chatbotAPI:

from fastapi import FastAPI

from pydantic import BaseModel

import uvicorn

import tensorflow

from tensorflow import keras

from keras import models

import json

import numpy as np

import pickle

import nltk

from nltk.corpus import stopwords

import random

import requests

import spacy

URL='http://127.0.0.1:8000/'

app = FastAPI()

listgreet=['greeting','goodbye','thanks','options']

class ChatBot(BaseModel):

    question:str

    PROCESS:int

    DISEASE:str

    MEDICINE:str

model=models.load\_model('ChatBotModel.h5')

words=pickle.load(open('word.pkl','rb'))

intents = json.loads(open('intents.json').read())

processdata={}

classes = pickle.load(open('classes.pkl','rb'))

lemmatizer=nltk.WordNetLemmatizer()

nlp= spacy.load(r"./output/model-last")

#load the best model

def clean\_up\_sentence(sentence):

    # tokenize the pattern - split words into array

    sentence\_words = nltk.word\_tokenize(sentence)

    # stem each word - create short form for word

    sentence\_words = [lemmatizer.lemmatize(word.lower()) for word in sentence\_words]

    return sentence\_words

def bow(sentence, words, show\_details=True):

    # tokenize the pattern

    sentence\_words = clean\_up\_sentence(sentence)

    # bag of words - matrix of N words, vocabulary matrix

    bag = [0]\*len(words)

    for s in sentence\_words:

        for i,w in enumerate(words):

            if w == s:

                # assign 1 if current word is in the vocabulary position

                bag[i] = 1

                if show\_details:

                    print ("found in bag: %s" % w)

    return np.array(bag)

def predict\_class(sentence, model):

    # filter out predictions below a threshold

    p = bow(sentence, words,show\_details=False)

    res = model.predict(np.array([p]))[0]

    print(res)

    ERROR\_THRESHOLD = 0.1

    results = [[i,r] for i,r in enumerate(res) if r>ERROR\_THRESHOLD]

    # sort by strength of probability

    results.sort(key=lambda x: x[1], reverse=True)

    return\_list = []

    for r in results:

        return\_list.append({"intent": classes[r[0]], "probability": str(r[1])})

    return return\_list

def getResponse(ints, intents\_json):

    tag = ints[0]['intent']

    list\_of\_intents = intents\_json['intents']

    for i in list\_of\_intents:

        if(i['tag']== tag):

            result = random.choice(i['responses'])

            break

    return result

def EntityDiseasePrediction(question):

    doc=nlp(question)

    response={}

    for d in doc.ents:

        if d.label\_=='DISEASE':

            response[d.label\_]=d.text

            return response['DISEASE']

        else:

            return ""

def IntentPrediction(question):

    ints =  predict\_class(question,model)

    print(ints)

    res,tag = getResponse(ints, intents)

    print(res,tag)

    return res,tag

def IntentConfirmation(res1,tag):

    intent=""

    intent1=""

    for k,v in res1.items():

        if k != 'DISEASE':

            intent1=k

        if intent1=="" and tag!="":

            intent=tag

        if tag=="" and intent1!="":

            intent=intent1

        if tag!="" and intent1!="":

            intent=tag

        if tag=="" and intent1=="":

            question=input("do you whant to know the cause ,prevention,treatment or medicine fro this disease")

            intent=tag

        disease=res1['DISEASE']

        print(disease,intent)

        return [disease,intent],1

@app.post('/chatbot/first')

async def entitydiseaseprediction(answer:ChatBot):

    data=answer.dict()

    print(data)

    question=data['question']

    PROCESS=data['PROCESS']

    DISEASE=data['DISEASE']

    MEDICINE=data['MEDICINE']

    if PROCESS==1:

        DISEASE=question

        PROCESS=2

        return "ASK",PROCESS,question,MEDICINE

    if PROCESS==0:

        ints =  predict\_class(question,model)

        print(ints)

        res = getResponse(ints, intents)

        PROCESS=0

        if ints[0]['intent']=='notknowing':

            PROCESS=1

            return 'notknowing',PROCESS,DISEASE,MEDICINE

        if ints[0]['intent'] in ['cause','prevention','medicine','treatment']:

            if DISEASE=='':

                return forCausePrevMedTreat(ints=ints,question=question,res=res,medicine=MEDICINE,dis=DISEASE)

            if DISEASE!='':

                if ints[0]['intent']=='cause':

                    PROCESS=2

                if ints[0]['intent']=='prevention':

                    PROCESS=5

                if ints[0]['intent']=='medicine':

                    PROCESS=4

                if ints[0]['intent']=='treatment':

                    PROCESS=3

            return res,PROCESS,DISEASE,MEDICINE

        return res,PROCESS,DISEASE,MEDICINE

    if PROCESS==2 or PROCESS==3 or PROCESS==4 or PROCESS==5 or PROCESS==6 or PROCESS==7 :

        ints =  predict\_class(question,model)

        if ints[0]['intent']!='accept':

            res = getResponse(ints, intents)

            if ints[0]['intent'] in ['cause','prevention','medicine','treatment']:

                return forCausePrevMedTreat(ints=ints,question=question,res=res,medicine=MEDICINE,dis=DISEASE)

            return res,0,DISEASE,MEDICINE

        if PROCESS==2 and ints[0]['intent']=='accept':

                PROCESS=3

                ints[0]['intent']=""

                return "treatment",PROCESS,DISEASE,MEDICINE

        if PROCESS==3 and ints[0]['intent']=='accept':

                ints[0]['intent']=""

                PROCESS=4

                return "medicine",PROCESS,DISEASE,MEDICINE

        if PROCESS==4 and ints[0]['intent']=='accept':

                ints[0]['intent']=""

