A PROJECT REPORT ON

"MBBS – Med Bay Bot System"

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE

BACHELOR OF ENGINEERING

In

INFORMATION TECHNOLOGY

Of

SAVITRIBAI PHULE PUNE UNIVERSITY

By

AUDARYA UTTARWAR	**B150238635**
RAJAT GUPTA	**B150238557**
PRANJAL BHAGADE	**B150238521**
KALYANI MAGDUM	**B150238588**

Under the guidance of
Dr. Bharati P. Vasgi

DEPARTMENT OF INFORMATION TECHNOLOGY
SINHGAD COLLEGE OF ENGINEERING,Pune-41
Accredited by NACC

2020-21

Sinhgad Technical Education Society,

Sinhgad College of Engineering ,
Pune-41 Department of Computer
Engineering

Sinhgad Institutes

Date:

CERTIFICATE

This is to certify that the project report entitled

“MBBS - Med Bay Bot System”

Submitted by

AUDARYA UTTARWAR	**B150238635**
RAJAT GUPTA	**B150238557**
PRANJAL BHAGADE	**B150238521**
KALYANI MAGDUM	**B150238588**

is a bonafide work carried out by him/her under the supervision of Dr. Bharati P. Vasgi and it is approved for the partial fulfillment of the requirements of Savitribai Phule Pune University , Pune for the award of the degree of Bachelor of Engineering (Information Technology) during the year 2020-21.

Dr. Bharati P. Vasgi
Guide
Department of Information Technology

Prof.G. R. Pathak
Head
Department of Information Technology

Dr. S. D. Lokhande
Principal
Sinhgad College of Engineering

Acknowledgment

We feel great pleasure in expressing our deepest sense of gratitude and sincere thanks to our guide **Dr. Bharati P. Vasgi** for her valuable guidance during the Project work, without which it would have been a very difficult task. I have no words to express my sincere thanks for valuable guidance, extreme assistance and cooperation extended to all the **Staff Members** of our Department.

This acknowledgement would be incomplete without expressing our special thanks to **Prof.G. R. Pathak Head of the Department (Information Technology)** for his support during the work We would also like to extend our heartfelt gratitude to our **Principal, Dr. S. D. Lokhande** who provided a lot of valuable support, mostly being behind the veils of college bureaucracy.

Last but not least we would like to thank all the Teaching, Non- Teaching staff members of our Department, our parents and colleagues who helped us directly or indirectly for completing this Project successfully.

ABSTRACT

Detection of Diseases is one of the preliminary steps in the treatment of a disease whether it is common cold or cancer. Detection of Diseases is solely based on symptoms which may be common for many diseases. Early detection of a disease is often regarded as the job half done while treating the disease. Detection of disease may seem like a simple and straightforward procedure but it is rather a very complex game. The problem of having diseases with common symptoms lead to a confusion for both the doctors and patients to identify it and be sure that the prescribed treatment is the right one. Also, detection of diseases requires the patient to go through a series of tests which may be time consuming as well as not a cost effective method for a patient.The proposed system is to create an alternative to this conventional method of visiting a hospital and making an appointment with a doctor to get diagnosis. This research intends to apply the concepts of natural language processing and machine learning to create a chatbot application. People can interact with the chatbot just like they do with another human and through a series of queries, the chatbot will identify the symptoms of the user and thereby, predict the disease and recommend treatment. Executing this proposed framework can help people avoid the time-consuming method of visiting hospitals by using this free of cost application, wherever they are.The system can be used by a person with restricted medical knowledge as well with ease and can come handy in early disease detection and diagnosis. It can also benefit users that are reluctant to visit hospitals on the onset of minor symptoms. This will provide them with a basic idea of the severity of the disease.

List of Figures

List of Tables

Abbreviation

ML	Machine Learning
GUI	Graphical User Interface
SDLC	Software Development Life Cycle
SRS	Software Requirement Specification
LR	Logistic Regression
KNN	K-Nearest Neighbour
MBBS	Med-Bay Bot System

TABLE OF CONTENTS

Chapter 1
Introduction

1.1 Background and Basics

Most Artificial Intelligence's main goal is to make Human interaction with computers and other electronic devices much easier and practical . Nowadays Personal assistants who can carry out tasks required for daily needs with just a meaningful phrase is a fast growing area . Many companies have used the dialogue systems technology to establish various kinds of Virtual Personal Assistants(VPAs) based on their applications and areas, such as Microsoft's Cortana, Apple's Siri, Amazon Alexa, Google Assistant, etc.Virtual Assistants are known for being great at many things and can make a positive contribution to the success of your SME. They can also assist with your organization skills and productivity by freeing up your time for you to focus on other important activities. The idea of an intelligent machine engaging in human interactions was first theorized by Alan Turing in 1950.Shortly after, automated computer programs, referred to as "bots", were created to simulate human conversation. For example, ELIZA[1] in 1966 matched user prompts to scripted responses, and Artificial Linguistic Internet Computer Entity (ALICE) [2] in 1995 introduced natural language processing (NLP) to interpret user input. Chabot now exists in various messaging platforms, such as Facebook Messenger, Skype, and Kik, largely for customer service purposes.There are many disease prediction systems available such as heart disease prediction [3], neurological disorders prediction [4], and skin disease prediction [5]. But a universal prediction system for diseases based on symptoms is rarely in practice. It is very helpful for doctors or medical experts to diagnose diseases at an early stage based on symptoms. When a query is given, probable diseases are suggested to the user based on the highest probability and scores.With the use of the internet and all resources available to the user, proper diseases are used, and based on that proper medication is done which is very beneficial to all human beings. It is very helpful for doctors and patients to know better about the disease without any medical tests or anything else.The main objective is to develop a novel architecture that could accept and handle such type of user queries by employing techniques like query expansion using a thesaurus, synonym matching, and symptom suggestion that will allow disease prediction with greater accuracy based on user input. We have scraped data from the web and generated a dataset which can be used in future research. Query search retrieval and matching are used in such problems to achieve prediction.

1.2 Literature Survey

Sr No	Paper Name	Author	Publication	Published Year	Outline	Algorithm used	Advantage
1.	Chatbot for Disease Prediction and Treatment Recommendation using Machine Learning [6]	Rohit Binu Mathew, Sandra Varghese, Sera Elsa Joy, Swanthana Susan Alex	Saintgits College of Engineering Kottayam, Kerala	10-Oct-2019	The proposed system is to create an alternative to this conventional method of visiting a hospital and making an appointment with a doctor to get diagnosis.	KNN	It creates a good human-like conversational environment for interaction and talks about their health and it is a great way for the users to regulate the healthy lifestyle.
2.	Designing Disease Prediction Model Using Machine Learning Approach [7]	Dhiraj Dahiwade, Prof. Gajanan Patle, Prof. Ektaa Meshram	Abha Gaikwad Patil College of Engineering, Nagpur	29-Aug-2019	It proposed general disease prediction based on symptoms of the patient.	CNN and KNN	Accuracy of CNN algorithm is more than KNN algorithm.
3.	Contextual Chatbot for Healthcare Purposes (using Deep Learning) [8]	Prathmesh Kandpal, Kapil Jasnani,Ritesh Raut,Siddharth Bhorge	World Conference on Smart Trends in System Security and Sustainability (WorldS4)	1 October 2019	It uses Machine Learning and Artificial Intelligence techniques to store and process the training models which help the chatbot to give better and appropriate	NLP, TF-IDF(Term frequency-inverse data (frequency)	Once it is able to identify the disease it can then provide the user with simple remedies to follow.overcome the weaknesses

					response.		of each modality and exploit the full strengths of combined modes.
4.	The ChatBot Feels You - A Counselling Service Using Emotional Response Generation [9]	R. Augustian Isaac, Abishek Narayanan	International Conference on Big Data and Smart Computing (BigComp)	2017	The major milestone that this project tries to achieve is that it increases the accuracy of the text software.	Hidden Markov Model	provides conversational mental healthcare service based on emotion recognition methods and chat assistant platform which consist of the context sensitive advanced natural language-based technique.
5.	Chatbots meet eHealth:automatizing healthcare [10]	Flora Amato, Stefano Marrone, Gabriele Piantadosi Antonio Picariello, and Carlo Sansone	DIETI - University of Naples Federico II, via Claudio 21, 80125 Napoli (Italy)	8-Aug-2017	The aim of this work is to investigate the effectiveness of novel human-machine interaction paradigms for eHealth applications.. transform the whole healthcare process to become more efficient, less expensive and higher quality	eHealth, Big Data, Deep Learning, Watson, Decision Support System	The e-Health application developed during this study can help to improve work place efficiency, reduce cost of health care delivery.
6.	Chat-Bot Utilization for Health Consultancy [11]	Prof. A. B. Gadewar1, Ammara Shaikh2,	KFUPM	1-June-2014	OntBot uses appropriate mapping technique for transformation of ontologies into	Pattern matching,sequence to	

		Samiksha Bhalerao3 , Bhagyashree Nawsagre			relational databases which drives the chat-bot. Ontologies means the study of nature and existence,determination	seque nce model	

Table 1.2 : Literature Survey

1.3 Project Undertaken

1.3.1 Problem Definition:

To develop a user friendly Chatbot which can be used for disease prediction based on provided symptoms, obtain disease information and any medicinal drug related information.

1.3.2 Scope Statement:

We can achieve the following : -

1. Users can interact with the chatbot in natural language (english) by providing it with symptoms.
2. Based on initial symptoms provided by the user, the chatbot provides more co-occurring symptoms to choose for better accuracy and prediction and once all the symptoms are provided a list of top 5 diseases based on likeness will be displayed to the user.
3. Users can ask for any drug information in natural language which will be processed and then the uses, benefits and side effects of the drug will be provided to the user instantly.
4. Users can also ask for any disease information which will be provided by the chatbot at the same interface.

1.4 Organization of the Project Report

The Report is divided into Four Chapters. Chapter 2, discusses Project Planning and Management that covers System Overview and functional and non-functional requirements. Chapter 3 discusses Analysis and Design. Chapter 4 about Implementation and Coding, Chapter 5 about Testing, Chapter 6 about Results and discussion, Chapter 7 about Conclusion and Chapter 8 about Future work.

Chapter 2
PROJECT PLANNING & MANAGEMENT

2.1 Introduction

This chapter covers the project planning and management details. It also covers System Requirement specifications. SRS is considered as the base for the effort estimations and project scheduling.

2.2 System Requirement Specification

- **System Overview**

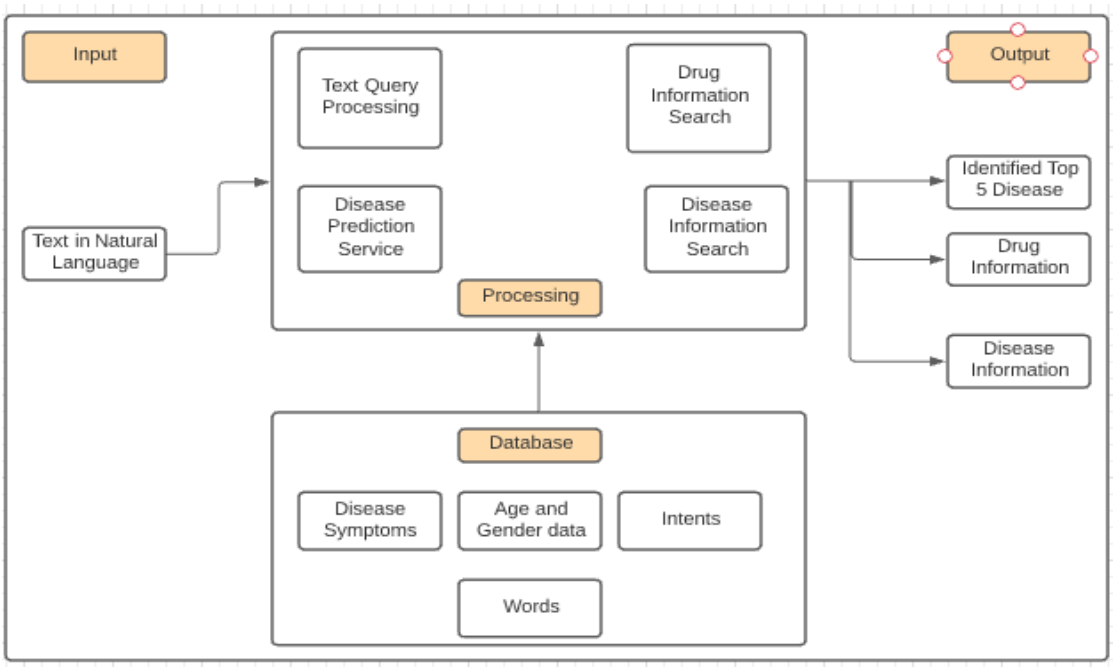


Figure 2.2. Architectural Diagram

Detailed SRS:

The input to the system is text entered by the user. The text is compared with predefined intents on which the chatbot is trained. If there is a specific call for a particular function then that function is called. If the system asks for more data i.e. symptoms then the user must enter required data if any. E.g. If the user wants to know about disease and he/she particularly entered some symptoms and wants to call the disease prediction facility of the chatbot assistant then that disease prediction facility function gets called only when the system is satisfied with the user input. The system does so

by prompting the user to input some more symptoms.

