IBM FLASK ASSIGNMENT -2

1.**QUERIES**

COMMAND :CREATE TABLE table\_name (  
    column1 datatype,  
    column2 datatype,  
    column3 datatype,  
   ...);

EXAMPLE: “CREATE TABLE USER(email varchar(30),username varchar(20),rollno int(20),password varchar(10));

insert into user values((‘ibm@gmail.com’,’ibm’,001,’ibm’)(‘abc@gmail.com’,’abc’,123,’abcd’));

**2.QUERIES**

COMMEND: UPDATE table\_name SET column1 = value1, column2 = value2,.. WHERE condition;

EXAMPLE:UPDATE user SET password=’abc’ where rollno=123;

COMMAND :DELETE FROM table\_name WHERE condition;

EXAMPLE :DELETE FROM user WHERE password=’abc’;

**3.CONNECT PYTHON CODE TO DB2**

from flask import Flask, render\_template, request, redirect, url\_for, session

import ibm\_db

import re

app = Flask(\_\_name\_\_)

app.secret\_key = &#39;a&#39;

conn = ibm\_db.connect(&quot;DATABASE=bludb;HOSTNAME=b70af05b-76e4-4bca-a1f523dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32716;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=jzc43091;PWD=PI8VtGRvZlSVT65A&quot;,&#39;&#39;,&#39;&#39;)

@app.route(&#39;/&#39;)

def homer():

return render\_template(&#39;home.html&#39;)

@app.route(&#39;/login&#39;,methods =[&#39;GET&#39;, &#39;POST&#39;])

def login():

global userid

msg = &#39;&#39;if request.method == &#39;POST&#39; :

username = request.form[&#39;username&#39;]

password = request.form[&#39;password&#39;]

sql = &quot;SELECT \* FROM users WHERE username =? AND password=?&quot;

stmt = ibm\_db.prepare(conn, sql)

ibm\_db.bind\_param(stmt,1,username)

ibm\_db.bind\_param(stmt,2,password)

ibm\_db.execute(stmt)

account = ibm\_db.fetch\_assoc(stmt)

print (account)

if account:

session[&#39;loggedin&#39;] = True

session[&#39;id&#39;] = account[&#39;USERNAME&#39;]

userid= account[&#39;USERNAME&#39;]

session[&#39;username&#39;] = account[&#39;USERNAME&#39;]

msg = &#39;Logged in successfully !&#39;

msg = &#39;Logged in successfully !&#39;

return render\_template(&#39;dashboard.html&#39;, msg = msg)

else:

msg = &#39;Incorrect username / password !&#39;

return render\_template(&#39;login.html&#39;, msg = msg)

@app.route(&#39;/register&#39;, methods =[&#39;GET&#39;, &#39;POST&#39;])

def registet():

msg = &#39;&#39;

if request.method == &#39;POST&#39; :

username = request.form[&#39;username&#39;]

email = request.form[&#39;email&#39;]

password = request.form[&#39;password&#39;]

sql = &quot;SELECT \* FROM users WHERE username =?&quot;

stmt = ibm\_db.prepare(conn, sql)

ibm\_db.bind\_param(stmt,1,username)

ibm\_db.execute(stmt)

account = ibm\_db.fetch\_assoc(stmt)

print(account)

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 = ‘name must contain only characters and numbers !’

else:

insert\_sql = “INSERT INTO users VALUES (?, ?, ?)”

prep\_stmt = ibm\_db.prepare(conn, insert\_sql)

ibm\_db.bind\_param(prep\_stmt, 1, username)

ibm\_db.bind\_param(prep\_stmt, 2, email)

ibm\_db.bind\_param(prep\_stmt, 3, password)

ibm\_db.execute(prep\_stmt)

msg = ‘You have successfully registered !’

elif request.method == ‘POST’:

msg = ‘Please fill out the form !’

return render\_template(‘register.html’, msg = msg)

@app.route(‘/dashboard’)

Def dash():

Return render\_template(‘dashboard.html’)

@app.route(‘/apply’,methods =[‘GET’, ‘POST’])

Def apply():

Msg = ‘’

If request.method == ‘POST’ :

Username = request.form[‘username’]

Email = request.form[‘email’]

Qualification= request.form[‘qualification’]

Skills = request.form[‘skills’]

Jobs = request.form[‘s’]

Sql = “SELECT \* FROM users WHERE username =?”

Stmt = ibm\_db.prepare(conn, sql)

Ibm\_db.bind\_param(stmt,1,username)

Ibm\_db.execute(stmt)

Account = ibm\_db.fetch\_assoc(stmt)

Print(account)

If account:

Msg = ‘there is only 1 job position! For you’

Return render\_template(‘apply.html’, msg = msg)

Insert\_sql = “INSERT INTO job VALUES (?, ?, ?, ?, ?)”

