**Project Code-CST2191G6ML02**

A Project Report

On

**“Chatbot for Healthcare System Using AI”**

**An Internship Training Project report submitted in partial fulfilment of the requirements for the**



**Under the guidance of**

**Nithin KS**

**Project Manager**

**Compsoft Technologies**

**Submitted By**

**Team Name: Elite**

**M. VARSHA.          3BR19CS082**

**NIVEDITA.M        3BR19CS098**

**RAHUL B H R.        3BR19CS121**

**ABSTRACT**

To start a good life healthcare is more important. But it is very difficult to obtain the consultation with the doctor in case of any health issues. The proposed idea is to create a health care Chabot system using Artificial Intelligence that can diagnose the disease and provide basic details about the disease before consulting a doctor. The system provides text (or) voice assistance that means user can use own convenient language. Bot will provide which type of disease you have based on user symptoms and appeared doctor details respective to user disease. The Chabot will clarify the users symptoms with series of questions and the symptom confirmation will be done. The disease will be categorized as minor and major disease. Chatbot will reply whether it is a major or minor disease. If it is a major disease user will be suggested with the doctor details and analgesics for further treatment and also provides food suggestion that means which type of food you have to take. The user can achieve the real benefit of a chatbot only when it can diagnose all kind of disease and provide necessary information. A text-to-text diagnosis Bot engages patients in conversation about their medical issues and provides a personalized diagnosis based on their symptoms. Hence, people will have an idea about their health and have the right protection.

**TABLE OF CONTENTS**

**Chapter No. Chapter Name**

Abstract

Table of Contents

List of Figures

1. Preamble

1.1. Introduction

1.2. Problem Statement

2 System Design

3 Software Requirement Specifications

3.1 Functional Requirements

3.2 Non Functional Requirements

3.3 Software Description

4 System Analysis

4.1. Existing System

4.2. Proposed System

5 Module Description

6 Implementation/Methodology

7 Software Testing

8 Code

9 Conclusion

10 References

**LIST OF FIGURES**

Figure No. Figure Name

**Figure 1.** Chatbot Architecture

**Figure 2.** System Architecture

**Figure 3.**  Data Flow Diagram

**CHAPTER 1**

**PREAMBLE**

**1.1. INTRODUCTION**

Nowadays, health care is very important in our life. Today’s people are busy with their works at home, office works and more addicted to Internet. They are not concerned about their health. So they avoid to go to hospitals for small problems, this may result in a major problem. So we can create a health care Chabot system using AI that can diagnosis the disease and provide basic information about the disease before consulting a doctor, which helps the patients to know more about their disease and improves their health. The system application uses question and answer protocol in the form of chatbot to answer user queries. The response to the question will be replied based on the user query. The significant keywords are fetched from the sentence and answers are given according to those keywords. If match is discovered or significant answer will be given or similar answers will be displayed. Bot will diagnosis which type of disease you have based on user symptoms and also gives doctor details of particular disease. It may reduce their health issues by using this application system. The system is developed to reduce the healthcare cost and time of the users as it is not possible for the users to visit the doctors or experts when immediately needed.

Artificial Intelligence is based on how any device perceives its Environment and takes actions based on the perceived data to achieve the result successfully. It is the study of intelligent agents. The term "artificial intelligence" is applied when a machine mimics "cognitive" functions that humans associate with other human minds, such as “learning” and “problem solving”. Artificial Intelligence gives the supreme power to mimic the human way of thinking and behaving to a computer. A chatbot (also known as a talk bot, chatterbot, Bot, IMbot, interactive agent, or Artificial Conversational Entity) is a computer program which conducts a conversation via auditory or textual methods. These programs are designed to provide a clone of how a human will chat and thereby it acts as a conversational partner rather than humans. For various practical purposes like customer service or information acquisition, , chatbot is being used in the dialog system. Mostly chatbots uses natural language processing for interpreting the user input and generating the corresponding response but certain simpler systems search for the keyword within the text and then provides a reply based on the matching keywords or certain pattern. Today, chatbots are part of virtual assistants such as Google Assistant, and are accessed via many organization apps, websites, and on instant messaging platforms.

Chatbots are such kind of computer programs that interact with users using natural languages. Chatbots work basically on Artificial intelligence so using this capability we have decided to add some contribution to the Health Informatics. The high cost of our healthcare system can often be attributed to the lack of patient engagement after they leave clinics or hospitals. Various surveys in this area have proved that that chatbot can provide healthcare in low costs and improved treatment if the doctors and the patient keep in touch after their consultation.