If a user wants to access features like disease information or drug information then the user can ask for it and their respective functions will be called.

2.2.1 Functional Requirements:

1. Performance: The performance of the system will give results instantly. After processing symptoms, the system detects the top 5 most possible diseases.
2. Capacity: Capacity of project depends on the dataset.
3. Availability: Users can interact with the system anytime, no registration or authentication required.
4. Reliability: System is reliable for prediction of the disease in terms of likeliness, obtaining proper drug and disease information .
5. Security: The system is secure because any sensitive information of the user is not stored permanently, the system has the information only as long as the user is interacting with the chatbot .

2.2.2 Non - Functional Requirements:

1. Performance Requirements:
    a. There should be proper internet connection throughout the interaction.
    b. Users should be able to define at least some of the symptoms to the chatbot.
2. Safety Requirements:
    a. Users input data should be transferred to the server without any loss.
    b. The final task should be analyzed on the basis of all respective input
3. Software Quality Attributes:
    a. Usability: The system should be user friendly and self-explanatory. Proposed system is Flexible, Robust, and easily usable. The proposed system will be helpful to users in order to know possible diseases earlier so that they can take treatment accordingly.
    b. Accuracy: The system will give accurate results related to the query.
    c. Openness: The system should be extensible to guarantee that it is useful for community systems.
    d. Reliability: The system should have accurate results and fast responses to users.

2.2.3 External Interface Requirement:

1. User Interfaces
    a. Chatbot interface to accept text in natural language and process it.
    b. Prediction system which decides the possible diseases based on symptoms provided to the chatbot interface.
    c. Result display in the form of chat in natural language.
2. Software Interfaces
    a. NLTK library for processing user entered natural text.
    b. Beautiful Soup to scrape disease and drug data from the internet.
    c. Keras for training prediction model.

2.2.4 Other Requirements:

Software Requirements

1. Operating System : Windows 7 or above, Ubuntu 16.04 or above.
2. Language : Python
3. Libraries:Beautiful Soup 4, Keras, Tensorflow, NLTK, Sklearn
4. UI : Angular
5. Database : CSV files

Hardware Requirements

1. Processor - Dual core/Intel i3 or above
2. RAM - 2 GB (Min)
3. Key Board - Standard Keyboard
4. Mouse - Two or Three Button Mouse
5. Monitor /LCD - SVGA/LED
6. Internet connection

2.3. Project Process Modeling

A software process model is a simplified representation of a software process. This software would use an agile process model.

Agile Model:

1. Agile method proposes incremental and iterative approach to software design.
2. The agile process is broken into individual models that designers work on as this application will also be broken into many parts like data scraping, chatbot model, prediction model, UI development, creating APIs and frontend backend integration.
3. Error can be fixed in the middle of the project.
4. It requires close communication with developers and together analyzing requirements and planning.

Chapter 3
ANALYSIS & DESIGN

3.1 Introduction

This chapter covers the analysis and design of the considered system.

3.2. Architecture Diagram

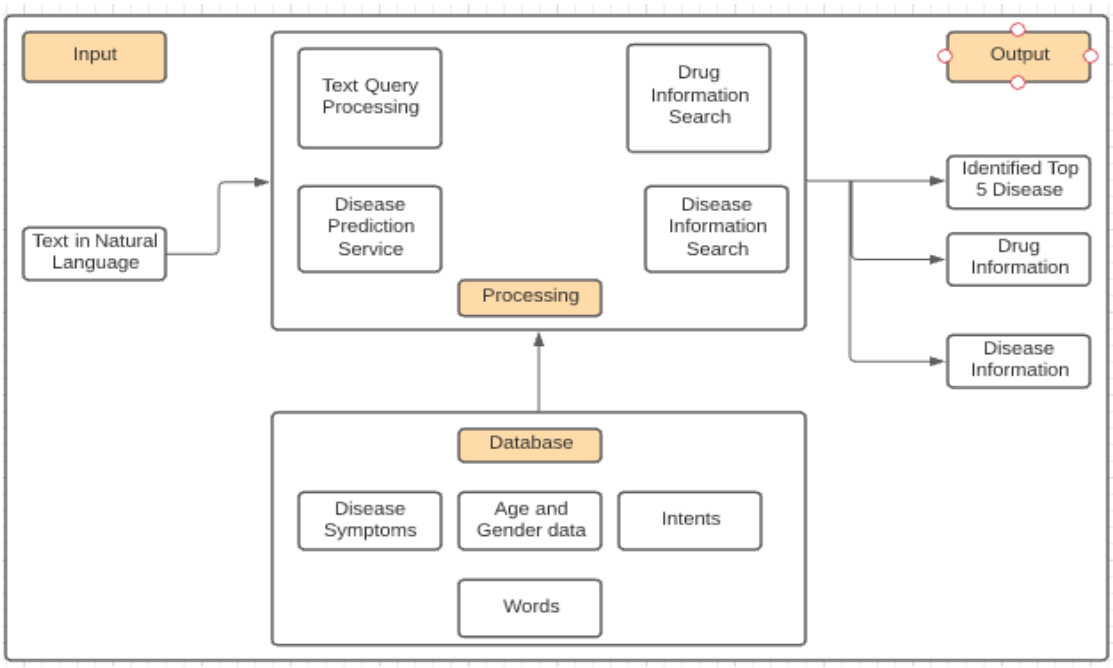


Fig.3.2 Architecture Diagram

The architecture diagram gives the abstract view of the system which involves the main components and entities of the system.

The input to the system is the text entered by the user in natural language. The text goes to the text processing module, where it's intent is identified and an appropriate response from the list of responses of the particular intent is provided to the user in the form of chat conversation.

If the intent is recognised as a disease prediction intent then symptoms from the text are extracted, stored in a list and passed to the machine learning model for prediction. Now, more symptoms are required for the prediction for accurate results. So, the bot prompts the user with a list of symptoms

repetitively which are relatable to the initial symptoms entered.

If the intent is recognised as disease information intent, the disease is searched in the vast disease information dataset and the appropriate information is displayed to the user.

If the intent is recognised as drug information intent, the drug information is scraped from the internet and appropriate information is displayed to the user.

3.3 UML Diagram

3.3.1 Use Case Diagram

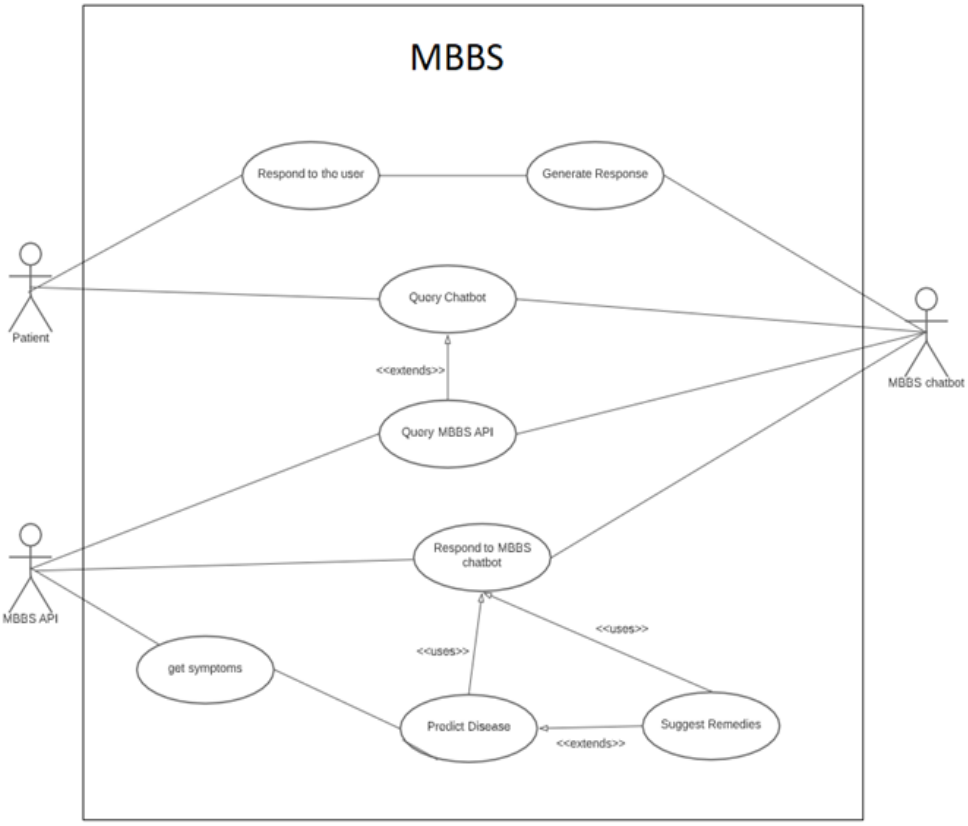


Fig 3.3.1 : Use Case Diagram

Use-Case diagram represents the functional requirements of the system. Used to get an outside view of a system.The most important aspect is to capture the dynamic behavior.

- Users can interact with the chatbot in natural language (english) by providing it with symptoms.

- Users have to first provide basic information like age and gender which will be used to narrow down the search scope of the disease.
- User gives Symptoms as input, based on initial symptoms provided by the user, the chatbot provides more co-occurring symptoms to choose .
- Once all the symptoms are provided a list of top 5 diseases based on likeness will be displayed to the user.

3.3.2 Activity Diagram

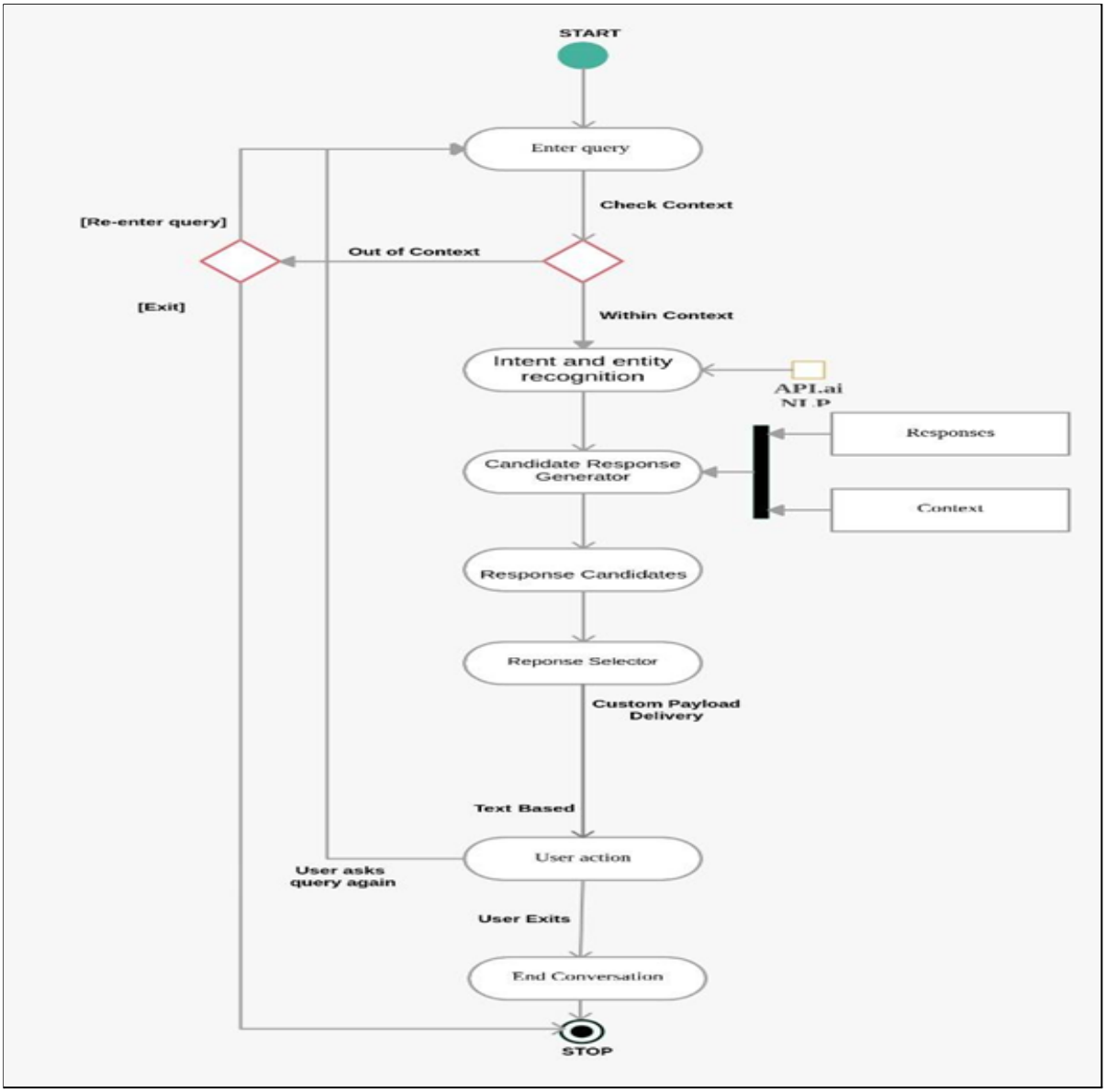


Fig 3.3.2 : Activity Diagram

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single

The purpose of an activity diagram can be described as :

- Draw the activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and concurrent flow of the system.

