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

import ibm\_db

from flask\_mail import Mail, Message

from random import randint

import requests

from werkzeug.utils import secure\_filename

from datetime import datetime as dt

connectionstring="DATABASE=bludb;HOSTNAME=55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=31929;PROTOCOL=TCPIP;UID=tgv79601;PWD=hedIlL8ICZwxQhwP;SECURITY=SSL;"

connection = ibm\_db.connect(connectionstring, '', '')

print(ibm\_db.active(connection))

global account

class user:

def \_\_init\_\_(self,id,name,password):

self.id=id

self.name=name

self.password=password

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

app.secret\_key="123456789"

app.config['IMAGE\_FOLDER'] = 'static/userfoodimage/'

global otp

app.config["MAIL\_SERVER"] = 'smtp.gmail.com'

app.config["MAIL\_PORT"] = 465

app.config["MAIL\_USERNAME"] = '2k19cse060@kiot.ac.in'

app.config['MAIL\_PASSWORD'] = 'uxcvcgxchojpwtfd'

app.config['MAIL\_USE\_TLS'] = False

app.config['MAIL\_USE\_SSL'] = True

mail = Mail(app)

@app.before\_request

def before\_request():

global account

if 'id' in session:

sql = "SELECT \* FROM user WHERE UserID =?"

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

ibm\_db.bind\_param(stmt, 1, session['id'])

ibm\_db.execute(stmt)

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

if account:

g.user = account

def calculate():

age=g.user['AGE']

weight=g.user['WEIGHT']

height=g.user['HEIGHT']

waist=g.user['WAIST']

bmi=0

whr=0

bmr=0

bfp=0

gender=1

if(str(g.user['GENDER']) == "Male"):

gender=0

print(g.user['GENDER'])

bmi = weight / (height/100)\*\*2

whr = waist/height

if gender==0: bmr=((10\*weight)+(6.25\*height)-(5\*age)+5)

if gender==1: bmr=((10\*weight)+(6.25\*height)-(5\*age)-161)

bmr=round(bmr)

if age<18 and gender==1:

bfp=(1.15\*bmi)-(0.70\*age)-2.2

if age<18 and gender==0:

bfp=(1.15\*bmi)-(0.70\*age)+1.4

if age>=18 and gender==1:

bfp=(1.20\*bmi)-(0.23\*age)-5.4

if age>=18 and gender==0:

bfp=(1.20\*bmi)-(0.23\*age)-16.2

bfp=round(bfp,1)

g.bmi=bmi

g.whr=whr

g.bmr=bmr

g.bfp=bfp

g.gender=gender

#index & homepage

@app.route('/')

@app.route('/index')

def root():

global account

if 'id' in session:

sql = "SELECT \* FROM user WHERE UserID =?"

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

ibm\_db.bind\_param(stmt, 1, session['id'])

ibm\_db.execute(stmt)

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

if account:

g.user = account

calculate()

return render\_template('home.html')

return render\_template('Index.html')

@app.route('/home')

def home():

global account

if 'id' in session:

sql = "SELECT \* FROM user WHERE UserID =?"

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

ibm\_db.bind\_param(stmt, 1, session['id'])

ibm\_db.execute(stmt)

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

if account:

g.user = account

try:

calculate()

return render\_template('Home.html')

except:

return render\_template('Home.html')

#signup module work

@app.route('/signup')

def signup():

return render\_template("autent/Signup.html")

@app.route('/validation')

def validation():

return render\_template("autent/Validation.html")

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

def adduser():

global name

global email

global password

global phone

global otp

if request.method == 'POST':

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

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

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

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

sql = "SELECT \* FROM user WHERE email =? AND phone=?"

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

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

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

ibm\_db.execute(stmt)

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

if account:

return render\_template('autent/Signup.html', msg="Email or Phone Number already exist, Unique detail.")

else:

session['regmail'] = email

otp = randint(000000, 999999)

vemail = email

msg = Message(subject='Verfication Code For NutriAssist', sender='2k19cse060@kiot.ac.in',recipients=[vemail])

msg.body = "You have succesfully registered on Nutritional Assist!\n\nUse the OTP given below to verify your email ID.\n\t\n\t" + str(otp)

mail.send(msg)

return render\_template("autent/Validation.html", resendmsg="OTP has been sent", msg="OTP has been sent")

elif ("regmail" in session):

if request.method == 'GET':

otp = randint(000000, 999999)

msg = Message(subject='OTP', sender='2k19cse060@kiot.ac.in',recipients=[session['regmail']])

msg.body = "You have succesfully registered on Nutritional Assist!\nUse the OTP given below to verify your email ID.\n\t\t" + str(otp)

mail.send(msg)

return render\_template("autent/Validation.html", resendmsg="OTP has been resent")

else:

return redirect('/')

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

def validate():

if request.method == 'POST':

global name

global email

global password

global phone

global otp

ID = 0

newuser=0

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

if(fotp == otp):

while True:

ID = randint(00000, 99999)

sql = "SELECT \* FROM user WHERE UserID =?"

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

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

ibm\_db.execute(stmt)

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

if account: continue

else: break

insert\_sql = "INSERT INTO USER(UserID,NAME,PHONE,EMAIL,PASSWORD,NEWUSER) VALUES (?,?,?,?,?,?)"

