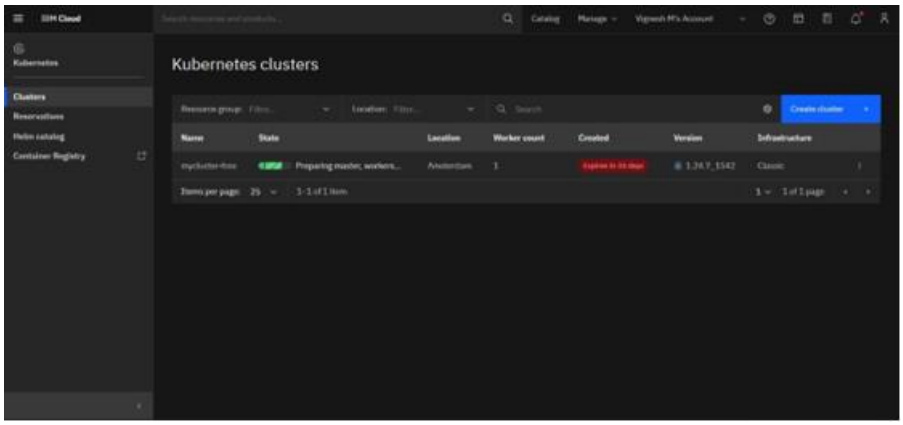
3. Create a IBM container registry and deploy helloworld app or jobportalapp.

```
C:\Users\SSN>ibmcloud cr region-set global
The region is set to 'global', the registry is 'icr.io'.

OK
```

```
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
```