Initially, the system is in idle mode. As it receives any text input from the user it begins execution. The received text is identified by checking the intents and decision is made for the task to be performed. After the question is being answered or the task is being performed, the system waits for another command. This continues until the user exits the interface.

3.3.3 Class Diagram

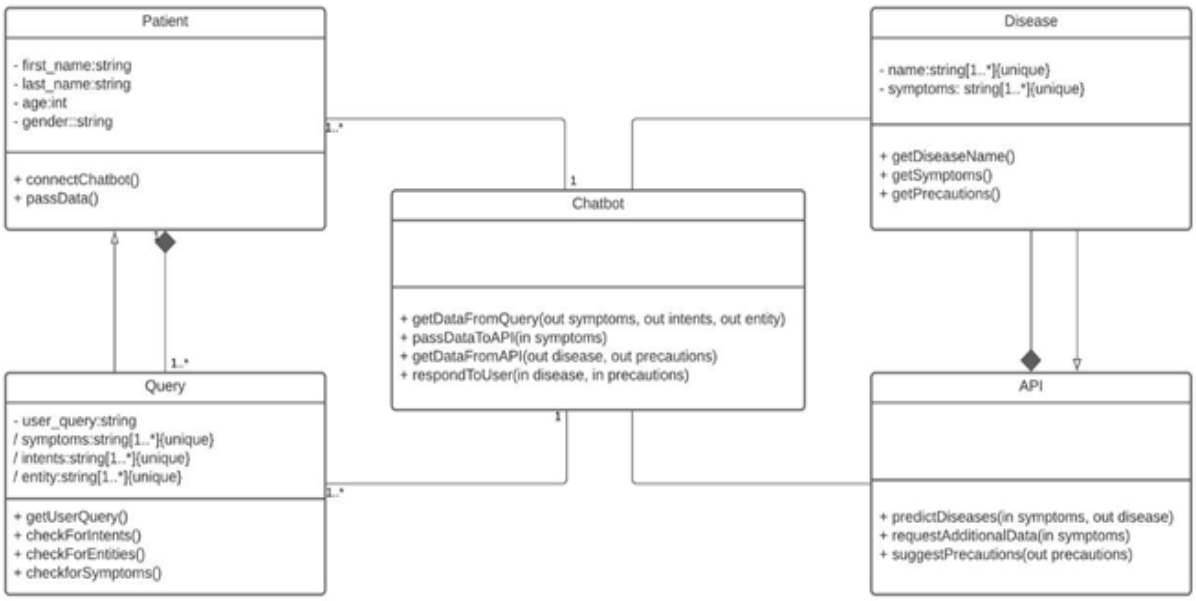


Fig 3.3.3 : Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

The purpose of the class diagram can be summarized as :

- Analysis and design of the static view of an application.
- Describe responsibilities of a system.
- Base for component and deployment diagrams.

3.3.4 Sequence Diagram

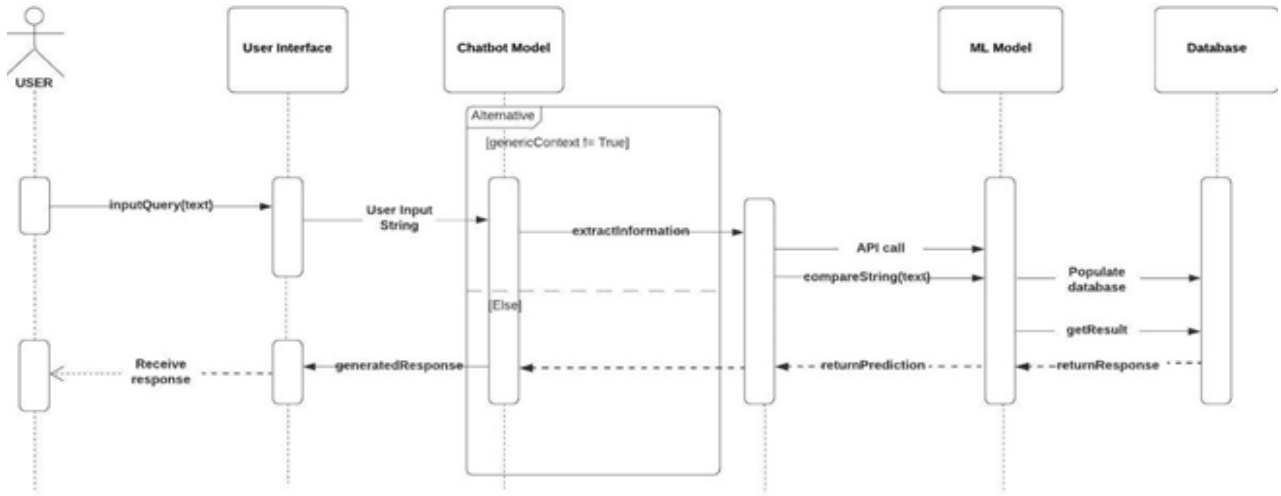


Fig 3.3.4 : Sequence Diagram

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. This interaction is a part of dynamic behavior of the system.Sequence diagram emphasizes on time sequence of messages and collaboration diagram emphasizes on the structural organization of the objects that send and receive messages.

The purpose of interaction diagram is −

- To capture the dynamic behaviour of a system.
- To describe the message flow in the system.
- To describe the structural organization of the objects.
- To describe the interaction among objects.

The above sequence diagram shows how an answer asked by the user is being fetched from the internet. The input is interpreted and sent to the Web scraper. The web scraper searches and finds the answer.The user sends commands to chatbot. The command is passed to the interpreter. It identifies what the user has asked and directs it to the task executor If the task is missing some info, the chatbot asks the user back about it. The received information is sent back to task and it is accomplished.

After execution feedback is sent back to the user.

3.3.5 Data Flow Diagram

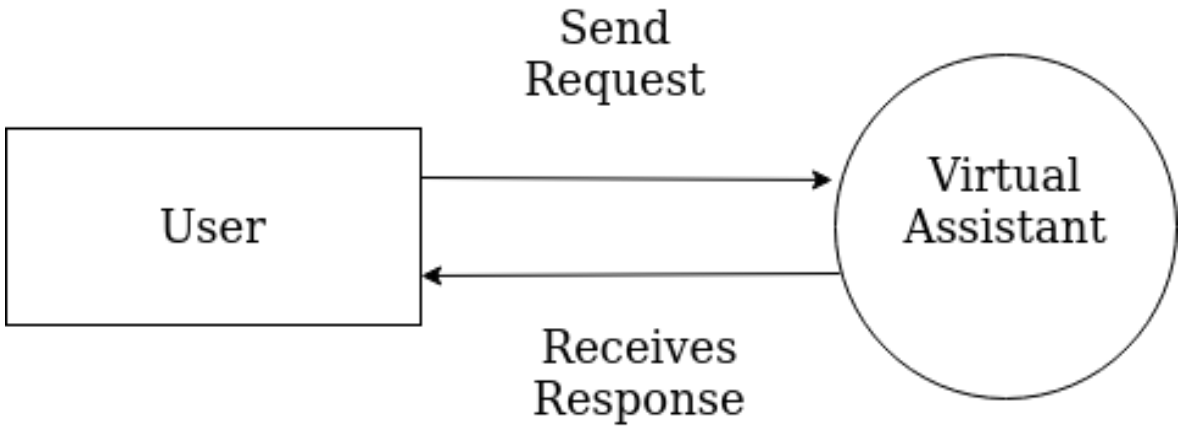


Fig 3.3.5 : DFD level-0 Diagram

Data flow diagram is a graphical representation of flow of data in an information system. It is capable of depicting incoming data flow, outgoing data flow and stored data. The DFD does not mention anything about how data flows through the system.

Level 0 - Highest abstraction level DFD is known as Level 0 DFD, which depicts the entire information system as one diagram concealing all the underlying details. Level 0 DFDs are also known as context level DFDs

Chapter 4
DATASET COLLECTION AND CLEANING

4.1 Introduction

This chapter involves one of the most important aspects of the project i.e. the dataset collection and cleaning process. For any machine learning model to give it's best accuracy, the dataset on which it is trained decides the quality of the model.

4.2 Importance

Data collection is the process of gathering and measuring information from countless different sources. In order to use the data we collect to develop practical artificial intelligence (AI) and machine learning solutions, it must be collected and stored in a way that makes sense for the business problem at hand.

Collecting data allows us to capture a record of past events so that we can use data analysis to find recurring patterns. From those patterns, we can build predictive models using machine learning algorithms that look for trends.

Predictive models are only as good as the data from which they are built, so good data collection practices are crucial to developing high-performing models. The data need to be error-free (garbage in, garbage out) and contain relevant information for the task at hand. For example, a loan default model would not benefit from tiger population sizes but could benefit from gas prices over time.

4.3 Collection Process

For the disease prediction model to work, we need a few things in the dataset:

- List of diseases
- List of symptoms for each disease
- Age range in which the disease can affect

- Gender for which the disease is prevalent

First step is to get a list of available diseases. For this we referred to a website called NHS Inform [12]. We scraped the website using BeautifulSoup4 library and collected around **315** disease names. Then we manually searched the internet for more diseases and collected around **234** more diseases. So, finally after merging and removing redundant diseases we are left with around **506** total diseases.

Next step is to collect symptoms for each of the 506 diseases. For this we wrote a script that will do a google search for that disease and scrape the symptoms from the google infobox.
The symptoms are stored in a list corresponding to the disease. So the dataset is in json format where the key is the disease name and the value is the list of symptoms for that disease.