Prep\_stmt = ibm\_db.prepare(conn, insert\_sql)

Ibm\_db.bind\_param(prep\_stmt, 1, username)

Ibm\_db.bind\_param(prep\_stmt, 2, email)

Ibm\_db.bind\_param(prep\_stmt, 3, qualification)

Ibm\_db.bind\_param(prep\_stmt, 4, skills)

Ibm\_db.bind\_param(prep\_stmt, 5, jobs)

Ibm\_db.execute(prep\_stmt)

Msg = ‘You have successfully applied for job !’

Session[‘loggedin’] = True

TEXT = “Hello,a new application for job position” +jobs+”is requested”

Elif request.method == ‘POST’:

Msg = ‘Please fill out the form !’

Return render\_template(‘apply.html’, msg = msg)

@app.route(‘/display’)

Def display():

Print(session[“username”],session[‘id’])

Cursor = mysql.connection.cursor()

Cursor.execute(‘SELECT \* FROM job WHERE userid = % s’, (session[‘id’],))

Account = cursor.fetchone()

Print(“accountdislay”,account)

Return render\_template(‘display.html’,account = account)

@app.route(‘/logout’)

Def logout():

Session.pop(‘loggedin’, None)

Session.pop(‘id’, None)

Session.pop(‘username’, None)

Return render\_template(‘home.html’)

If \_\_name\_\_ == ‘\_\_main\_\_’:

App.run(host=’0.0.0.0’)

4. **create a flask app with registration page,login page & welcome page.By default load registration page once the user enter all field store data in database and navigate to the login page authenticate username and password.If the user is valid show the welcome page.**

project/\_\_init\_\_.py

from flask import Flask

from flask\_sqlalchemy import SQLAlchemy

db = SQLAlchemy()

def create\_app():

app = Flask(\_\_name\_\_)

app.config['SECRET\_KEY'] = 'secret-key-goes-here'

app.config['SQLALCHEMY\_DATABASE\_URI'] = 'sqlite:///db.sqlite'

db.init\_app(app)

from .auth import auth as auth\_blueprint

app.register\_blueprint(auth\_blueprint

from .main import main as main\_blueprint

app.register\_blueprint(main\_blueprint)

return app;

project/main.py

from flask import Blueprint

from . import db

main = Blueprint('main', \_\_name\_\_)

@main.route('/')

def index():

return 'Index'

@main.route('/profile')

def profile():

return 'Profile';

project/auth.py

from . import db

auth = Blueprint('auth', \_\_name\_\_)

@auth.route('/login')

def login():

return 'Login'

@auth.route('/signup')

def signup():

return 'Signup'

@auth.route('/logout')

def logout():

return 'Logout'

OUTPUT:

project/templates/base.html

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Flask Auth Example</title>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bulma/0.7.2/css/bulma.min.css" />

</head><body>

<section class="hero is-primary is-fullheight">

<div class="hero-head">

<nav class="navbar">

<div class="container">

<div id="navbarMenuHeroA" class="navbar-menu">

<div class="navbar-end">

<a href="{{ url\_for('main.index') }}" class="navbar-item">

Home

</a>

<a href="{{ url\_for('main.profile') }}" class="navbar-item">

Profile

</a>

<a href="{{ url\_for('auth.login') }}" class="navbar-item">

Login

</a>

<a href="{{ url\_for('auth.signup') }}" class="navbar-item">

Sign Up

</a>

<a href="{{ url\_for('auth.logout') }}" class="navbar-item">

Logout

</a>

</div>

</div>

</div>

</nav>

</div>

<div class="hero-body">

<div class="container has-text-centered">

{% block content %}

{% endblock %}

</div>

</div>

</section>

</body>

</html>

project/templates/index.html

{% extends "base.html" %}

{% block content %}

<h1 class="title">

Flask Login Example

</h1>

<h2 class="subtitle">

Easy authentication and authorization in Flask.

</h2>

{% endblock %}

project/templates/login.html

{% extends "base.html" %}

{% block content %}

<div class="column is-4 is-offset-4">

<h3 class="title">Login</h3>

<div class="box">

<form method="POST" action="/login">

<div class="field">

<div class="control">

<input class="input is-large" type="email" name="email" placeholder="Your Email" autofocus="">

</div>

</div>

<div class="field">

<div class="control">

<input class="input is-large" type="password" name="password" placeholder="Your Password">