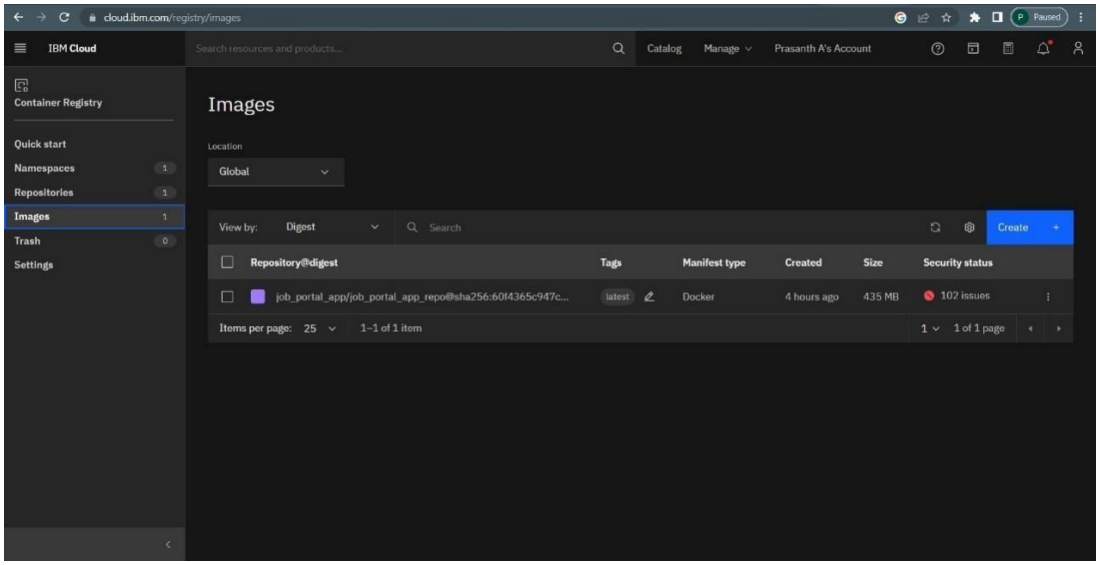


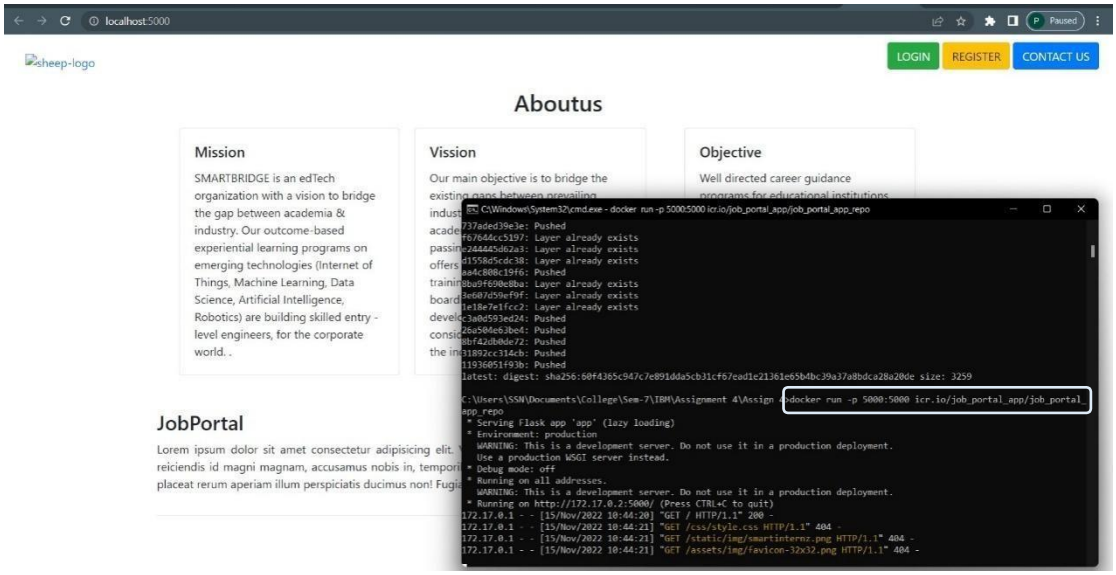
```
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign 4>docker tag flask-app  icr.io/job-app/job-app-repo

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

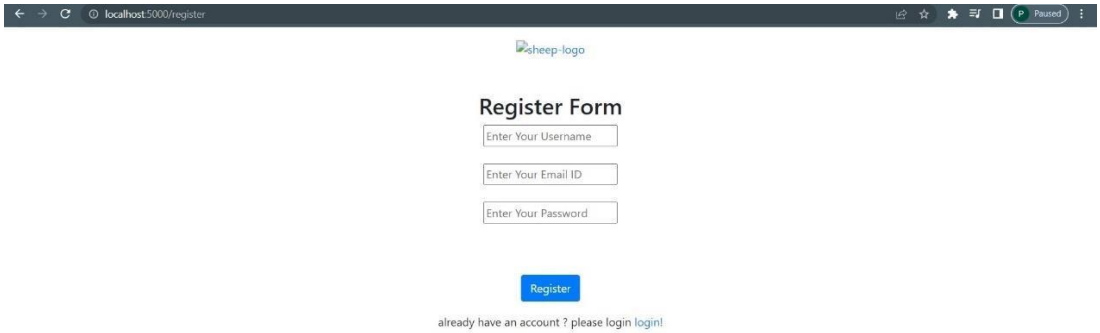


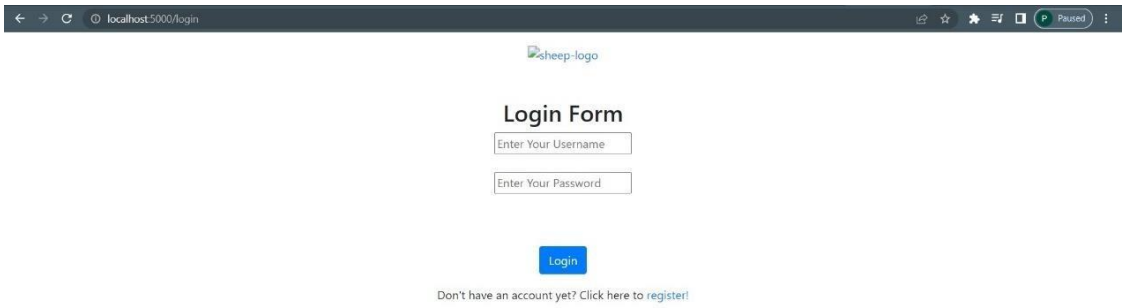

```
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign 4>ibmcloud cr login
Logging 'docker' in to 'icr.io'...
Logged in to 'icr.io'.