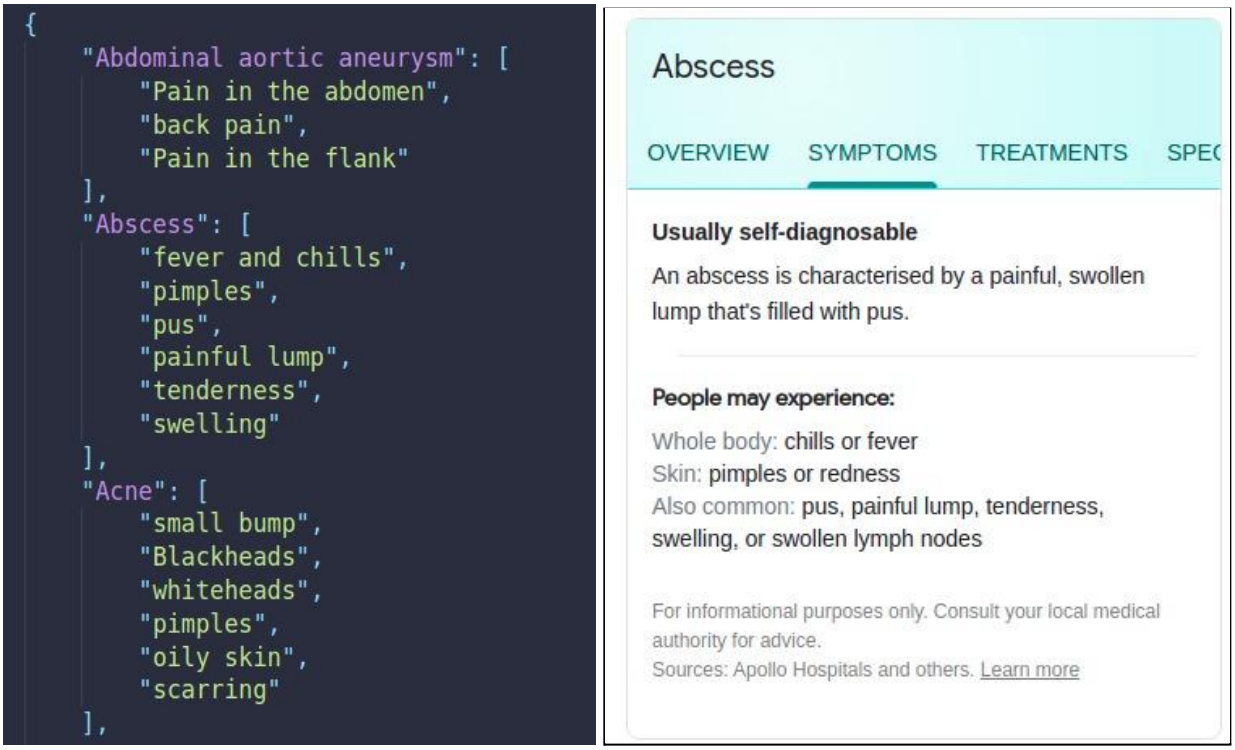
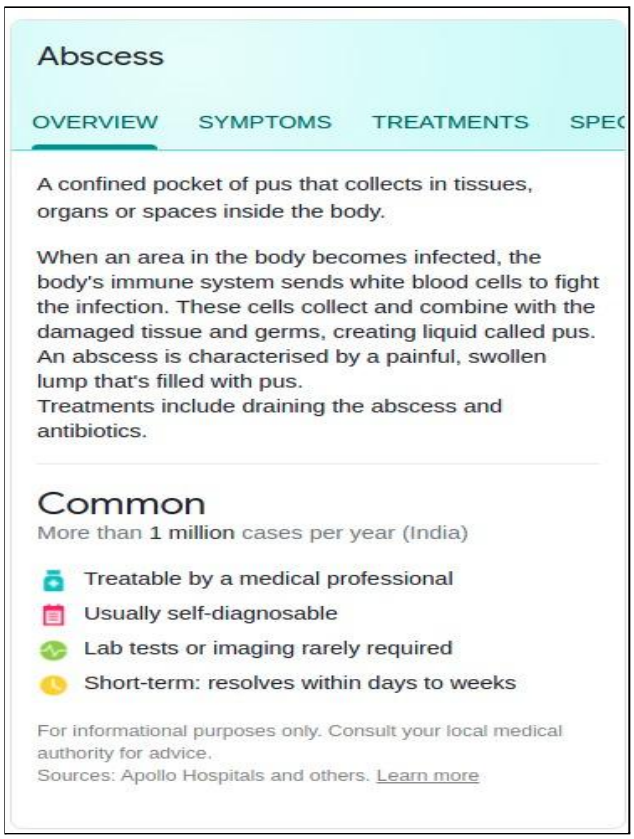
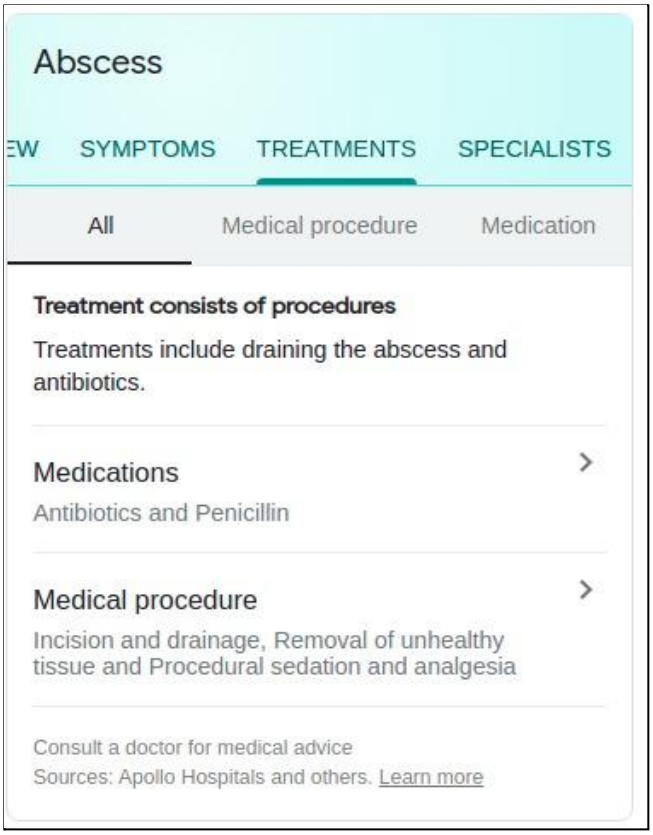


Figure 4.3.1 Disease Symptoms data collection

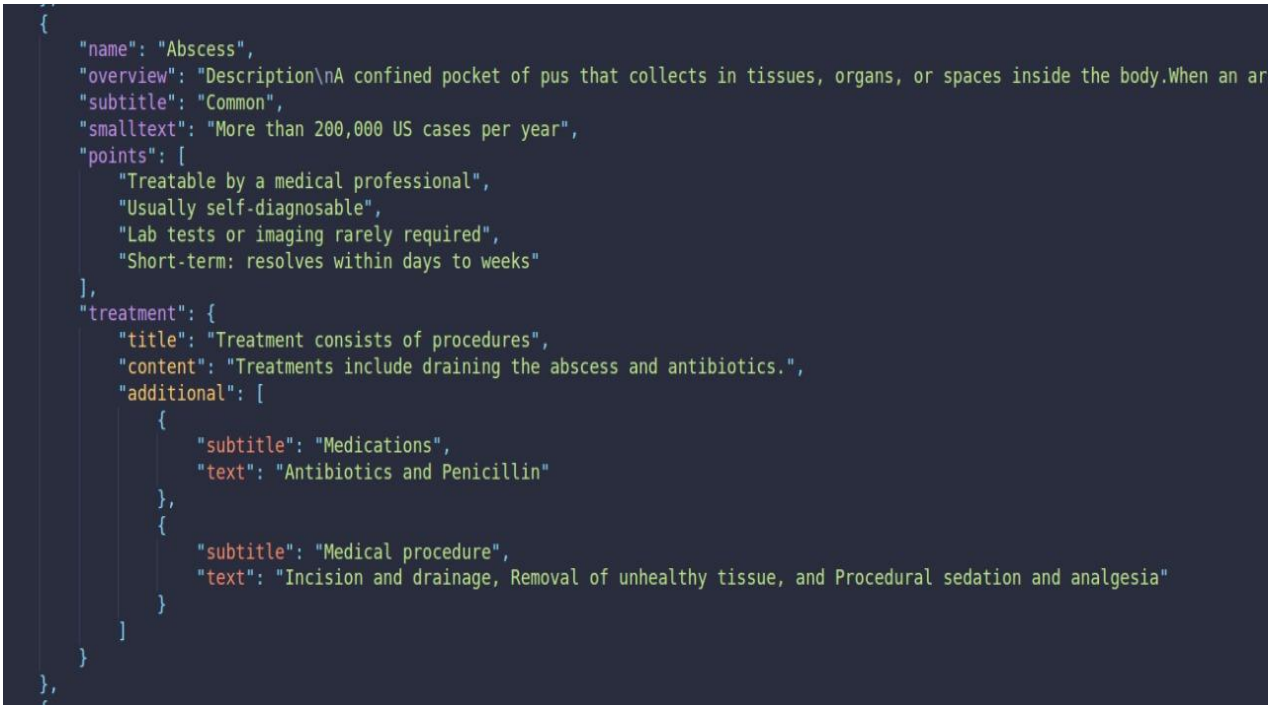
We also wrote a separate script for scraping the disease information, treatment from the google infobox and stored it in json format.



(a) Disease Info in Google info box



(b) Disease Treatment in Google Info box



```
{
    "name": "Abscess",
    "overview": "Description\nA confined pocket of pus that collects in tissues, organs, or spaces inside the body.When an ar
    "subtitle": "Common",
    "smalltext": "More than 200,000 US cases per year",
    "points": [
        "Treatable by a medical professional",
        "Usually self-diagnosable",
        "Lab tests or imaging rarely required",
        "Short-term: resolves within days to weeks"
    ],
    "treatment": {
        "title": "Treatment consists of procedures",
        "content": "Treatments include draining the abscess and antibiotics.",
        "additional": [
            {
                "subtitle": "Medications",
                "text": "Antibiotics and Penicillin"
            },
            {
                "subtitle": "Medical procedure",
                "text": "Incision and drainage, Removal of unhealthy tissue, and Procedural sedation and analgesia"
            }
        ]
    }
},
```

(c) Disease Info in JSON data format

Figure 4.3.2 (a,b,c) Disease Info and Treatments data collection

We also manually created age and gender dataset for each disease.

```
{
    "Acute cholecystitis": {
        "age_group": [
            "65-150"
        ],
        "gender": [
            "MALE",
            "FEMALE",
            "PRIORITYM"
        ]
    },
    "Acne": {
        "age_group": [
            "10-18",
            "18-45"
        ],
        "gender": [
            "MALE",
            "FEMALE",
            "PRIORITYM",
            "PRIORITYF"
        ]
    },
```

Figure 4.3.3. Age and Gender data in JSON data format

Chapter 5
ALGORITHM AND METHODOLOGY

5.1 Introduction

This chapter covers all the ML algorithms used in the project and the methodology of the project.

5.2 Logistic Regression

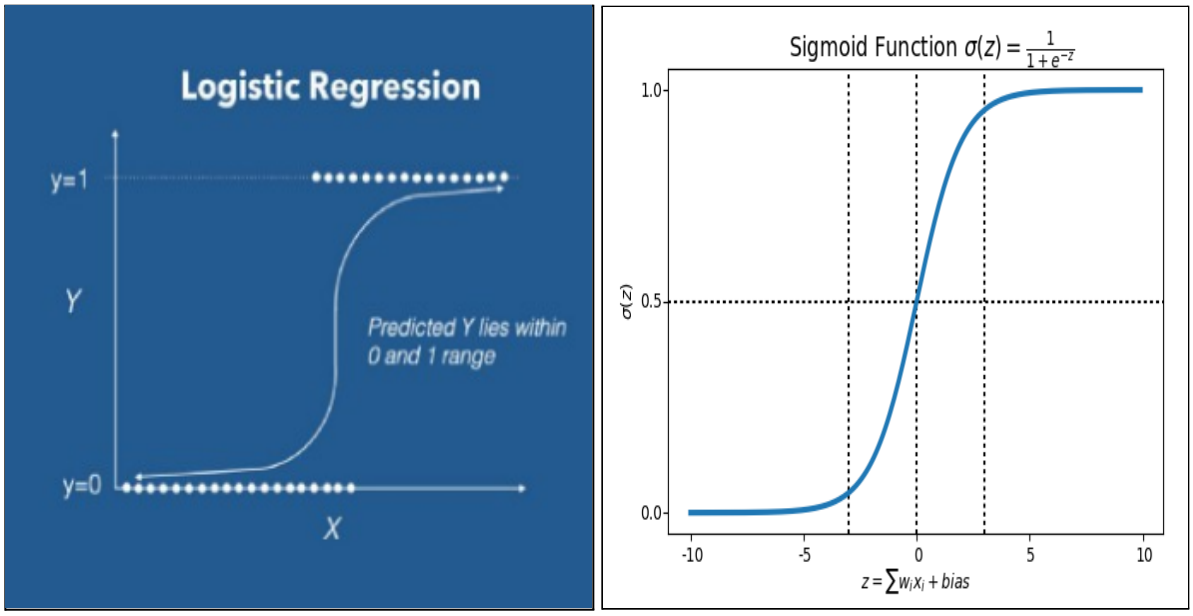


Figure 5.2 Logistic Regression Graphs

Logistic Regression is a Machine Learning algorithm which is used for classification problems, it is a predictive analysis algorithm and based on the concept of probability. We can call a Logistic Regression a Linear Regression model but the Logistic Regression uses a more complex cost function, this cost function can be defined as the Sigmoid Function or also known as the logistic function instead of a linear function.

The hypothesis of logistic regression tends to limit the cost function between 0 and 1. Therefore linear functions fail to represent it as it can have a value greater than 1 or less than 0 which is not possible as per the hypothesis of logistic regression.

