

# IBM Assignment -2

Project Title : Personal Expense Tracker Application

Team ID : PNT2022TMID32657

Name: Nishanthi.R

Registration No : 813819104062

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

The screenshot displays the IBM Db2 on Cloud web interface. The 'Tables' tab is active, showing a list of tables. The 'USER' table is selected, and its definition is shown on the right. The table has four columns: EMAIL, USERNAME, ROLLNO, and PASSWORD, all of type VARCHAR with a length of 32 and a nullable status of Y.

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

The screenshot displays the IBM Db2 on Cloud web interface. The 'Data objects' tab is active, showing the 'USER' table. The 'SQL' tab is active, showing a script with four INSERT statements. The script has been executed, and the results are shown in the 'History' tab.

```
1 INSERT INTO USER VALUES('thiyagu@gmail.com', 'thiyagarajan', '221111', '2346');
2 INSERT INTO USER VALUES('sabari@gmail.com', 'sabari', '221079', '2346');
3 INSERT INTO USER VALUES('sujitha@gmail.com', 'sujitha', '221103', '2346');
4 INSERT INTO USER VALUES('nishanthi@gmail.com', 'nishanthi', '221062', '2346');
```

Script	Date	Status	Runtime
INSERT INTO USER VALUES('thiyagu@gmail.com', 'thiyagar...		✓	0.005 s
INSERT INTO USER VALUES('sabari@gmail.com', 'sabari', ...		✓	0.003 s
INSERT INTO USER VALUES('sujitha@gmail.com', 'sujitha'...		✓	0.003 s
INSERT INTO USER VALUES('nishanthi@gmail.com', 'nisan...		✓	0.003 s

## 2) A) Perform UPDATE Query with user table

The screenshot shows the IBM Db2 on Cloud SQL editor interface. The left sidebar displays the 'Data objects' tab with a search filter 'SZT06394'. The main editor area shows a query in a file named '\*Untitled - 1':

```
UPDATE USER SET password='654321' WHERE email='thiyagu@gmail.com';
```

The 'History' tab at the bottom shows the execution history of the query:

Script	Date	Status	Runtime
Untitled - 1	Nov 1, 2022 7:52:08 PM	✓ 1	0.005 s
UPDATE USER SET password='654321' WHERE email='thiyagu@g...		✓	0.005 s

The screenshot shows the IBM Db2 on Cloud SQL editor interface displaying the results of the UPDATE query. The left sidebar shows the 'Data objects' tab with a search filter 'SZT06394.USER'. The main editor area shows the results of the query in a table:

EMAIL	USERNAME	ROLLNO	PASSWORD
nishanthi@gmail.com	nishanthi	221062	2346
sabari@gmail.com	sabari	221079	2346
sujitha@gmail.com	sujitha	221103	2346
thiyagu@gmail.com	thiyagarajan	221111	654321

The 'History' tab at the bottom shows the execution history of the query.

## B) Perform DELETE Query with user table

The screenshot displays the IBM Db2 on Cloud web interface. The top section shows a SQL editor with a query to delete a user from the 'USER' table. The query is: `DELETE FROM USER WHERE email='thiyagu@gmail.com';`. The query has been executed successfully, as indicated by the 'Run all' button and the 'History' tab showing a successful execution with a status of '1' and a runtime of 0.004 s.

The bottom section shows the 'Tables' tab with the 'SZT06394.USER' table selected. The table contains the following data:

EMAIL	USERNAME	ROLLNO	PASSWORD
nishanthi@gmail.com	nishanthi	221062	2346
sabari@gmail.com	sabari	221079	2346
sujitha@gmail.com	sujitha	221103	2346

- 3) Connect python code to db2.

#### Connect.py

```
import ibm_db

def get_db_connection():
    try:
        conn = ibm_db.connect("DATABASE=BLUDB;\
                               HOSTNAME=<hostname>;\
                               PORT=30376;\
                               Security=SSL;\
                               SSLServerCertificate=DigiCertGlobalRootCA.crt;\
                               UID=<username>;\
                               PWD=<password>;", "", "")
        print("Connected to DB")
        return conn
    except:
        print("error while connecting ", ibm_db.conn_errormsg())
        return 0
```

