

Assignment-2

PROJECT NAME :	Containment Zone Alerting App
NAME :	V. NATESH KRISHNAN
ROLL NO :	92172019104102

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

```
CREATE TABLE user (  
    roll_number int,  
    username varchar(300),  
    email varchar(300),  
    password varchar(300)  
);
```

```
CREATE TABLE users (  
    roll_number int,  
    username varchar(300),  
    email varchar(300),  
    password varchar(300)  
);
```

Output

SQL query successfully executed. However, the result set is empty.

2. Perform UPDATE, DELETE Queries with user table

INSERT Statement:

INSERT INTO user

```
( roll_number, username ,email, password) VALUES (1,  
'Raja  lingam', 'rajalingam@gmail.com','raju987'), (2,  
'Ajay', 'ajay@gmail.com','ajay654'),  
(3, 'Anton', 'anton@gmail.com','anton321'),  
(4, 'Prasanth', 'prasanth@gmail.com','prasanth123');
```

```
INSERT INTO user  
  ( roll_number, username ,email, password) VALUES  
  (1, 'Raja  lingam', 'rajalingam@gmail.com','raju987'),  
  (2, 'Ajay', 'ajay@gmail.com','ajay654'),  
  (3, 'Anton', 'anton@gmail.com','anton321'),  
  (4, 'Prasanth', 'prasanth@gmail.com','prasanth123');
```

Output

Available Tables

5	Delivered	1
---	-----------	---

User

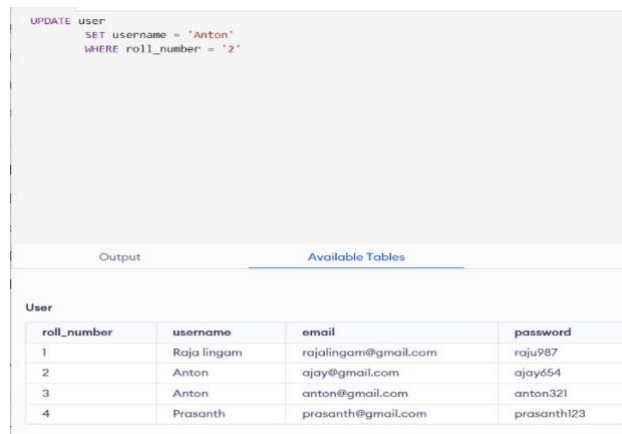
roll_number	username	email	password
1	Raja lingam	rajalingam@gmail.com	raju987
2	Ajay	ajay@gmail.com	ajay654
3	Anton	anton@gmail.com	anton321
4	Prasanth	prasanth@gmail.com	prasanth123

UPDATE Statement:

UPDATE users

SET username = 'Anton'

WHERE roll_number = '2'



The screenshot shows a database interface with a text area containing the following SQL statement:

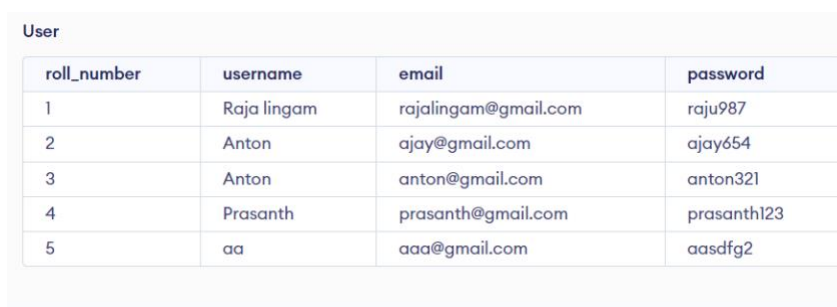
```
UPDATE user
SET username = 'Anton'
WHERE roll_number = '2'
```

Below the text area, there are two tabs: "Output" and "Available Tables". The "Available Tables" tab is selected, showing a table named "User".

roll_number	username	email	password
1	Raja lingam	rajalingam@gmail.com	raju987
2	Anton	ajay@gmail.com	ajay654
3	Anton	anton@gmail.com	anton321
4	Prasanth	prasanth@gmail.com	prasanth123

Insert Statement:

insert into users values(5,'aa','aaa@gmail.com','aasdfg2') ;



The screenshot shows a database interface with a table named "User".

roll_number	username	email	password
1	Raja lingam	rajalingam@gmail.com	raju987
2	Anton	ajay@gmail.com	ajay654
3	Anton	anton@gmail.com	anton321
4	Prasanth	prasanth@gmail.com	prasanth123
5	aa	aaa@gmail.com	aasdfg2

DELETE Statement:

delete from users where roll_number='5'

User

roll_number	username	email	password
1	Raja lingam	rajalingam@gmail.com	raju987
2	Anton	ajay@gmail.com	ajay654
3	Anton	anton@gmail.com	anton321
4	Prasanth	prasanth@gmail.com	prasanth123

3. Connect python code to db2.

```
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=30119;Security=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=lbs14903;PWD=1N4walQ5ywwiwp7c;", "", "")
```

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.

login.html

```
<!DOCTYPE html>

<html>

<head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">


    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-
awesome/4.7.0/css/font-awesome.min.css">


    <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css')
}}">


    <title>Login Form</title>

</head>

<body>

    <div class="container">

        <form action="{{ url_for('login') }}" method="POST"
class="login-email">

            <p class="login-text" style="font-size: 2rem; font-
weight: 800;">Login</p>

            <div class="input-group">

                <input id="username" type="text"
placeholder="Enter username" name="username">

            </div>

            <div class="input-group">

                <input id="password" type="password"
placeholder="Password" name="password">
```

```

        </div>

        <div class="input-group">
class="btn">Login</button>            <button name="submit"

        </div>

        <p class="login-register-text">Don't have an account?
<a href="{{ url_for('register')}}">Register Here</a>.</p>

        </form>

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

