Assignment -2

Python to db2.

Assignment Date	19 September 2022
Student Name	DEVANAND V
Student Roll Number	7376191CS151
Maximum Marks	2 Marks

Question:

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

2. Perform UPDATE, DELETE Queries with user table

3. Connect python code to db2.

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

Program:

db2_config.py

The configrations for db2.

```
import ibm_db
import sys

conn = ''
def get_connection():
    db_name = "bludb"
    db_host_name = "6667d8e9-9d4d-4ccb-ba32-
21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud"
    db_port = "30376"
    db_protocol = "tcpip"
    db_username = "jzt12971"
    db_password = "dAmITuIrVMzd1jkp"

    try:
        connection_str =
f"database={db_name};hostname={db_host_name};port={db_port};protocol={db_protocol
};uid={db_username};pwd={db_password};security=ssl"
        # conn_str =
f"DATABASE={db_name};HOSTNAME={db_host_name};PORT={db_port};SECURITY=SSL;SSLServe
```

```
rCertificate=<FULL_PATH_TO_SERVER_CERTIFICATE>;UID={db_username};PWD={db_password
}"
        connection = ibm_db.connect(connection_str, '', '')

        # sql = " INSERT INTO  'JZT12971'.'USERS'
('ID','USERNAME','EMAIL','ROLLNO','PASSWORD') VALUES( ?, ?, ?, ?,?)"
        return connection
    except:
        print("Connection failed:", ibm_db.conn_errormsg())
        sys.exit(1)
```

db2_operation.py

To perform db2 operations.

```
import ibm_db
import json


def insert_user_data(conn, details):
    sql = 'INSERT INTO  "JZT12971"."USERS"
("ID","USERNAME","EMAIL","ROLLNO","PASSWORD") VALUES(seq_user.nextval, ?, ?, ?
,?);'
    stmt = ibm_db.prepare(conn, sql)
    # for i in range(1, len(details)):
    #      print(i, details[i - 1])
    for i in range(0, len(details)):
        ibm_db.bind_param(stmt, i + 1, details[i])
    ibm_db.execute(stmt)


def isAuthenticate(conn, username, 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)
    return ibm_db.fetch_assoc(stmt)

def isUserExists(conn, username):
    sql = "SELECT * FROM users where username = ?;"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt, 1, username)
    ibm_db.execute(stmt)
    acc = ibm_db.fetch_assoc(stmt)
    if acc == False:
        return acc
    else:
        print(acc)
        return (username == acc['USERNAME'].strip())
    # print(acc['USERNAME'], ".")
    # print(username == acc['USERNAME'].strip())

    return acc
    # return ibm_db.fetch_assoc(stmt)
```

auto_increment_id.sql

To increment user id by 1 for every new user entry.

```
CREATE SEQUENCE seq_user
MINVALUE 1
START WITH 1
INCREMENT BY 1
CACHE 100;
```

app.py

```
import traceback
from flask import Flask, request, render_template, session, redirect, url_for,
flash
from db2_config import get_connection
from db2_operation import insert_user_data, isAuthenticate, isUserExists
from flask_debugtoolbar import DebugToolbarExtension
app = Flask(__name__)
app.config['SECRET_KEY'] = 'a'
conn = get_connection()


@app.route('/')
@app.route('/registration', methods=["GET", "POST"])
def registration():
    msg = ""
    details = []
    if request.method == "POST":
        name = request.form.get("name")
        email = request.form.get("email")
        rollno = request.form.get("rollno")
        password = request.form.get("password")
        details = [name, email, rollno, password]
        # conn = get_connection(details=details)
        try:
            acc = isUserExists(conn=conn, username=name)
            print("register acc:", acc)
            if acc == False:
                insert_user_data(conn=conn, details=details)
                return redirect(url_for('login'))
            else:
                msg = f"User {name} already exists"
                flash(msg)
        except Exception as exp:
            print("insert failed", exp.__traceback__)
            traceback.print_exc()
            # return "Your details is " + name + " " + email + " " + number
    # return render_template("form.html", data = details)
    return render_template('register.html')
# details = []


@app.route('/login', methods=["GET", "POST"])
def login():
    global user_id
    msg = ''
    print("lg")
    if request.method == 'POST':
        print("post method")
        username = request.form.get("name")
        password = request.form.get('password')
        print(username, password)
        try:
            account = isAuthenticate(conn=conn, username=username,
password=password)
            print("statement executed")
            # account = isAuthenticate(username=username, password=password)
            print("login satus", account)
            if (account):
                print(f"acc user name = {account['USERNAME']}")
                session['loggedin'] = True
                session['id'] = account['USERNAME']
                session['USERNAME'] = account['USERNAME']
                user_id = account['USERNAME']

                msg = "Login sucessfull"
                return redirect(url_for('welcome'))
            else:
                msg = "login failed"
            print(msg)
            flash(msg)
        except Exception as exp:
            print("Authentication failed", exp.__traceback__)
        # account = isAuthenticate(username=username, password=password)
      #   return msg
    return render_template('login.html', msg=msg)


@app.route('/welcome')
def welcome():
    return f"<h1>Welcome {user_id}!</h1>"


if __name__ == '__main__':
    # toolbar = DebugToolbarExtension(app)
    app.debug = True
    app.run()
```

