

ASSIGNMENT 2

Date	12 October 2022
Student Name	Barathkumar P
Student Roll no	621319106008
Maximum Marks	2 Marks

1. Create User table with email, username, roll number, password.

Create user table:

The screenshot shows the IBM Db2 on Cloud web interface. The SQL editor contains the following SQL statement:

```
1 CREATE TABLE user_details(  
2     s1_no INT GENERATED BY DEFAULT AS IDENTITY NOT NULL,  
3     user_name VARCHAR(150) NOT NULL,  
4     user_email VARCHAR(255) NOT NULL,  
5     roll_no INT NOT NULL,  
6     password VARCHAR(100),  
7     PRIMARY KEY (s1_no),  
8 );  
9
```

The History tab is active, showing the execution of the script:

Script	Date	Status	Runtime
Assignment 2	Oct 17, 2022 9:46:21 PM	✓ 1	0.111 s
CREATE TABLE user_details(s1_no INT GENERATED BY DEFAULT AS IDENTITY NOT NULL, user_name VARCHAR(150) ...		✓	0.111 s

Insert details:

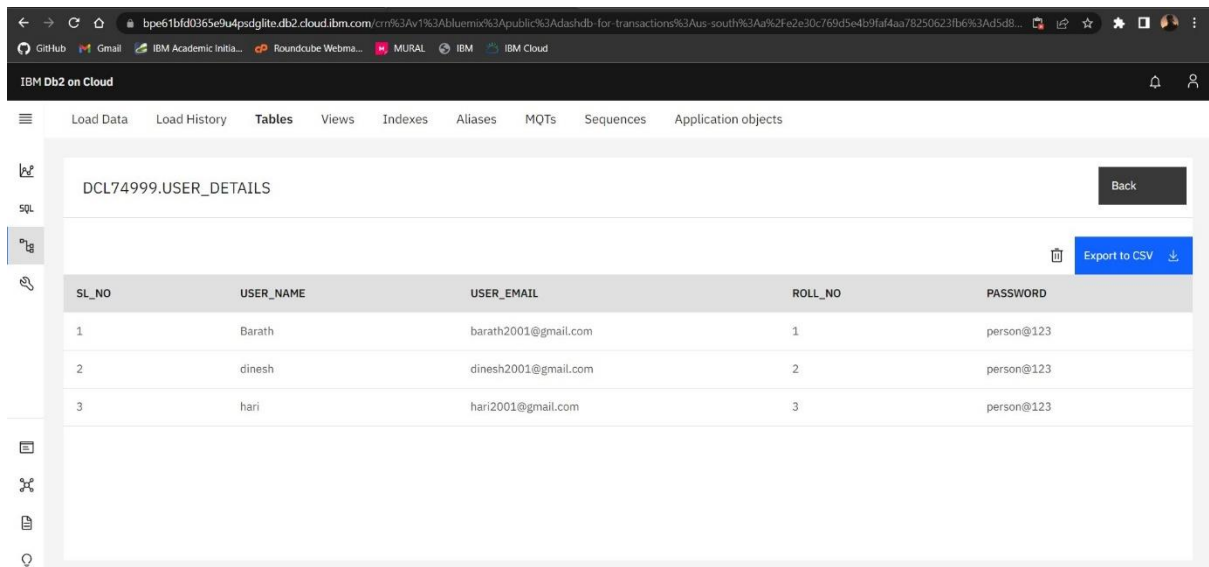
The screenshot shows the IBM Db2 on Cloud web interface. The SQL editor contains the following SQL statements:

```
14  
15  
16  
17  
18 insert into user_details(user_name,user_email,roll_no,password) values('Barath','barath2001@gmail.com',01,'person@123');  
19 insert into user_details(user_name,user_email,roll_no,password) values('dinesh','dinesh2001@gmail.com',02,'person@123');  
20 insert into user_details(user_name,user_email,roll_no,password) values('hari','hari2001@gmail.com',03,'person@123');  
21
```

The History tab is active, showing the execution of the scripts:

Script	Date	Status	Runtime
Assignment 2	Oct 17, 2022 9:49:00 PM	✓ 3	0.098 s
insert into user_details(user_name,user_email,roll_no,password) values('Barath','barath2001@gmail.com...		✓	0.020 s
insert into user_details(user_name,user_email,roll_no,password) values('dinesh','dinesh2001@gmail.com...		✓	0.059 s
insert into user_details(user_name,user_email,roll_no,password) values('hari','hari2001@gmail.com',03...		✓	0.019 s
Assignment 2	Oct 17, 2022 9:46:21 PM	✓ 1	0.111 s

User Table:

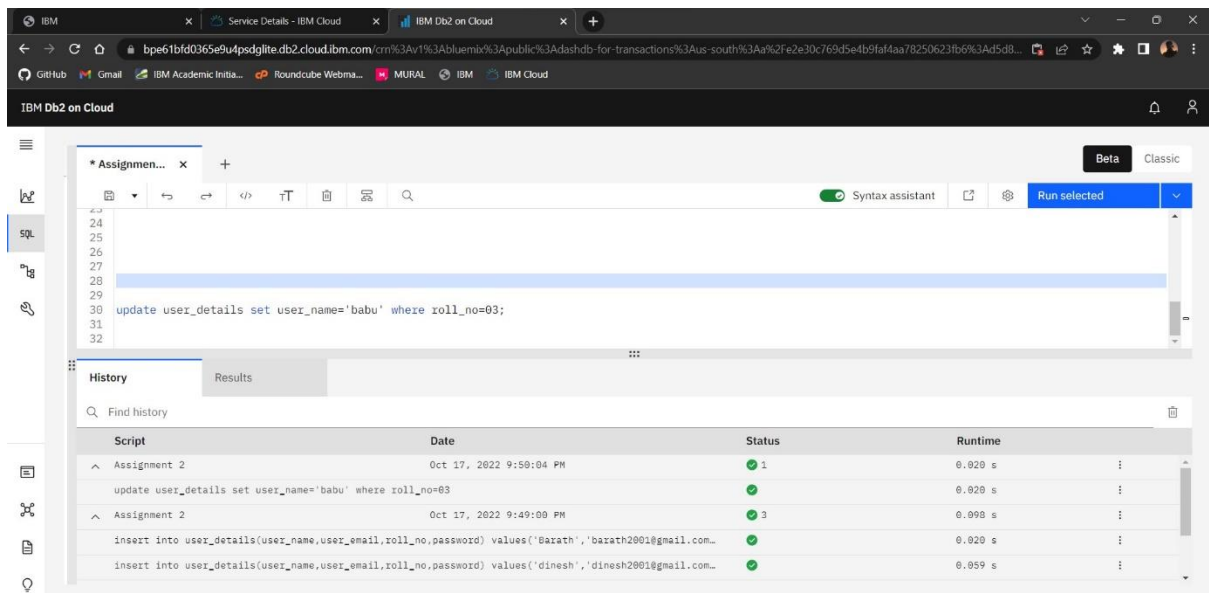


The screenshot shows the IBM Db2 on Cloud console interface. The top navigation bar includes 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The 'Tables' tab is selected, displaying a table named 'DCL74999.USER_DETAILS'. A 'Back' button is in the top right. Below the table name, there is an 'Export to CSV' button. The table structure is as follows:

SL_NO	USER_NAME	USER_EMAIL	ROLL_NO	PASSWORD
1	Barath	barath2001@gmail.com	1	person@123
2	dinesh	dinesh2001@gmail.com	2	person@123
3	hari	hari2001@gmail.com	3	person@123

2. Perform UPDATE, DELETE Queries with user table

Update:



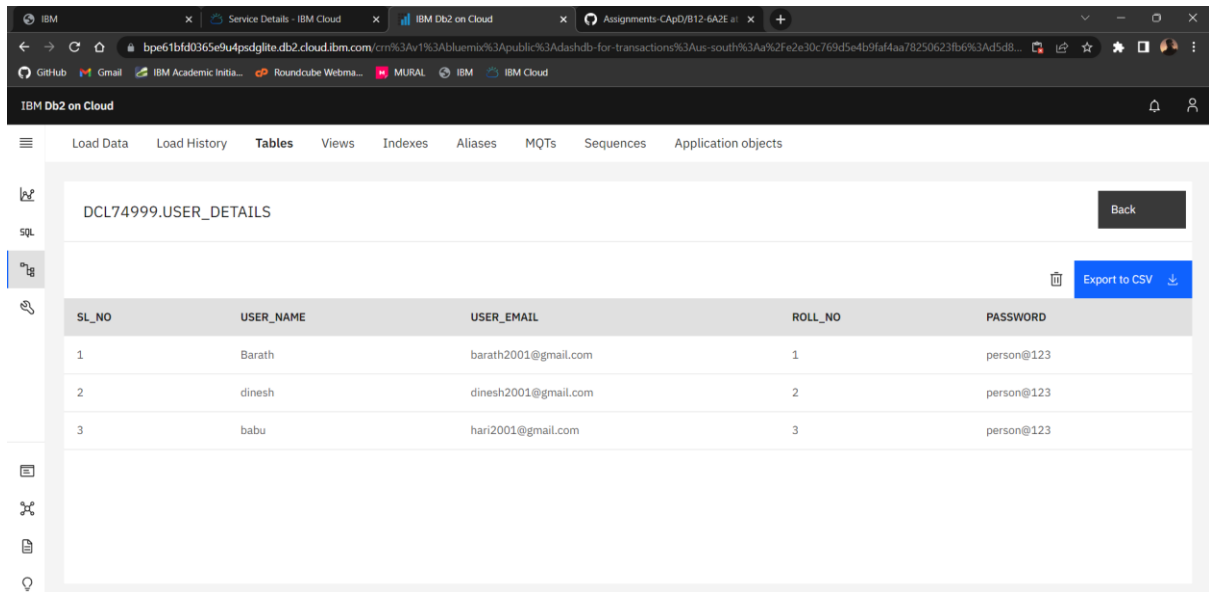
The screenshot shows the IBM Db2 on Cloud console with an SQL editor and a history table. The SQL editor contains the following query:

```
update user_details set user_name='babu' where roll_no=03;
```

The history table shows the execution of this query and other related queries:

Script	Date	Status	Runtime
Assignment 2	Oct 17, 2022 9:59:04 PM	1	0.020 s
update user_details set user_name='babu' where roll_no=03			0.020 s
Assignment 2	Oct 17, 2022 9:49:00 PM	3	0.090 s
insert into user_details(user_name,user_email,roll_no,password) values('Barath','barath2001@gmail.com...			0.020 s
insert into user_details(user_name,user_email,roll_no,password) values('dinesh','dinesh2001@gmail.com...			0.059 s

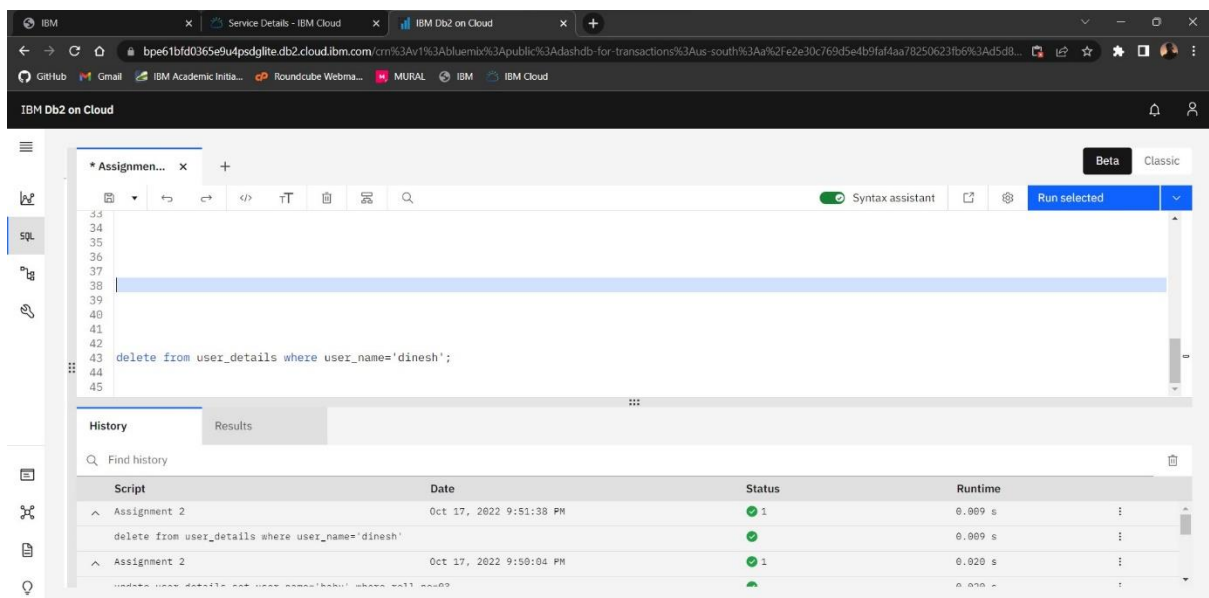
Updated Table:



The screenshot shows the IBM Db2 on Cloud web interface. The top navigation bar includes 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The 'Tables' tab is selected, displaying a table named 'DCL74999.USER_DETAILS'. A 'Back' button is in the top right. An 'Export to CSV' button is in the top right corner of the table area. The table has five columns: 'SL_NO', 'USER_NAME', 'USER_EMAIL', 'ROLL_NO', and 'PASSWORD'. It contains three rows of data.

SL_NO	USER_NAME	USER_EMAIL	ROLL_NO	PASSWORD
1	Barath	barath2001@gmail.com	1	person@123
2	dinesh	dinesh2001@gmail.com	2	person@123
3	babu	hari2001@gmail.com	3	person@123

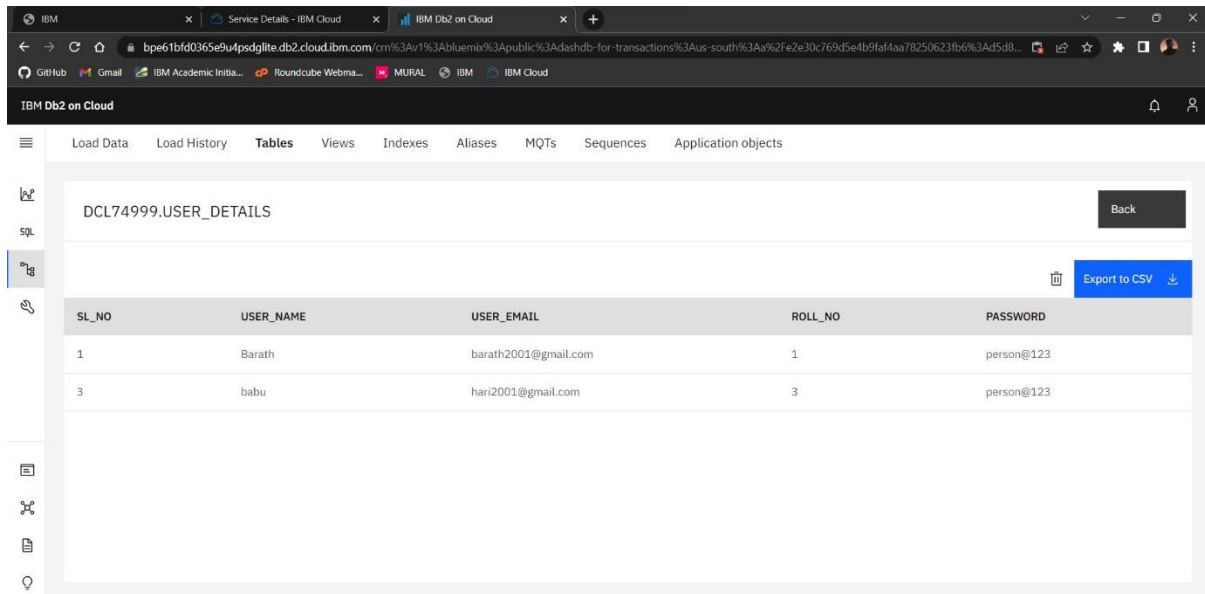
Delete:



The screenshot shows the IBM Db2 on Cloud web interface with a SQL script being executed. The script is: `delete from user_details where user_name='dinesh';`. The 'History' tab is selected, showing a table of execution history. The table has four columns: 'Script', 'Date', 'Status', and 'Runtime'. It shows three execution records, all with a status of '1' and a runtime of 0.009 s or 0.020 s.

