FRONT-END

<!DOCTYPE html>

<html>

<head>

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

<style>

body {font-family: Arial, Helvetica, sans-serif;}

/\* Full-width input fields \*/

input[type=text], input[type=password] {

width: 100%;

padding: 12px 20px;

margin: 8px 0;

display: inline-block;

border: 1px solid #ccc;

box-sizing: border-box;

}

/\* Set a style for all buttons \*/

button {

background-color: #4CAF50;

color: white;

padding: 14px 20px;

margin: 8px 0;

border: none;

cursor: pointer;

width: 100%;

}

button:hover {

opacity: 0.8;

}

/\* Extra styles for the cancel button \*/

.cancelbtn {

width: auto;

padding: 10px 18px;

background-color: #f44336;

}

/\* Center the image and position the close button \*/

.imgcontainer {

text-align: center;

margin: 24px 0 12px 0;

position: relative;

}

img.avatar {

width: 40%;

border-radius: 50%;

}

.container {

padding: 16px;

}

span.psw {

float: right;

padding-top: 16px;

}

/\* The Modal (background) \*/

.modal {

display: none; /\* Hidden by default \*/

position: fixed; /\* Stay in place \*/

z-index: 1; /\* Sit on top \*/

left: 0;

top: 0;

width: 100%; /\* Full width \*/

height: 100%; /\* Full height \*/

overflow: auto; /\* Enable scroll if needed \*/

background-color: rgb(0,0,0); /\* Fallback color \*/

background-color: rgba(0,0,0,0.4); /\* Black w/ opacity \*/

padding-top: 60px;

}

/\* Modal Content/Box \*/

.modal-content {

background-color: #fefefe;

margin: 5% auto 15% auto; /\* 5% from the top, 15% from the bottom and centered \*/

border: 1px solid #888;

width: 80%; /\* Could be more or less, depending on screen size \*/

}

/\* The Close Button (x) \*/

.close {

position: absolute;

right: 25px;

top: 0;

color: #000;

font-size: 35px;

font-weight: bold;

}

.close:hover,

.close:focus {

color: red;

cursor: pointer;

}

/\* Add Zoom Animation \*/

.animate {

-webkit-animation: animatezoom 0.6s;

animation: animatezoom 0.6s

}

@-webkit-keyframes animatezoom {

from {-webkit-transform: scale(0)}

to {-webkit-transform: scale(1)}

}

@keyframes animatezoom {

from {transform: scale(0)}

to {transform: scale(1)}

}

/\* Change styles for span and cancel button on extra small screens \*/

@media screen and (max-width: 300px) {

span.psw {

display: block;

float: none;

}

.cancelbtn {

width: 100%;

}

}

</style>

</head>

<body>

<button onclick="document.getElementById('id01').style.display='block'" style="width:auto;">Login</button>

<div id="id01" class="modal">

<form class="modal-content animate" action="/action\_page.php">

<div class="imgcontainer">

<span onclick="document.getElementById('id01').style.display='none'" class="close" title="Close Modal">&times;</span>

<img src="img\_avatar2.png" alt="Avatar" class="avatar">

</div>

<div class="container">

<label for="uname"><b>Username</b></label>

<input type="text" placeholder="Enter Username" name="uname" required>

<label for="psw"><b>Password</b></label>

<input type="password" placeholder="Enter Password" name="psw" required>

<button type="submit">Login</button>

</div>

<div class="container" style="background-color:#f1f1f1">

<button type="button" onclick="document.getElementById('id01').style.display='none'" class="cancelbtn">Cancel</button>

</div>

</form>

</div>

<script>

// Get the modal

var modal = document.getElementById('id01');

// When the user clicks anywhere outside of the modal, close it

window.onclick = function(event) {

if (event.target == modal) {

modal.style.display = "none";

