

Assignment 2:

731319104041 – T.Vimalraj

731319104004 – S.Balachandru

731319104012 – N.Gokulakannan

731319104025 – P.Nandhakumar

1. Create User table with user with email,usernameand password.

The screenshot shows the IBM Db2 on Cloud console interface. The 'Tables' tab is selected, displaying a list of tables in the 'YGG09863' schema. The 'USERS' table is highlighted. The 'Table definition' panel on the right shows the structure of the 'USERS' table:

Name	Data type	Nullable	Length	Scale
USERNAME	VARCHAR	Y	32	0
EMAIL	VARCHAR	Y	32	0
PASSWORD	VARCHAR	Y	32	0

The table is currently empty, with approximately 0 rows and 0 KB of data. A 'View data' button is visible at the bottom of the table definition panel.

2. Perform UPDATE,DELETE Queries with user table

The screenshot shows the IBM Db2 on Cloud console interface with the 'SQL' tab selected. The 'My script' panel displays a SQL script:

```
1 insert into users values('maryada','sample123','maryada@student.tce.edu');
```

The 'History' panel shows the execution history of the script:

Script	Date	Status	Runtime
Untitled - 1	Oct 3, 2022 10:22:41 AM	✓ 1	0.024 s
insert into users values('maryada','sample123','maryada@student...		✓	0.024 s
Untitled - 1	Oct 3, 2022 10:21:51 AM	✓ 1	0.039 s
delete from users where username='maryada'		✓	0.039 s

IBM Db2 on Cloud

Data objects | My script

Filter objects

YGG09863

*Untitled - 1

1 delete from users where username='maryada';

Run all

History

Find history

Script	Date	Status	Runtime
Untitled - 1	Oct 3, 2022 10:23:07 AM	✓ 1	0.020 s
delete from users where username='maryada'		✓	0.020 s
Untitled - 1	Oct 3, 2022 10:22:41 AM	✓ 1	0.024 s
insert into users values('maryada','sample123','maryada@stude...		✓	0.024 s
Untitled - 1	Oct 3, 2022 10:21:51 AM	✓ 1	0.039 s

IBM Db2 on Cloud

Data objects | My script

Filter objects

YGG09863

*Untitled - 1

1 UPDATE users
2 SET password='sample1234'
3 WHERE username='maryada';

Run all

History

Find history

Script	Date	Status	Runtime
Untitled - 1	Oct 3, 2022 10:24:55 AM	✓ 1	0.024 s
UPDATE users SET password='sample1234' WHERE username='maryad...		✓	0.024 s
Untitled - 1	Oct 3, 2022 10:24:35 AM	✓ 1	0.019 s
insert into users values('maryada','sample123','maryada@stude...		✓	0.019 s
Untitled - 1	Oct 3, 2022 10:23:07 AM	✓ 1	0.020 s

3. Connect python code to db2. (Done in question 4)

4. Create a flask app with registration page, login page and welcome page. By default load the registration page once the user enters all the fields store the data in database and navigate to login page authenticate user username and password. If the user is valid show the welcome page

app.py:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
```

"""

Created on Sun Oct 2 12:32:54 2022

@author: maryada

"""

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

```
app=Flask(__name__)
app.secret_key='a'
conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=9938aec0-8105-433e-8bf9-
0fbb7e483086.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32459;Security=SSL;SSL
ServerCertificate=DigiCertGlobalRootCA.crt;UID=ygg09863;PWD=dQXLEcjtaCqXsJUt","","")
```

```
@app.route('/')
@app.route('/login',methods=['GET','POST'])
def login():
    global userid;
    msg=" "
```

```
    if request.method == 'POST' and 'username' in request.form and 'password' in request.form:
```

```
        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!"
            return render_template("index.html", msg=msg)
        else :
            msg="Incorrect username and password"
```

```
    return render_template("login.html", msg=msg)
```

```
@app.route('/logout')
def logout():
    session.pop('loggedin', None)
    session.pop('id', None)
    session.pop('username', None)
    return redirect(url_for('login'))
```

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

```

def register():
    msg=""
    if request.method == 'POST' and 'username' in request.form and 'password' in request.form and
'email' in request.form :
        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 = 'Username must contain only characters and numbers !'
        elif not username or not password or not email:
            msg = 'Please fill out the form !'
        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 logged in!"

    elif request.method=="POST":
        msg="Please fill out the form"
    return render_template('register.html', msg=msg)

if __name__=="_main_":
    app.run(host='0.0.0.0')

```

templates folder: index.html

<!-- Store this code in 'index.html' file inside the 'templates' folder-->

```

<html>
    <head>
        <meta charset="UTF-8">
        <title> Index </title>
        <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">

    </head>
    <body><br></br></br></br></br>
        <div align="center">
            <div align="center" class="border">

```

```

        <div class="header">
            <h1 class="word">Index</h1>
        </div></br></br></br>
        <h1 class="bottom">
            Hi {{ session.username }}!!</br></br> Welcome to the index
page...