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

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

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

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

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

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

ibm\_db.bind\_param(prep\_stmt, 6, newuser)

ibm\_db.execute(prep\_stmt)

msg = Message(subject='Welcome to NutriAssist', sender='2k19cse060@kiot.ac.in',recipients=[email])

msg.body = "You have succesfully registered on NutriAssist!\n\nYour NutriAssist ID is:"+ str(ID) +"\n\nKindly fill up the profile page to for more informational details.\n\t\n\t"

mail.send(msg)

return render\_template("autent/Login.html")

else:

return render\_template("autent/Validation.html", resendmsg="OTP not match")

else:

return render\_template("autent/Signup.html", resendmsg="POST is not working")

#login module work

@app.route('/signin')

def signin():

return render\_template('autent/Login.html')

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

def login():

if request.method == 'POST':

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

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

sql = "SELECT \* FROM user WHERE email =?"

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

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

ibm\_db.execute(stmt)

global account

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

if account:

if (email == str(account['EMAIL']).strip() and password == str(account['PASSWORD']).strip()):

session['id'] = account['USERID']

g.record=1

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

else:

g.record=0

if g.record!=1:

flash("Username or Password Mismatch...!!!",'danger')

return render\_template('autent/Login.html', msg="Email is invalid")

else:

flash("Account doesn't exist...!!!",'danger')

return render\_template('autent/Login.html', msg="Enter detail again or signup for new account")

else:

return render\_template('autent/Login.html',msg="Retry")

@app.route('/user')

def user():

if not g.user:

return render\_template('autent/Login.html')

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

#profile module work

@app.route('/profile')

def profile():

try:

calculate()

return render\_template("Profile.html")

except:

return render\_template("Profile.html")

@app.route('/profileinfo')

def info():

return render\_template('autent/profileInfo.html')

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

def profileupdate():

if request.method == 'POST':

userid=g.user['USERID']

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

gender = str(request.form.get('gender'))

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

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

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

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

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

g.gender,g.age,g.height,g.weight,g.waist=gender,age,height,weight,waist

newuser=1

sql = "UPDATE user SET(phone,gender,dob,age,height,weight,waist,newuser)=(?,?,?,?,?,?,?,?) where userid =?"

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

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

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

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

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

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

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

ibm\_db.bind\_param(stmt, 7, waist)

ibm\_db.bind\_param(stmt, 8, newuser)

ibm\_db.bind\_param(stmt, 9, userid)

ibm\_db.execute(stmt)

sql = "SELECT \* FROM user WHERE userid =?"

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

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

ibm\_db.execute(stmt)

global account

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

g.user= account

calculate()

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

# else:

# return redirect(url\_for('profileinfo'))

#food detection page

@app.route('/fdp')

def fdp():

return render\_template('fdp.html')

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

def work():

image = request.files['file']

image1 = request.files['file']

if (bool(request.files)):

url = "https://spoonacular-recipe-food-nutrition-v1.p.rapidapi.com/food/images/analyze"

headers = {

"X-RapidAPI-Key": "4910966cf9msh95e8f19b1e26643p14be06jsn1c7184794096",

"X-RapidAPI-Host": "spoonacular-recipe-food-nutrition-v1.p.rapidapi.com"

}

files = {'file': ('Image.png', image, 'image/\*', {'Expires': '10'}) }

response = requests.request("POST", url, files=files, headers=headers)

data=response.json()

print(data)

now = dt.now()

dtstring = dt.isoformat(now)

image1.save(app.config['IMAGE\_FOLDER']+secure\_filename(dtstring+"\_"+image1.filename))

image\_path = app.config['IMAGE\_FOLDER']+secure\_filename(dtstring+"\_"+image1.filename)

image\_place = "/"+image\_path

if ((data['category']['probability']) > 0.8): probabilityText= 'Im almost certain!'

elif (data['category']['probability'] > 0.6): probabilityText= 'I am rather confident in that.'

elif (data['category']['probability'] > 0.4): probabilityText= 'Not really sure but looks like it.'

elif (data['category']['probability'] > 0.2): probabilityText= 'Maybe - maybe not though.'

else: probabilityText= 'I am really unsure about that!'

calories=float((float(data['nutrition']['calories']['value'])\*float(630.0))/float(800.0))

protein=float((float(data['nutrition']['protein']['value'])\*float(630))/float(30))

carbs=float((float(data['nutrition']['carbs']['value'])\*float(630))/float(40))

fat=float((float(data['nutrition']['fat']['value'])\*float(630))/float(30))

print(image\_path)

print(image\_place)

return render\_template("fdp.html",data=data,probabilityText=probabilityText,calories=calories,fat=fat,protein=protein,carbs=carbs,image=image\_place)

else:

return render\_template("fdp.html",msg="NO file has uploaded")

#history module work

#logout work

@app.route('/logout')

@app.route('/signout')

def signout():

session.clear()

g.record=0

return render\_template("Index.html")

#main & docker

if \_\_name\_\_ == '\_\_main\_\_':

app.run(host="0.0.0.0", port=5000, debug=True)