5.2.1 Sigmoid Function

In order to map predicted values to probabilities, we use the Sigmoid function. The function maps any real value into another value between 0 and 1. In machine learning, we use sigmoid to map predictions to probabilities.

$$f(x) = \frac{1}{1+e^{-(x)}}$$

Equation 5.2.1 (a) Sigmoid Function

When using linear regression we use formula of the hypothesis :

$h\Theta(x) = \beta_0 + \beta_1 X$

For logistic regression we modify it a little bit :

$\sigma(Z) = \sigma(\beta_0 + \beta_1 X)$

Now our hypothesis will give values between 0 and 1.

$$h\theta(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

Equation 5.2.1 (b) Sigmoid Function

5.3 Neural Networks

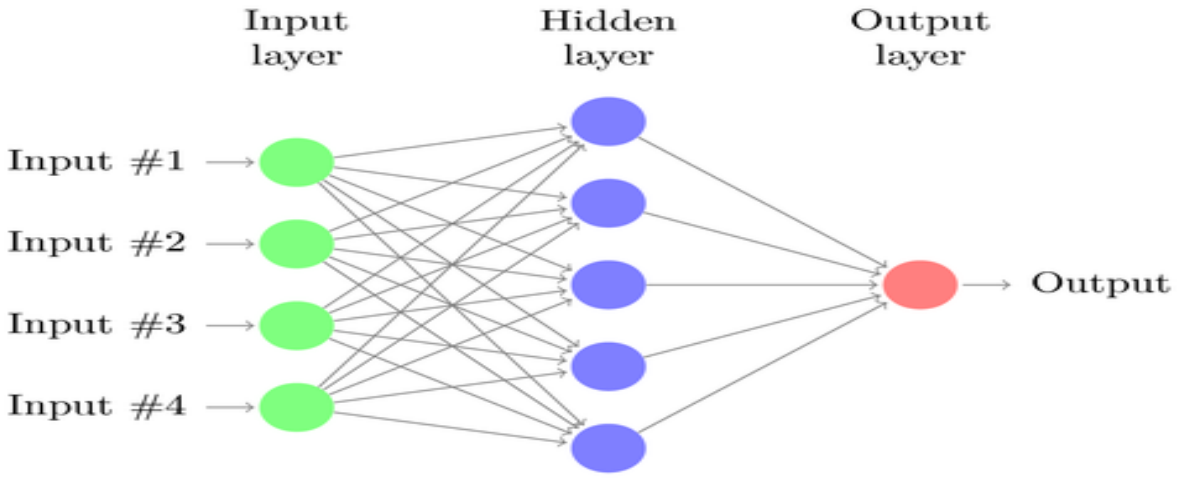


Figure 5.3 Neural Networks

Artificial Neural Networks (ANNs) are composed of a large number of highly interconnected processing elements (neurons) working together to solve specific problems.

ANNs, like humans, learn by example. An ANN is configured for a specific application, such as pattern recognition or text classification, image recognition, voice recognition through a learning process.

The most common type of ANN consists of three groups, or layers, of units: a layer of **input** units is connected to a layer of **hidden** units, which is connected to a layer of **output** units.

- **Input units**: The activity of the input units represents the raw information that is fed into the network. This is also called the input layer.
- **Hidden units**: The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units. This is also called a hidden layer.
- **Output units**: The behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units. This is also called the output layer.

5.3.1 Activation Function

The purpose of the activation function is to introduce non-linearity into the output of a neuron. This is important because most real world data is non linear and we want neurons to learn these nonlinear

representations. Every activation function takes a single number and performs a certain fixed mathematical operation on it.

5.3.2 Dropout

Dropout refers to ignoring units (neurons) during the training phase of a certain set of neurons which are chosen at random. By ignoring, meaning these units are not considered during a particular forward or backward pass.

A fully connected layer occupies most of the parameters, and hence, neurons develop co-dependency amongst each other during training which restricts the individual power of each neuron leading to over-fitting of training data.

5.4 Methodology

When the user visits the MBBS web page, they will see a welcome message. Now the user has to enter all his queries in the text box provided. After pressing the enter or send button, this query is sent to the API. The chatbot model trained using the Neural Networks then classifies the user intent and returns a response from the available list of responses. In this way the user will be able to chat with MBBS chatbot.

Now if the user enters that “I am not feeling well” or some sentence with the similar intent or directly provides the chatbot with the symptoms that he/she is experiencing then the chatbot will make the decision to send these symptoms to the MBBS function ‘get_symptoms()’. But if before sending this message, the user has not provided the chatbot with some personal details like name, age and gender, then the chatbot will first prompt the user to provide these details and only then will the user be able to move forward.

The ‘get_symptom()’ function will then return a list of symptoms, matching with the symptoms provided by the user. Now, how will the chatbot find the matching symptoms from our dataset? For this the bot will process the user entered symptoms with some NLP functions. After this step each symptom is broken down into its basic stem form or list of stem words (due to tokenization) if the initial symptom contains more than one word. If the symptom contains only one stem then it is searched in all the symptoms available in the dataset using regular expression pattern searching.

Once all the matching symptoms are found they are returned to the user. But if the symptom is a list of stems (obtained after tokenizing) then for every stem the same process of pattern matching is performed and a list of symptoms which contain all the stems is compiled. This list is then sent to the user.

The chatbot will then prompt the user to select one or more symptoms from this list of symptoms. Once the user has selected all the symptoms, it is once again sent to the chatbot API. The chatbot model again makes the decision of calling another one of the MBBS functions named ‘get_cooccurring_symptoms’. This function then makes a list of all the co-occurring symptoms with the symptoms selected by the user. This list is then again shown to the user using the GUI.

The user is now allowed to select the symptoms from this new list. Once the user has selected all the symptoms the user is given two options: 1. To get another list of co-occurring symptoms. 2. Done. These are all the symptoms that I have. If the user selects the first option then symptoms selected until now are sent back to the MBBS function through the chatbot. Now the list of co-occurring symptoms will be compiled based on all the selected symptoms till that point. There is one more functionality added here, where all the symptoms that the user didn't select from the list of co-occurring symptoms are saved in a separate list called the ‘skip_symptoms_list’. This list is used in the backend to make sure that no symptoms are shown to the user twice.

Now if the user chooses the second option then the final list of symptoms is sent to the chatbot API. The chatbot once again makes the decision to send these symptoms to the MBBS model which is trained using Logistic Regression. This model then returns a list of probability values for all the diseases (between 0 and 1). After sorting this list in the descending order, the top 5 diseases with the highest values are taken. Now for these top 5 diseases, the age and gender provided by the user are compared. If in the list, there exists a disease which does not happen in the case of the gender provided by the user then it is removed from the list. Then according to the age group, the user is shown whether there is a chance for the user to have that disease or not.

Once the classification of the disease is done, user is shown the list of diseases with a likeness percentage which is calculated using the formula,

$$\frac{n\left((\text{Final symptoms selected by the user}) \cap (\text{Total Symptoms for that disease})\right)}{n(\text{Total Symptoms for that Disease})}$$

Equation 5.4 Calculating Likeness for a disease

This list of diseases also has an option to view more info about the disease. If the user clicks on it then he is shown a detailed description of the disease and various treatment options for it.

Chapter 6
IMPLEMENTATION & CODING

6.1 Introduction

This chapter covers the role of various subsystems/modules/classes along with implementation details listing of the code for the major functionalities.

6.2. Code Listing

6.2.1. Dataset

```
# Read each disease and symptom list, convert into dictionary and add to dataframe
for key, values in final_dis.items():
    key = str.encode(key).decode('utf-8')
    row_norm = dict({x: 0 for x in symp_list})
    for sym in values:
        row_norm[sym.lower()] = 1
    row_norm['label_dis'] = key

    df_norm = df_norm.append(pd.Series(row_norm), ignore_index=True)

print(df_norm.shape)

# Export the dataset into CSV files
# df_comb.to_csv("/content/drive/MyDrive/MedBayBot/dis_sym_dataset_comb_v2.csv",index=None)
df_norm.to_csv("static/Dataset/dis_sym_dataset_norm.csv", index=None)

# Export disease symptoms into TXT file for better visibility
with open('static/Dataset/dis_symp_dict.txt', 'w') as f:
    for key, value in final_dis.items():
        print([key]+value, file=f)
print("done")
```

6.2.2. Data Scraping

```
headers = {
    'User-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.77 Safari/537.36'}
cnt = 1
f_cnt = 1
notFound = []
# disease = {}
for dis in dis_list:
    query = dis + ' symptoms'
    URL = 'https://www.google.com/search?q={}'.format(query)
    # print(URL)
    # print(URL)
    time.sleep(2)
    page = requests.get(URL, headers=headers)
    page = BeautifulSoup(page.content, 'html5lib')
    with open('page.html', 'w') as f:
        print(page, file=f)
```

```
    try:
        page = page.find('ul', class_='i8Z77e')
        # print(page)
        sym = []
        # try:
        symp1 = page.find_all('li', class_='TrT0Xe')
        # print(symp1)
        for element in symp1:
            sym.append(element.get_text())
        # except:
        #   print(cnt, dis)
        #   notFound.append(dis)
        #   cnt += 1

        disease[dis] = {}
        disease[dis]['symptoms'] = sym
        # found[dis] = result.get_text()
        print(f_cnt, dis, ":", disease[dis])
        f_cnt += 1
    except:
        # print(cnt, dis)
        notFound.append(dis)
        cnt += 1

# cnt = 1
# for key,value in disease.items():
#   print(cnt,key,value['symptoms'])
#   cnt += 1

with open('static/Dataset/dis_json.json', 'w') as f:
    json.dump(disease, f, indent=4)
f.close()
```

6.2.3. Disease Prediction Modules

```
        return list(found_symptoms)

    def predict_disease(self, limit, selected_symptoms):
        sample_x = [0 for x in range(0, len(self.DATASET_SYMPTOMS))]
        for symptom in selected_symptoms:
            sample_x[self.DATASET_SYMPTOMS.index(symptom)] = 1
        prediction = NORM_MODEL.predict_proba([sample_x])
        return prediction[0].argsort()[-(limit):][::-1]

    def get_common_symptoms(self, top_10_diseases, selected_symptoms, skip_symptoms):
        match_symp = []
        for idx, index in enumerate(top_10_diseases):
            row = self.df_norm.loc[self.df_norm['label_dis']
                                   == self.DISEASES[index]].values.tolist()
            row[0].pop(0)
            for idx, val in enumerate(row[0]):
                if val != 0 and self.DATASET_SYMPTOMS[idx] not in selected_symptoms and self.DATASET_SYMPTOMS[idx] not in skip_symptoms:
                    match_symp.append(self.DATASET_SYMPTOMS[idx])
        if len(match_symp) == 0:
            return match_symp
        return Counter(match_symp)


    def get_symptoms_dict(self, dict_symp):
        symp_dictionary = {}
        symp_dictionary['co_occuring'] = []
        dict_symp_tup = sorted(
            dict_symp.items(), key=operator.itemgetter(1), reverse=True)
        for obj in dict_symp_tup:
            temp_dict = {}
            temp_dict['symptom'] = obj[0]
            temp_dict['frequency'] = obj[1]
            symp_dictionary['co_occuring'].append(temp_dict)
        return symp_dictionary
```

6.2.4 User Interface

```
generateBotMessage(res){
    var msg;

    console.log('response '+JSON.stringify(res));
    if(res.drug_info_flag) {
      this.drugInfo = res.drug_info_flag;
    }
    if(!this.username && res.name) {
      this.username = res.name;
    }
    if(!this.age && res.age) {
      this.age = res.age;
    }
    if(!this.gender && res.gender) {
      this.gender = res.gender;
    }
    if(Array.isArray(res.response) ) { // string of messages
      res.response.forEach((text) => {
        msg = {
          message:text,
          symptoms:[],
          owner:'bot'}
          this.sendBotMessage(msg);
      });
    } else { //single message
      msg = {
        message:res.response,
        symptoms:[],
        owner:'bot'}
        this.sendBotMessage(msg);
    }
    this.messagesChanged.next(this.messages);
        if(res.gender_query && !this.gender) {
          msg = {
            genderQuery : res.gender_query,
            owner : 'bot'}
          this.sendBotMessage(msg);
        }
        if(res.found_symptoms && res.found_symptoms.length>0){
          msg = {
            symptoms:res.found_symptoms,
            owner:'bot'}
            this.sendBotMessage(msg);
        } else if (res.co_occuring){
          let symptomsList= this.getCoocurringSymptoms(res.co_occuring);
          msg = {
            finalSymptoms:symptomsList,
            owner:'bot'}
            this.sendBotMessage(msg);
        } else if(res.diseases_list) {
          msg = {
            message:'Based on the data provided you may have : ',
            diseaseList:res.diseases_list,
            owner:'bot'}
```

Chapter 7
TESTING

7.1 Introduction

This chapter covers the testing approach used and the test cases.

7.2 Unit Testing

It is a technique using which individual modules are tested to determine if there are any issues by the developer himself.

Do not create test cases for everything. Instead, focus on the tests that impact the behavior of the system.

Isolate the development environment from the test environment.

Use test data that is close to that of production.

Before fixing a defect, write a test that exposes the defect. Why? First, you will later be able to catch the defect if you do not fix it properly. Second, your test suite is now more comprehensive. Third, you will most probably be too lazy to write the test after you have already fixed the defect.

Write test cases that are independent of each other. For example, if a class depends on a database, do not write a case that interacts with the database to test the class. Instead, create an abstract interface around that database connection and implement that interface with a mock object.

Aim at covering all paths through the unit. Pay particular attention to loop conditions. Make sure you are using a version control system to keep track of your test scripts. In addition to writing cases to verify the behavior, write cases to ensure the performance of the code.