        </h1></br></br></br>
        <a href="{{ url_for('logout') }}" class="btn">Logout</a>

    </div>
</div>
</body>
</html>

```

login.html:

<!-- Store this code in 'login.html' file inside the 'templates' folder -->

```

<html>
    <head>
        <meta charset="UTF-8">
        <title> Login </title>
        <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
    </head>
    <body></br></br></br></br></br>
        <div align="center">
            <div align="center" class="border">
                <div class="header">
                    <h1 class="word">Login</h1>
                </div></br></br></br>
                <h2 class="word">
                    <form action="{{ url_for('login') }}" method="post">
                        <div class="msg">{{ msg }}</div>
                        <input id="username" name="username" type="text"
placeholder="Enter Your Username" class="textbox"/></br></br>
                        <input id="password" name="password" type="password"
placeholder="Enter Your Password" class="textbox"/></br></br></br>
                        <input type="submit" class="btn" value="Sign In"></br></br>
                    </form>
                </h2>
                <p class="bottom">Don't have an account? <a class="bottom"
href="{{ url_for('register') }}"> Sign Up here</a></p>
            </div>
        </div>
    </body>
</html>

```

register.html:

<!-- Store this code in 'register.html' file inside the 'templates' folder -->

```

<html>
    <head>
        <meta charset="UTF-8">

```

```

<title> Register </title>
<link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body><br><br><br><br><br>
<div align="center">
<div align="center" class="border">
<div class="header">
<h1 class="word">Register</h1>
</div><br><br><br>
<h2 class="word">
<form action="{{ url_for('register') }}" method="post">
<div class="msg">{{ msg }}</div>
<input id="username" name="username" type="text"
placeholder="Enter Your Username" class="textbox"/><br><br>
<input id="password" name="password" type="password"
placeholder="Enter Your Password" class="textbox"/><br><br>
<input id="email" name="email" type="text"
placeholder="Enter Your Email ID" class="textbox"/><br><br>
<input type="submit" class="btn" value="Sign Up"><br>
</form>
</h2>
<p class="bottom">Already have an account? <a class="bottom"
href="{{ url_for('login') }}"> Sign In here</a></p>
</div>
</div>
</body>
</html>

```

static folder:
style.css

/* Store this code in 'style.css' file inside the 'static' folder*/

```

.header{
padding: 5px 120px;
width: 150px;
height: 70px;
background-color: #236B8E;
}

.border{
padding: 80px 50px;
width: 400px;
height: 450px;
border: 1px solid #236B8E;
border-radius: 0px;
background-color: #9AC0CD;
}

.btn {
padding: 10px 40px;
background-color: #236B8E;
}

```

```
        color: #FFFFFFF;
        font-style: oblique;
        font-weight: bold;
        border-radius: 10px;
    }