}

}

</script>

</body>

</html>

BACK-END

import pymysql

from app import app

#from tables import Results

from db\_config import mysql

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

from werkzeug import generate\_password\_hash, check\_password\_hash

from hashlib import md5

from flask\_table import Table, Col, LinkCol

class Results(Table):

stud\_id = Col('ID', show=False)

studno = Col('Student Number')

fname = Col('First Name')

lname = Col('Last Name')

contact = Col('Contact Number')

gender = Col('Gender')

bday = Col('Birthday')

program = Col('Program')

edit = LinkCol('Update', 'updatestudentsview', url\_kwargs=dict(id='stud\_id'))

delete = LinkCol('Delete', 'deletestudent', url\_kwargs=dict(id='stud\_id'))

@app.route('/')

def index():

if 'username' in session:

username\_session = escape(session['username']).capitalize()

return render\_template('homepage.html', session\_user\_name=username\_session)

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

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

def login():

error = None

if 'username' in session:

return redirect(url\_for('index'))

if request.method == 'POST':

username\_form = request.form['uname']

password\_form = request.form['psw']

conn = mysql.connect()

cursor = conn.cursor()

cursor.execute("SELECT COUNT(1) FROM users WHERE uname = %s;", [username\_form])

if cursor.fetchone()[0]:

cursor.execute("SELECT pass FROM users WHERE uname = %s;", [username\_form])

for row in cursor.fetchall():

passx =(password\_form).encode()

if md5(passx).hexdigest() == row[0]:

session['username'] = request.form['uname']

return redirect(url\_for('index'))

else:

error = "Invalid Credential"

else:

error = "Invalid Credential"

return render\_template('index.html', error=error)

@app.route('/logout')

def logout():

session.pop('username', None)

return redirect(url\_for('index'))

@app.route('/new\_student')

def add\_student():

return render\_template('addstudent.html')

#@app.route('/list\_student')

#def list\_student():

# return render\_template('listofstuds.html')

@app.route('/search\_student')

def search\_student():

return render\_template('searchstuds.html')

@app.route('/add', methods=['POST'])

def addstudent():

try:

\_fname = request.form['firstname']

\_lname = request.form['lastname']

\_contact = request.form['contactno']

\_gender = request.form['gender']

\_month = request.form['month']

\_day = request.form['day']

\_year = request.form['year']

\_studno = request.form['studno']

\_progg = request.form['program']

if \_fname and \_lname and \_contact and \_gender and \_month and \_day and \_year and \_studno and \_progg and request.method == 'POST':

bday = \_month + " " + \_day + ", " + \_year

sql = "INSERT INTO students(fname,lname,contact,gender,bday,studno,program) VALUES (%s,%s,%s,%s,%s,%s,%s)"

data = (\_fname,\_lname,\_contact,\_gender,bday,\_studno,\_progg)

conn = mysql.connect()

cursor = conn.cursor()

cursor.execute(sql,data)

conn.commit()

flash('Student added successfully!')

return redirect('/')

else:

return 'Error while adding user'

except Exception as e:

print(e)

finally:

cursor.close()

conn.close()

@app.route('/list')

def liststudents():

try:

conn = mysql.connect()

cursor = conn.cursor(pymysql.cursors.DictCursor)

cursor.execute("SELECT \* FROM students")

rows = cursor.fetchall()

table = Results(rows)

table.border = True

return render\_template('listofstuds.html',table=table)

except Exception as e:

print(e)

finally:

cursor.close()

conn.close()

@app.route('/updatestudent/<int:id>')

def updatestudentsview(id):

#try:

conn = mysql.connect()

cursor = conn.cursor(pymysql.cursors.DictCursor)

cursor.execute("SELECT \* FROM students WHERE stud\_id=%s", id)

row = cursor.fetchone()

if row:

return render\_template('updatestuds.html', row=row)

else:

return 'Error loading #{id}'.format(id=id)

#except Exception as e:

#print(e)

#finally:

#cursor.close()

#conn.close()

@app.route('/update', methods=['POST'])

def updatestudent():

#try:

\_fname = request.form['ufname']