                PROCESS=5

                return "prescription",PROCESS,DISEASE,MEDICINE

        if PROCESS==5 and ints[0]['intent']=='accept':

                ints[0]['intent']=""

                PROCESS=6

                return "effects",PROCESS,DISEASE,MEDICINE

        if PROCESS==6 and ints[0]['intent']=='accept':

                ints[0]['intent']=""

                PROCESS=7

                return "prevention",PROCESS,DISEASE,MEDICINE

        if PROCESS==7 and ints[0]['intent']=='accept':

                ints[0]['intent']=""

                PROCESS=0

                return "followup",PROCESS,DISEASE,MEDICINE

def forCausePrevMedTreat(ints,question,res,medicine,dis):

        disease=EntityDiseasePrediction(question)

        if disease!='':

            DISEASE=disease

            print('disease',disease)

        else:

            DISEASE=dis

            print('disease',dis)

        if ints[0]['intent']=='cause':

            PROCESS=2

            return 'cause',PROCESS,DISEASE,medicine

        if ints[0]['intent']=='prevention':

            PROCESS=5

            return 'prevention',PROCESS,DISEASE,medicine

        if ints[0]['intent']=='medicine':

            PROCESS=4

            return 'medicine',PROCESS,DISEASE,medicine

        if ints[0]['intent']=='treatment':

            PROCESS=3

            return 'treatment',PROCESS,DISEASE,medicine

        return res,PROCESS,DISEASE,medicine

def sendrequest(url,body,method):

    if method=='POST':

         response=requests.post(url,body)

         return response.json()

def cleansymptoms(symp):

    sym=symp.strip()

    symptoms=sym.replace(" ","\_")

    body={

            "symptoms":symptoms

            }

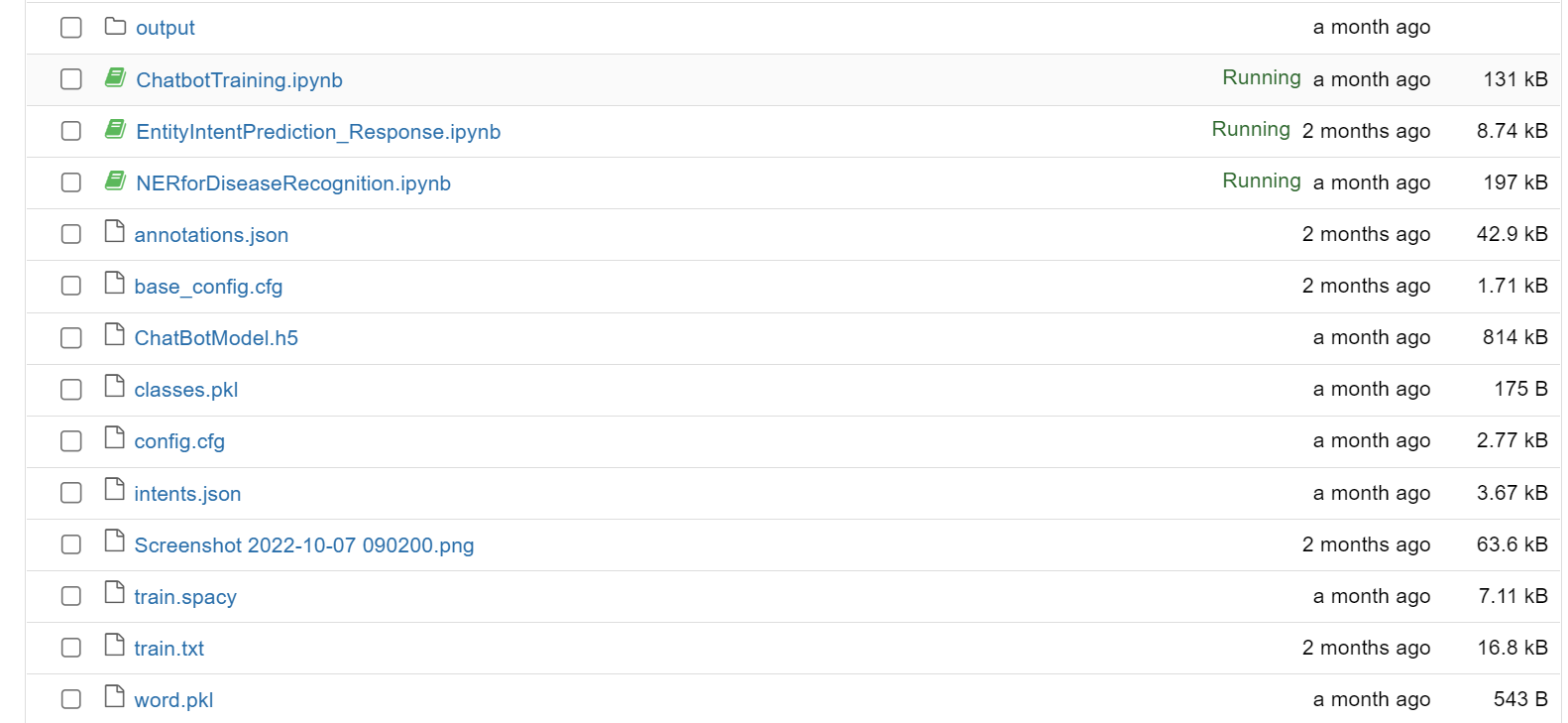
    app\_json = json.dumps(body)

    url='http://127.0.0.1:8001/predict/diseases'

    return url,app\_json

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

    uvicorn.run(app,host='0.0.0.0',port=8002)