The main purpose of the scheme is to build the language gap between the user and health providers by giving immediate replies to the Questions asked by the user. Establishing question answer forums is becoming a simple way to answer those queries rather than browsing through the list of potentially relevant document from the web. Many of the existing systems have some limitation such as there is no instant response given to the patient they have to wait for experts acknowledgement for a long time. Some of the processes may charge amount to perform live chat or telephony communication with doctors online. This system allows computer for communication between human and computer by using natural language processing (NLP). There are three analyses which understand natural language i.e., identification of main linguistic relations is completed to parse subject into object of the sentences. After that Chatbot is an Entity which imitate human discussion in its particular accepted set-up together with a text or vocal language with techniques such as Natural Language Processing (NLP). The aim of this system is to replicate a person’s discussion. The development of chatbot application can be done with making a user interface to send input and receive response. It is a system that interact with user by keeping the track of the state of interaction and recollecting the preceding commands to give functionality. The medical chat-bots can be developed by using artificial algorithms that scrutinize user's queries and recognize it and give reply to related query.

Most of the disease can be identified by common symptoms so the disease can be predicted if the patient body is analyzed periodically. The system give response by use of an efficient Graphical User Interface such that if actual person is chatting with the user. chatterbot that can be used in various fields like education, healthcare, and route assistance The central part of the chat-bots includes MySQL. It is an interactive system solve users query regarding medicine. so they can get correct guidance for treatment through android app by using Google API.

Computers give us information; they engage us and help us in a lot of manners. A chatbot is a program intended to counterfeit smart communication on a text or speech. Yet, this paper concentrates only on text.

These systems can learn themselves and restore their knowledge using human assistance or using web resources. This application is incredibly fundamental since knowledge is stored in advance. The system application uses the question and answer protocol in the form of a chatbot to answer user queries. This system is developed to reduce the healthcare cost and time of the users, as it is not possible for the users to visit the doctors or experts when immediately needed.

The response to the question will be replied based on the user query and knowledge base. The significant keywords are fetched from the sentence and answer to those sentences. If the match is discovered or the significant answer will be given or similar answers will be displayed.

The complex questions and answers present in the database are viewed and answered by an expert. Here the users can personally ask any questions regarding healthcare, as not much time will be wasted by the user for consulting a doctor. The input sentence of the chat pattern is stored in an RDBMS. The chatbot would coordinate the input sentence from the user question with the knowledge base. Each query is compared with the knowledge database of the chatbot. The important keywords are extracted from the given input sentence and the sentence similarity is found. The keyword ranking and sentence similarity are found using the N-gram, TF-IDF, and cosine similarity. The interfaces are standalone built using the JAVA programming language.

**WHAT IS HEALTHCARE CHATBOT?**

In general terms, a bot is nothing but a software that will perform automatic tasks. In other terms, a bot is a computer program that is designed to communicate with human users through the internet. The most natural definition of a chatbot is – a developed a program that can have a discussion/conversation with a human. Healthcare chatbots are software integrated into either a website or mobile app and possess any combination of the following attributes:

**Natural language processing** : The ability for a chatbot or other form of artificial intelligence to understand aspects of human language including syntax and semantics from users of various backgrounds.

**Contextual AI**: contextual AI takes into consideration the intent of the user as they ask questions and submit responses, thus ensuring the best relevant conversation takes place.

**Chatbot Architecture**

Previously chat bots solely supported a single adjacency pair, also known as a one-shot conversation. However, modern chatbots can sustain multiple adjacency pairs, remembering states

and contexts between conversations and have the capability to associate data in different adjacency pairs which is related. This is the bot’s ability to preserve the conversation. A chatbot consists of four main parts: front- end, knowledge-base, back-end and corpus which is the training data.

The front end is accountable for enabling communication between the bot and the user. The NLU utilises Artificial intelligence methods to identify the intent and context of the user input.

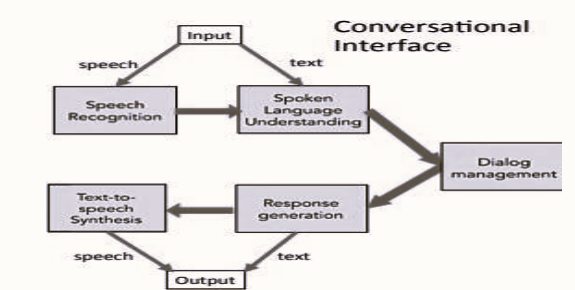


Fig 1. Chatbot Architecture

An appropriate response is generated from the user’s intent. The knowledge base defines the chatbot knowledge, which is created within the NLU and supported by the back-end, the back-end applies the domains corpus to produce the knowledge base. Input can be supplied to the chatbot in the form of text or speech. The Input is sent to the dialog management system which is the NLU in this case, which determines an appropriate response and amends the chatbots state accordingly to carry out the required action. The chatbot will produce responses in the form of both text and speech.