Script	Date	Status	Runtime
Assignment 2 delete from user_details where user_name='dinesh'	Oct 17, 2022 9:51:38 PM	1	0.009 s
Assignment 2 delete from user_details where user_name='dinesh'	Oct 17, 2022 9:50:04 PM	1	0.020 s
Assignment 2 delete from user_details where user_name='dinesh'	Oct 17, 2022 9:50:04 PM	1	0.020 s

Table after performed delete query



The screenshot shows the IBM Db2 on Cloud web interface. The table 'DCL74999.USER_DETAILS' is displayed with the following data:

SL_NO	USER_NAME	USER_EMAIL	ROLL_NO	PASSWORD
1	Barath	barath2001@gmail.com	1	person@123
3	babu	hari2001@gmail.com	3	person@123

3. Connect python code to db2.

```
import ibm_db
import bcrypt
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=31321;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;PROTOCOL=TCP IP;UID=dcl74999;PWD=Px6tqEmXL9AcQigs","")
```

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 with username and password. If the user is valid show the welcome page

App.py

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

```
import ibm_db
import bcrypt
```

```
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=31321;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;PROTOCOL=TCP;IP;UID=dcl74999;PWD=Px6tqEmXL9AcQigs","")
```

```
# url_for('static', filename='style.css')
```

```
app = Flask(__name__)
app.secret_key = b'_5#y2L"F4Q8z\n\xec]/'
```

```
@app.route("/", methods=['GET'])
def home():
    if 'email' not in session:
        return redirect(url_for('login'))
    return render_template('home.html', name='Home')
```

```
@app.route("/register", methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        email = request.form['email']
        username = request.form['username']
        rollNo = request.form['rollNo']
        password = request.form['password']

        if not email or not username or not rollNo or not password:
            return render_template('register.html', error='Please fill all fields')
```

```
        hash=bcrypt.hashpw(password.encode('utf-8'),bcrypt.gensalt())
