|  |
| --- |
| from flask import \*#importing flask (Install it using python -m pip install flask) |
|  | import os |
|  | import ibm\_db |
|  | import bcrypt |
|  | #import send\_from\_directory |
|  | app = Flask(\_\_name\_\_) #initialising flask |
|  | app.secret\_key = '' |
|  | PEOPLE\_FOLDER = os.path.join('static', 'people\_photo') |
|  |  |
|  | import ibm\_db |
|  | conn = ibm\_db.connect("DATABASE=bludb;HOSTNAME=54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32733;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;PROTOCOL=TCPIP;UID=phb43134;PWD=mTTnIE45Raj9A0rr",'','') |
|  |  |
|  |  |
|  | @app.route("/") #defining the routes for the home() funtion (Multiple routes can be used as seen here) |
|  | @app.route("/home") |
|  |  |
|  | def home(): |
|  | return render\_template("home.html") #rendering our home.html contained within /templates |
|  |  |
|  | @app.route("/login") |
|  | def login(): |
|  | if request.method == 'POST': |
|  | email = request.form['email'] |
|  | password = request.form['password'] |
|  | if not email or not password: |
|  | return render\_template('login.html',error='Please fill all fields') |
|  |  |
|  | query = "SELECT \* FROM USER WHERE email=?" |
|  | stmt = ibm\_db.prepare(conn, query) |
|  | ibm\_db.bind\_param(stmt,1,email) |
|  | ibm\_db.execute(stmt) |
|  | isUser = ibm\_db.fetch\_assoc(stmt) |
|  | print(isUser,password) |
|  |  |
|  | if not isUser: |
|  | return render\_template('login.html',error='Invalid Credentials') |
|  |  |
|  | isPasswordMatch = bcrypt.checkpw(password.encode('utf-8'),isUser['PASSWORD'].encode('utf-8')) |
|  |  |
|  | if not isPasswordMatch: |
|  | return render\_template('login.html',error='Invalid Credentials') |
|  |  |
|  | session['email'] = isUser['EMAIL'] |
|  | return redirect(url\_for('user\_dashboard')) |
|  |  |
|  | return render\_template('login.html',name='Home') |
|  |  |
|  |  |
|  |  |
|  |  |
|  | app.config["UPLOAD\_DIR"] = "uploads" |
|  | @app.route("/register",methods=['GET','POST']) |
|  | def register(): |
|  | if request.method == 'POST': |
|  | email = request.form['email'] |
|  | password = request.form['password'] |
|  | firstname=request.form['firstname'] |
|  | middlename=request.form['middlename'] |
|  | lastname=request.form['lastname'] |
|  | dob=request.form['dob'] |
|  | age=request.form['age'] |
|  | drno=request.form['drno'] |
|  | landmark1=request.form['lnmk1'] |
|  | landmark2=request.form['lnmk2'] |
|  | village =request.form['village'] |
|  | district=request.form['district'] |
|  | pincode=request.form['pincode'] |
|  | country=request.form['country'] |
|  | country\_code=request.form['countryCode'] |
|  | phone=request.form['phone'] |
|  | schlname10=request.form['schlname10'] |
|  | maxmark10=request.form['maxmark10'] |
|  | actmark10=request.form['actmark10'] |
|  | schlname212=request.form['schlname212'] |
|  | maxmark212=request.form['maxmark212'] |
|  | actmark212=request.form['actmark212'] |
|  | clgname=request.form['clgname'] |
|  | clgmaxmark=request.form['clgmaxmark'] |
|  | clgactmark=request.form['clgactmark'] |
|  | stream=request.form['stream'] |
|  | graduation=request.form['graduation'] |
|  | grayear=request.form['grayear'] |
|  | branch=request.form['branch'] |
|  | skills=request.form['form[]'] |
|  | q1=request.form['Q1'] |
|  | q2=request.form['Q2'] |
|  | q3=request.form['Q3'] |
|  | q4=request.form['Q4'] |
|  | q5=request.form['Q5'] |
|  | position=request.form['position'] |
|  | experience=request.form['expyears'] |
|  | designation=request.form['designation'] |
|  | tum = request.files['resume'] |
|  | tum.save(os.path.join(app.config['UPLOAD\_DIR'],tum.filename)) |
|  |  |
|  |  |
|  |  |
|  | hash=bcrypt.hashpw(password.encode('utf-8'),bcrypt.gensalt()) |
|  |  |
|  | query = "SELECT EMAIL FROM JOBSEEKER WHERE EMAIL=?" |
