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

import ibm\_db

import re

app = Flask(\_name\_)

# for connection

# conn= ""

app.secret\_key = 'a'

print("Trying to connect...")

conn = ibm\_db.connect("DATABASE=bludb;HOSTNAME=824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=30119;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=qvk70423;PWD=saDlGasU4iQy1yvk;", '', '')

print("connected..")

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

def signup():

global userid

msg = ''

if request.method == 'POST':

username = request.form['username']

name = request.form['name']

email = request.form['email']

phn = request.form['phn']

password = request.form['pass']

repass = request.form['repass']

print("inside checking")

print(name)

if len(username) == 0 or len(name) == 0 or len(email) == 0 or len(phn) == 0 or len(password) == 0 or len(repass) == 0:

msg = "Form is not filled completely!!"

print(msg)

return render\_template('signup.html', msg=msg)

elif password != repass:

msg = "Password is not matched"

print(msg)

return render\_template('signup.html', msg=msg)

elif not re.match(r'[a-z]+', username):

msg = 'Username can contain only small letters and numbers'

print(msg)

return render\_template('signup.html', msg=msg)

elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):

msg = 'Invalid email'

print(msg)

return render\_template('signup.html', msg=msg)

elif not re.match(r'[A-Za-z]+', name):

msg = "Enter valid name"

print(msg)

return render\_template('signup.html', msg=msg)

elif not re.match(r'[0-9]+', phn):

msg = "Enter valid phone number"

print(msg)

return render\_template('signup.html', msg=msg)

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 = 'Acccount already exists'

else:

userid = username

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, name)

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

ibm\_db.bind\_param(prep\_stmt, 4, phn)

ibm\_db.bind\_param(prep\_stmt, 5, password)

ibm\_db.execute(prep\_stmt)

print("successs")

msg = "succesfully signed up"

return render\_template('dashboard.html', msg=msg, name=name)

else:

return render\_template('signup.html')

@app.route('/dashboard')

def dashboard():

return render\_template('dashboard.html')

@app.route('/')

def base():

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

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

def login():

global userid

msg = ''

if request.method == 'POST':

username = request.form['username']

userid = username

password = request.form['pass']

if userid == 'admin' and password == 'admin':

print("its admin")

return render\_template('admin.html')

else:

sql = "select \* from agents where username = ? and password = ?"

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['Loggedin'] = True

session['id'] = account['USERNAME']

userid = account['USERNAME']

session['username'] = account['USERNAME']

msg = 'logged in successfully'

# for getting complaints details

sql = "select \* from complaints where assigned\_agent = ?"

complaints = []

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

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

ibm\_db.execute(stmt)

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

while dictionary != False:

complaints.append(dictionary)

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

print(complaints)

return render\_template('agentdash.html', name=account['USERNAME'], complaints=complaints)

sql = "select \* from users where username = ? and password = ?"

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['Loggedin'] = True

session['id'] = account['USERNAME']

userid = account['USERNAME']

session['username'] = account['USERNAME']

msg = 'logged in successfully'

# for getting complaints details

sql = "select \* from complaints where username = ?"

complaints = []

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

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

ibm\_db.execute(stmt)

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

while dictionary != False:

# print "The ID is : ", dictionary["EMPNO"]

# print "The Name is : ", dictionary[1]

complaints.append(dictionary)

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

print(complaints)

return render\_template('dashboard.html', name=account['USERNAME'], complaints=complaints)

else:

msg = 'Incorrect user credentials'

return render\_template('dashboard.html', msg=msg)

else:

return render\_template('login.html')

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

def add():

if request.method == 'POST':

title = request.form['title']

des = request.form['des']

try:

sql = "insert into complaints(username,title,complaint) values(?,?,?)"

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

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

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

ibm\_db.bind\_param(stmt, 3, des)

ibm\_db.execute(stmt)

except:

print(userid)

print(title)

print(des)

print("cant insert")

sql = "select \* from complaints where username = ?"

complaints = []

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

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

ibm\_db.execute(stmt)

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

while dictionary != False:

# print "The ID is : ", dictionary["EMPNO"]

# print "The Name is : ", dictionary[1]

complaints.append(dictionary)

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

print(complaints)

return render\_template('dashboard.html', name=userid, complaints=complaints)

@app.route('/agents')

def agents():

sql = "select \* from agents"

agents = []

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

ibm\_db.execute(stmt)

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

while dictionary != False:

agents.append(dictionary)

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

return render\_template('agents.html', agents=agents)

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

def addagent():

if request.method == 'POST':

username = request.form['username']

name = request.form['name']

email = request.form['email']

phone = request.form['phone']

domain = request.form['domain']

password = request.form['password']

try:

sql = "insert into agents values(?,?,?,?,?,?,2)"

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

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

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

ibm\_db.bind\_param(stmt, 3, email)

ibm\_db.bind\_param(stmt, 4, phone)

ibm\_db.bind\_param(stmt, 5, password)

ibm\_db.bind\_param(stmt, 6, domain)

ibm\_db.execute(stmt)

except:

print("cant insert")

sql = "select \* from agents"

agents = []

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

ibm\_db.execute(stmt)

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

while dictionary != False:

agents.append(dictionary)

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

return render\_template('agents.html', agents=agents)

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

def updatecomplaint():

if request.method == 'POST':

cid = request.form['cid']

solution = request.form['solution']

try:

sql = "update complaints set solution =?,status=1 where c\_id = ? and assigned\_agent=?"

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

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

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

ibm\_db.bind\_param(stmt, 3, userid)

ibm\_db.execute(stmt)

sql = "update agents set status =3 where username=?"

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

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

ibm\_db.execute(stmt)

except:

print("cant insert")

sql = "select \* from complaints where assigned\_agent = ?"

complaints = []

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

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

ibm\_db.execute(stmt)

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

while dictionary != False:

complaints.append(dictionary)

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

# print(complaints)

return render\_template('agentdash.html', name=userid, complaints=complaints)

@app.route('/tickets')

def tickets():

sql = "select \* from complaints"

complaints = []

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

ibm\_db.execute(stmt)

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

while dictionary != False:

complaints.append(dictionary)

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

sql = "select username from agents where status <> 1"

freeagents = []

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

ibm\_db.execute(stmt)

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

while dictionary != False:

freeagents.append(dictionary)

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

print(freeagents)

return render\_template('tickets.html', complaints=complaints, freeagents=freeagents)

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

def assignagent():

if request.method == "POST":

ccid = request.form['ccid']

agent = request.form['agent']

print(ccid)

print(agent)

try:

sql = "update complaints set assigned\_agent =? where c\_id = ?"

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

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

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

ibm\_db.execute(stmt)

sql = "update agents set status =1 where username = ?"

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

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

ibm\_db.execute(stmt)

except:

print("cant update")

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

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

app.run(debug=False,host='0.0.0.0')