```

```
        query = "SELECT * FROM USER WHERE email=? OR rollNo=?"
        stmt = ibm_db.prepare(conn, query)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.bind_param(stmt, 2, rollNo)
        ibm_db.execute(stmt)
        isUser = ibm_db.fetch_assoc(stmt)
```

```
        if not isUser:
            insert_sql = "INSERT INTO USER(EMAIL, USERNAME, ROLLNO, PASSWORD)
VALUES (?, ?, ?, ?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prepare_stmt, 1, email)
            ibm_db.bind_param(prepare_stmt, 2, username)
```

```

        ibm_db.bind_param(prepare_stmt, 3, rollNo)
        ibm_db.bind_param(prepare_stmt, 4, hash)
        ibm_db.execute(prepare_stmt)
        return render_template('register.html',success="You can login")
    else:
        return render_template('register.html',error='Invalid Credentials')

return render_template('register.html',name='Home')

@app.route("/login",methods=['GET','POST'])
def login():
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']

        if not email or not password:
            return render_template('login.html',error='Please fill all fields')
        query = "SELECT * FROM USER WHERE email=?"
        stmt = ibm_db.prepare(conn, query)
        ibm_db.bind_param(stmt,1,email)
        ibm_db.execute(stmt)
        isUser = ibm_db.fetch_assoc(stmt)
        print(isUser,password)

        if not isUser:
            return render_template('login.html',error='Invalid Credentials')

        isPasswordMatch = bcrypt.checkpw(password.encode('utf-8'),isUser['PASSWORD'].encode('utf-8'))

        if not isPasswordMatch:
            return render_template('login.html',error='Invalid Credentials')

        session['email'] = isUser['EMAIL']
        return redirect(url_for('home'))

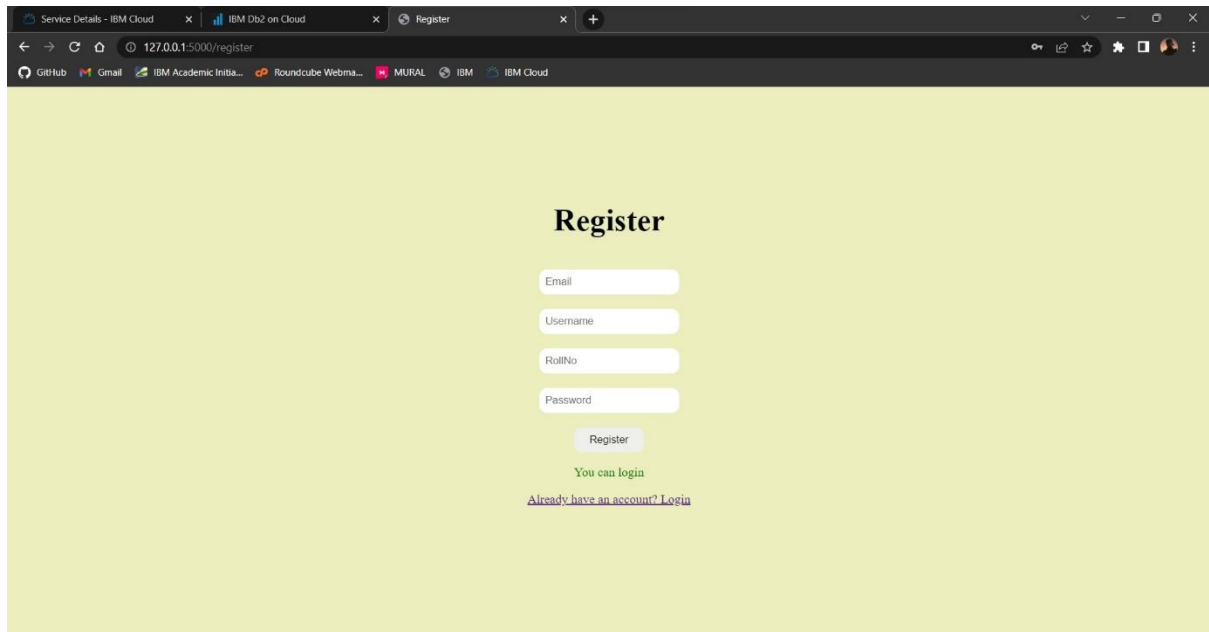
return render_template('login.html',name='Home')

@app.route('/logout')
def logout():
    session.pop('email', None)
    return redirect(url_for('login'))
if __name__ == "__main__":
    app.run(debug=True)