|  | stmt = ibm\_db.prepare(conn, query) |
|  | ibm\_db.bind\_param(stmt,1,email) |
|  | ibm\_db.execute(stmt) |
|  | isUser = ibm\_db.fetch\_assoc(stmt) |
|  |  |
|  | if not isUser: |
|  | insert\_sql = "INSERT INTO JOBSEEKER (EMAIL, PASSWORD, FIRSTNAME, MIDDLENAME, LASTNAME, DOB, AGE, DOOR\_NO, LANDMARK1, LANDMARK2,VILLAGE, DISTRICT, PINCODE, COUNTRY, COUNTRY\_CODE, PHONE, SCHOOL\_NAME, SC\_MAX\_MARK, SC\_ACT\_MARK,HSC\_SCHOOL\_NAME,HSC\_MAX\_MARK, HSC\_ACT\_MARK,COLLEGE\_NAME,COLLEGE\_MAX,COLLEGE\_ACT, STREAM, GRADUATION, GRADUATION\_YEAR, BRANCH, SKILLS, PRESENTATION, HACKATHON, CODING, WON, PRIZE\_WON, POSITION, EXPYEARS,DESIGNATION, RESUME) VALUES (?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)" |
|  | prep\_stmt = ibm\_db.prepare(conn, insert\_sql) |
|  | ibm\_db.bind\_param(prep\_stmt, 1, email) |
|  | ibm\_db.bind\_param(prep\_stmt, 2, password) |
|  | ibm\_db.bind\_param(prep\_stmt, 3, firstname) |
|  | ibm\_db.bind\_param(prep\_stmt, 4, middlename) |
|  | ibm\_db.bind\_param(prep\_stmt, 5, lastname) |
|  | ibm\_db.bind\_param(prep\_stmt, 6, dob) |
|  | ibm\_db.bind\_param(prep\_stmt, 7, age) |
|  | ibm\_db.bind\_param(prep\_stmt, 8, drno) |
|  | ibm\_db.bind\_param(prep\_stmt, 9, landmark1) |
|  | ibm\_db.bind\_param(prep\_stmt, 10, landmark2) |
|  | ibm\_db.bind\_param(prep\_stmt, 11, village) |
|  | ibm\_db.bind\_param(prep\_stmt, 12, district) |
|  | ibm\_db.bind\_param(prep\_stmt, 13, pincode) |
|  | ibm\_db.bind\_param(prep\_stmt, 14, country) |
|  | ibm\_db.bind\_param(prep\_stmt, 15, country\_code) |
|  | ibm\_db.bind\_param(prep\_stmt, 16, phone) |
|  | ibm\_db.bind\_param(prep\_stmt, 17, schlname10) |
|  | ibm\_db.bind\_param(prep\_stmt, 18, maxmark10) |
|  | ibm\_db.bind\_param(prep\_stmt, 19, actmark10) |
|  | ibm\_db.bind\_param(prep\_stmt, 20, schlname212) |
|  | ibm\_db.bind\_param(prep\_stmt, 21, maxmark212) |
|  | ibm\_db.bind\_param(prep\_stmt, 22, actmark212) |
|  | ibm\_db.bind\_param(prep\_stmt, 23, clgname) |
|  | ibm\_db.bind\_param(prep\_stmt, 24, clgmaxmark) |
|  | ibm\_db.bind\_param(prep\_stmt, 25, clgactmark) |
|  | ibm\_db.bind\_param(prep\_stmt, 26, stream) |
|  | ibm\_db.bind\_param(prep\_stmt, 27, graduation) |
|  | ibm\_db.bind\_param(prep\_stmt, 28, grayear) |
|  | ibm\_db.bind\_param(prep\_stmt, 29, branch) |
|  | ibm\_db.bind\_param(prep\_stmt, 30, skills) |
|  | ibm\_db.bind\_param(prep\_stmt, 31, q1) |
|  | ibm\_db.bind\_param(prep\_stmt, 32, q2) |
|  | ibm\_db.bind\_param(prep\_stmt, 33, q3) |
|  | ibm\_db.bind\_param(prep\_stmt, 34, q4) |
|  | ibm\_db.bind\_param(prep\_stmt, 35, q5) |
|  | ibm\_db.bind\_param(prep\_stmt, 36, position) |
|  | ibm\_db.bind\_param(prep\_stmt, 37, experience) |
|  | ibm\_db.bind\_param(prep\_stmt, 38, designation) |
|  | ibm\_db.bind\_param(prep\_stmt, 39, tum) |
|  | ibm\_db.execute(prep\_stmt) |
|  | return render\_template('register.html') |
|  | else: |
|  | return render\_template('register.html',error='Invalid Credentials') |
|  |  |
|  | return render\_template('register.html',name='Home') |
|  |  |
|  |  |
|  | @app.route("/orgregister") |
|  | def orgregister(): |
|  | return render\_template("orgregister.html") #rendering our register.html contained within /templates |
|  |  |
|  |  |
|  | @app.route("/user\_dashboard") |
|  | def user\_dashboard(): |
|  |  |
|  | return render\_template("user\_dashboard.html") |
|  |  |
|  | @app.route('/logout') |
|  | def logout(): |
|  | session.pop('email', None) |
|  | return redirect(url\_for('login')) |
|  |  |
|  |  |
|  | if \_\_name\_\_ == "\_\_main\_\_": #checking if \_\_name\_\_'s value is '\_\_main\_\_'. \_\_name\_\_ is an python environment variable who's value will always be '\_\_main\_\_' till this is the first instatnce of app.py running |
|  | app.run(debug=True,port=8080) #running flask (Initalised on line 4) |