**1.2. Problem Statement**

To design and develop Healthcare Chatbot system using knowledgeable database and interpreter

which will be employed as a function of pattern matching.

Healthcare Chatbot helps to improve the patient experience and overall functioning of healthcare

organization.

For once, medical chatbots reduce healthcare professionals’ workload by reducing hospital visits,

reducing unnecessary treatment procedures, decreasing hospital admissions and readmissions as

treatment compliance and knowledge about their symptoms improve. For patients, this comes with

a lot of benefits:

* Less time spent commuting to the doctor’s office
* Less money spent on unnecessary treatments and tests
* Easy access to the doctor at the push of a button

Goal and Objectives:

* To minimize the time required to solve the queries.
* To give response to the user based on queries.
* To simplify communication between user and machine.

**CHAPTER 2**

**SYSTEM DESIGN**

INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing. It can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

OBJECTIVES

1.Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus, the objective of input design is to create an input layout that is easy to follow.

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through output. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system‘s relationship to help user in decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system. The output form of an information system should accomplish one or more of the following objectives.

* Convey information about past activities, current status or projections of the Future.
* Signal important events, opportunities, problems, or warnings.
* Trigger an action.
* Confirm an action.

SYSTEM ARCHITECTURE



Fig 2. system Architecture

The above Figure proceeds with the user can start their conversation with the chatbot like user friendly and it will be stored in the database for future reference. The chatbot will clarify the user’s symptoms with serious of questions and the symptom conformation will be done. The disease will be categorized as minor and major disease. Chatbot will reply whether it’s a major or minor disease. If it’s a major one user will be suggested with the doctor details nearby you for further treatment and display the analgesics and also provides food suggestions that means which food you have to take more to recover the disease. The chatbot user interface can chat with like user friendly.by using chatbot don‘t go to hospitals for even small problems.

DATA FLOW DIAGRAM

Fig 3. Data Flow Diagram

The chatbot will take the input from the user and then processing the input by using algorithms. Bot will apply the algorithms on whatever the user give the input to the bot.it will understand the input by using algorithms, set of symptoms in the database. The chatbot will clarify the user’s symptoms with serious of questions and the symptom conformation will be done. The disease will be categorized as minor and major disease. Chatbot will reply whether it’s a major or minor disease. If it’s a major one user will be suggested with the doctor details nearby you for further treatment and display the analgesics and also provides food suggestions that means which food you have to take more to recover the disease.

**CHAPTER 3**

**SOFTWARE REQUIREMENT SPECIFICATIONS**

**3.1 Functional Requirements**

**Hardware Requirement**

Processor: INTEL CORE i7 2.60 GHz

RAM: 16GB or Higher

Hard Disk: 1TB

**Software Requirements**

OS: Windows 11(64-bit operating system)

Browser Used: Google Chrome

Libraries and frameworks for Artificial intelligence used: Tensorflow, Scikit-learn, NLTK,

PyTorch, Keras, Pandas

**3.2 Non-Functional Requirements**

Software Quality Attributes

* Robustness
* Reliability
* Better learning methods
* Acquiring good accuracy results

**3.3 Software Description**

**Python language:**

Python is the popular programming language. In-fact you will see the code of python language as English language. Python 2 and Python 3 are the two major versions of Python. But, Python Installation for both the versions are different. In addition, Commands and path environment variables for both the versions are different. You can choose your python version according to your requirement. Beginners always faces challenges of installing Python in their operating system.

Python is a popular object-oriented programming language having the capabilities of high-level programming language. It’s easy to learn syntax and portability capability makes it popular these days. The followings facts give us the introduction to Python.

Python was developed by Guido van Rossum at Stichting Mathematisch Centrum in the Netherlands. It was written as the successor of programming language named ABC. It’s first version was released in 1991. The name Python was picked by Guido van Rossum from a TV show named Monty Python‘s Flying Circus. It is an open source programming language which means that we can freely download it and use it to develop programs. It can be downloaded from www.python.org.. Python programming language is having the features of Java and C both. It is having the elegant ‗C‘ code and on the other hand, it is having classes and objects like Java for object-oriented programming. It is an interpreted language, which means the source code of Python program would be first converted into bytecode and then executed by Python virtual machine.