7.3 Test Cases

Test Case Name	Disease Prediction - Providing all the symptoms for a disease
Prerequisite	User must have provided all the correct symptoms
Steps to be executed	1. Get the symptoms for calling prediction function 2. Get the actual disease names related with those symptoms 3. Pass these symptoms to the prediction function and get the predicted disease names 4. Compare the actual disease names and predicted disease names
Expected Results	Disease names (e.g. Abdominal aortic aneurysm)
Actual Results	Disease names
Pass/Fail	Pass
Remark	This test case was used to check the accuracy of the model by randomly passing the symptoms for 100 diseases

Test Case Name	Disease Prediction - Providing some of the symptoms for a disease
Prerequisite	User must have provided all the correct symptoms
Steps to be executed	1. Get the symptoms for calling prediction function 2. Get the actual disease names related with those symptoms 3. Pass these symptoms to the prediction function and get the predicted disease names 4. Compare the actual disease names and predicted disease names
Expected Results	Disease names (e.g. Abdominal aortic aneurysm)
Actual Results	Disease names

Pass/Fail	Pass
Remark	This test case was also used to check the accuracy of the model by randomly passing the symptoms for 100 diseases

Test Case Name	Chatbot Response
Prerequisite	User must write a text query in english
Steps to be executed	1. A text query will be sent to the chatbot model. 2. Chatbot will then return a intent and a response for that query 3. Then we check the classified intent and expected intent.
Expected Results	Intent: Greeting for the “Hello” message.
Actual Results	Intent: Greeting
Pass/Fail	Pass
Remark	This test case was also used to check the accuracy of the model by passing text queries

Table 6.2 Unit Testing Testcases

Chapter 8
RESULTS & DISCUSSIONS

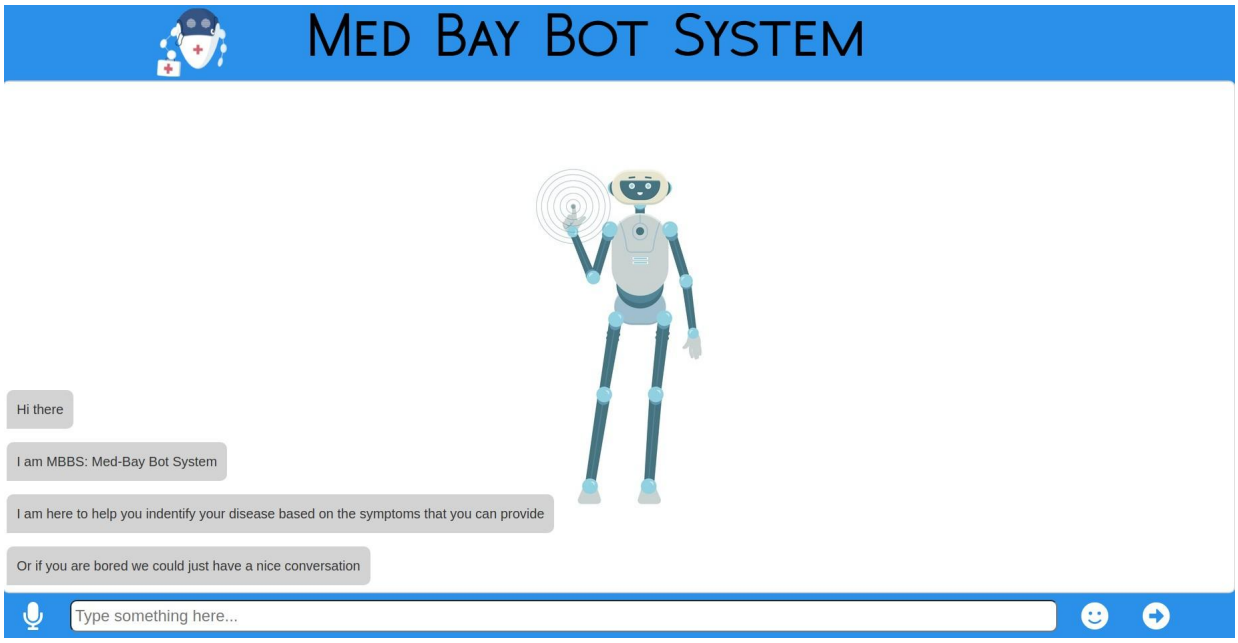


Figure 8. System Starts

8.1. Talk back service:

This feature collects primary information of the user like name,age ,gender,etc.Also user can chat with the system like “What is your name?”,”What is your work?”.Users information will be helpful to correctly diagnose the disease.

This is the most necessary and mandatory part to move further and for correct prediction of disease.

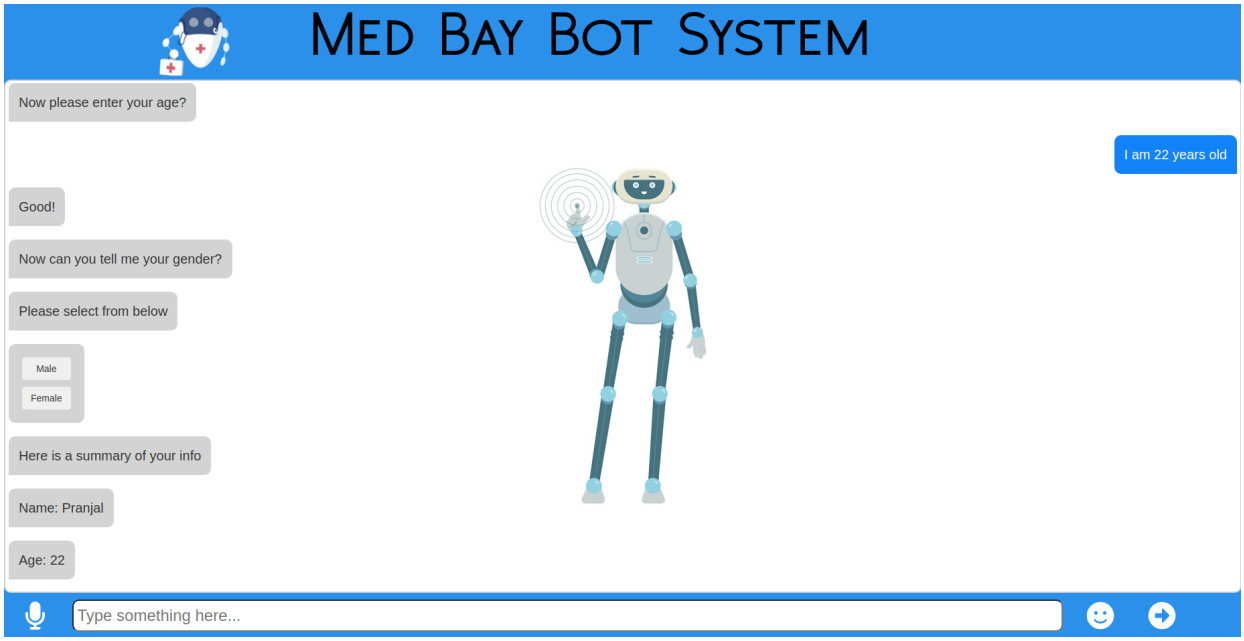


Figure 8.1. Talk with user

8.2. Symptoms Selection:

Here the user first enters Symptoms they have. Then System suggests some topmost matching symptoms to user.If users have symptoms from the suggested list they can select among them otherwise they can select “Done.But I have more symptoms”.Then again it suggests symptoms based on previous symptoms.Once all symptoms are covered user can select “Done!These are all the symptoms I have”.

In this way the system can collect all symptoms from the user. e.g In below pictures user entered experiencing fever (Low-grade fever). Then the system again suggests more symptoms related to that.This chain will continue until the user selects :”Done!These are all the symptoms I have”.

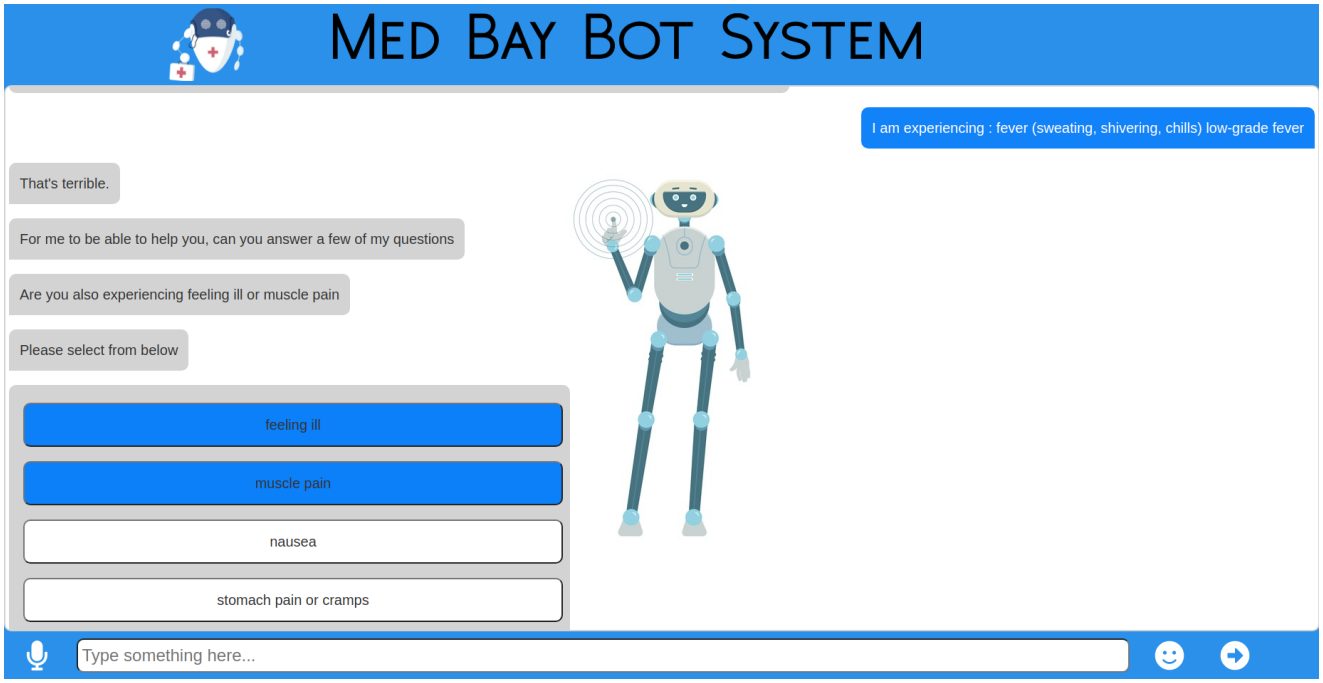


Figure 8.2.1. User Enter Symptoms

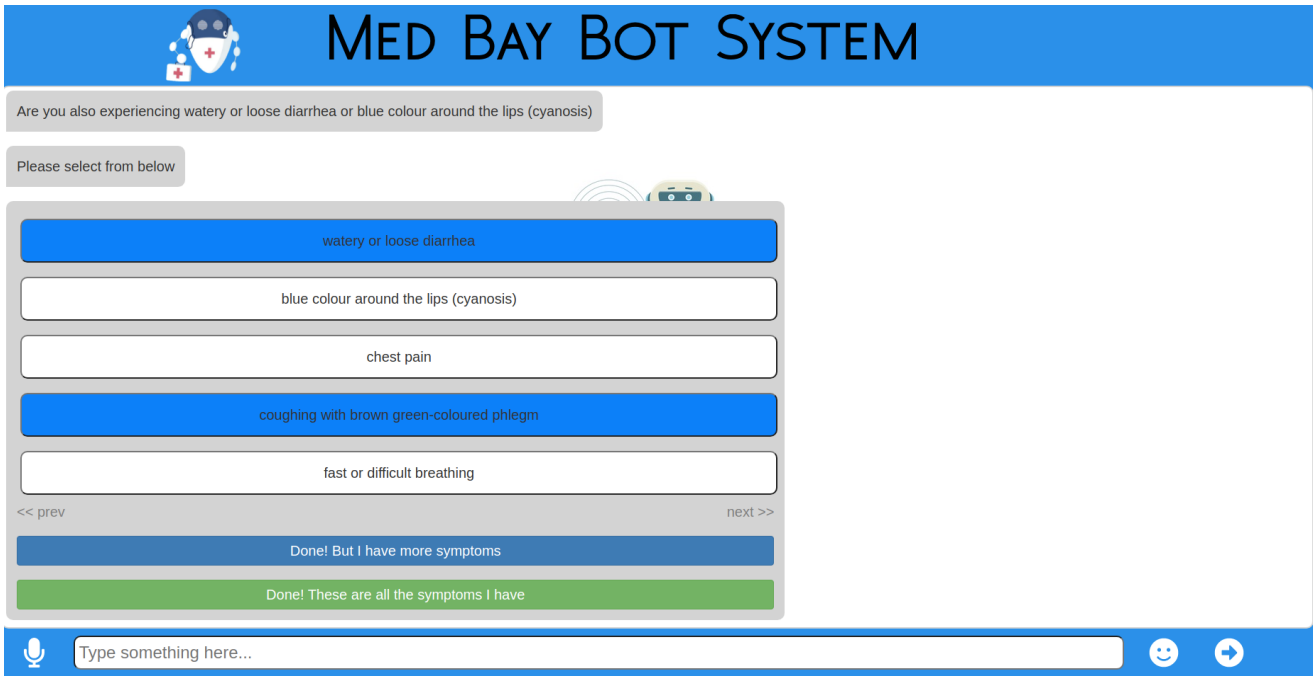


Figure 8.2.2. More Symptoms Suggestions

8.3. Disease Prediction:

Once all symptoms are entered, the system can suggest upto top 5 disease.e.g here in picture user enter symptoms like low grade fever,muscle pain,feeling ill,loose motion,coughing,etc. This system suggests Norovirus and Chest Infection topmost occurring diseases.