\_lname = request.form['ulname']

\_contact = request.form['ucontact']

\_progg = request.form['uprogram']

\_id = request.form['id']

if \_fname and \_lname and \_contact and \_progg and \_id and request.method == 'POST':

sql = "UPDATE students SET fname=%s,lname=%s,contact=%s,program=%s WHERE stud\_id=%s"

data = (\_fname,\_lname,\_contact,\_progg,\_id)

conn = mysql.connect()

cursor = conn.cursor()

cursor.execute(sql,data)

conn.commit()

flash('Student updated successfully!')

return redirect('/list')

else:

return 'Error while updating user'

#except Exception as e:

#print(e)

#finally:

#cursor.close()

#conn.close()

@app.route('/deletestudent/<int:id>')

def deletestudent(id):

#try:

conn = mysql.connect()

cursor = conn.cursor()

cursor.execute("DELETE FROM students WHERE stud\_id=%s", id)

conn.commit()

flash('User deleted successfully!')

return redirect('/list')

#except Exception as e:

#print(e)

#finally:

#cursor.close()

#conn.close()

# @app.route('/update', methods=['POST'])

# def update\_user():

# try:

# \_name = request.form['inputName']

# \_email = request.form['inputEmail']

# \_password = request.form['inputPassword']

# \_id = request.form['id']

# # validate the received values

# if \_name and \_email and \_password and \_id and request.method == 'POST':

# #do not save password as a plain text

# \_hashed\_password = generate\_password\_hash(\_password)

# # save edits

# sql = "UPDATE tbl\_user SET user\_name=%s, user\_email=%s, user\_password=%s WHERE user\_id=%s"

# data = (\_name, \_email, \_hashed\_password, \_id,)

# conn = mysql.connect()

# cursor = conn.cursor()

# cursor.execute(sql, data)

# conn.commit()

# flash('User updated successfully!')

# return redirect('/')

# else:

# return 'Error while updating user'

# except Exception as e:

# print(e)

# finally:

# cursor.close()

# conn.close()

# @app.route('/delete/')

# def delete\_user(id):

# try:

# conn = mysql.connect()

# cursor = conn.cursor()

# cursor.execute("DELETE FROM tbl\_user WHERE user\_id=%s", (id,))

# conn.commit()

# flash('User deleted successfully!')

# return redirect('/')

# except Exception as e:

# print(e)

# finally:

# cursor.close()

# conn.close()

if \_\_name\_\_ == "\_\_main\_\_":

app.run()

from app import app

from flaskext.mysql import MySQL

mysql = MySQL()

# MySQL configurations

app.config['MYSQL\_DATABASE\_USER'] = 'root'

app.config['MYSQL\_DATABASE\_PASSWORD'] = 'p@ssw0rd'

app.config['MYSQL\_DATABASE\_DB'] = 'studentdb'

app.config['MYSQL\_DATABASE\_HOST'] = 'localhost'

mysql.init\_app(app)

from app import app

from flaskext.mysql import MySQL

mysql = MySQL()

# MySQL configurations

app.config['MYSQL\_DATABASE\_USER'] = 'root'

app.config['MYSQL\_DATABASE\_PASSWORD'] = 'p@ssw0rd'

app.config['MYSQL\_DATABASE\_DB'] = 'studentdb'

app.config['MYSQL\_DATABASE\_HOST'] = 'localhost'

mysql.init\_app(app)

from flask import Flask

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

app.secret\_key = "secret key"

DATABASE:

create schema studentdb;

use studentdb;

create table users (

uname varchar(100),

pass varchar(100)

);

insert into users (uname, pass) values ('admin', '21232f297a57a5a743894a0e4a801fc3')

create table students(

stud\_id int auto\_increment,

fname varchar(100),

lname varchar(100),

contact varchar(100),

gender varchar(100),

bday nvarchar(100),

studno varchar(100),

program varchar(100),

primary key (stud\_id)

) AUTO\_INCREMENT