</div>

</div>

<div class="field">

<label class="checkbox">

<input type="checkbox" name="remember">

Remember me

</label>

</div>

<button class="button is-block is-info is-large is-fullwidth">Login</button>

</form>

</div>

</div>

{% endblock %}

project/templates/signup.html

{% extends "base.html" %}

{% block content %}

<div class="column is-4 is-offset-4">

<h3 class="title">Sign Up</h3>

<div class="box">

<form method="POST" action="/signup">

<div class="field">

<div class="control">

<input class="input is-large" type="email" name="email" placeholder="Email" autofocus="">

</div>

</div>

<div class="field">

<div class="control">

<input class="input is-large" type="text" name="name" placeholder="Name" autofocus="">

</div>

</div>

<div class="field">

<div class="control">

<input class="input is-large" type="password" name="password" placeholder="Password">

</div> </div> <button class="button is-block is-info is-large is-fullwidth">Sign Up</button> </form>

</div>

</div>

{% endblock %}

project/templates/profile.html

{% extends "base.html" %}

{% block content %}

<h1 class="title">

Welcome, Anthony!

</h1>

{% endblock %}

project/main.py

from flask import Blueprint, render\_template

@main.route('/')

def index():

return render\_template('index.html')

@main.route('/profile')

def profile():

return render\_template('profile.html')

project/auth.py

from flask import Blueprint, render\_template

...

@auth.route('/login')

def login():

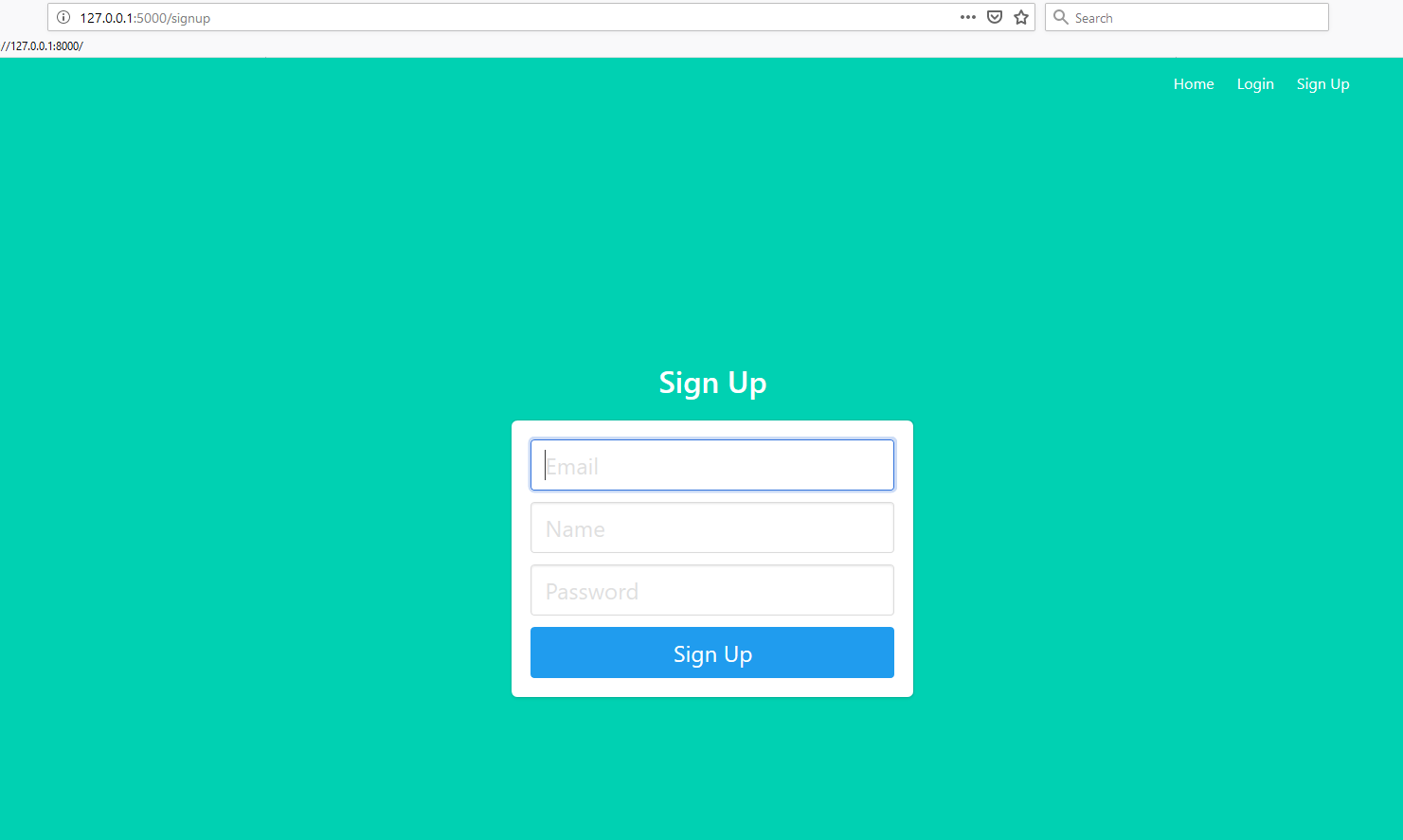
return render\_template('login.html')