Templates:

base.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>{% block title %}{% endblock %}</title>
    <link href="{{ url_for('static', filename='css/stylesheet.css') }}"
rel="stylesheet">
</head>
<body>
{% block main %}
{% endblock %}
</body>
</html>
```

Login.html

```
{% extends 'base.html' %}
{% block title %}Login{% endblock %}
{% block main %}
<!-- <h1>{{msg}}</h1> -->
{% include 'flash.html' %}
<form action="{{ url_for('login')}}" method="post">
    <fieldset>
        <legend>Login Form:</legend>
        <label for="name">User Name:</label><br>
        <input type="text" name="name" id="name"><br>
        <label for="password">Password:</label><br>
        <input type="password" name="password" id="password"><br>
        <button type="submit">submit</button>
    </fieldset>
</form>
{% endblock %}
```

register.html

```
{% extends 'base.html' %}
{% block title %}Register{% endblock %}
{% block main %}
{% include 'flash.html' %}
<form action="{{ url_for('registration')}}" method="post">
    <fieldset>
        <legend>Registration Form:</legend>
        <label for="name">User Name:</label><br>
        <input type="text" name="name" id="name"><br>
        <label for="email">Email</label><br>
        <input type="email" name="email" id="email"><br>
        <label for="rollno">Roll No:</label> <br>
        <input type="text" name="rollno" id="rollno"><br>
        <label for="password">password</label><br>
        <input type="password" name="password" id="password"><br>
        <button type="submit">submit</button>
    </fieldset>
</form>
<div>
    <!-- <a href="{{url_for('login')}}"><button>Login</button></a> -->
</div>
{% endblock %}
```

flash.html

```
<div>
    {% for message in get_flashed_messages() %}
    <h2>{{ message }}</h2>
    {% endfor %}
</div>
```

Login Page:

```
{% extends 'base.html' %}
{% block title %}Login{% endblock %}
{% block main %}
<!-- <h1>{{msg}}</h1> -->
{% include 'flash.html' %}
<form action="{{ url_for('login')}}" method="post">
    <fieldset>
        <legend>Login Form:</legend>
        <label for="name">User Name:</label><br>
        <input type="text" name="name" id="name"><br>
        <label for="password">Password:</label><br>
        <input type="password" name="password" id="password"><br>
        <button type="submit">submit</button>
    </fieldset>
</form>
{% endblock %}
```

Static:

Stylesheet.css