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>_
```

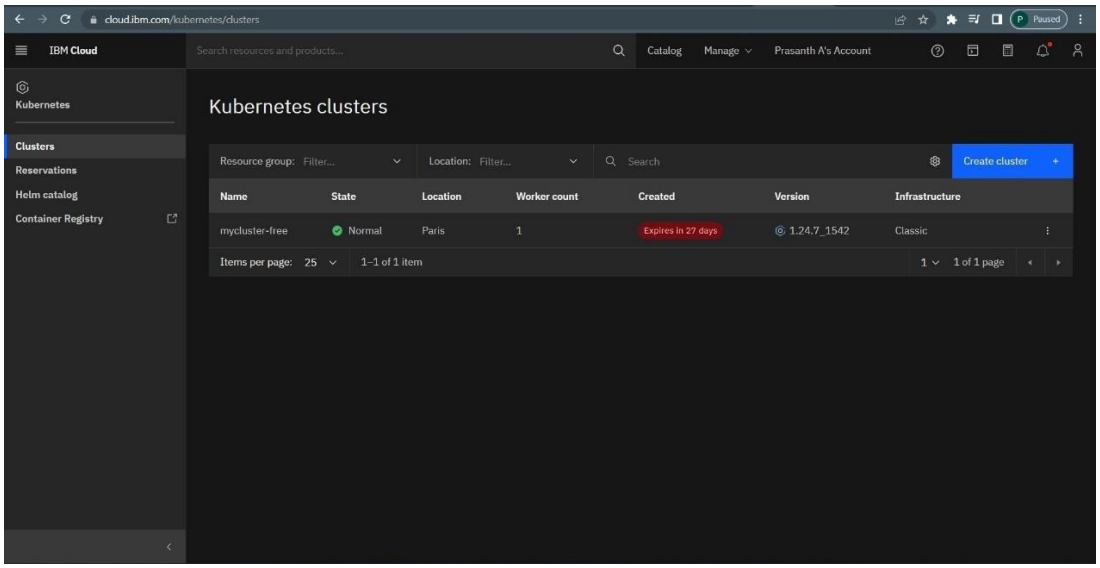

cloud.ibm.com/registry/images
IBM Cloud
Search resources and products...
Catalog
Manage
Prasanth A's Account
Container Registry
Quick start
Namespaces 1
Repositories 1
Images 1
Trash 0
Settings
Images
Location
Global
View by: Digest
Search
Create
Repository@digest
Tags
Manifest type
Created
Size
Security status
job_portal_app/job_portal_app_repo@sha256:60f4365c947c...
latest
Docker
4 hours ago
435 MB
102 issues
Items per page: 25
1–1 of 1 item
1 of 1 page


localhost:5000
sheep-logo
LOGIN
REGISTER
CONTACT US
Aboutus
Mission
SMARTBRIDGE is an edTech organization with a vision to bridge the gap between academia & industry. Our outcome-based experiential learning programs on emerging technologies (Internet of Things, Machine Learning, Data Science, Artificial Intelligence, Robotics) are building skilled entry - level engineers, for the corporate world. .
Vission
Our main objective is to bridge the
Objective
Well directed career guidance
JobPortal
Lorem ipsum dolor sit amet consectetur adipisicing elit.
reiciendis id magni magnam, accusamus nobis in, tempori
placeat rerum aperiam illum perspiciatis ducimus non! Fugi
C:\Windows\System32\cmd.exe - docker run -p 5000:5000 icr.io/job_portal_app/job_portal_app_repo
f37a0ed39e3e: Pushed
f67644cc5197: Layer already exists
244445d62a3: Layer already exists
d1558d5cdc38: Layer already exists
aa4c808c19f6: Pushed
8ba9f690e8ba: Layer already exists
9e607d59ef9f: Layer already exists
1e18e7e1fcc2: Layer already exists
c3a0d593ed24: Pushed
26a504e63be4: Pushed
8bf42db0de72: Pushed
31892cc314cb: Pushed
11936051f93b: Pushed
latest: digest: sha256:60f4365c947c7e891dda5cb31cf67ead1e21381e65b4bc39a37a8bdca28a20de size: 3259
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign > docker run -p 5000:5000 icr.io/job_portal_app/job_portal_app_repo
* Serving Flask app 'app' (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on all addresses.
WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://172.17.0.2:5000/ (Press CTRL+C to quit)
172.17.0.1 - - [15/Nov/2022 10:44:20] "GET / HTTP/1.1" 200 -
172.17.0.1 - - [15/Nov/2022 10:44:21] "GET /css/style.css HTTP/1.1" 404 -
172.17.0.1 - - [15/Nov/2022 10:44:21] "GET /static/img/smartinternz.png HTTP/1.1" 404 -
172.17.0.1 - - [15/Nov/2022 10:44:21] "GET /assets/img/favicon-32x32.png HTTP/1.1" 404 -


localhost:5000/register
sheep-logo
Register Form
Enter Your Username
Enter Your Email ID
Enter Your Password
Register
already have an account ? please login login!



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

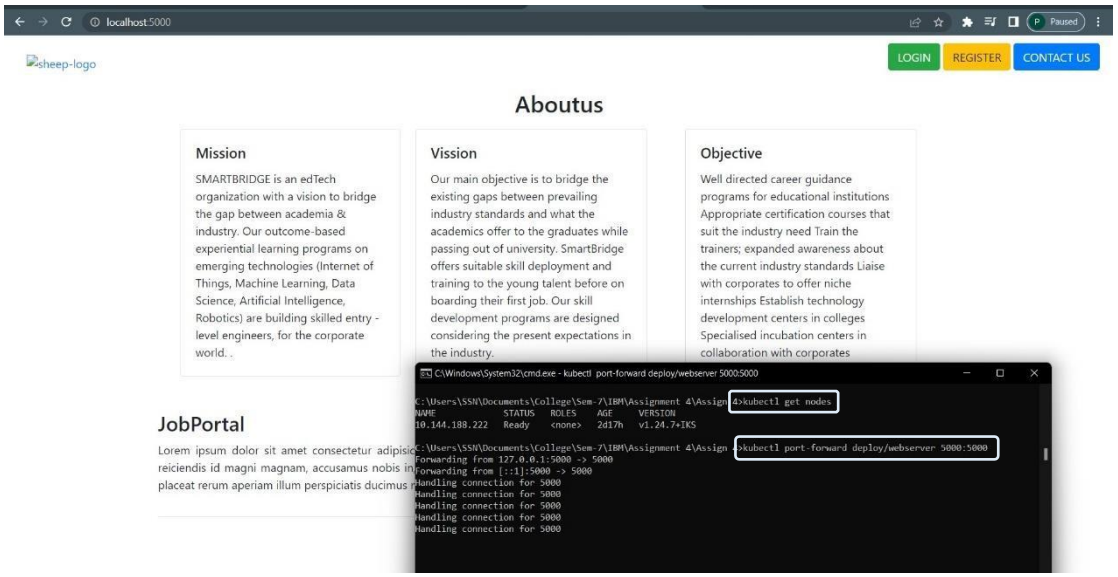