Disease Prediction API:

from fastapi import FastAPI

from pydantic import BaseModel

import pickle

import uvicorn

import numpy as np

from tensorflow.keras.utils import load\_img, img\_to\_array

from keras.applications.vgg16 import preprocess\_input

from keras.preprocessing import image

from keras.models import load\_model

from scipy import spatial

app = FastAPI()

bagFeature=pickle.load(open("models/bagFeature.pkl","rb"))

symptoms\_features=pickle.load(open("models/symptoms\_features.pkl","rb"))

disease\_symptoms=pickle.load(open("models/diseas\_features.pkl","rb"))

Heartmodel=pickle.load(open('models/heart\_disease\_model.pck','rb'))

Pneumoniamodel=load\_model('models/Pneumoniadetection.h5')

Diabetemodel = pickle.load(open('models/diabetsModel.pck', 'rb'))

class DiabetesDiseases(BaseModel):

    pregnancies:float

    glucose:float

    bloodPressure:float

    skinThickness:float

    insulin:float

    bmI:float

    diabetesPedigreeFunction:float

    age:float

class HeartDiseases(BaseModel):

    age:float

    sex:float

    cp:float

    trestbps:float

    chol:float

    fbs:float

    restecg:float

    thalach:float

    exang:float

    oldpeak:float

    slope:float

    ca:float

    thal:float

class PneumoniaDiseases(BaseModel):

    path:str

class GeneralDiseases(BaseModel):

    symptoms:str

@app.post('/predict/diabetes')

async def predict\_species(values: DiabetesDiseases):

        data=values.dict()

        input\_data = (

            data['pregnancies'], data['glucose'], data['bloodPressure'], data['skinThickness'],

            data['insulin'], data['bmI'], data['diabetesPedigreeFunction'],data['age']

        )

        input\_data\_as\_numpy\_array = np.asarray(input\_data)

        # reshape the array as we are predicting for one instance

        input\_data\_reshaped = input\_data\_as\_numpy\_array.reshape(1,-1)

        prediction = Diabetemodel.predict(input\_data\_reshaped)

        print(prediction)

        if (prediction[0] == 0):

            return 'Peson not Diabetic'

        else:

            return 'Peson  Diabetic'

@app.post('/predict/heart')

async def predict\_species(values: HeartDiseases):

    data=values.dict()

    input\_data = (

            data['age'], data['sex'], data['cp'], data['trestbps'],

            data['chol'], data['fbs'], data['restecg'],data['thalach'],

             data['exang'],data['oldpeak'],data['slope'],data['ca'],

             data['thal']

        )

    input\_data\_as\_numpy\_array = np.asarray(input\_data)

    input\_data\_reshaped = input\_data\_as\_numpy\_array.reshape(1,-1)

    prediction = Heartmodel.predict(input\_data\_reshaped)

    print(prediction)

    if (prediction[0]== 0):

        return 'The Person does not have a Heart Disease'

    else:

        return 'The Person has Heart Disease'

@app.post('/predict/pneumonia')

async def predict\_species(imgPath: PneumoniaDiseases):

    data=imgPath.dict()

    #C:/Users/jaola/Documents/GitHub/Project CP3/normalPneu.jpeg

    img=load\_img(data['path'],target\_size=(224,224))

    x=img\_to\_array(img)

    x=np.expand\_dims(x, axis=0)

    img\_data=preprocess\_input(x)

    classes=Pneumoniamodel.predict(img\_data)

    result=int(classes[0][0])

    if result==0:

       return "Person is Affected By PNEUMONIA"

    else:

       return "Result is Normal"

@app.post('/predict/diseases')

async def predict\_species(symptoms: GeneralDiseases):

    data=symptoms.dict()

    bag=createbag(data['symptoms'])

    diagnosis\_percent={}

    for i in range(len(bagFeature)):

        result = 1 - spatial.distance.cosine(bagFeature[i][1], bag)

        diagnosis\_percent[bagFeature[i][0]]=result\*100

    diagnosis\_percent=sorted(diagnosis\_percent.items(),key=lambda x:x[1],reverse=True)

    n=4

    disease={}

    diseaselist=[]

    percentage=[]

    for i in range(n):

        diseaselist.append(diagnosis\_percent[i][0])

        percentage.append(diagnosis\_percent[i][1])

    disease={

      'diseaselist':diseaselist,

      'percentage':percentage,

      'statut':'accepted'

    }

    if percentage[0]<80:

        print(diseaselist,percentage)

        sympto=reduceDiseasePercentage(data['symptoms'],diseaselist,percentage)

        quest=askmoreSympto(list(sympto))

        value={

            "question":quest,

            "curentSymptom":data['symptoms'],

            'diseaselist':diseaselist,

            'percentage':percentage,

            'statut':'refused'

        }

        return value

    disease=dict(disease)

    return  disease

def reduceDiseasePercentage(symp,diseaselist,percentage):

    symptom=symp.split(',')

    sympts=[]

    for d in diseaselist:

        diseaseSympt=disease\_symptoms[d];

        for s in symptom:

            if s in diseaseSympt:

                diseaseSympt.remove(s)

        sympts.extend(diseaseSympt)

    print(set(sympts))

    return set(sympts)

def askmoreSympto(sympto):

    d=[]

    m=[]

    n=5

    for i in range(len(sympto)):

        d.append(sympto[i])

        if len(d)==n:

            m.append(d)

            d=[]

    return m

def createbag(symptoms):

      symptoms=symptoms.split(',')

      bag=[]

      for val in symptoms\_features:

          bag.append(1) if val in symptoms else bag.append(0)

      return bag

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

    uvicorn.run(app, host="0.0.0.0", port=8001)

Healthcare:

Django Projects:

Views.py:

from http.client import HTTPResponse

import random

from django.shortcuts import render,redirect

from django.http import HttpResponse

from django.http import JsonResponse

from main import form

from twilio.rest import Client

import json

from django.core.files.storage import FileSystemStorage

import requests

from django.contrib.auth.models import User

from django.contrib.auth import login, authenticate,logout

import pickle as pickle

import numpy as np

from django.views.decorators.csrf import csrf\_exempt

from main.form import PatientForm,MedicineForm,DiseasesForm,DoctorForm,RegistrationForm,loginForm,PneumoniaForm,AppointmentsForm

from main.models import AppointmentsModel, DiseasesModel,DoctorModel,HistoryModel,MedicineModel,PatientsModel,Profile,Discussion

from main.search import DiseaseSearch

DISEASE\_HOST='http://127.0.0.1:8001/'

CHATBOT\_HOST='http://127.0.0.1:8002/'

account\_sid = 'AC91ee2b42485471796a886124787de050'

auth\_token = '4dfc801c9fa59e200f29b4b0a57a3caf'

def registration(request):

    if request.method=='POST':

        print(request.POST)

        form=RegistrationForm(request.POST)

        if form.is\_valid():

            user=form.save()

            user.refresh\_from\_db()

            user.profile.fullname= form.cleaned\_data.get('fullname')

            user.profile.phone= form.cleaned\_data.get('phone')

            user.save()

            username = form.cleaned\_data.get('username')

            password = form.cleaned\_data.get('password1')

            user = authenticate(username=username, password=password)

            login(request, user)

            return redirect('prediction')

        else:

            print('login Error')

    form=RegistrationForm()

    return render (request,'userPage/registration.html',{'form':form})

def loginView(request):

    #User.objects.filter(is\_superuser=True).delete()

    if request.method=='POST':

        username=request.POST['username']

        password=request.POST['password1']

        user = authenticate(request, username=username, password=password)

        if user is not None:

            login(request, user)

            retractProfile(request,username)

            return redirect('prediction')

    form=loginForm()

    return render(request,'userPage/login.html',{'form':form})

def retractProfile(request,username):

    USERNAME=username

    u=User.objects.filter(username=USERNAME).values()[0]

    p=Profile.objects.get(user=u['id'])

    PHONE=p.phone

    FULLNAME=p.fullname

    request.session['uid']=u['id']

    request.session['username']=USERNAME

    request.session['phone']=PHONE

    request.session['fullname']=FULLNAME

    request.session['PROCESS']=0

    request.session['DISEASE']=''

    request.session['MEDICINE']=''

def index(request):

    return render(request,'userPage/home.html',{'page\_title':'Home'})

def profileview(request):

    return render(request,'userPage/profile.html',{})

def dashboard(request):

    disease=DiseasesModel.objects.all()

    medicine=MedicineModel.objects.all()

    patient=PatientsModel.objects.all()

    doctor=DoctorModel.objects.all()

    numdisease=len(disease)

    nummedicine=len(medicine)

    numpatient=len(patient)

    numdoctor=len(doctor)

    data={

        'page\_title':'Dashboard',

        'numdisease':numdisease,

        'nummedicine':nummedicine,

        'numpatient':numpatient,

        'numdoctor':numdoctor

        }

    return render(request,'dashboard.html',data)

def patient(request):

    form=PatientForm()

    data={

    'form':form,

     'page\_title':'Add Patient'

    }

    return render(request,'patientForm.html',data)

def medicine(request):

    form=MedicineForm()

    data={

    'form':form,

     'page\_title':'Add Medicine '

    }

    return render(request,'medicineForm.html',data)

def disease(request):

    form=DiseasesForm()

    data={

    'form':form,

     'page\_title':'Add disease '

    }

    return render(request,'diseasForm.html',data)

def doctors(request):

    form=DoctorForm()

    data={

    'form':form,

     'page\_title':'Add Doctors '

    }

    return render(request,'doctorForm.html',data)

def medicineManage(request):

    medicine=MedicineModel.objects.all()

    print(medicine.values())

    data={

     'results':medicine,

     'page\_title':'Manage Medicines '

    }

    if request.method=='POST':

        if request.POST['id']:

            print(request.POST)

            id =request.POST['id']

            print('-------------yu')

            med=MedicineModel.objects.get(pk=id)

            med.name=request.POST['name']

            med.definition=request.POST['definition']

            med.warnings=request.POST['warnings']

            med.prescription=request.POST['prescription']

            med.effects=request.POST['effects']

            med.save()

            print('update')

            med=MedicineModel.objects.all()

            data={

            'results':med,

            'page\_title':'Manage Medicines '

            }

            return render(request,'medicineManage.html',data)

        else:

            print('no id')

            med=MedicineForm(request.POST)

            if med.is\_valid():

                med.save()

                print('saved')

                med=MedicineModel.objects.all()

                print(med.values())

                data={

                'results':med,

                'page\_title':'Manage Medicines '

                }

                return render(request,'medicineManage.html',data)

    return render(request,'medicineManage.html',data)

def diseaseManage(request):

    disease=DiseasesModel.objects.all()

    data={

     'results':disease,

     'page\_title':'Manage Disease '

    }

    if request.method=='POST':

        if request.POST['id']:

            print(request.POST)

            id =request.POST['id']

            print('-------------yu')

            disease=DiseasesModel.objects.get(pk=id)

            disease.name=request.POST['name']

            disease.definition=request.POST['definition']

            disease.causes=request.POST['causes']

            disease.causes=request.POST['treatments']

            disease.save()

            print('update')

            disease=DiseasesModel.objects.all()

            data={

            'results':disease,

            'page\_title':'Manage Disease '

            }

            return render(request,'diseaseManage.html',data)

        else:

            print('no id')

            disease=DiseasesForm(request.POST)

            print(request.POST)

            if disease.is\_valid():

                disease.save()

                print('saved')

                disease=DiseasesModel.objects.all()

                print(disease.values())

                data={

                'results':disease,

                'page\_title':'Manage Disease '

                }

                return render(request,'diseaseManage.html',data)

    return render(request,'diseaseManage.html',data)

def patientManage(request):

    patient=PatientsModel.objects.all()

    data={

     'results':patient,

     'page\_title':'Manage Patient '

    }

    if request.method=='POST':

        if request.POST['id']:

            id =request.POST['id']

            print('-------------yu')

            patient=PatientsModel.objects.get(pk=id)

            patient.age=request.POST['age']

            patient.doctor=request.POST['doctor']

            patient.address=request.POST['address']

            patient.weight=request.POST['weight']

            if len(request.FILES)>0:

                            patient.image=request.FILES['image']

            patient.save()

            print('update')

            patient=PatientsModel.objects.all()

            print(patient.values())

            data={

            'results':patient,

            'page\_title':'Manage Patient '

            }

            return render(request,'patientManage.html',data)

        else:

            print('no id')

            patient=PatientForm(request.POST,request.FILES)

            print(request.POST,request.FILES)

            if patient.is\_valid():

                patient.save()

                print('saved')

                patient=PatientsModel.objects.all()

                print(patient.values())

                data={

                'results':patient,

                'page\_title':'Manage Patient '

                }

                return render(request,'patientManage.html',data)

            else:

                print('not valid')

    return render(request,'patientManage.html',data)

def doctorsManage(request):

    doctor=DoctorModel.objects.all()

    print(doctor.values())

    data={

     'results':doctor,

     'page\_title':'Manage Doctor '

    }

    if request.method=='POST':

        if request.POST['id']:

            print(request.POST)

            id =request.POST['id']

            print('-------------yu')

            doc=DoctorModel.objects.get(pk=id)

            doc.name=request.POST['name']

            doc.specialty=DiseasesModel.objects.get(pk=request.POST['specialty'])

            if len(request.FILES)>0:

                            doc.image=request.FILES['image']

            doc.save()

            print('update')

            doc=DoctorModel.objects.all()

            data={

            'results':doc,

            'page\_title':'Manage Doctor '

            }

            return render(request,'doctorManage.html',data)

        else:

            print('no id')

            doc=DoctorForm(request.POST,request.FILES)

            if doc.is\_valid():

                doc.save()

                print('saved')

                doc=DoctorModel.objects.all()

                print(doc.values())

                data={

                'results':doc,

                'page\_title':'Manage Doctors '

                }

                return render(request,'doctorManage.html',data)

    return render(request,'doctorManage.html',data)

def patientedit(request,id):

    form=PatientForm()

    print(id)

    data={

        'id':id,

        'form':form,

    }

    return render(request,'patientEdit.html',data)

def patientdelete(request,id):

    patient=PatientsModel.objects.get(pk=id)

    patient.delete()

    patient=PatientsModel.objects.all()

    print(patient.values())

    data={

     'results':patient,

     'page\_title':'Manage Patient '

    }

    return render(request,'patientManage.html',data)

def diseaseedit(request,id):

    form=DiseasesForm()

    print(id)

    data={

        'id':id,

        'form':form,

    }

    return render(request,'diseaseEdit.html',data)

def diseasedelete(request,id):

    disease=DiseasesModel.objects.get(pk=id)

    disease.delete()

    disease=DiseasesModel.objects.all()

    print(disease.values())

    data={

     'results':disease,

     'page\_title':'Manage Patient '

    }

    return render(request,'diseaseManage.html',data)

def doctoredit(request,id):

    form=DoctorForm()

    print(id)

    data={

        'id':id,

        'form':form,

    }

    return render(request,'doctorEdit.html',data)

def doctordelete(request,id):

    doctor=DoctorModel.objects.get(pk=id)

    doctor.delete()

    doctor=DoctorModel.objects.all()

    print(doctor.values())

    data={

     'results':doctor,

     'page\_title':'Manage Doctor '

    }

    return render(request,'doctorManage.html',data)

def medicineedit(request,id):

    form=MedicineForm()

    print(id)

    data={

        'id':id,

        'form':form,

    }

    return render(request,'medicineEdit.html',data)

def medicinedelete(request,id):

    med=MedicineModel.objects.get(pk=id)

    med.delete()

    med=MedicineModel.objects.all()

    print(med.values())

    data={

     'results':med,

     'page\_title':'Manage Medicine '

    }

    return render(request,'medicineManage.html',data)

def predictiondiabetes(request):

    if request.method=='POST':

        pregnancies=request.POST['pregnancies']

        glucose=request.POST['glucose']

        bloodPressure=request.POST['bloodPressure']

        skinThickness=request.POST['skinThickness']

        insulin=request.POST['insulin']

        bMI=request.POST['bmI']

        diabetesPedigreeFunction=request.POST['diabetesPedigreeFunction']

        age=request.POST['age']

        body={

        "pregnancies": int(pregnancies),

        "glucose": int(glucose),

        "bloodPressure": int(bloodPressure),

        "skinThickness": int(skinThickness),

        "insulin": int(insulin),

        "bmI": int(bMI),

        "diabetesPedigreeFunction": int(diabetesPedigreeFunction),

        "age": int(age)

        }

        print(body)

        app\_json = json.dumps(body)

        url='http://127.0.0.1:8001/predict/diabetes'

        response=requests.post(url,app\_json)

        print(response.json())

        data={

            'result':response.json(),

             'username':request.session['username'],

            'phone':request.session['phone'],

            'fullname':request.session['fullname'],

        }

        return render(request,'actions/diabetePrediction.html',data)

    data={

            'username':request.session['username'],

            'phone':request.session['phone'],

            'fullname':request.session['fullname'],

    }

    return render(request,'actions/diabetePrediction.html',data)

def predictionheart(request):

    if request.method=='POST':

        age=request.POST['age']

        sex=request.POST['sex']

        cp=request.POST['cp']

        trestbps=request.POST['trestbps']

        chol=request.POST['chol']

        fbs=request.POST['fbs']

        restecg=request.POST['restecg']

        thalach=request.POST['thalach']

        exang=request.POST['exang']

        oldpeak=request.POST['oldpeak']

        slope=request.POST['slope']

        ca=request.POST['ca']

        thal=request.POST['thal']

        body={

        "age": int(age),

        "sex": int(sex),

        "cp": int(cp),

        "trestbps": int(trestbps),

        "chol": int(chol),

        "fbs": int(fbs),

        "restecg": int(restecg),

        "thalach": int(thalach),

        "exang": int(exang),

        "oldpeak": int(oldpeak),

        "slope": int(slope),

        "ca": int(ca),

        "thal": int(thal)

        }

        print(body)

        app\_json = json.dumps(body)

        url='http://127.0.0.1:8001/predict/heart'

        response=requests.post(url,app\_json)

        print(response.json())

        data={

            'result':response.json(),

            'username':request.session['username'],

            'phone':request.session['phone'],

            'fullname':request.session['fullname'],

        }

        return render(request,'actions/heartdiseasePrediction.html',data)

    data={

           'username':request.session['username'],

            'phone':request.session['phone'],

            'fullname':request.session['fullname'],

         }

    return render(request,'actions/heartdiseasePrediction.html',data)

@csrf\_exempt

def prediction(request):

    if request.method=='POST':

        symp=request.POST['symptoms']

        sym=symp.strip()

        symptoms=sym.replace(" ","\_")

        body={

             "symptoms":symptoms

            }

        app\_json = json.dumps(body)

        url='http://127.0.0.1:8001/predict/diseases'

        response=requests.post(url,app\_json)

        print(response.json())

        data={

            'result':response.json(),

            'form': form,

            'username':request.session['username'],

            'phone':request.session['phone'],

            'fullname':request.session['fullname'],

            }

        return render(request,'actions/generaldisease.html',data)

    user = User.objects.get(username=request.session['username'])

    print(user)

    data={

           'username':request.session['username'],

            'phone':request.session['phone'],

            'fullname':request.session['fullname'],

    }

    return render(request,'actions/generaldisease.html',data)

@csrf\_exempt

def sendSymptoms(request):

     if request.method=='POST':

        symp=request.POST['symptom']

        sym=symp.strip()

        symptoms=sym.replace(" ","\_")

        symptoms=symptoms.split(",")

        symptoms=symptoms[:len(symptoms)-1]

        symptoms=",".join(symptoms)

        print(symptoms)

        body={

             "symptoms":symptoms

            }

        response=sendrequest(body=body,host=DISEASE\_HOST,url='predict/diseases',method='POST')

        print(response)

        data={

            'res':response,

        }

        return JsonResponse({'statut':200,'data':data})

def pneumonia(request):

    if request.method == 'POST':

        form = PneumoniaForm(request.POST, request.FILES)

        print(request.POST, request.FILES)

        if form.is\_valid():

            form.save()

            print(request.FILES['document'])

            urlfile="C:/Users/jaola/Documents/GitHub/Project CP3/HealthCare/"

            media='media/documents/'

            filename=request.FILES['document']

            urlpath=urlfile+media+str(filename)

            print(urlpath)

            body={

             "path":urlpath

            }

            app\_json = json.dumps(body)

            url='http://127.0.0.1:8001/predict/pneumonia'

            response=requests.post(url,app\_json)

            print(response.json())

            data={

                'result':response.json(),

                 'form': form,

                'username':request.session['username'],

                'phone':request.session['phone'],

                'fullname':request.session['fullname'],

            }

            return render(request, 'actions/pneumoniaPrediction.html',data)

    else:

        form = PneumoniaForm()

        data={

           'username':request.session['username'],

            'phone':request.session['phone'],

            'fullname':request.session['fullname'],

            'form': form,

        }

    return render(request, 'actions/pneumoniaPrediction.html',data )

def chatbot(request):

    p=Profile.objects.get(user=request.session['uid'])

    patient=PatientsModel.objects.get(profile=p.id)

    doctor=patient.doctor

    data={

           'username':request.session['username'],

            'room':str(doctor.profile.user.id)+"/"+str(patient.id),

        }

    return render(request,"actions/chatbot.html",data)

#request id

def doctorchatpatient(request,id,patentid):

    patient=PatientsModel.objects.filter(doctor\_id=id)

    doctor=DoctorModel.objects.get(id=id)

    doc={}

    doc['idprofile']=doctor.profile.id

    doc['idname']=doctor.profile.fullname

    patient=list(patient)

    datapatient=[]

    for p in patient:

        d={}

        profile=Profile.objects.get(id=p.profile\_id)

        u=User.objects.filter(username=profile).values()[0]

        d['name']=u['username']

        d['pid']=p.id

        d['did']=id

        d['image']=PatientsModel.objects.get(id=p.id).image.url

        datapatient.append(d)

    data={

     'patients':datapatient,

     'doctor':doc,

      'doctorid':id,

      'patentid':patentid,

       'room':str(doctor.profile.user.id)+"/"+str(patentid),

       'username':request.session['username'],

    }

    return render(request, 'doctor/doctorchatbot.html',data )

def doctorhome(request,id):

    doctor=DoctorModel.objects.get(id=id)

    doc={}

    doc['idprofile']=doctor.profile.id

    doc['idname']=doctor.profile.fullname

    doc['image']=doctor.image.url

    doc['specialty']=doctor.specialty

    from datetime import date

    today = date.today()

    print(today)

    patient=PatientsModel.objects.filter(doctor=id)

    todayappointment=[]

    appointment=[]

    for p in patient:

        app=AppointmentsModel.objects.filter(patient=p.id,date=today)

        if len(app)>0:

            appointment.append(app)

    print(appointment)

    for ap in appointment:

        appoint={}

        appoint['time']=ap[0].slot

        appoint['date']=ap[0].date

        appoint['age']=ap[0].age

        appoint['disease']=ap[0].disease

        appoint['patient']=ap[0].patient.id

        appoint['image']=ap[0].patient.image.url

        appoint['name']=ap[0].patient.profile.fullname

        todayappointment.append(appoint)

    print(todayappointment)

    data={

        'doctor':doc,

        'id':id,

        'todayappointment':todayappointment

    }

    return render(request,'doctor/doctorhome.html',data)

def doctorpatients(request,id):

    patient=PatientsModel.objects.filter(doctor=id)

    print('values')

    print(patient.values())

    doctor=DoctorModel.objects.get(id=id)

    doc={}

    doc['idprofile']=doctor.profile.id

    doc['idname']=doctor.profile.fullname

    doc['image']=doctor.image.url

    doc['specialty']=doctor.specialty

    patlist=[]

    for p in patient:

        pat={}

        pat['id']=p.id

        pat['name']=p.profile.fullname

        pat['image']=p.image.url

        pat['age']=p.age

        patlist.append(pat)

    print(patlist)

    data={

        'doctor':doc,

        'id':id,

        'patient':patlist

    }

    return render(request,'doctor/doctorpatients.html',data)

def patientchat(request):

    p=Profile.objects.get(user=request.session['uid'])

    patient=PatientsModel.objects.get(profile=p.id)

    doctor=patient.doctor

    data={

           'username':request.session['username'],

            'room':str(doctor.profile.user.id)+"/"+str(patient.id),

        }

    return render(request,'actions/patientchat.html',data)

def doctorappointment(request):

    if request.method=='POST':

        form=AppointmentsForm(request.POST)

        print(request.POST)

        if form.is\_valid():

            print('form accepted')

            form.save()

            data={

            'form':form

            }

            return render(request,'doctor/doctorappointment.html',data)

        else:

            print('form invalid')

    form=AppointmentsForm()

    data={

   'form':form

    }

    return render(request,'doctor/doctorappointment.html',data)

@csrf\_exempt

def chatappointment(request):

    if request.method=='POST':

        p=Profile.objects.get(user=request.session['uid'])

        patient=PatientsModel.objects.get(profile=p.id)

        print(request.POST)

        disease=request.POST['disease']

        age=request.POST['age']

        date=request.POST['date']

        slot=request.POST['slot']

        app=AppointmentsModel.objects.create(patient=patient,disease=disease,age=age,date=date,slot=slot)

        print(app)

        print('Appo saved')

        data={

            'res':'Appointment Submitted'

        }

        return JsonResponse({'statut':'200','data':data})

@csrf\_exempt

def doctorchat(request):

    if request.method=='POST':

        p=Profile.objects.get(user=request.session['uid'])

        patient=PatientsModel.objects.get(profile=p.id)

        doctor=patient.doctor

        data={'doctorname':doctor.profile.fullname,'id':doctor.profile.user.id,

        'patientid':patient.id,

        'patUsername':patient.profile.user.username,

        'docUsername':doctor.profile.user.username,

        }

        print(data)

        return JsonResponse({'statut':'200','data':data})

@csrf\_exempt

def doctorappointmentdate(request):

    if request.method=="POST":

        date=request.POST['date']

        print(request.POST)

        slots=[

                '9:00','10:00','11:00','12:00','14:00','15:00'

            ]

        og=AppointmentsModel.objects.all()

        obj=AppointmentsModel.objects.filter(date=date)

        print(obj)

        for o in obj:

            slots.remove(o.slot)

            print(o.slot)

        data={

        "slots":slots

        }

        print(data)

        return JsonResponse({'statut':'200','data': data})

@csrf\_exempt

def chatbotsend(request):

    if request.method=='POST':

        print(request.POST)

        message=request.POST['message']

        PROCESS=request.session['PROCESS']

        DISEASE=request.session['DISEASE']

        MEDICINE=request.session['MEDICINE']

        body={

            "question": message,

            "PROCESS": PROCESS,

            "DISEASE":DISEASE,

            'MEDICINE':MEDICINE

            }

        response,process,disease,medicine=sendrequest(CHATBOT\_HOST,'chatbot/first',body,'POST')

        print(response)

        request.session['MEDICINE']=medicine

        request.session['PROCESS']=process

        request.session['DISEASE']=disease

        p=0

        if response=='ASK':

            p=1

            response=showDisease(disease)

        if response=='cause':

            p=2

            response=causemessage(disease,DiseaseSearch(disease).causes())

        if response=='treatment':

            p=2

            response=treatmentmessage(disease,DiseaseSearch(disease).treatments())

        if response=='medicine':

            p=2

            response=medicinemessage(disease,DiseaseSearch(disease).medicine())

            MEDICINE=DiseaseSearch(disease).medicinename()

        if response=='prevention':

            p=2

            response=preventionmessage(disease,DiseaseSearch(disease).preventions())

        if response=='followup':

            p=1

            response=followupmessage(2)

        if response=='prescription':

            p=2

            response=prescriptionmessage(DiseaseSearch(disease).medicineprecription())

        if response=='effects':

            p=2

            response=effectsmessage(DiseaseSearch(disease).medicineeffects())

        print(response,PROCESS,DISEASE)

        data={

            'res':response,

            'position':p

        }

        return JsonResponse({'statut':200,'data':data})

def sendrequest(host,url,body,method):

    if method=='POST':

         app\_json = json.dumps(body)

         uri=host+url

         response=requests.post(uri,app\_json)

         return response.json()

def sendtowatsapp(body,tonumber):

    client = Client(account\_sid, auth\_token)

    message = client.messages.create(

                              from\_='whatsapp:+14155238886',

                              body=body,

                              to=tonumber

                          )

def causemessage(disease,cause):

    p1='Oh, you have {} .{} .Would you like to know the treatment of {}'.format(disease,cause,disease)

    p2='I am sorry you have {}.{} .Don\'t you   like to know how to treat of {}'.format(disease,cause,disease)

    return random.choice([p1,p2])

def treatmentmessage(disease,treatment):

    p1="Here are the treatment for {} {} .Would you like to know what medicine to take ? ".format(disease,treatment)

    p2="for {} you should {}.Would you want to take medicine?".format(disease,treatment)

    return random.choice([p1,p2])

def medicinemessage(disease,medicine):

    p1='for {} medicine you should take {} would you like to have prescriptions for it '.format(disease,medicine)

    p2='I recommend you to take {} .It is very effective for {}. do you want to know ho to use it ?'.format(medicine,disease)

    return random.choice([p1,p2])

def prescriptionmessage(prescr):

    p1='{} .Would you like to know the sides effects for it?'.format(prescr)

    p2='{} .Would you like to know what happen if you overdose ?'.format(prescr)

    return random.choice([p1,p2])

def effectsmessage(effects):

    p1='{} .Always make sure you follow doctors prescription? Would you like to know how to prevent this disease ?'.format(effects)

    p2='{} .Please take it according to the appropriate dosage ? Would you like to know how to prevent this disease?'.format(effects)

    return random.choice([p1,p2])

def preventionmessage(disease,prevention):

    p1='To prevent {} .The preventions are {}.Would you like me to follow up ?'.format(disease,prevention)

    p2='for {} preventions. You should {}. Would you like me to follow up ?'.format(disease,prevention)

    return random.choice([p1,p2])

def followupmessage(no):

    p1='Take care of yourself.After {} days follow up would be necessary. In case you should take an oppointment '.format(no)

    p2='After {} days .If  no result then you should go to the doctor'.format(no)

    return random.choice([p1,p2])

def showDisease(d):

    p1='you might be suffering from {} . which service would you like to take'.format(d)

    p2='you have {} . which service would you like to take'.format(d)

    return random.choice([p1,p2])

def logoutview(request):

    logout(request)

    return render(request,'userPage/home.html',{'page\_title':'Home'})

Urls.py:

from django.urls import path

from . import views

urlpatterns = [

    path("",views.index,name='index'),

     path("dashboard/",views.dashboard,name='dashboard'),

    path("patient/",views.patient,name='patient'),

     path("medicine/",views.medicine,name='medicine'),

     path("disease/",views.disease,name='disease'),

     path("doctors/",views.doctors,name='doctors'),

      path("profile",views.profileview,name='profile'),

      #manage

    path("medicine/manage",views.medicineManage,name='medicinemanage'),

     path("disease/manage",views.diseaseManage,name='diseasemanage'),

      path("patient/manage",views.patientManage,name='patientmanage'),

        path("doctors/manage",views.doctorsManage,name='doctorsmanage'),

        path("patient/manage/edit/<int:id>",views.patientedit,name='patientedit'),

        path("patient/manage/delete/<int:id>",views.patientdelete,name='patientdelete'),

         path("disease/manage/edit/<int:id>",views.diseaseedit,name='diseaseedit'),

        path("disease/manage/delete/<int:id>",views.diseasedelete,name='diseasedelete'),

         path("doctors/manage/edit/<int:id>",views.doctoredit,name='doctoredit'),

        path("doctors/manage/delete/<int:id>",views.doctordelete,name='doctordelete'),

         path("medicine/manage/edit/<int:id>",views.medicineedit,name='medicineedit'),

        path("medicine/manage/delete/<int:id>",views.medicinedelete,name='medicinedelete'),

        #registration

         path("patient/registration",views.registration,name='registration'),

          path("login/",views.loginView,name='login'),

          path("logout/",views.logoutview,name='logout'),

           path("disease/prediction/diabetes",views.predictiondiabetes,name='predictiondiabetes'),

            path("disease/prediction/heart",views.predictionheart,name='predictionheart'),

            path("disease/prediction/pneumonia",views.pneumonia,name='pneumonia'),

            path("disease/prediction/",views.prediction,name='prediction'),

            path("disease/prediction/sendSymptoms",views.sendSymptoms,name='sendSymptoms'),

            path("disease/prediction/chatbot",views.chatbot,name='chatbot'),

             path("disease/prediction/chatbotsend",views.chatbotsend,name='chatbotsend'),

            path('doctors/patients/appointment',views.doctorappointment,name='doctorappointment'),

             path('doctors/patients/slots',views.doctorappointmentdate,name='doctorappointmentdate'),

            path('doctors/chatbot/<int:id>/<int:patentid>',views.doctorchatpatient,name='doctorchatpatient'),

            path('doctors/home/<int:id>',views.doctorhome,name='doctorhome'),

            path('doctors/patients/<int:id>',views.doctorpatients,name='doctorpatients'),

            path('disease/prediction/appointment',views.chatappointment, name='chatappointment'),

            path('disease/prediction/slots',views.doctorappointmentdate, name='patientappointmentdate'),

            path('disease/prediction/chat',views.doctorchat, name='doctorchat'),

            path('disease/prediction/patientchat',views.patientchat, name='patientchat'),