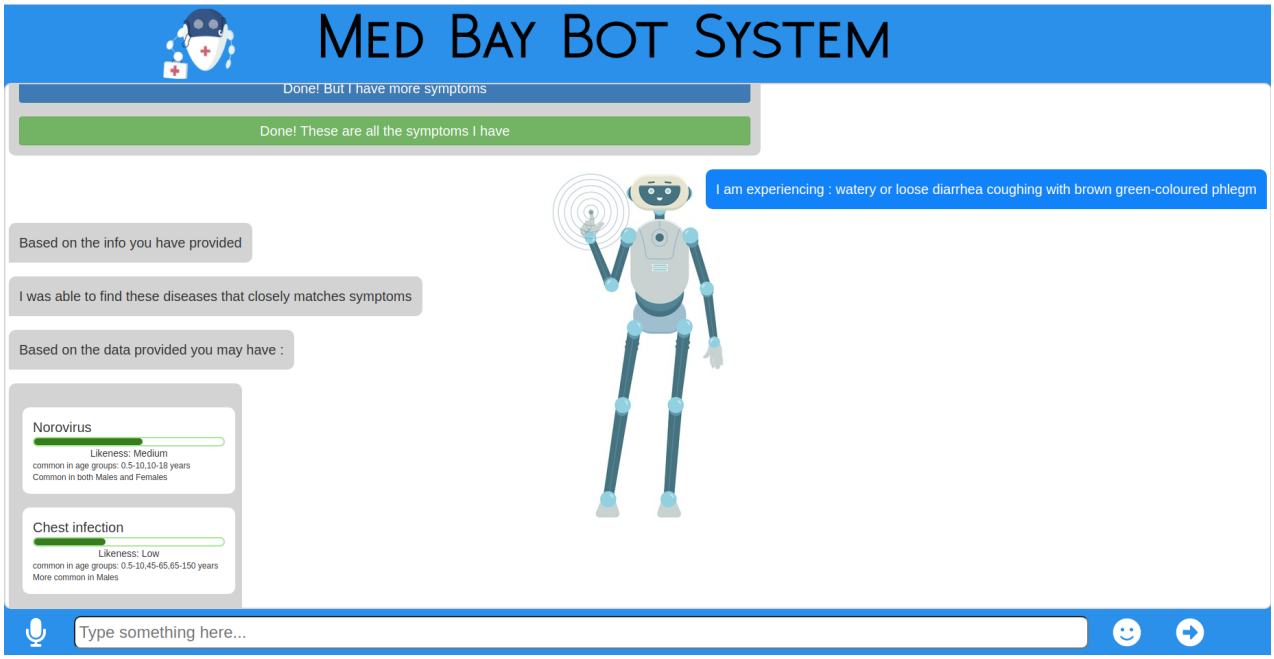


Figure 8.3. Disease prediction with symptoms

8.4. Drug Information:

This feature gives us Drug information.If user want to know about some medicines what they used for,what it's contents,it's benefits and side effect they can enter name of that medicine and system will suggest some related drug names.User can select option which they want.e.g If User entered Crocin then system can suggest related options like crocin advance,crocin pain relief,crocin240 DS suspension mixed fruit,Crocin baby drop,crocin 1000mg etc. so user can select any one for more information.

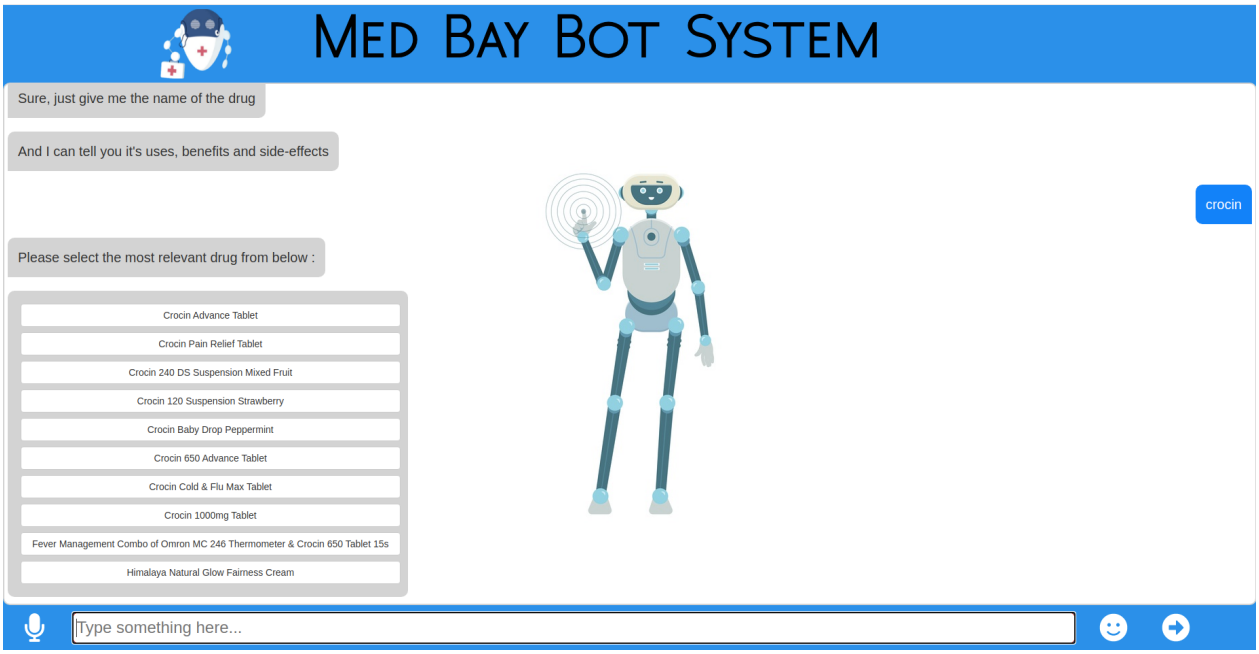


Figure 8.4.1. Most relevant drug

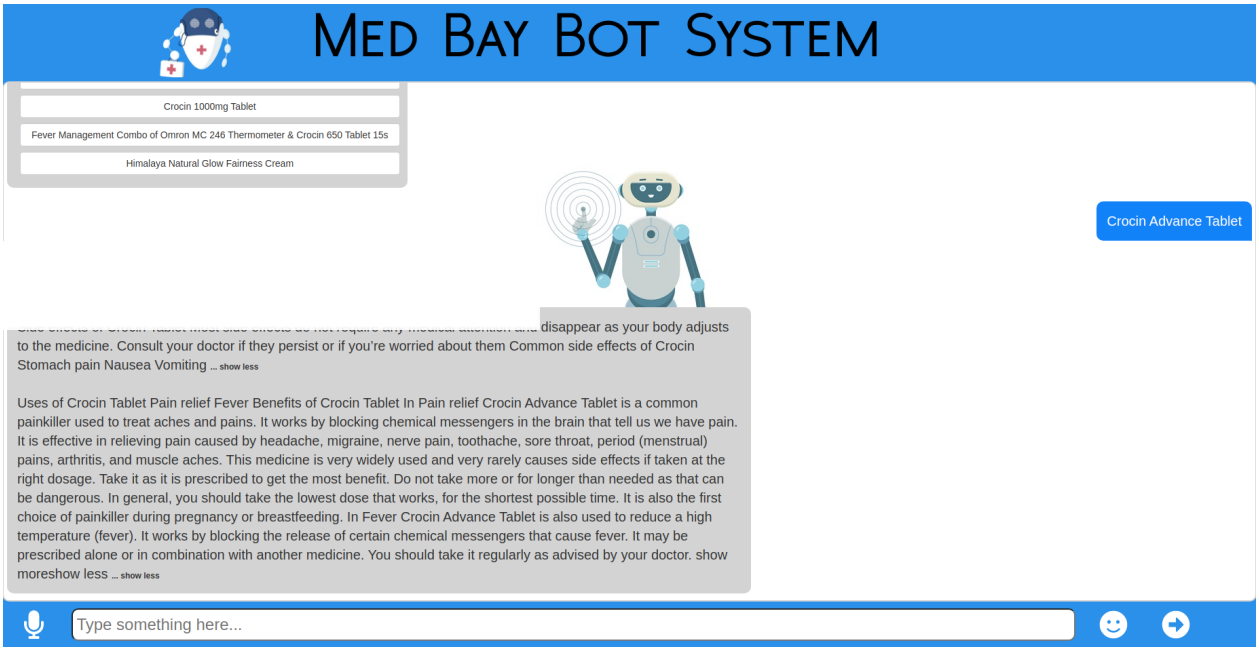


Figure 8.4.2. Drug Information

8.5. Disease Information:

In this feature according to symptoms the system predicts 5 topmost possible diseases. If

user want more information about that diseases then user can click on that particular diseases and required all information will get e.g in given picture gives information of abdominal aortic aneurysm