@auth.route('/signup')

def signup():

return render\_template('signup.html')

OUTPUT:



project/models.py

from . import db

class User(db.Model):

id = db.Column(db.Integer, primary\_key=True) # primary keys are required by SQLAlchemy

email = db.Column(db.String(100), unique=True)

password = db.Column(db.String(100))

name = db.Column(db.String(1000))

project/auth.py

from flask import Blueprint, render\_template, redirect, url\_for

...@auth.route('/signup')

def signup():

return render\_template('signup.html')

@auth.route('/signup', methods=['POST'])

def signup\_post():

return redirect(url\_for('auth.login'))

auth.py

from .models import User

from . import db

.@auth.route('/signup', methods=['POST'])

def signup\_post():

email = request.form.get('email')

name = request.form.get('name')

password = request.form.get('password')

user = User.query.filter\_by(email=email).first()

if user:

return redirect(url\_for('auth.signup'))

new\_user = User(email=email, name=name, password=generate\_password\_hash(password, method='sha256'))

db.session.add(new\_user)

db.session.commit()

return redirect(url\_for('auth.login'))

project/auth.py

from flask import Blueprint, render\_template, redirect, url\_for, request, flash

@auth.route('/signup', methods=['POST'])

def signup\_post():

... if user: flash('Email address already exists')

return redirect(url\_for('auth.signup'))

signup.html

{% with messages = get\_flashed\_messages() %}

{% if messages %}

<div class="notification is-danger">

{{ messages[0] }}. Go to <a href="{{ url\_for('auth.login') }}">login page</a>.

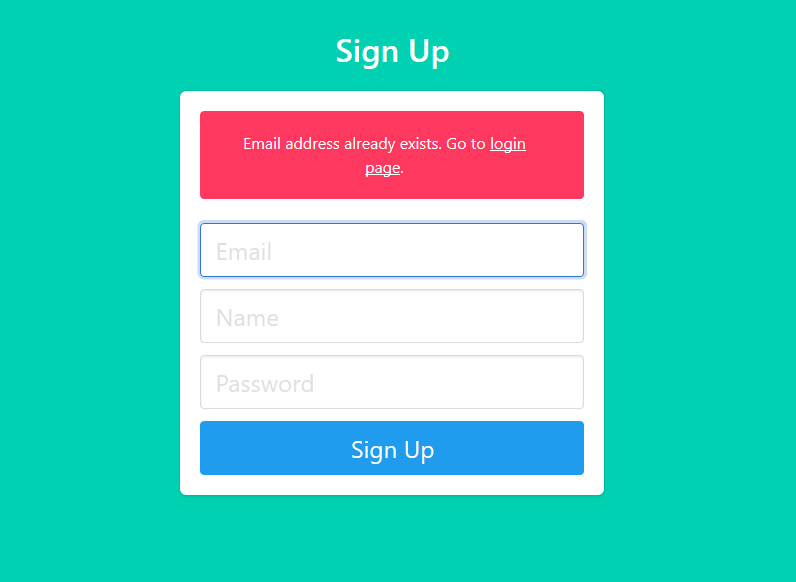
</div>

{% endif %}

{% endwith %}

<form method="POST" action="/signup">

OUTPUT:



project/auth.py

@auth.route('/login', methods=['POST'])

def login\_post():

# login code goes here

email = request.form.get('email')

password = request.form.get('password')

remember = True if request.form.get('remember') else False

user = User.query.filter\_by(email=email).first()

if not user or not check\_password\_hash(user.password, password):

flash('Please check your login details and try again.')

return redirect(url\_for('auth.login'))

return redirect(url\_for('main.profile'))

login.html

{% with messages = get\_flashed\_messages() %}

{% if messages %}

<div class="notification is-danger">

{{ messages[0] }}

</div>

{% endif %}

{% endwith %}

<form method="POST" action="/login">

models.py

from flask\_login import UserMixin

from . import db

class User(UserMixin, db.Model):

id = db.Column(db.Integer, primary\_key=True) # primary keys are required by SQLAlchemy

email = db.Column(db.String(100), unique=True)

password = db.Column(db.String(100))

name = db.Column(db.String(1000))

project/auth.py

from flask\_login import login\_user

from .models import User

from . import db

@auth.route('/login', methods=['POST'])

def login\_post():

login\_user(user, remember=remember)

return redirect(url\_for('main.profile'))

OUTPUT :