- 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, session, redirect, url_for
import ibm_db
from connect import get_db_connection

app = Flask(__name__)
app.secret_key = 'abcde'
con = 0

@app.route('/')
@app.route("/register", methods=['GET', 'POST'])
def register():
    if(request.method == 'GET'):
        if 'auth' in session:
            if session['auth']:
                return redirect(url_for('home'))
            return render_template("register_2.html")
    elif(request.method == 'POST'):
        name = request.form['username']
        email = request.form['email']
```

```

        password = request.form['password']
        rollno = request.form['rollno']
        # TODO: check if account already exist
        qry = "INSERT INTO USER VALUES(?,?,?,?)"
        stmt = ibm_db.prepare(con, qry)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.bind_param(stmt, 2, name)
        ibm_db.bind_param(stmt, 3, rollno)
        ibm_db.bind_param(stmt, 4, password)

        ibm_db.execute(stmt)
        return redirect(url_for('login'))
    else:
        return "404 not found"

@app.route('/login', methods=['GET', 'POST'])
def login():
    if(request.method == 'GET'):
        if 'auth' in session:
            if session['auth']:
                return redirect(url_for('home'))
            return render_template("login.html")
    elif(request.method == 'POST'):
        email=request.form['email']
        password=request.form['password']
        sql="SELECT * FROM USER WHERE email=? AND password=?"
        stmt=ibm_db.prepare(con, sql)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.bind_param(stmt, 2, password)
        ibm_db.execute(stmt)
        resp=ibm_db.fetch_assoc(stmt)
        print("resp - ", resp)
        if resp:
            session['auth'] = True
            session['email'] = email
            session['name'] = resp['USERNAME']
            # session['rollno'] = rollno

            return redirect(url_for('home'))
        else:
            return redirect(url_for('login'))
    else:
        return "404 not found"

@app.route('/home')
def home():
    if 'auth' in session:
        if not session['auth']:

```

```

        return redirect(url_for('login'))
    return render_template("home.html")

@app.route('/logout')
def logout():
    session.pop('email', None)
    session.pop('name', None)
    session['auth'] = False
    return redirect(url_for('login'))

if(__name__=="__main__"):
    con = get_db_connection()
    app.run(debug=True)

```

## home.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Home</title>
    <link rel="stylesheet" href="{{url_for('static', filename =
'style2.css')}}">
</head>
<body>
    <nav class="navbar">
        <h1 class="logo">MyMoney</h1>
        <ul class="nav-ul">
            <li class="nav-li"><a href="/logout">Logout</a></li>
        </ul>
    </nav>
    <div class="container">
        <h1 class="head">Welcome {{session['name']}} !!!</h1>
    </div>
</body>
</html>

```

## Register.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

```

```

    <link rel="stylesheet" href="{url_for('static', filename =
'style2.css')}}">
    <title>Register</title>
</head>
<body>
    <nav class="navbar">
        <h1 class="logo">MyMoney</h1>
        <ul class="nav-ul">
            <li class="nav-li"><a href="/login">Login</a></li>
        </ul>
    </nav>
    <div class="container">
        <h1 class="head">Register for a New Account</h1>
        <form action="/register" method="post" class="register">
            <label for="name">UserName</label>
            <input type="text" name="username" class="inp">

            <label for="name">Rollno</label>
            <input type="text" name="rollno" class="inp">

            <label for="email">Email</label>
            <input type="email" name="email" class="inp">

            <label for="password">Password</label>
            <input type="password" name="password" class="inp">

            <input type="submit" value="Register" class="inp-btn">

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

```

## Login.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="{url_for('static', filename =
'style2.css')}}">
    <title>Register</title>
</head>
<body>
    <nav class="navbar">
        <h1 class="logo">MyMoney</h1>

```

```

        <ul class="nav-ul">
            <li class="nav-li"><a href="/register">Register</a></li>
        </ul>
    </nav>
    <div class="container">
        <h1 class="head">Login to your Account</h1>
        <form action="/login" method="post" class="register">

            <label for="email">Email</label>
            <input type="email" name="email" class="inp">

            <label for="password">Password</label>
            <input type="password" name="password" class="inp">

            <input type="submit" value="Login" class="inp-btn">

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

```

## Style2.css

```

@import
url('https://fonts.googleapis.com/css2?family=Montserrat:wght@400;500;600&disp
lay=swap');
@import url('http://fonts.cdnfonts.com/css/lemon-tuesday');
*{
    margin: 0%;
    padding: 0%;
    box-sizing: border-box;
    font-family: 'Montserrat', sans-serif;
}

.navbar{
    display: flex;
    flex-direction: row;
    justify-content: space-between;
    align-items: center;
    /* background-color: #6c73ee; */
    background-color: #7C4CFB;
    color: rgb(234, 237, 240);
}

.nav-ul{
    display: flex;
    flex-direction: row;
    list-style: none;
}

```



```
.nav-li{
  padding: 1rem;
  font-weight: bold;
  font-size: 20px;
}

.nav-li:hover{
  /* background-color: #545CB3; */
  background-color: #9900FF;
  cursor: pointer;
}

.logo{
  font-family: 'Lemon Tuesday', sans-serif;
  font-size: 2rem;
  padding-left: 1rem;
  padding-bottom: 0.6rem;
}

.container{
  display: grid;
  grid-template-columns: 1fr 1fr 1fr;
}

.center{
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  height: 92vh;
}

.left{
  display: flex;
  flex-direction: column;
  align-items: flex-end;
  height: 92vh;
  overflow: hidden;
}

.right{
  display: flex;
  flex-direction: column;
  align-items: flex-end;
  height: 92vh;
  overflow: hidden;
}
```

```
.img-div-right{
  height: 100vh;
  margin-top: 13.5rem;
}

.img-div-left{
  height: 100vh;
  margin-top: 15.5rem;
}

.register{
  display: flex;
  flex-direction: column;
}

label{
  margin-top: 20px;
  font-size: 20px;
  font-weight: 600;
}

.inp{
  width: 20vw;
  height: 5vh;
  border: 2px solid #120E43;
  border-radius: 6px;
}

.inp-btn{
  margin-top: 15px;
  width: 20vw;
  height: 5vh;
  background-color: #6c73ee;
  color: #ffffff;
  font-size: 20px;
  font-weight: 600;
  border-radius: 6px;
}

.inp-btn:hover{
  background-color: #545CB3;
  cursor: grab;
}

.head{
  color: black;
}

.company{
  color: #ffffff;
  margin-bottom: 15vh;
}
```

```
.main-img-left{
  transform: scale(0.90, 0.90);
}

.main-img-right{
  transform: scale(0.80, 0.80);
}
```

The screenshot shows a web browser window with multiple tabs. The active tab is 'Register' at the URL 'http://127.0.0.1:5000/register'. The browser's address bar and tabs are visible at the top. The website's header is purple with the 'MyMoney' logo on the left and a 'Login' link on the right. The main content area has a dark blue background with the heading 'Register for a New Account' in white. Below the heading is a registration form with the following fields: 'UserName' (containing 'Rhubika'), 'RollNo' (containing '221076'), 'Email' (containing 'rhubika@gmail.com'), and 'Password' (containing '.....'). A blue 'Register' button is positioned below the password field.

IBM-Project-17227-1 x Cloud Object Storage x Service Details - IBM x Assignments-CAPD// x IBM x Register x WhatsApp x +

← → ↻ ⌂ http://127.0.0.1:5000/register

Course Introductio... My-OSCP Notes Chegg expert login OSCP Web Dev tools OFF-C Placement Resources Finance CP Placement plan LCO Pro CTS Strivers A2Z DSA C...

**MyMoney** Login

## Register for a New Account

UserName  
Rhubika

RollNo  
221076

Email  
rhubika@gmail.com

Password  
.....

Register

The screenshot shows the same web browser window, but the active tab is 'Login' at the URL 'http://127.0.0.1:5000/login'. The website's header is purple with the 'MyMoney' logo on the left and a 'Register' link on the right. The main content area has a dark blue background with the heading 'Login to your Account' in white. Below the heading is a login form with the following fields: 'Email' (containing 'rhubika@gmail.com') and 'Password' (containing '.....'). A blue 'Login' button is positioned below the password field.

IBM-Project-17227-1 x Cloud Object Storage x Service Details - IBM x Assignments-CAPD// x IBM x Register x WhatsApp x +

← → ↻ ⌂ http://127.0.0.1:5000/login

Course Introductio... My-OSCP Notes Chegg expert login OSCP Web Dev tools OFF-C Placement Resources Finance CP Placement plan LCO Pro CTS Strivers A2Z DSA C...

**MyMoney** Register

## Login to your Account

Email  
rhubika@gmail.com

Password  
.....

Login