```
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assignment 4>ibmcloud plugin install container-service
Looking up 'container-service' from repository 'IBM Cloud'...
Plug-in 'container-service[kubernetes-service/ks] 1.0.459' found in repository 'IBM Cloud'
Attempting to download the binary file...
 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 a
ccount.
 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 N
ame   Provider
mycluster-free    cdntg4tf02end88h9tl0   deploying   37 minutes ago   1         par        1.24.7_1542   Default
      classic

C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assignment 4>ibmcloud ks cluster config --cluster cdntg4tf02end88h
9tl0
OK
The configuration for cdntg4tf02end88h9tl0 was downloaded successfully.

Added context for cdntg4tf02end88h9tl0 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 synchron
izes.

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
```


localhost:5000
sheep-logo
LOGIN
REGISTER
CONTACT US
Aboutus
Mission
SMARTBRIDGE is an edTech organization with a vision to bridge the gap between academia & industry. Our outcome-based experiential learning programs on emerging technologies (Internet of Things, Machine Learning, Data Science, Artificial Intelligence, Robotics) are building skilled entry - level engineers, for the corporate world. .
Vission
Our main objective is to bridge the existing gaps between prevailing industry standards and what the academics offer to the graduates while passing out of university. SmartBridge offers suitable skill deployment and training to the young talent before on boarding their first job. Our skill development programs are designed considering the present expectations in the industry.
Objective
Well directed career guidance programs for educational institutions Appropriate certification courses that suit the industry need Train the trainers; expanded awareness about the current industry standards Liaise with corporates to offer niche internships Establish technology development centers in colleges Specialised incubation centers in collaboration with corporates
JobPortal
Lorem ipsum dolor sit amet consectetur adipisci
reiciendis id magni magnam, accusamus nobis in
placeat rerum aperiam illum perspiciatis ducimus
C:\Windows\System32\cmd.exe - kubectl port-forward deploy/webserver 5000:5000
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign 4>kubectl get nodes
NAME STATUS ROLES AGE VERSION
10.144.188.222 Ready <none> 2d17h v1.24.7+IKS
C:\Users\SSN\Documents\College\Sem-7\IBM\Assignment 4\Assign >kubectl port-forward deploy/webserver 5000:5000
Forwarding from 127.0.0.1:5000 -> 5000
Forwarding from [::1]:5000 -> 5000
Handling connection for 5000
Handling connection for 5000
Handling connection for 5000
Handling connection for 5000
Handling connection for 5000