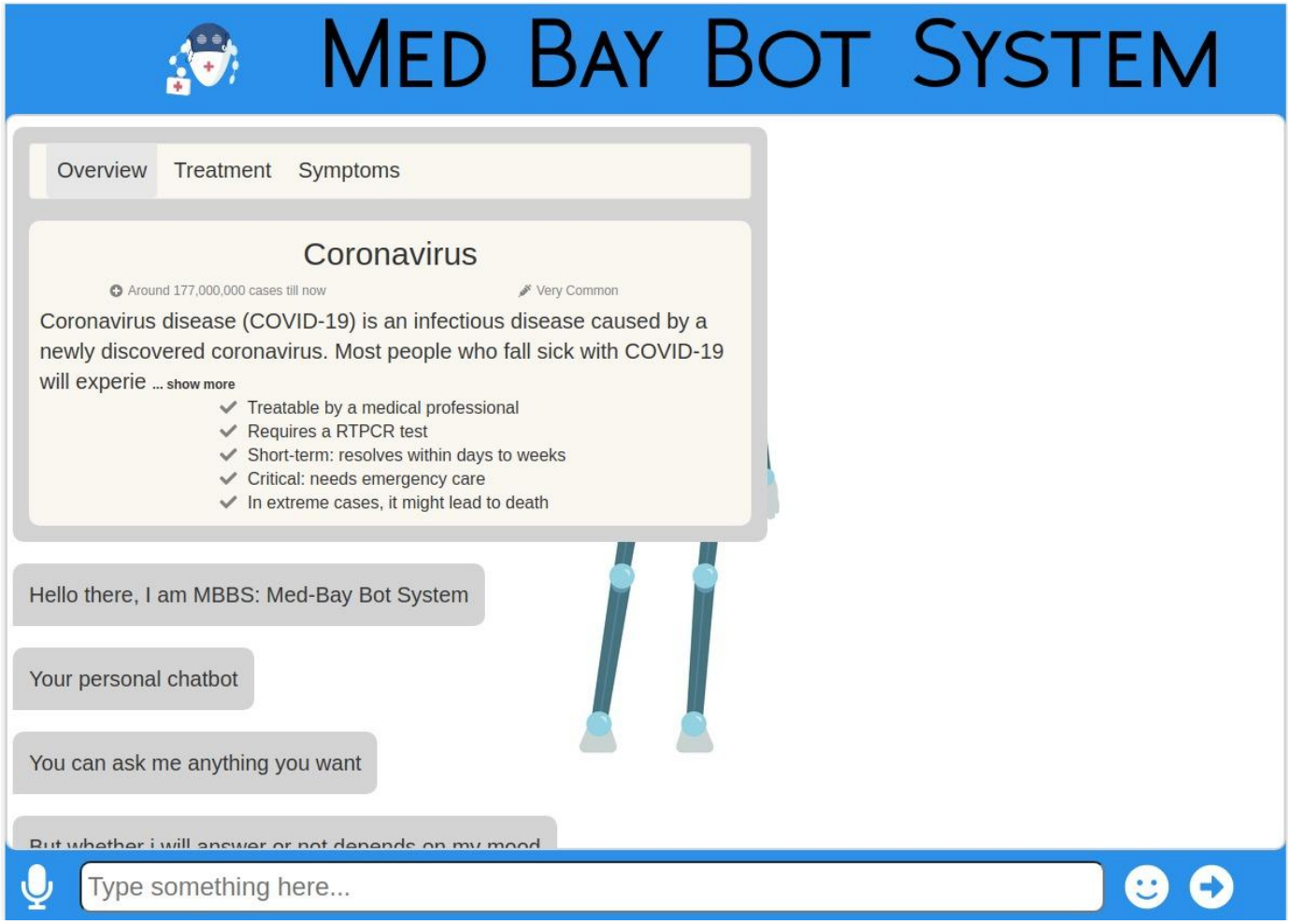


Figure 8.5. Disease Information

8.6. Accuracy

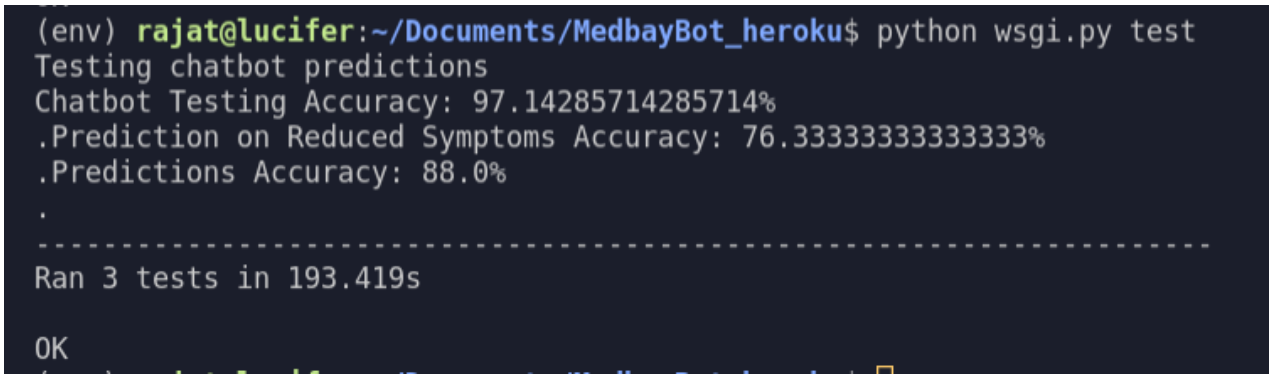


Figure 8.6. Accuracy

8.7. Comparison with other ML models

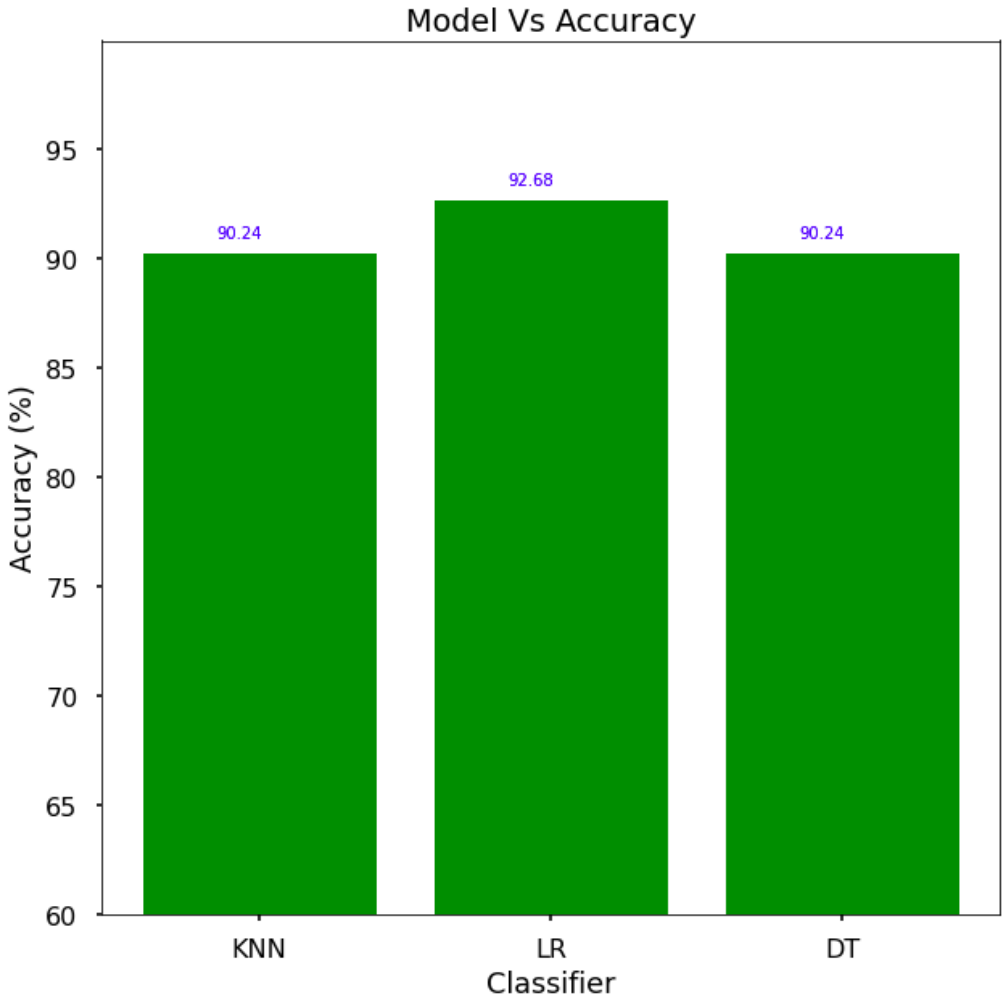


Figure 8.7. Comparison Graph

Above graph shows the comparison between accuracy of three different ML models which are KNN, Logistic Regression and Decision Tree classifiers. As we can see from the plotted graph that Logistic regression gives slightly better accuracy than the other two algorithms. Hence, we chose to train our model using Logistic Regression

8.8. Result Comparison with WebMD Symptom Checker

As the accuracy score shown above only represents the accuracy of our model and not the accuracy of a particular disease, we decided to check our model predictions against a well established symptom checker available on the internet. We chose to compare our model against WebMD Symptom Checker. Entering almost the same symptoms on both (WebMD and our model) we got the same results most of the time. Below fig. shows the prediction of our model vs WebMD predictions.

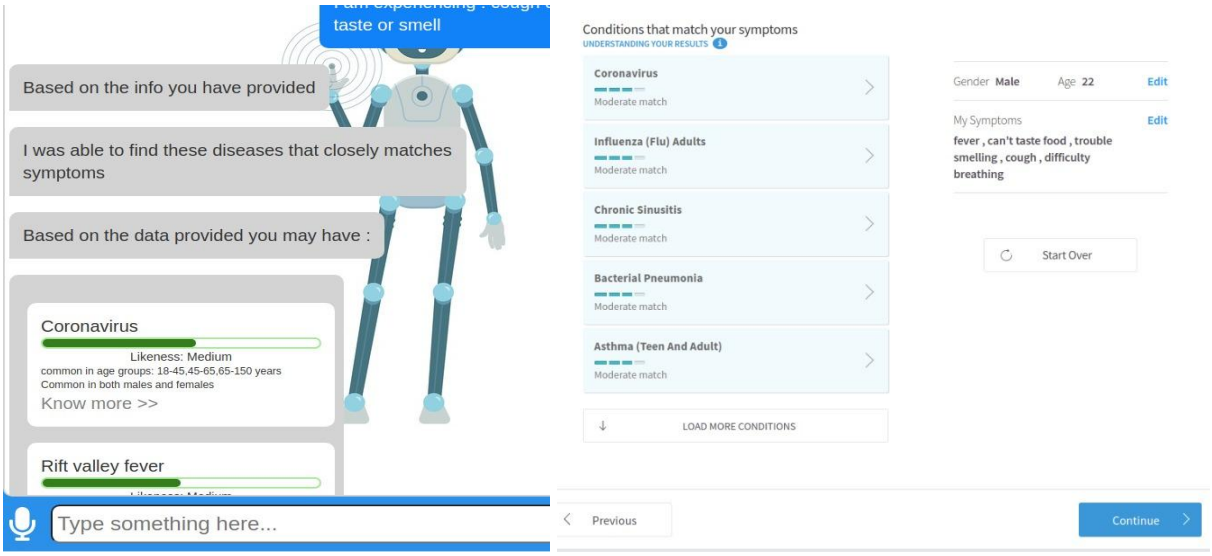


Figure 8.8 Result comparison with WebMD

Chapter 9
CONCLUSION AND FUTURE WORK

9.1. Conclusion

This project is an adequate solution for many services. The main key feature of this project is disease prediction. We can easily and precisely predict the disease on the basis of symptoms. So the user gets to know about the respective disease as early as possible and according to that user can consult a doctor.It is useful for other services like Disease information,Drug information.It gives information about particular disease and drugs.

The application is developed for obtaining a fast response from the bot which implies with minimal delay it provides the correct result to the user. It is found that chatbot is user friendly even for the naive user. Chatbot provides personalised diagnosis supported symptoms.

The system was able to suggest a diagnosis using a direct approach of the question and answering technique. The new development in artificial intelligence and the new way of thinking have the potential to entirely change the experience of customers. Especially in the field of medicine, a medical based chatbot offers a personalized analysis based upon symptoms.

9.2. Future Work

Our aim is to implement a full fledged interactive system which can later help the user in scheduling doctors appointments, finding relevant doctors upon detection of a disease, which are practicing near the user's location. An interactive speech to text and text to speech element can also be added for the differently abled individuals.

The chatbot can also suggest drugs related to the diagnosis it has made. A system can be implemented where the chatbot can remind the user about their scheduled drug dosage and scheduled doctor appointments.

REFERENCES

1. Joseph Weizenbaum. 1966, “ELIZA—a computer program for the study of natural language communication between man and machine”, Commun. ACM 9, 1 (Jan. 1966), 36–45. DOI:https://doi.org/10.1145/365153.365168
2. Shah, Huma. (2005). A.L.I.C.E.: an ACE in Digitaland. 4.
3. A. H. Chen, S. Y. Huang, P. S. Hong, C. H. Cheng and E. J. Lin, "HDPS: Heart disease prediction system," 2011 Computing in Cardiology, 2011, pp. 557-560.
4. S. Kaur, H. Aggarwal and R. Rani, "Neurological disease prediction using ensemble Machine Learning Model," 2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), 2020, pp. 410-414, doi: 10.1109/PDGC50313.2020.9315758.
5. Patnaik, Sourav Kumar, et al. "Automated Skin Disease Identification using Deep Learning Algorithm." Biomedical and Pharmacology Journal, vol. 11, no. 3, 2018, p. 1429+. Gale OneFile: Health and Medicine, link.gale.com/apps/doc/A592785486/HRCA?u=anon~1d7bf681&sid=googleScholar&xid=3de369bd. Accessed 13 June 2021.
6. R. B. Mathew, S. Varghese, S. E. Joy and S. S. Alex, "Chatbot for Disease Prediction and Treatment Recommendation using Machine Learning," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), 2019, pp. 851-856, doi: 10.1109/ICOEI.2019.8862707.
7. D. Dahiwade, G. Patle and E. Meshram, "Designing Disease Prediction Model Using Machine Learning Approach," 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), 2019, pp. 1211-1215, doi: 10.1109/ICCMC.2019.8819782.
8. P. Kandpal, K. Jasnani, R. Raut and S. Bhorge, "Contextual Chatbot for Healthcare Purposes (using Deep Learning)," 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), 2020, pp. 625-634, doi: 10.1109/WorldS450073.2020.9210351.
9. Dongkeon Lee, Kyo-Joong Oh and Ho-Jin Choi, "The chatbot feels you - a counseling

service using emotional response generation," 2017 IEEE International Conference on Big Data and Smart Computing (BigComp), 2017, pp. 437-440, doi: 10.1109/BIGCOMP.2017.7881752.

10. Amato, F. et al. “Chatbots Meet eHealth: Automatizing Healthcare.”, 2017, WAIAH@AI*IA.
11. Prof. A. B. Gadewar, Ammara Shaikh, Samiksha Bhalerao, Bhagyashree Nawsagre, “Chat-Bot Utilization for Health Consultancy”, 2020, International Research Journal of Engineering and Technology (IRJET), pp. 1832-1834
12. https://www.nhsinform.scot