```

OUTPUT:

Registering:



The screenshot shows a web browser window with the address bar displaying "127.0.0.1:5000/register". The browser's tab bar includes "Service Details - IBM Cloud", "IBM Db2 on Cloud", and "Register". The page has a light green background and a central "Register" heading. Below the heading are four input fields: "Email", "Username", "RollNo", and "Password". A "Register" button is positioned below these fields. Underneath the button, there is a green link "You can login" and a purple link "Already have an account? Login".

Register

Email

Username

RollNo

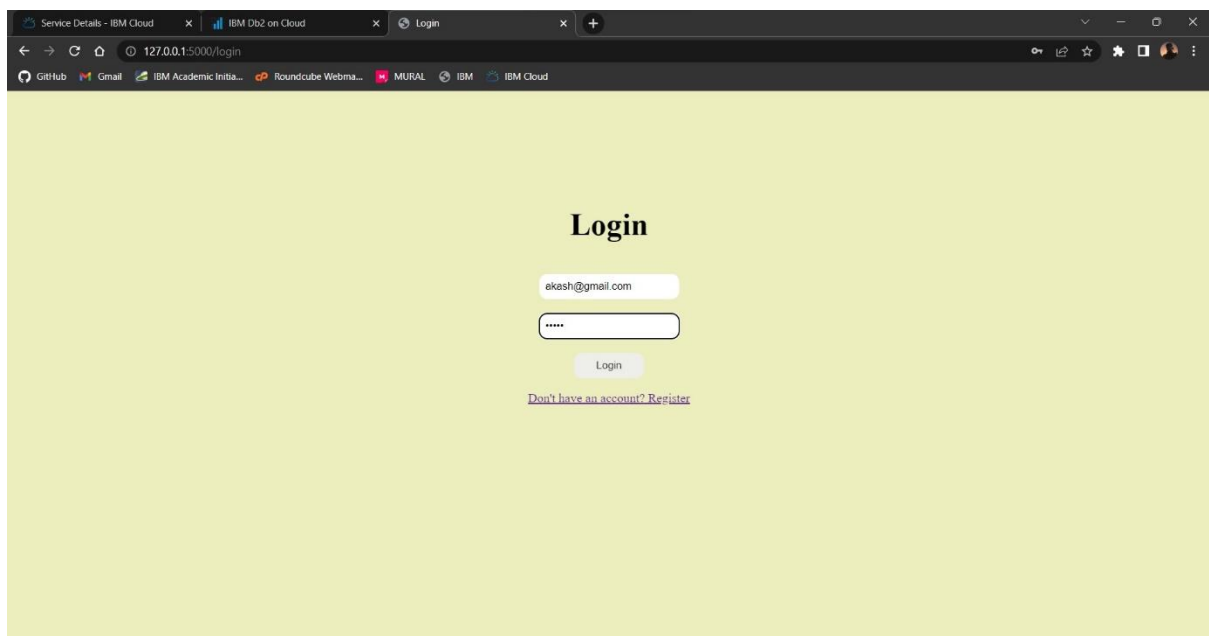
Password

Register

[You can login](#)

[Already have an account? Login](#)

Logging in:



The screenshot shows a web browser window with the address bar displaying "127.0.0.1:5000/login". The browser's tab bar includes "Service Details - IBM Cloud", "IBM Db2 on Cloud", and "Login". The page has a light green background and a central "Login" heading. Below the heading are two input fields: one for the email address (containing "ekash@gmail.com") and one for the password (displayed as "*****"). A "Login" button is positioned below these fields. Underneath the button, there is a purple link "Don't have an account? Register".

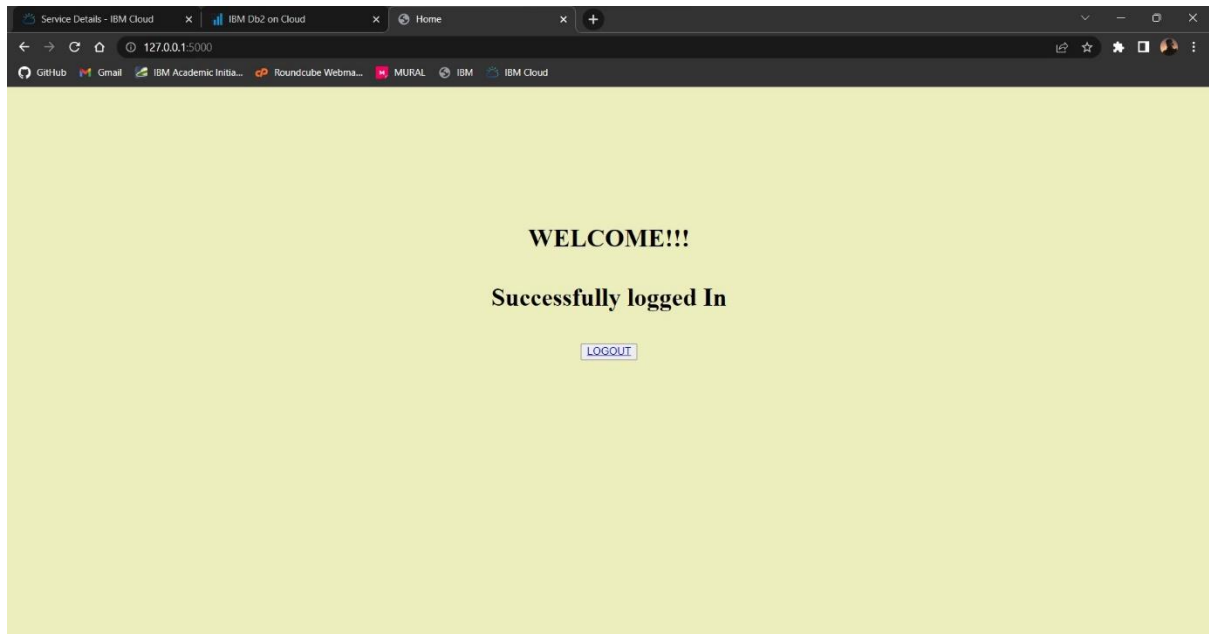
Login

ekash@gmail.com

Login

[Don't have an account? Register](#)

Home page:



Database:

