**ABSTRACT**

Healthcare is very important to lead a good life. However, it is very difficult to obtain consultation with the doctor for every health problem. The idea is to create a medical chatbot using Artificial Intelligence that can diagnose the disease and provide basic details about the disease before consulting a doctor. Through chatbots one can communicate with text or voice interface and get replies through artificial intelligence

Typically, a chat bot will communicate with a real person. Chat bots are used in applications such as

1. E-commerce customer service,
2. Call centers,
3. Internet gaming.

Chatbots are programs built to automatically engage with received messages. Chatbots can be programmed to respond the same way each time, to respond differently to messages containing certain keywords and even to use machine learning to adapt their responses to fit the situation. A developing number of hospitals, nursing homes, and even private centers, presently utilize online Chatbots for human services on their sites. These bots connect with potential patients visiting the site, helping them discover specialists, booking their appointments, and getting them access to the correct treatment.

**Table of Contents**

1. Preamble

1.1 Introduction

1.2 Problem statement

2. System design

3. SRS

4. Implementation

5. Software testing

6. Code

7. Conclusion

8. References

**PREAMBLE**

**a.) INTRODUCTION**

A chatbot is an artificial intelligence (AI) software that can simulate a conversation (or a chat) with a user in natural language through messaging applications, websites, mobile apps or through the telephone. A chatbot is often described as one of the most advanced and promising expressions of interaction between humans and machines. However, from a technological point of view, a chatbot only represents the natural evolution of a Question Answering system leveraging Natural Language Processing (NLP). Formulating responses to questions in natural language is one of the most typical Examples of Natural Language Processing applied in various enterprises’ end-use applications.

In some cases, health chatbots are also able to connect patients with clinicians for diagnosis or treatment, but that is already one step further down the line. The general idea is that in the future, these talking or texting smart algorithms might become the first contact point for primary care. Patients will not get in touch with physicians or nurses or any medical professional with every one of their health questions but will turn to chatbots first. If the little medical helper could not comfortably respond to the raised issues, it will transfer the case to a real-life doctor.

**b.) PROBLEM STATEMENT**

As in today’s world, the number of patients daily is increasing rapidly with the change in lifestyle. The queues in hospitals & local doctor’s residences are therefore on a rapid increase. Patients with busy schedules have to spend a lot of time waiting to meet the doctor. Some diseases take a lot of time to cure, thus some working patients may be forced to take a leave.

Most people in this new generation don’t pay attention to health. Performing medical check- ups regularly is very much mandatory. This helps in early detection of diseases such as Cancer, AIDS etc. which can help perform early diagnosis. Detection of disease at a later stage can be very risky and diagnosis at this stage can be very expensive.

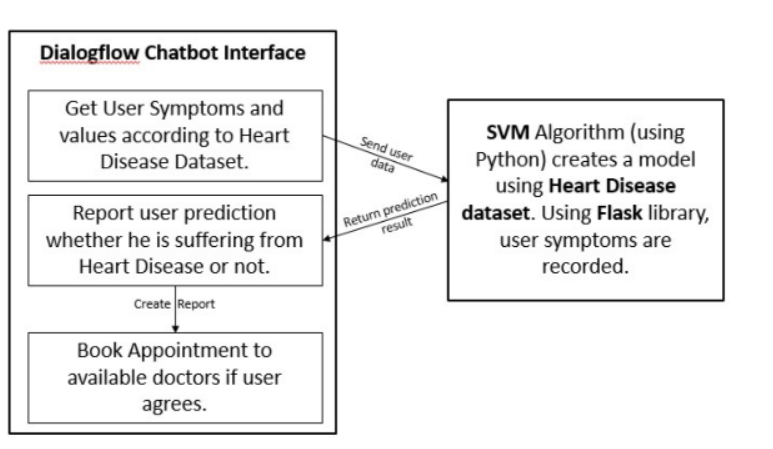
Heart Disease is a very common problem worldwide. Every day, many people, young and old, die of a heart attack. Taking care of the Heart is very important and as mentioned above, early detection of such a disease is necessary. However, there are limited doctors available in this field. Some of them charge reasonable amount of consultation charges which some poor people cannot afford. Hence, there is a need for a platform which can help in detection of heart disease without the presence of a doctor, free of cost and user-friendly to use for old as well as young people.

Most Medical Chatbot’s available are for general purposes i.e., not specific to a certain medical domain like Cardiology (heart), Gastroenterology etc. Most of these chatbot’s don’t have the feature of booking appointments with related doctors.

The objective is to develop a conversational system to predict heart diseases using dialogflow as front end, SVM algorithm to classify the dataset and predict whether the user suffers from heart disease or not.

The System will be able to get the patient reports, analyze it and conclude whether he/she is suffering from heart diseases or not. It also helps to get an early diagnosis. The system will also be trained to provide appointment booking facilities with related doctors. This healthcare boot system will help hospitals to provide healthcare support online 24 x 7.

**SYSTEM DESIGN**



**SRS**

**Hardware Requirements**

* Pentium Processor IV or higher
* Min 10 GB HOD
* RAM 512 MB or higher
* 2.4 GHz or faster processor

**Software Requirements**

* Windows Vista onwards
* Python compiler
* Dialogflow machine learning library
* Flask
* Pandas
* Sklearn
* mtplib
* Json

**IMPLEMENTATION**

The following Tools & Technologies were used to implement this project:

* SVM Algorithm
* Heart Disease Dataset
* Dialogflow (API.ai)
* ngrok
* Python
* Python Libraries used: Flask, pandas, sklearn, smtplib, json, os.
* Eclipse/Liclipse editor
* HTML, CSS, JS

**CODE**

from flask import Flask, render\_template, request, make\_response

import json, os

import SVM

import pygmail

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

speech, name, age1, gender = "","","",""

problem, sym\_duration = "",""

q1,q2,q3,q4,q5,q6 = 0,0,0,0,0,0

sym1,sym2,sym3 = "","",""

new\_report, tips, soln = "","", ""

doctor, date, time, uemail = "","","",""

@app.route('/Home')

def Home():

return render\_template('Home.html')

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

def Home\_Value():

global uemail

uemail = request.form['uemail']

return render\_template('Main.html')

@app.route('/')

def index():

if uemail == "": return render\_template('Home.html')

else: return render\_template('Main.html')

@app.route('/Form')

def Form():

if uemail == "": return render\_template('Home.html')

else: return render\_template('Form.html')

@app.route('/Chatbot')

def Chatbot():

if uemail == "": return render\_template('Home.html')

else: return render\_template('Chatbot.html')

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

def getvalue():

age = age1

sex = gender

if sex == 'Male': sex = 1

elif sex == 'Female': sex = 0

cp = request.form['cp']

if cp == 'Typical Angina': cp = 0

elif cp == 'Atypical Angina': cp = 1

elif cp == 'Non-anginal Pain': cp = 2

elif cp == 'Asymptomatic': cp = 3

trestbps = request.form['trestbps']

chol = request.form['chol']

fbs = request.form['fbs']

if fbs == 'Yes': fbs = 1

elif fbs == 'No': fbs = 0

restecg = request.form['restecg']

if restecg == 'Normal': restecg = 0

elif restecg == 'Having ST-T Wave Abnormality': restecg = 1

elif restecg == 'Left Ventricular Hyperthrophy': restecg = 2

thalach = request.form['thalach']

exang = request.form['exang']

if exang == 'Yes': exang = 1

elif exang == 'No': exang = 0

oldpeak = request.form['oldpeak']

slope = request.form['slope']

if slope == 'Upsloping': slope = 0

elif slope == 'Flat': slope = 1

elif slope == 'Downsloping': slope = 2

ca = request.form['ca']

thal = request.form['thal']

if thal == 'Normal': thal = 1

elif thal == 'Fixed Defect': thal = 2

elif thal == 'Reversible Defect': thal = 3

op = SVM.svm\_pred(age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang,

oldpeak, slope, ca, thal)

global speech

if op == 0:

opstr = "No Heart Disease"

speech = "Your Report Looks Fine."

if op == 1:

opstr = "Heart Disease Present"

speech = "You may be suffering from a Heart Disease/problem!"

return render\_template('pass.html', n=op, s=opstr)

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

def webhook():

req = request.get\_json(silent=True, force=True)

res = makeWebhookResult(req)

res = json.dumps(res, indent=4)

r = make\_response(res)

r.headers['Content-Type'] = 'application/json'

return r

def makeWebhookResult(req):

# Get all the Query Parameter

query\_response = req["queryResult"]

res = { "fulfillmentText": "", }

global name

global gender

global age1

global sym1

global sym2

global new\_report

# Patient\_Name

if query\_response.get("action") == "user\_name":

r = query\_response.get("parameters")

r1 = r.get("given-name")

global name

name = r1

if query\_response.get("action") == "user\_age":

r = query\_response.get("parameters")

r2 = r.get("age")

r1 = r2.get("amount")

global age1

age1 = int(r1)

# Checkup\_Patient\_gender

if query\_response.get("action") == "DefaultWelcomeIntent.DefaultWelcomeIntent-

custom.Checkup\_Patient-custom":

r = query\_response.get("parameters")

r1 = r.get("Gender")

global gender

gender = r1

a1 = "OK "+ name + "(" + str(age1) + "), Please fill and submit the report

form on the right side...."

res = { "fulfillmentText": a1, }

# Checkup\_Patient\_filling

if query\_response.get("action") == "DefaultWelcomeIntent.DefaultWelcomeIntent-

custom.Checkup\_Patient-custom.Checkup\_Patient\_gender-custom":

a2 = "Thanks " + name + ", after analysing the information you have given us,

The System Predicts that "+ speech +" Please note, this is not a diagnosis. Always

visit a doctor if you are in doubt, or if your symptoms get worse or don't improve. If

your situation is serious, always call the emergency services. Do you want to book an

appointment with a doctor?"

res = { "fulfillmentText": a2, }

############ Suffering Patient ##########################

# Suffering\_Patient

if query\_response.get("action") == "DefaultWelcomeIntent.DefaultWelcomeIntent-

custom":

# Main Problem

global problem

global soln

r = query\_response.get("parameters")

problem = r.get("Symptoms")

if problem == "Chest Pain": soln = "A pain reliever, such as aspirin, can help

alleviate the heart/chest pain associated with less severe cases. When heart pain

strikes, lying down immediately with the head elevated above the body may bring some

relief. A slightly upright position helps when the pain is due to reflux."

elif problem == "High Blood Pressure": soln = "Blood pressure often increases

as weight (Obesity) increases. Regular physical activity such as 150 minutes a week

can lower your blood pressure by about 5 to 8 mm Hg if you have high blood pressure.

Eating a diet that is rich in whole grains, fruits, vegetables and low-fat dairy

products and skimps on saturated fat and cholesterol can lower your blood pressure by

up to 11 mm Hg if you have high blood pressure. Chronic stress and also smoking may

contribute to high blood pressure, so avoid that."

elif problem in("breathing problems", "breathlessness"): soln = "Breathing-in

deeply through the abdomen and also ursed-lip breathing can help to manage your

breathlessness. Finding a comfortable and supported position to stand or lie in can

help to relax and catch your breath. Inhaling steam can help to keep nasal passages

clear, which can help to breathe more easily. Drinking black coffee may help to treat

breathlessness, reducing tiredness in the airway muscles. Being overweight also can

cause disrupted breathing while you sleep (sleep apnea)."

elif problem in("sleeping", "sleep"): soln = "Set yourself up for restful

sleep: Stick to a regular sleep/wake schedule. Turn off the TV, computer, and other

devices before bedtime. Keep your bedroom cool and dark. Avoid alcohol before bedtime

and caffeine in the afternoon or evening. Exercise every morning."

# Suffering\_Patient\_symp\_dur

if query\_response.get("action") == "DefaultWelcomeIntent.DefaultWelcomeIntent-

custom.Suffering\_Patient-custom":

# Duration

global sym\_duration

sym\_duration = query\_response.get("queryText")

# Suffering\_Patient\_Q2

if query\_response.get("action") == "DefaultWelcomeIntent.DefaultWelcomeIntent-

custom.Suffering\_Patient-custom.Suffering\_Patient\_symp\_dur-custom":

# Q. Heart Disease

r = query\_response.get("parameters")

r1 = r.get("Confirmation")

global q1

if r1 == 'Yes': q1 = 1

# Suffering\_Patient\_Q3

if query\_response.get("action") == "DefaultWelcomeIntent.DefaultWelcomeIntent-

custom.Suffering\_Patient-custom.Suffering\_Patient\_symp\_dur-

custom.Suffering\_Patient\_Q2-custom":

# Q. Diabetes

r = query\_response.get("parameters")

r1 = r.get("Confirmation")

global q2

if r1 == 'Yes': q2 = 1

# Suffering\_Patient\_Q4

if query\_response.get("action") == "DefaultWelcomeIntent.DefaultWelcomeIntent-

custom.Suffering\_Patient-custom.Suffering\_Patient\_symp\_dur-

custom.Suffering\_Patient\_Q2-custom.Suffering\_Patient\_Q3-custom":

# Q. High Blood Pressure

r = query\_response.get("parameters")

r1 = r.get("Confirmation")

global q3

if r1 == 'Yes': q3 = 1

# Suffering\_Patient\_Q5

if query\_response.get("action") == "DefaultWelcomeIntent.DefaultWelcomeIntent-

custom.Suffering\_Patient-custom.Suffering\_Patient\_symp\_dur-

custom.Suffering\_Patient\_Q2-custom.Suffering\_Patient\_Q3-custom.Suffering\_Patient\_Q4-

custom":

# Q. Chronic Obstructive Lung Disease/Asthma

r = query\_response.get("parameters")

r1 = r.get("Confirmation")

global q4

if r1 == 'Yes': q4 = 1

# Suffering\_Patient\_Q6

if query\_response.get("action") == "DefaultWelcomeIntent.DefaultWelcomeIntent-

custom.Suffering\_Patient-custom.Suffering\_Patient\_symp\_dur-

custom.Suffering\_Patient\_Q2-custom.Suffering\_Patient\_Q3-custom.Suffering\_Patient\_Q4-

custom.Suffering\_Patient\_Q5-custom":

# Q. Smoking

r = query\_response.get("parameters")

r1 = r.get("Confirmation")

global q5

if r1 == 'Yes': q5 = 1

# Suffering\_Patient\_sym1

if query\_response.get("action") == "DefaultWelcomeIntent.DefaultWelcomeIntent-

custom.Suffering\_Patient-custom.Suffering\_Patient\_symp\_dur-

custom.Suffering\_Patient\_Q2-custom.Suffering\_Patient\_Q3-custom.Suffering\_Patient\_Q4-

custom.Suffering\_Patient\_Q5-custom.Suffering\_Patient\_Q6-custom":

# Q. brain stroke/ Overweight/Obese / Kidney Disease

r = query\_response.get("parameters")

r1 = r.get("Confirmation")

global q6

if r1 == 'Yes': q6 = 1

# Suffering\_Patient\_sym2

if query\_response.get("action") == "symp1":

global sym1

sym1 = query\_response.get("queryText")

# Suffering\_Patient\_sym3

if query\_response.get("action") == "Suffering\_Patient\_sym2.Suffering\_Patient\_sym2-

custom":

global sym2

sym2 = query\_response.get("queryText")

# Suffering\_Patient\_sym\_final

if query\_response.get("action") == "Suffering\_Patient\_sym2.Suffering\_Patient\_sym2-

custom.Suffering\_Patient\_sym3-custom":

global sym3

sym3 = query\_response.get("queryText")

###### Create Report ############

new\_report = ""

new\_report += "To Summerize: You had "+problem+" for "+sym\_duration+"

duration."

if q1 == 1: new\_report += "You had Heart Disease before, "

if q2 == 1: new\_report += "You have/had Diabetes, "

if q3 == 1: new\_report += "You have/had High Blood Pressure, "

if q4 == 1: new\_report += "You've suffered from Asthma OR Chronic Obstructive

lung disease, "

if q5 == 1: new\_report += "You're Smoking (or smoked before), "

if q6 == 1: new\_report += "You have/had: Brain Stroke OR Kidney Disease OR

Obesity Problem, "

if sym1 not in("none","None") or sym2 not in("none","None") or sym3 not

in("none","None"): new\_report += "and also have symptoms like "

if sym1 not in("none","None"): new\_report += sym1+", "

if sym2 not in("none","None"): new\_report += sym2+", "

if sym3 not in("none","None"): new\_report += sym3

if (q1,q2,q3,q4,q5,q6) == 0: new\_report += "You did'nt have any of the above

mentioned symptoms!"

# Suffering\_Patient\_sym\_report\_filling

if query\_response.get("action") == "Suffering\_Patient\_sym2.Suffering\_Patient\_sym2-

custom.Suffering\_Patient\_sym3-custom.Suffering\_Patient\_sym\_final-

custom.Suffering\_Patient\_sym\_report\_yes-custom":

r = query\_response.get("parameters")

name = r.get("given-name")

gender = r.get("Gender")

r2 = r.get("age")

r1 = r2.get("amount")

age1 = int(r1)

# Suffering\_Patient\_sym\_report\_results

if query\_response.get("action") == "Suffering\_Patient\_sym2.Suffering\_Patient\_sym2-

custom.Suffering\_Patient\_sym3-custom.Suffering\_Patient\_sym\_final-

custom.Suffering\_Patient\_sym\_report\_yes-custom.Suffering\_Patient\_sym\_report\_filling-

custom":

a2 = "Thanks " + name + ", after analysing the information you have given us,

The System Predicts that "+ speech +" "+ new\_report +". Please note, this is not a

diagnosis. Always visit a doctor if you are in doubt, or if your symptoms get worse or

don't improve. If your situation is serious, always call the emergency services. Do

you want to book an appointment with a doctor?"

res = { "fulfillmentText": a2, }

# Suffering\_Patient\_sym\_report\_no

if query\_response.get("action") == "Suffering\_Patient\_sym2.Suffering\_Patient\_sym2-

custom.Suffering\_Patient\_sym3-custom.Suffering\_Patient\_sym\_final-custom":

r = "OK, "+ new\_report + ". Some ways how you can avoid this problem are: " +

soln + "--> Please note, this is not a diagnosis. Always visit a doctor if you are in

doubt, or if your symptoms get worse or don't improve. If your situation is serious,

always call the emergency services. Do you want to book an appointment with a doctor?"

res = { "fulfillmentText": r, }

######## DOCTOR SECTION ##########

# app\_date\_time

if query\_response.get("action") == "doctors\_list.doctors\_list-custom":

r1 = query\_response.get("queryText")

global doctor

doctor = r1

# app\_booked

if query\_response.get("action") == "doctors\_list.doctors\_list-

custom.app\_date\_time-custom":

r = query\_response.get("parameters")

Doctor Chatbot

Department of Computer Engineering, S.R.I.E.I.T. Page 41

r1 = r.get("date")

r2 = r.get("time")

global date

global time

date = r1

time = r2

pygmail.sendEmail( uemail, doctor, date, time, name, new\_report )

return res

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

port = int(os.getenv('PORT', 5000))

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

Format of JSON Request: for input “I have chest pain”.

Request:

{

"responseId": "37969871-e938-48f1-89e2-afcfeab59b85-266f04e0",

"queryResult": {

"queryText": "i have chest pain",

"action": "DefaultWelcomeIntent.DefaultWelcomeIntent-custom",

"parameters": {

"Symptoms": "Chest Pain"

},

"allRequiredParamsPresent": true,

"fulfillmentText": "For how long have you been suffering from it?",

"fulfillmentMessages": [

{

"text": {

"text": [

"For how long have you been suffering from it?"

]

}

}

],

"outputContexts": [

{

"name": "projects/doctor-chatbot-kepbfr/agent/sessions/11ad2821-98f2-

6b77-2373-a59ae7535e46/contexts/symp\_duration",

"lifespanCount": 5,

"parameters": {

"Symptoms": "Chest Pain",

"Symptoms.original": "chest pain"

}

},

{

"name": "projects/doctor-chatbot-kepbfr/agent/sessions/11ad2821-98f2-

6b77-2373-a59ae7535e46/contexts/suffering\_patient-followup",

"lifespanCount": 2,

"parameters": {

"Symptoms": "Chest Pain",

"Symptoms.original": "chest pain"

}

},

{

"name": "projects/doctor-chatbot-kepbfr/agent/sessions/11ad2821-98f2-

6b77-2373-a59ae7535e46/contexts/defaultwelcomeintent-followup",

"lifespanCount": 2,

"parameters": {

"Symptoms": "Chest Pain",

"Symptoms.original": "chest pain"

}

},

{

"name": "projects/doctor-chatbot-kepbfr/agent/sessions/11ad2821-98f2-

6b77-2373-a59ae7535e46/contexts/\_\_system\_counters\_\_",

"parameters": {

"no-input": 0.0,

"no-match": 0.0,

"Symptoms": "Chest Pain",

"Symptoms.original": "chest pain"

}

}

],

"intent": {

"name": "projects/doctor-chatbot-kepbfr/agent/intents/02b4271f-818f-4b64-

94bc-0b2c3e51d350",

"displayName": "Suffering\_Patient"

},

"intentDetectionConfidence": 1.0,

"languageCode": "en"

},

"originalDetectIntentRequest": {

"payload": {}

},

"session": "projects/doctor-chatbot-kepbfr/agent/sessions/11ad2821-98f2-6b77-2373-

a59ae7535e46"

}

{'queryText': 'i have chest pain', 'action':

'DefaultWelcomeIntent.DefaultWelcomeIntent-custom', 'parameters': {'Symptoms': 'Chest

Pain'}, 'allRequiredParamsPresent': True, 'fulfillmentText': 'For how long have you

been suffering from it?', 'fulfillmentMessages': [{'text': {'text': ['For how long

have you been suffering from it?']}}], 'outputContexts': [{'name': 'projects/doctor-

chatbot-kepbfr/agent/sessions/11ad2821-98f2-6b77-2373-

a59ae7535e46/contexts/symp\_duration', 'lifespanCount': 5, 'parameters': {'Symptoms':

'Chest Pain', 'Symptoms.original': 'chest pain'}}, {'name': 'projects/doctor-chatbot-

kepbfr/agent/sessions/11ad2821-98f2-6b77-2373-a59ae7535e46/contexts/suffering\_patient-

followup', 'lifespanCount': 2, 'parameters': {'Symptoms': 'Chest Pain',

'Symptoms.original': 'chest pain'}}, {'name': 'projects/doctor-chatbot-

kepbfr/agent/sessions/11ad2821-98f2-6b77-2373-

a59ae7535e46/contexts/defaultwelcomeintent-followup', 'lifespanCount': 2,

'parameters': {'Symptoms': 'Chest Pain', 'Symptoms.original': 'chest pain'}}, {'name':

'projects/doctor-chatbot-kepbfr/agent/sessions/11ad2821-98f2-6b77-2373-

a59ae7535e46/contexts/\_\_system\_counters\_\_', 'parameters': {'no-input': 0.0, 'no-

match': 0.0, 'Symptoms': 'Chest Pain', 'Symptoms.original': 'chest pain'}}], 'intent':

{'name': 'projects/doctor-chatbot-kepbfr/agent/intents/02b4271f-818f-4b64-94bc-

0b2c3e51d350', 'displayName': 'Suffering\_Patient'}, 'intentDetectionConfidence': 1.0,

'languageCode': 'en'}

{ "fulfillmentText": "" }

**SVM.py**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn import svm

def svm\_pred(age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak,

slope, ca, thal):

data = pd.read\_csv("heart.csv")

train = data.drop('target', axis = 1)

target = data.target

X\_train, X\_test, y\_train, y\_test = train\_test\_split( train, target, test\_size =

0.3, random\_state = 109 )

clf = svm.SVC( kernel = 'linear' )

clf.fit(X\_train, y\_train)

NewData = [[38,1,2,138,175,0,1,173,0,0,2,4,2], [age, sex, cp, trestbps, chol, fbs,

restecg, thalach, exang, oldpeak, slope, ca, thal]]

result = clf.predict(NewData)[1]

return result

pygmail.py

import smtplib

def sendEmail( mailto, doctor, date, time, name, report ):

gmailaddress = "doctor.chatbot.project@gmail.com"

gmailpassword = "chatbot2020"

sub = "Confirmed: Doctor Appointment Booked"

if (len(report) == 0): report = "Checkup"

msag = "Hey " + name + ",\n\n Your Appointment has been Successfully Booked with

Dr." + doctor + "\n\nDate : " + date + "\nTime : " + time + "\nProblem : " + report +

"\n\nThank you for using Doctor Chatbot."

msg = 'Subject: {}\n\n{}'.format(sub, msag)

sub2 = "Appointment Booked with Doctor "+ doctor + " on "+ date

msag2 = "Patient Email: "+ mailto + "\n\nReport: " + report

msg2 = 'Subject: {}\n\n{}'.format(sub2, msag2)

mailServer = smtplib.SMTP('smtp.gmail.com' , 587)

mailServer.starttls()

mailServer.login(gmailaddress , gmailpassword)

mailServer.sendmail(gmailaddress, mailto , msg)

print("--------------------\nUser Email Sent!\n--------------------")

mailServer.sendmail(gmailaddress, gmailaddress , msg2)

print("\n--------------------\nAdmin Email Sent!\n--------------------")

mailServer.quit()

Return

**Home.html**

<!DOCTYPE html>

<html>

<head>

<title> Welcome to Doctor Chatbot </title>

</head>

<style type="text/css">

body, html {

height: 100%;

}

\* {

box-sizing: border-box;

}

.bg-img {

/\* The image used \*/

background-image: url("{{ url\_for('static', filename='home.jpg' ) }}");

/\* Control the height of the image \*/

min-height: 640px;

/\* Full-width input fields \*/

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

width: 100%;

padding: 15px;

margin: 5px 0 22px 0;

border: none;

background: #f1f1f1;

}

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

background-color: #ddd;

outline: none;

}

/\* Set a style for the submit button \*/

.btn {

background-color: #e60000;

color: white;

padding: 16px 20px;

border: none;

cursor: pointer;

width: 100%;

opacity: 0.9;

}

.btn:hover {

opacity: 1;

}

/\* Modal Content \*/

.modal-content {

background-color: #fefefe;

margin: auto;

padding: 20px;

border: 1px solid #888;

width: 80%;

}

/\* The Close Button \*/

.close {

color: #aaaaaa;

float: right;

font-size: 28px;

font-weight: bold;

}

</style>

<body>

<!-- Modal -->

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

<div class="modal-content">

<span class="close">&times;</span>

<p>

<h3 align="center"> Doctor Chatbot Privacy Policy </h3>

This Privacy Policy applies to the practices of Doctor Chatbot concerning information

that is obtained by and through your use of our website, web application or mobile

application and the services provided through the Application. Doctor Chatbot is

committed to respecting your privacy and recognizing your need for appropriate

protection and management of personally identifiable information (“PII”) and the

information you share. The purpose of this Privacy Policy is to explain the types of

information This Project obtains about users of our Application and/or Services, how

the information is obtained, how it is used, how it is disclosed, how you can get

access to this information, and the choices you have regarding our use of, and your

ability to review and correct, the information. By using the Application and/or the

Services, you are accepting and consenting to the use of your information as described

in this Policy and you agree to be bound by this Policy. Please review this Policy

carefully. <br>

This Privacy Policy applies only to this Application and Services and becomes

effective as soon as you use the Application. Any other services rendered by any

independent provider are subject to that particular provider’s own privacy standards

or policies.

</p>

<p>

<h3 align="center"> Terms and Conditions </h3>

The only PII that the Application obtains is information that you voluntarily provide

or authorize. When you use our Services, we collect PII such as your name, age,

gender, email address etc. We also may collect PII or Protected Health Information

(“PHI”) from you when you make a request to be connected with an independent medical

practice, which may provide an on-call licensed physician to provide non-emergency

general adult and pediatric healthcare services at your location; create a profile; or

comment on articles or complete surveys. <br>

The information that we collect varies depending upon how you use our Services. When

you choose to Enroll; you will need to provide us with health-related information that

a provider will need to determine whether he/she is willing to provide services to

you. That information will be associated with your profile. Any PHI that you provide

through the Application briefly describing your symptoms (or those of any minor for

whom you request Services) will be sent to our employed medical provider or an

independent alternate provider to be used for treatment and processing your payment

for the Visit, and other services. All providers are bound to protect all PHI of

patients in accordance with HIPAA rules, as outlined in our Notice of Privacy

Practices.

</p>

</div>

</div>

<!-- Form -->

<div class="bg-img">

<form action="{{ url\_for('Home') }}" method="post" class="container">

<h1>Doctor Chatbot</h1>

<label for="email"><b>Please Enter Email to Continue</b></label>

<input type="text" placeholder="Enter Valid Email" name="uemail" required>

<label>

<input type="checkbox" name="remember" style="margin-bottom:15px" required> I

Agree <a href="#" id="myBtn"> Terms & Conditions </a>

</label>

<button type="submit" class="btn">Chat with Doctor</button>

</form>

</div>

<!-- Modal Script -->

<script>

var modal = document.getElementById("myModal");

var btn = document.getElementById("myBtn");

var span = document.getElementsByClassName("close")[0];

btn.onclick = function() {

modal.style.display = "block";

}

span.onclick = function() {

modal.style.display = "none";

}

window.onclick = function(event) {

if (event.target == modal) {

modal.style.display = "none";

}

}

</script>

</body>

</html>

Main.html

<!DOCTYPE html>

<html>

<head>

<title>Doctor Chatbot</title>

</head>

<frameset cols="35% ,65%">

<frame src="http://127.0.0.1:5000/Chatbot" frameborder="0">

<frame src="http://127.0.0.1:5000/Form" frameborder="1">

</frameset>

</html>

**Chatbot.html**

<!DOCTYPE html>

<html>

<head>

<style>

#snackbar {

visibility: hidden;

min-width: 250px;

margin-left: -125px;

background-color: #333;

color: #fff;

text-align: center;

border-radius: 2px;

padding: 16px;

position: fixed;

z-index: 1;

left: 50%;

bottom: 30px;

font-size: 17px;

}

#snackbar.show {

visibility: visible;

-webkit-animation: fadein 0.5s, fadeout 0.5s 2.5s;

animation: fadein 0.5s, fadeout 0.5s 2.5s;

}

@-webkit-keyframes fadein {

from {bottom: 0; opacity: 0;}

to {bottom: 30px; opacity: 1;}

}

@keyframes fadein {

from {bottom: 0; opacity: 0;}

to {bottom: 30px; opacity: 1;}

}

@-webkit-keyframes fadeout {

from {bottom: 30px; opacity: 1;}

to {bottom: 0; opacity: 0;}

}

@keyframes fadeout {

from {bottom: 30px; opacity: 1;}

to {bottom: 0; opacity: 0;}

}

</style>

</head>

<body onload="myFunction()">

<iframe

allow="microphone;"

width="450"

height="630"

src="https://console.dialogflow.com/api-client/demo/embedded/0eee2d04-83be-47bc-

9580-4a5e517e0990">

</iframe>

<div id="snackbar">Say Hi to Doctor Chatbot</div>

<script>

function myFunction() {

var x = document.getElementById("snackbar");

x.className = "show";

setTimeout(function(){ x.className = x.className.replace("show", ""); }, 3000);

}

</script>

</body>

</html>

**Form.html**

<!DOCTYPE html>

<html>

<head>

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

<style>

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

/\* Full-width input fields \*/

input[type=text], select {

width: 100%;

padding: 12px;

border: 1px solid #ccc;

border-radius: 4px;

resize: vertical;

}

label {

padding: 12px 12px 12px 0;

display: inline-block;

}

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

button {

background-color: #ff0000;

color: white;

padding: 14px 20px;

margin: 8px 0;

border: none;

cursor: pointer;

width: 58%;

}

button:hover {

opacity: 0.8;

}

.container {

padding: 16px;

}

span.psw {

float: right;

padding-top: 16px;

}

/\* 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: 60%; /\* 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;

}

.cancelbtn {

width: 100%;

}

}

/\* ToolTip \*/

.tooltip {

position: relative;

display: inline-block;

border-bottom: 1px dotted black;

}

</style>

</head>

<body style="background-image: url('{{ url\_for('static', filename='h.png' ) }}');

background-repeat: no-repeat; background-attachment: fixed;

background-size: cover;">

<div style="text-align: center;">

<h2 style="padding-top: 250px; padding-right: 4%">Click Here to get Form</h2>

<button onclick="document.getElementById('id01').style.display='block'"

style="width:auto; position: absolute; top: 45%; right: 48%; border: 2px solid

#000000; box-shadow: 0 12px 16px 0 rgba(0,0,0,0.24), 0 17px 50px 0

rgba(0,0,0,0.19);">Report Form</button>

</div>

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

<form class="modal-content animate" name="passdata" action="." method="post">

<div class="imgcontainer">

<h2> Report </h2>

<span onclick="document.getElementById('id01').style.display='none'"

class="close" title="Close Modal">&times;</span>

</div>

<div class="container">

<!--

<label for="age">Age</label>

<input type="text" style="width: 96%" id="Age" name="age" placeholder="Enter

Your Age.." required> <br>

<label for="gender">Gender</label>

<select id="gender" name="sex">

<option value="" disabled selected>Select Your Gender ..</option>

<option value="Male">Male</option>

<option value="Female">Female</option>

</select> <br>

-->

<label for="cpt">Chest Pain Types</label>

<div class="tooltip">(info)

<span class="tooltiptext">Select the type of chest-pain experienced by the

individual: caused when your heart muscle doesn’t get enough oxygen-rich blood.</span>

</div>

<select id="gender" name="cp" required>

<option value="" disabled selected>Select Type Of Chest Pain ..</option>

<option value="Typical Angina">Typical Angina</option>

<option value="Atypical Angina">Atypical Angina</option>

<option value="Non-anginal Pain">Non-anginal Pain</option>

<option value="Asymptomatic">Asymptomatic</option>

</select> <br>

<label for="rbp">Resting Blood Pressure</label>

<div class="tooltip">(info)

<span class="tooltiptext">Blood pressure value of an individual in mmHg

(unit): may be high due to obesity, high cholesterol or diabetes.</span>

</div>

<input type="text" style="width: 96%" id="Age" name="trestbps"

placeholder="Blood Pressure Value (mmHg).." required> <br>

<label for="sc">Serum Cholestoral</label>

<div class="tooltip">(info)

<span class="tooltiptext">Serum cholesterol in mg/dl (unit): A high level

of low-density lipoprotein (LDL) cholesterol (“bad” cholesterol).</span>

</div>

<input type="text" style="width: 96%" id="CS" name="chol"

placeholder="Cholestoral Value (mg/dl).." required> <br>

<label for="fbs" disabled selected>Fasting Blood Sugar</label>

<div class="tooltip">(info)

<span class="tooltiptext">Compares the fasting blood sugar value of an

individual with 120mg/dl. Not producing enough of a hormone secreted by your pancreas

(insulin).</span>

</div>

<select id="fps" name="fbs" required>

<option value="" disabled selected>Fasting Blood Sugar > 120mg/dl</option>

<option value="Yes">Yes</option>

<option value="No">No</option>

</select> <br>

<label for="recg" disabled selected>Resting ECG</label>

<div class="tooltip">(info)

<span class="tooltiptext">Resting electrocardiographic Results.</span>

</div>

<select id="fps" name="restecg" required>

<option value="" disabled selected>Resting ECG</option>

<option value="Normal">Normal</option>

<option value="Having ST-T Wave Abnormality">Having ST-T Wave

Abnormality</option>

<option value="Left Ventricular Hyperthrophy">Left Ventricular

Hyperthrophy</option>

</select> <br>

<label for="mhra">Max. Heart Rate Achieved</label>

<input type="text" style="width: 96%" id="CS" name="thalach"

placeholder="Max. Heart Rate .." required> <br>

<label for="eia">Exercise Induced Angina </label>

<select id="fps" name="exang" required>

<option value="" disabled selected>Exercise Induced Angina </option>

<option value="Yes">Yes</option>

<option value="No">No</option>

</select> <br>

<label for="stdie">ST Depression Induced by Exercise</label>

<input type="text" style="width: 96%" id="Age" name="oldpeak"

placeholder="Enter Value.." required> <br>

<label for="gender">Peak Exercise ST Segment</label>

<div class="tooltip">(info)

<span class="tooltiptext">A treadmill ECG stress test is considered

abnormal when there is a horizontal or down-sloping ST-segment depression ≥ 1 mm at

60–80 ms after the J point.</span>

</div>

<select id="fps" name="slope" required>

<option value="" disabled selected>Peak exercise ST Segment</option>

<option value="Upsloping">Upsloping</option>

<option value="Flat">Flat</option>

<option value="Downsloping">Downsloping</option>

</select> <br>

<label for="mvcf">Number of Major Vessels Colored by Flourosopy </label>

<input type="text" style="width: 96%" id="CS" name="ca" placeholder="Enter

value .." required> <br>

<label for="gender">thalassemia</label>

<select id="fps" name="thal" required>

<option value="" disabled selected>Thal </option>

<option value="Normal">Normal</option>

<option value="Fixed Defect">Fixed Defect</option>

<option value="Reversible Defect">Reversible Defect</option>

</select> <br>

<button type="submit" style="width: 100%">Submit Report</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>

Pass.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>pass</title>

</head>

<body>

<div style="text-align: center;">

<h1 style="padding-top: 250px;"> Thank You! Report Successfully Submitted. </h1>

<h4> Continue Chatting with Doctor Bot... </h4>

</div>

</body>

</html>

**CONCLUSION**

The main approach of this system is to detect & predict the presence of a heart disease with the best possible accuracy and speed which are considered as important characteristics of this project. From the surveys conducted on various predicting algorithms like KNN, ANN, SVM, I Bayes, Decision tree etc., various tests were conducted on the same dataset to check for accuracy of each algorithm. From the survey point of view, it was found that Support Vector Machine algorithm on a Heart Disease Dataset gives the best possible accuracy in competition with I Bayes,

Decision tree, KNN and more. An SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. SVM is a discriminative classifier formally defined by a separating hyperplane.

Using the Technology above, the objective is to build a System which will be able to get the patient reports, analyse it and concludes whether he/she is suffering from heart diseases or not. The process will be done in a conversational manner using Dialogflow Platform. It also helps to get early diagnosis. The system will also be trained to provide an appointment booking facility with related doctors. This healthcare bot system will help hospitals to provide healthcare support online 24 x 7. The Action Plan to implement the project is given in the table below. Since the project is built to mostly execute on desktop Web Browsers, it is recommended to supply input in text format and not voice so that wrong inputs (like in case of asked symptoms) can be avoided which may lead to error. The Bot also responds with a Disclaimer: "Please note, this is not a diagnosis. Always visit a doctor if you are in doubt, or if your symptoms get worse or don't improve.

If your situation is serious, always call the emergency services". This ensures that the user doesn't completely reply on the bots' recommendations, which in worst case can lead to death. The Target Audience of this project are people mostly from the age group 50 & above (elderly people). However, specific doctors can also use this as a tool. Expected benefits to the target groups/population are:

* They can get immediate access to the facility (heart health) free to access from anywhere.
* The best doctors are available to choose for treatment according to the patient’s heart health and according to the consultancy charges.
* Patients need not wait in a queue to consult doctor to know their heart health, saving time of both.
* Earlier diagnosis may be affordable than later diagnosis. The proposed system helps in early
* diagnosis. Similarly at the later stage costly medicines are required than at the earlier stage.
* It may reduce the travelling cost and time as the application works irrespective of geographical distance (24x7 online help).
* Doctors can get more clients online.

**REFERENCES**

1. Kyo-Joong Oh, DongKun Lee, ByungSoo Ko, Ho-Jin Choi; “A Chatbot for Psychiatric Counseling in Mental Healthcare Service Based on Emotional Dialogue Analysis and Sentence Generation”; 2375-0324/17 $31.00 ©2017 IEEE, 2017 IEEE 18th International Conference on Mobile Data Management.
2. Chin-Yuan Huang, Ming-Chin Yang, Chin-Yu Huang, Yu-Jui Chen, Meng-Lin Wu, Kai-Wen Chen; “A Chatbot-supported Smart Wireless Interactive Healthcare System for Weight Control and Health Promotion”; 978-1-5386-6786-6/18/$31.00 ©2018 IEEE, Proceedings of the 2018 IEEE IEEM.
3. Siddhant Rai, Akshayanand Raut, Akash Savaliya, Dr. Radha Shankarmani; “Darwin: Convolutional Neural Network based Intelligent Health Assistant”; 978-1-5386-0965-1/18/$31.00 ©2018 IEEE, Proceedings of the 2nd International conference on Electronics, Communication and Aerospace Technology (ICECA 2018).
4. Ankita Dewan, Prof. Meghna Sharma; “Prediction of Heart Disease Using a Hybrid Technique in Data Mining Classification”; 978-9-3805-441 6-8/15/$31.00 ©2015 IEEE, 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom).
5. Amiya Kumar Tripathy, Rebeck Carvalho, Ajit Puthenputhussery, Nikita Chhabhaiya, Bijoy Anthony; “MediAssistEdge – Simplifying diagnosis procedure & Improving patient doctor Connectivity”; 978-1-4799-8187-8/15/$31.00 ©2015 IEEE, 2015 International Conference on Technologies for Sustainable Development (ICTSD-2015), Feb. 04–06, 2015, Mumbai, India.