```
*, *:before, *:after {
    -moz-box-sizing: border-box;
    -webkit-box-sizing: border-box;
    box-sizing: border-box;
  }

  body {
    font-family: 'Nunito', sans-serif;
    color: #384047;
  }
```

```css
form {
  max-width: 300px;
  margin: 10px auto;
  padding: 10px 20px;
  background: #f4f7f8;
  border-radius: 8px;
}

h1,h2 {
  margin: 0 0 30px 0;
  text-align: center;
}

input[type="text"],
input[type="password"],
input[type="date"],
input[type="datetime"],
input[type="email"],
input[type="number"],
input[type="search"],
input[type="tel"],
input[type="time"],
input[type="url"],
textarea,
select {
  background: rgba(255,255,255,0.1);
  border: none;
  font-size: 16px;
  height: auto;
  margin: 0;
  outline: 0;
  padding: 15px;
  width: 100%;
  background-color: #e8eeef;
  color: #8a97a0;
  box-shadow: 0 1px 0 rgba(0,0,0,0.03) inset;
  margin-bottom: 30px;
}

input[type="radio"],
input[type="checkbox"] {
  margin: 0 4px 8px 0;
}
```

```css
select {
  padding: 6px;
  height: 32px;
  border-radius: 2px;
}

button {
  padding: 19px 39px 18px 39px;
  color: #FFF;
  background-color: #4bc970;
  font-size: 18px;
  text-align: center;
  font-style: normal;
  border-radius: 5px;
  width: 100%;
  border: 1px solid #3ac162;
  border-width: 1px 1px 3px;
  box-shadow: 0 -1px 0 rgba(255,255,255,0.1) inset;
  margin-bottom: 10px;
}

fieldset {
  margin-bottom: 30px;
  border: none;
}

legend {
  font-size: 1.4em;
  margin-bottom: 10px;
}

label {
  display: block;
  margin-bottom: 8px;
}

label.light {
  font-weight: 300;
  display: inline;
}

.number {
  background-color: #5fcf80;
  color: #fff;
  height: 30px;
```

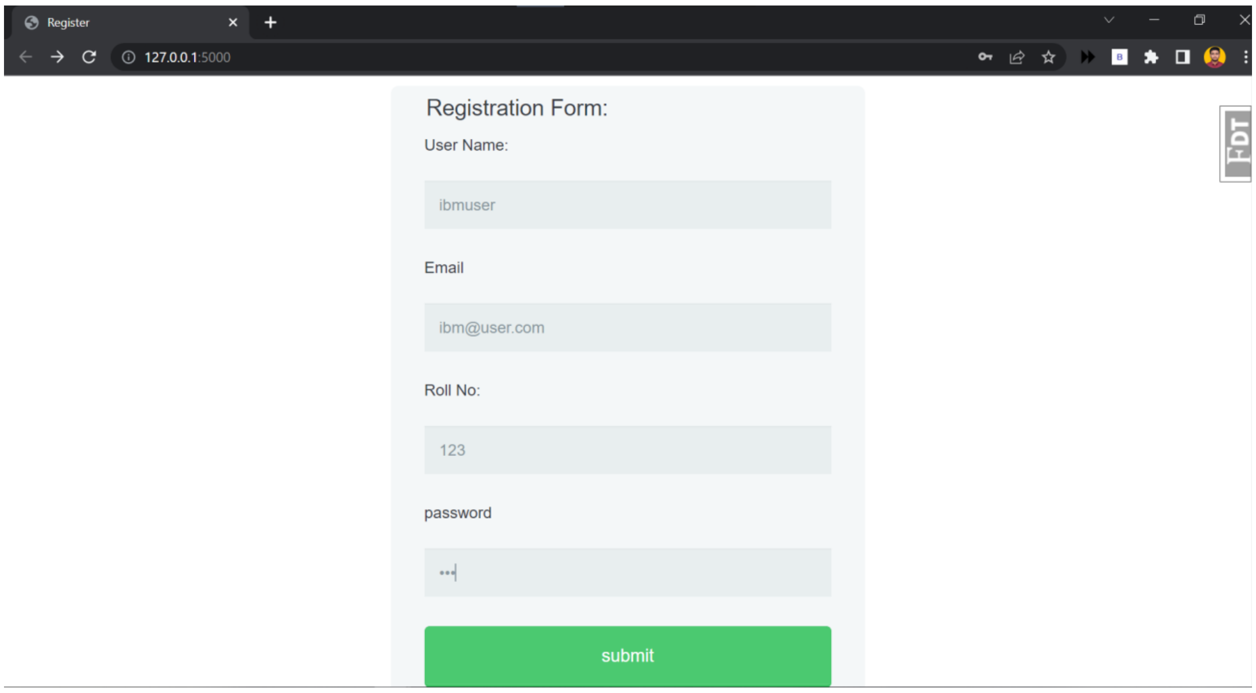
```
    width: 30px;
    display: inline-block;
    font-size: 0.8em;
    margin-right: 4px;
    line-height: 30px;
    text-align: center;
    text-shadow: 0 1px 0 rgba(255,255,255,0.2);
    border-radius: 100%;
  }