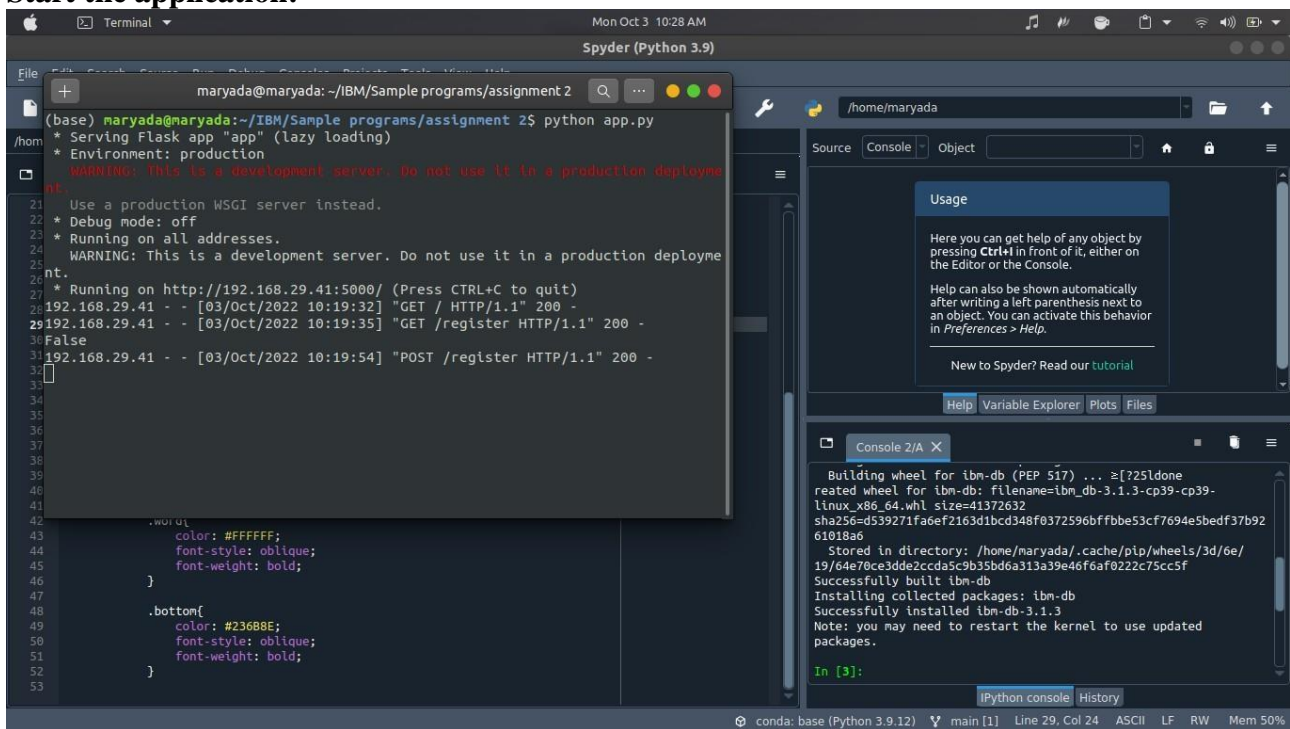
    .textbox{
        padding: 10px 40px;
        background-color: #236B8E;
        text-color: #FFFFFFF;
        border-radius: 10px;
    }

    ::placeholder {
        color: #FFFFFFF;
        opacity: 1;
        font-style: oblique;
        font-weight: bold;
    }

    .word{
        color: #FFFFFFF;
        font-style: oblique;
        font-weight: bold;
    }

    .bottom{
        color: #236B8E;
        font-style: oblique;
        font-weight: bold;
    }
```

Output: Start the application:



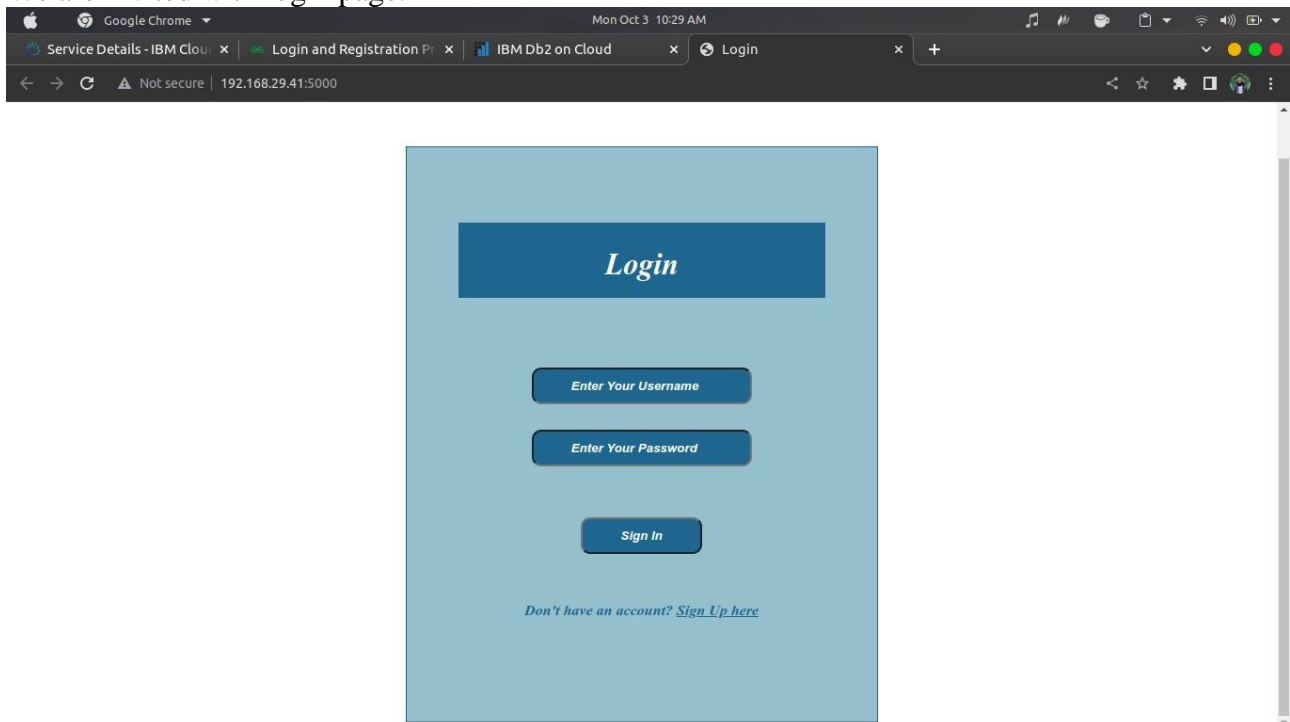
The screenshot shows the Spyder Python IDE interface. The left pane displays a code editor with a Flask application. The top-right pane shows the 'Console' tab with the following output:

```
(base) maryada@maryada: ~/IBM/Sample programs/assignment 2$ python app.py
* 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://192.168.29.41:5000/ (Press CTRL+C to quit)
192.168.29.41 - - [03/Oct/2022 10:19:32] "GET / HTTP/1.1" 200 -
192.168.29.41 - - [03/Oct/2022 10:19:35] "GET /register HTTP/1.1" 200 -
False
192.168.29.41 - - [03/Oct/2022 10:19:54] "POST /register HTTP/1.1" 200 -
```

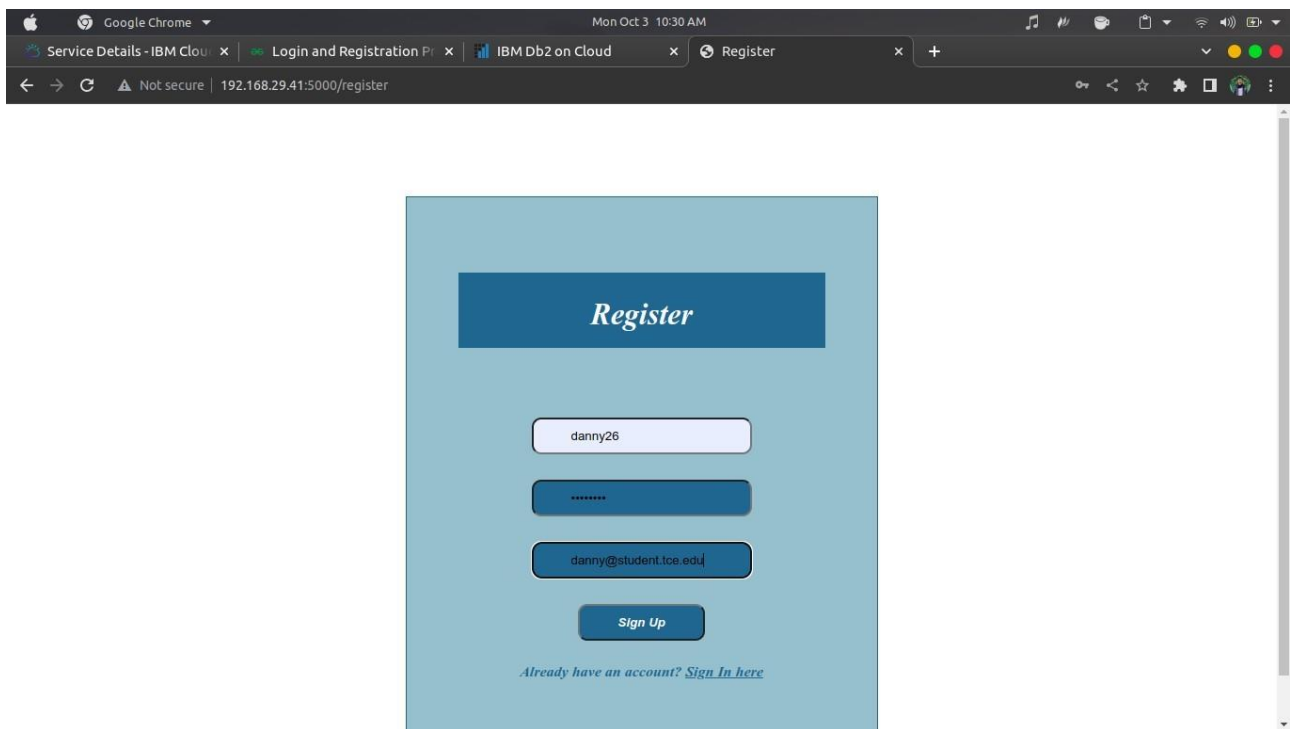
The bottom-right pane shows the 'Console 2/A' tab with the following output:

```
Building wheel for ibm-db (PEP 517) ... [?25ldone
reated wheel for ibm-db: filename=ibm_db-3.1.3-cp39-cp39-
linux_x86_64.whl size=41372632
sha256=d539271fa6ef2163d1bcd348f0372596bfbbe53cf7694e5bedf37b92
61018a6
Stored in directory: /home/maryada/.cache/pip/wheels/3d/6e/
19/64e70ce3dde2ccda5c9b35bd6a313a39e46fa022c75cc5f
Successfully built ibm-db
Installing collected packages: ibm-db
Successfully installed ibm-db-3.1.3
Note: you may need to restart the kernel to use updated
packages.
In [3]:
```

We are invited with login page.



Registration:



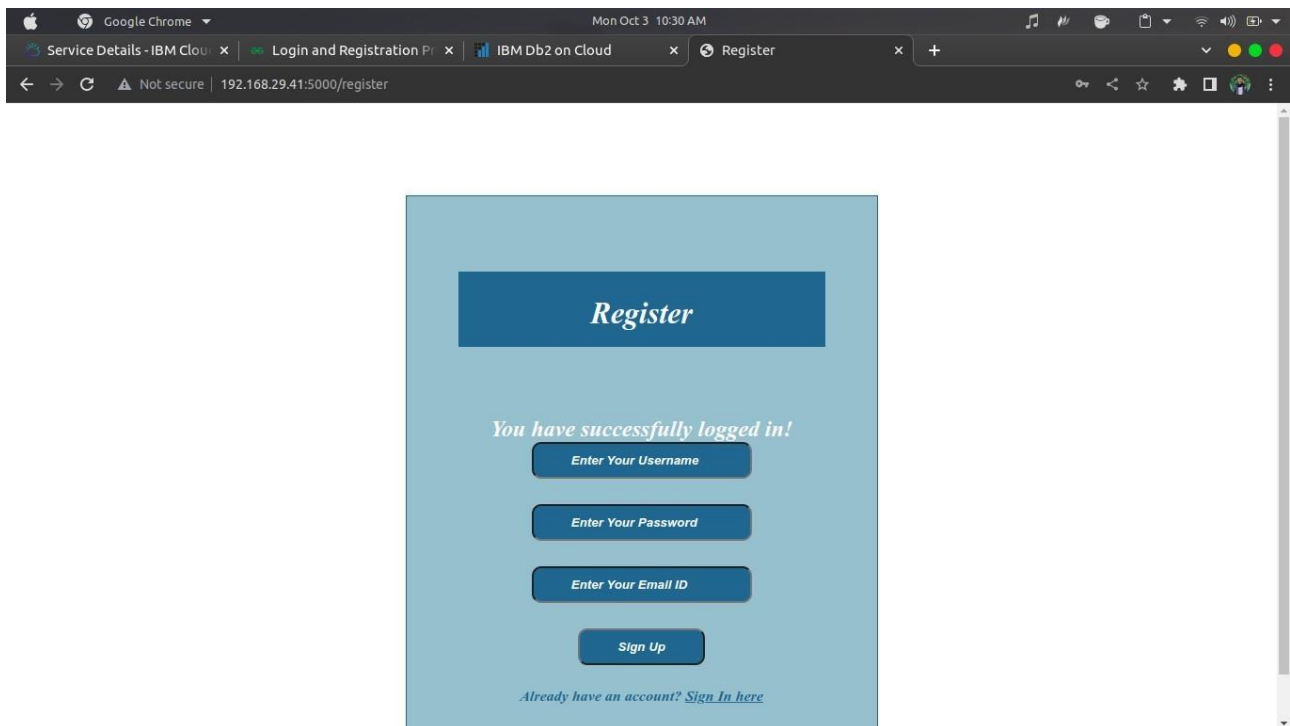
Register

danny26

danny@student.tce.edu

Sign Up

Already have an account? [Sign In here](#)



Register

You have successfully logged in!

Enter Your Username

Enter Your Password

Enter Your Email ID

Sign Up