```

register.html:

```

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-
awesome/4.7.0/css/font-awesome.min.css">

    <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css')
}}">

    <title>Register Form</title>

</head>
<body>
    <div class="container" >

```

```

        <form action="{{ url_for('register') }}" method="POST" class="login-email">
        <p class="login-text" style="font-size: 2rem; font-weight: 800;">Register</p>
            <div class="msg">{{ msg }}</div>
            <div class="input-group">
                <input type="text" placeholder="Username"
name="username">
            </div>
            <div class="input-group">
                <input type="email" placeholder="Email" name="email">
            </div>
            <div class="input-group">
                <input type="password" placeholder="Password"
name="password">
            </div>
            <div class="input-group">
                <button type="submit" class="btn">Register</button>
            </div>
            <p class="login-register-text">Have an account? <a
href="{{ url_for('login') }}">Login Here</a>.</p>
        </form>
    </div>
</body>
</html>

```

index.html:

```

<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">
page...      Hi {{session.username}}!!</br></br> Welcome to Our

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

  </div>
</div>

</body>

</html>
```

style.css:

```
.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: #FFFFFF;
```

```
font-style: oblique;
```

```
font-weight: bold;
```

```
border-radius: 10px;
```

```
}
```

```
.textbox{
```

```
padding: 10px 40px;
```

```
background-color: #236B8E;
```

```
text-color: #FFFFFF;
```

```
border-radius: 10px;
```

```
}
```

```
::placeholder {
```

```
color: #FFFFFF;
```

```
opacity: 1;
```

```
font-style: oblique;
```

```
font-weight: bold;
```

```
}
```

```
.word{
```

```
color: #FFFFFF;
```

```
font-style: oblique;
```

```
font-weight: bold;
```

```
}
```

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

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=824dfd4d-99de-440d-9991-  
629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=30119;Security=S  
SL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=lbs14903;PWD=1N4walQ5ywwi  
wP7c;", "", "")  
  
@app.route('/')  
@app.route('/login', methods =['GET', 'POST'])  
def login():  
    msg = "  
  
    if request.method == 'POST' and 'username' in request.form and 'password' in  
request.form:  
        username = request.form['username']  
        password = request.form['password']  
  
        stmt = ibm_db.prepare(conn,'SELECT * FROM users WHERE username = ? AND  
password = ?')  
  
        ibm_db.bind_param(stmt,1,username)  
        ibm_db.bind_param(stmt,2,password)  
  
        ibm_db.execute(stmt)
```

```

account = ibm_db.fetch_assoc(stmt)

if account:
    session['loggedin'] = True
    session['username'] = account['USERNAME']
    msg = 'Logged in successfully !'
    return render_template('index.html', msg = msg)
else:
    msg = 'Incorrect username / 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']
        password = request.form['password']
        email = request.form['email']

        stmt = ibm_db.prepare(conn,'SELECT * FROM users WHERE username = ?')
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        if account:
            msg = 'Account already exists !'

```

```
elif not re.match(r'^[^\s@]+@[^\s@]+\.[^\s@]+', 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:
    prep_stmt = ibm_db.prepare(conn,"INSERT INTO users VALUES(?, ?, ?)")
    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)

if __name__ == '__main__':
    app.debug = True
    app.run(host='0.0.0.0',port=8080)
```

Output:

The screenshot shows a web browser window with the address bar displaying `127.0.0.1:8080/register`. The browser's tab bar includes tabs for 'Error', 'Error', 'Register Form', and '+'. The address bar also shows several bookmarks: 'Gmail', 'YouTube', 'IBM-EPBL/IBM-Proj...', 'Programming In Ja...', 'IBM', 'Catalog - IBM Cloud', 'Service Details - IBM', and 'IBM Db2 on Cloud'. The main content area features a white 'Register' form on a gray background. The form has the title 'Register' and a message 'You have successfully registered !'. It contains three input fields: 'Username', 'Email', and 'Password'. Below these fields is a blue 'Register' button. At the bottom of the form, there is a link: 'Have an account? Login Here.' The Windows taskbar at the bottom shows the system clock as 10:12 PM and the temperature as 29°C.

Register

You have successfully registered !

Username

Email

Password

Register

Have an account? [Login Here.](#)

The screenshot shows a web browser window with the address bar displaying `127.0.0.1:8080/login`. The browser's tab bar includes tabs for 'IBM Db2 on Cloud', 'Error', 'Login Form', and '+'. The address bar also shows the same set of bookmarks as the previous screenshot. The main content area features a white 'Login' form on a gray background. The form has the title 'Login' and contains two input fields: 'ashwinkumar123' (for Username) and 'Password'. Below these fields is a blue 'Login' button. A Google Chrome extension notification bubble is visible over the 'Login' button, stating 'Use passwords saved in your Google Account'. At the bottom of the form, there is a link: 'Don't have an account? Register Here.' The Windows taskbar at the bottom shows the system clock as 10:28 PM and the temperature as 29°C.

Login

ashwinkumar123

Password

Use passwords saved in your Google Account

Login

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