  @media screen and (min-width: 480px) {

    form {
      max-width: 480px;
    }

  }
```

Registration page



Registration Form:

User Name:

ibmuser

Email

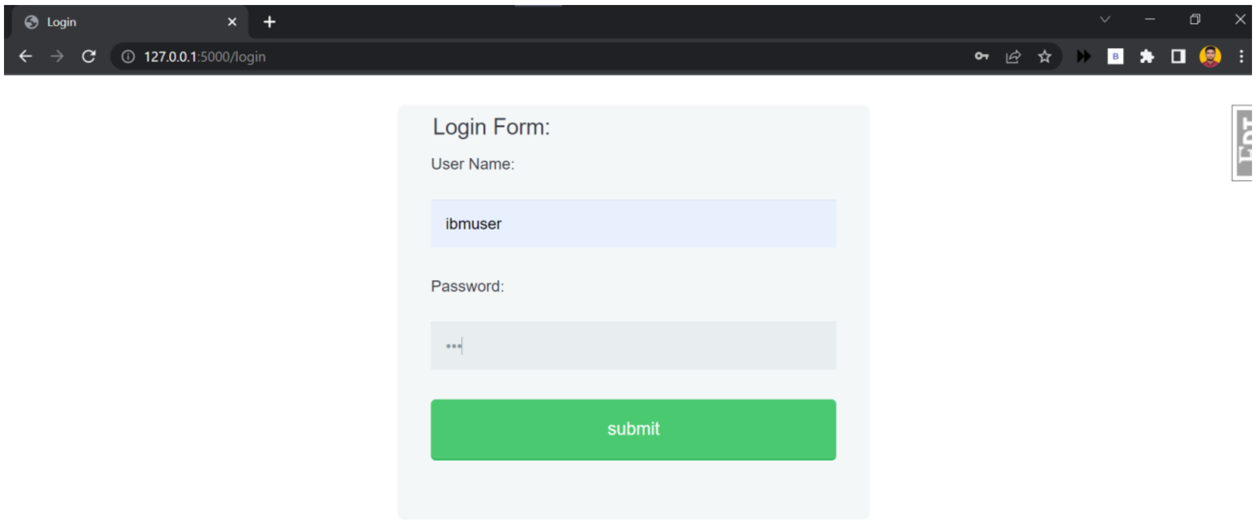
ibm@user.com

Roll No:

123

password

•••


Login

127.0.0.1:5000/login

Login Form:

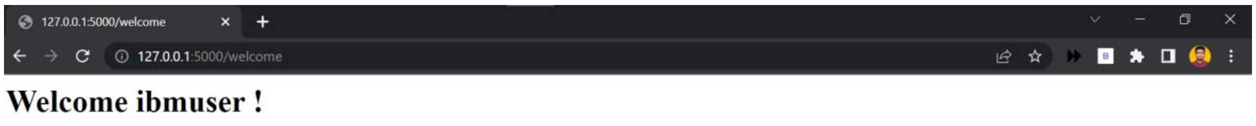
User Name:

ibmuser

Password:

•••

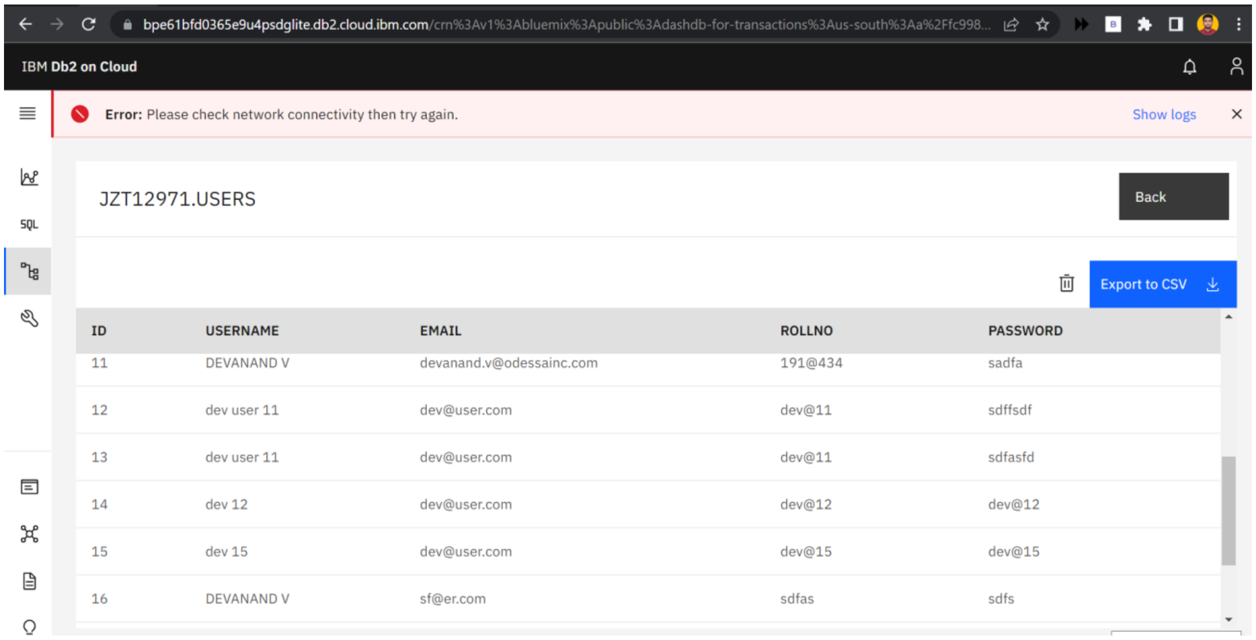
Welcome page:


127.0.0.1:5000/welcome

127.0.0.1:5000/welcome

Welcome ibmuser !

IBM Cloud user table.


bpe61bfd0365e9u4psdglite.db2.cloud.ibm.com/crn%3Av1%3Abluemix%3Apublic%3Adashdb-for-transactions%3Aus-south%3Aa%2Ffc998...

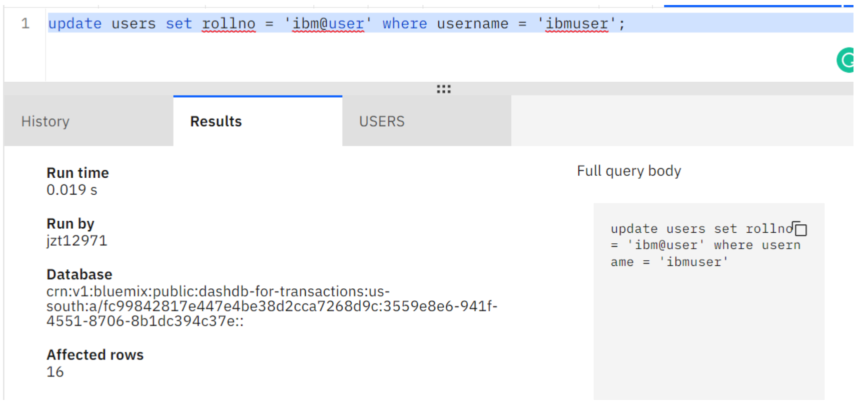
IBM Db2 on Cloud

Error: Please check network connectivity then try again.

JZT12971.USERS

ID	USERNAME	EMAIL	ROLLNO	PASSWORD
11	DEVANAND V	devanand.v@odessainc.com	191@434	sadfa
12	dev user 11	dev@user.com	dev@11	sdffsdf
13	dev user 11	dev@user.com	dev@11	sdfasfd
14	dev 12	dev@user.com	dev@12	dev@12
15	dev 15	dev@user.com	dev@15	dev@15
16	DEVANAND V	sf@er.com	sdfas	sdfs

Update statement:


```
update users set rollno = 'ibm@user' where username = 'ibmuser';
```

Run time
0.019 s

Run by
jzt12971

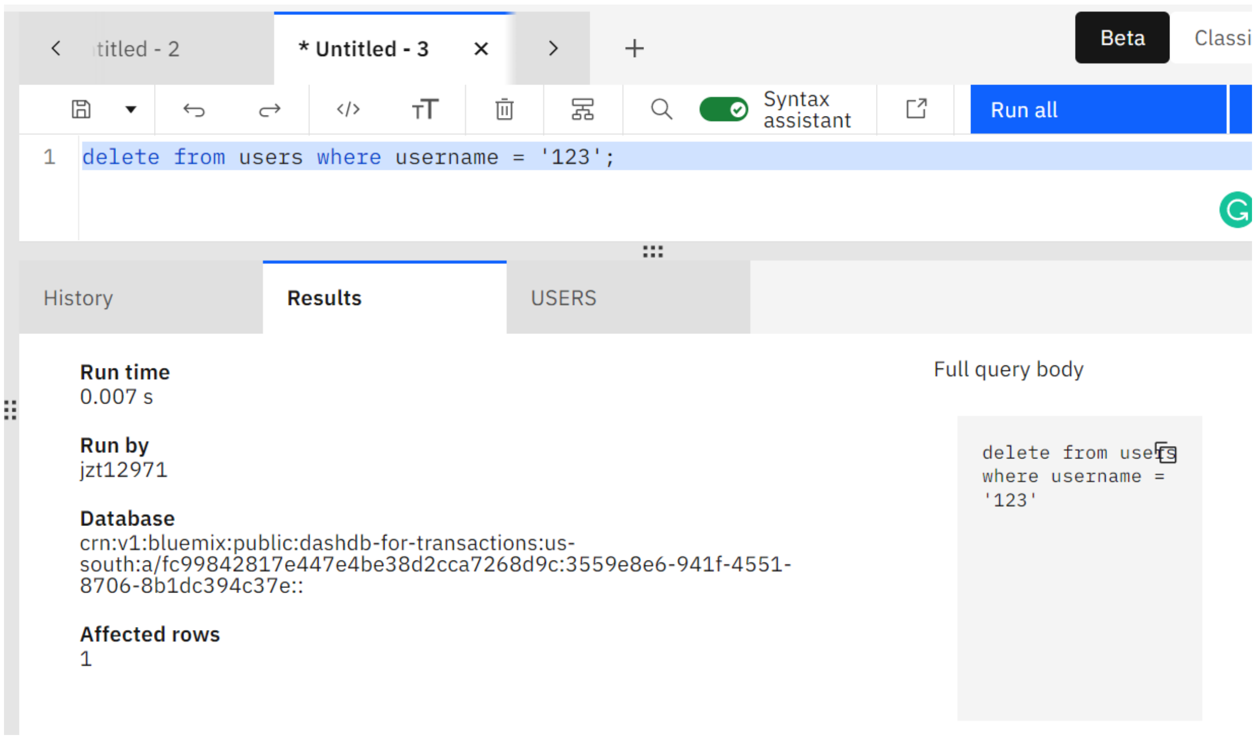
Database
crn:v1:bluemix:public:dashdb-for-transactions:us-south:a/fc99842817e447e4be38d2cca7268d9c:3559e8e6-941f-4551-8706-8b1dc394c37e::

Affected rows
16

Full query body

```
update users set rollno
= 'ibm@user' where usern
ame = 'ibmuser'
```

Delete statement:



```
delete from users where username = '123';
```

Run time
0.007 s

Run by
jzt12971

Database
crn:v1:bluemix:public:dashdb-for-transactions:us-south:a/fc99842817e447e4be38d2cca7268d9c:3559e8e6-941f-4551-8706-8b1dc394c37e::

Affected rows
1

Full query body

```
delete from users
where username =
'123'
```