Already have an account? [Sign In here](#)

Google Chrome | Mon Oct 3 10:31 AM

Service Details - IBM Cloud | Login and Registration Pr | IBM Db2 on Cloud | Register

bpe61bfd0365e9u4psdglite.db2.cloud.ibm.com/crn%3Av1%3Abluemix%3Apublic%3Adashdb-for-transactions%3Aus-south%3Aa%2Fbc3a4dafd57c42e38a173...

IBM Db2 on Cloud

Load Data | Load History | **Tables** | Views | Indexes | Aliases | MQTs | Sequences | Application objects

YGG09863.USERS

Back

Export to CSV

USERNAME	EMAIL	PASSWORD
danny26	danny@student.tce.edu	sample23
maryada	sample123	sample1234

Login and Logout:

Google Chrome | Mon Oct 3 10:32 AM

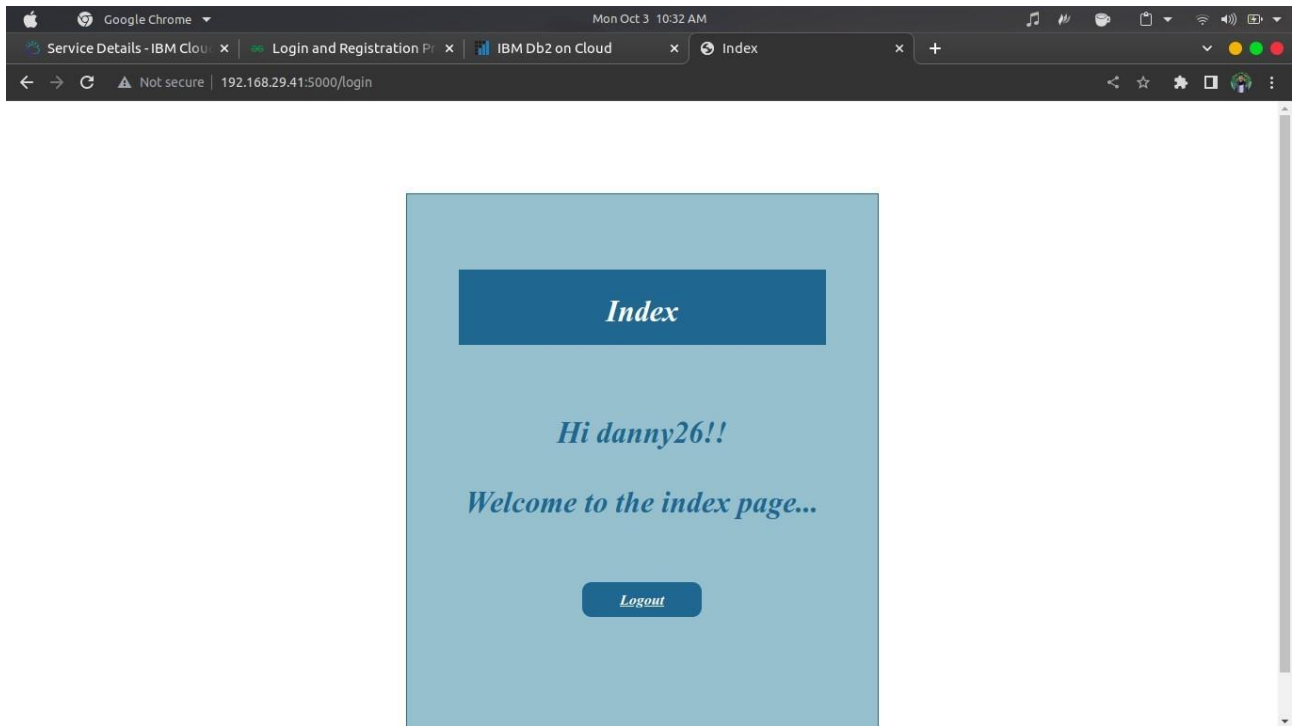
Service Details - IBM Cloud | Login and Registration Pr | IBM Db2 on Cloud | Login

Not secure | 192.168.29.41:5000/login

Login

Sign In

Don't have an account? [Sign Up here](#)



Logout:

