Assignment 3

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

```
CREATE TABLE USER (
  rollno INTEGER PRIMARY KEY,
  email TEXT NOT NULL,
  name TEXT NOT NULL,
  password TEXT NOT NULL
);

INSERT INTO USER      VALUES
(1,  'nirajkumar.cse2019@dscetac.in','niraj','123@45');
INSERT INTO USER VALUES (2, 'tharankumar.cse2019@dscet', 'tharankumar',
'765@67');
INSERT INTO USER VALUES (3, 'palrajr.cse2019@dscet.ac.in', 'palraj',
'6280@76');
INSERT INTO USER VALUES (4, 'rajkuamr.cse2019@dscet.ac.in', 'rajkuamr',
'310519@456');


SELECT * FROM USER;
```

2. **Perform UPDATE,DELETE Queries with user table**

```
CREATE TABLE USER (
  rollno INTEGER PRIMARY KEY,
  email TEXT NOT NULL,
  name TEXT NOT NULL,
  password TEXT NOT NULL
);

(1,  'nirajkumar.cse2019@dscetac.in','niraj','123@45');
INSERT INTO USER VALUES (2, 'tharankumar.cse2019@dscet', 'tharankumar',
'765@67');
INSERT INTO USER VALUES (3, 'palrajr.cse2019@dscet.ac.in', 'palraj',
'6280@76');
INSERT INTO USER VALUES (4, 'rajkuamr.cse2019@dscet.ac.in', 'rajkuamr',
'310519@456');


UPDATE USER
SET name='sukhit', email='niranjan888k@gmail.com'
WHERE rollno='1';

DELETE FROM USER WHERE rollno='2';

SELECT * FROM USER;
```

3. **Connect python code to db2.**

```
from ibm_db import connect
# Careful with the punctuation here - we have 3 arguments.
# The first is a big string with semicolons in it.
# (Strings separated by only whitespace, newlines included,
#  are automatically joined together, in case you didn't know.)
# The last two are emptry strings.
connection = connect('DATABASE=<database name>;'
                     'HOSTNAME=<database ip>;'  # 127.0.0.1 or localhost works if it's
local
                     'PORT=<database port>;'
                     'PROTOCOL=TCPIP;'
                     'UID=<database username>;'
                     'PWD=<username password>;', '', '')
```

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**

```
# Store this code in'app.py'file

from flask import Flask, render_template, request, redirect, url_for, session
from flask_mysqldb import MySQL
import MySQLdb.cursors
import re


app = Flask(__name__)


app.secret_key ='your secret key'

app.config['MYSQL_HOST'] ='localhost'
app.config['MYSQL_USER'] ='root'
app.config['MYSQL_PASSWORD'] ='your password'
app.config['MYSQL_DB'] ='geeklogin'

mysql = MySQL(app)

@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']
      cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
```

```
        cursor.execute('SELECT * FROM accounts WHERE username = % s AND
password = % s', (username, password,))
        account = cursor.fetchone()
        if account:
            session['loggedin'] = True
            session['id'] = account['id']
            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']
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute('SELECT * FROM accounts WHERE username = % s',
(username,))
        account = cursor.fetchone()
        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:
            cursor.execute('INSERT INTO accounts VALUES (NULL, % s, % s, % s)',
(username, password, email,))
            mysql.connection.commit()
            msg ='You have successfully registered !'
    elif request.method =='POST':
        msg ='Please fill out the form !'
    return render_template('register.html', msg = msg)
```

5. **Design a chatbot using IBM Watson assistant for hospital. Ex: User comes with query to know the branches for that hospital in your city. Submit the web URL of that chat bot as a assignment.**

```
<script
>
        window.watsonAssistantChatOptions = {
        integrationID: "3052c371-e736-4f66-a1b4-7630200ece6a", // The ID of this
        integration.
        region: "jp-tok", // The region your integration is hosted in.
        serviceInstanceID: "d8f44d27-bfbc-439e-b73e-ce2e30d0c235", // The ID of your
        service instance.
        onLoad: function(instance) { instance.render(); }
        };
        setTimeout(function(){
        const t=document.createElement('script');
        t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
        (window.watsonAssistantChatOptions.clientVersion || 'latest') +
        "/WatsonAssistantChatEntry.js";
        document.head.appendChild(t);
        });
        </script>
```

6. **Create Watson assistant service with 10 steps and use 3 conditions in it. Load that script in HTML page.**

```
 <script>
   window.watsonAssistantChatOptions = {
   integrationID: "3052c371-e736-4f66-a1b4-7630200ece6a", // The ID of this integration.
   region: "jp-tok", // The region your integration is hosted in.
   serviceInstanceID: "d8f44d27-bfbc-439e-b73e-ce2e30d0c235", // The ID of your service
   instance.
   onLoad: function(instance) { instance.render(); }
   };
   setTimeout(function(){
   const t=document.createElement('script');
   t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
   (window.watsonAssistantChatOptions.clientVersion || 'latest') +
   "/WatsonAssistantChatEntry.js";
   document.head.appendChild(t);
   });
   </script>
```