This article Python Installation tutorial is for them. Introduction to Python Programming article is a complete guide for beginners to know more about Python. You will know how to install python in Windows, Linux, MacOS. How to check your Python versions .

You make sure to install python carefully by reading all the steps described here otherwise it may be wrongly install.Python Installation Tutorial for Windows

In this section, you will know how to do Python installation in the Windows OS. First we will cover for the latest Python 3 version and then Python 2 version. You can either install both the versions or choose the version according to your end goal.

**Applications of Python**

1. Science and Numeric Applications

Python finds its place in the scientific community

For this, we have:

* **SciPy** – A collection of packages for mathematics, science, and engineering.
* **Pandas**– A data-analysis and -modeling library

**Pandas in python**

Pandas is an open-source, BSD-licensed Python library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.In this tutorial, we will learn the various features of Python Pandas and how to use them in practice.Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.

In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.Prior to Pandas, Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing sand analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

**Key Features of Pandas**

* Fast and efficient DataFrame object with default and customized indexing.
* Tools for loading data into in-memory data objects from different file formats.
* Data alignment and integrated handling of missing data.
* Reshaping and pivoting of date sets.
* Label-based slicing, indexing and subsetting of large data sets.
* Columns from a data structure can be deleted or inserted.
* Group by data for aggregation and transformations.
* High performance merging and joining of data.
* Time Series functionality.

Standard Python distribution doesn't come bundled with Pandas module. A lightweight alternative is to install NumPy using popular Python package installer, pip

**Pip install pandas**

* **IPython** – A powerful shell for easy editing and recording of work sessions. It also supports visualizations and parallel computing.
* **Software Carpentry Course** – It teaches basic skills for scientific computing and running bootcamps. It also provides open-access teaching materials.
* Also, **NumPy** lets us deal with complex numerical calculations.

2. Software Development

Software developers make use of Python as a support language. They use it for build-control and management, testing, and for a lot of other things:

* **SCons** – for build-control
* **Buildbot, Apache Gump** – for automated and continuous compilation and testing
* **Roundup, Trac** – for project management and bug-tracking.
* Roster of Integrated Development Environments

3. Database Access

With Python, you have:

* Custom and ODBC interfaces to MySQL, Oracle, PostgreSQL, MS SQL Server, and others. These are freely available for download.
* Object databases like Durus and ZODB
* Standard Database API

4. Artificial Intelligence and Machine Learning

The IT industry went far beyond the science fiction ideas and our future is moving towards Artificial Intelligence. People are keen to move to technologies where machines can mimic human behaviours and most of the human work is being automated by machines. There is also a need to process and analyze huge volumes of data.

Statistically speaking that Python is the most popular language for Artificial intelligence and Machine Learning.

The several libraries and frameworks for Artificial intelligence are listed below.

* Tensorflow
* Scikit-learn
* NLTK
* Caffee
* PyTorch
* Keras
* Pandas
* Accord.NET

**FLASK:**

Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website. Flask is part of the categories of the micro-framework. Micro-framework are normally framework with little to no dependencies to external libraries. This has pros and cons. Pros would be that the framework is light, there are little dependency to update and watch for security bugs, cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins. Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks. Flask offers suggestions, but doesn't enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that make adding new functionality easy.

**Why is Flask a good web framework choice?**

Flask is considered more Pythonic than the Django web framework because in common situations the equivalent Flask web application is more explicit. Flask is also easy to get started with as a beginner because there is little boilerplate code for getting a simple app up and running.

Install Flask

Within the activated environment, use the following command to install Flask:

**$ pip install Flask**

**HTML:**

HTML stands for Hypertext Markup Language. It allows the user to create and structure sections, paragraphs, headings, links, and blockquotes for web pages and applications.HTML is not a programming language, meaning it doesn‘t have the ability to create dynamic functionality. Instead, it makes it possible to organize and format documents, similarly to Microsoft Word.When working with HTML, we use simple code structures (tags and attributes) to mark up a website page. For example, we can create a paragraph by placing the enclosed text within a starting *<p>* and closing *</p>* tag.

**<b> :-**Forbolding or otherwise styling text without conveying any additional meaningsuch as with <em> and <strong> etc.,

**<base>** :-Use to set a base URL.

**<body>** :-Establishes the body of an HTML document.

**<br> :-**Single line break.

**<select>:-** Used with <option> for selecting a particular option.

**<span>:-**Useful for applying styles to or around text, especially inline.

**<style>:-**For declaring style sheets within document.

**<table>:-**For making tables in a HTML document.

**<td>:-**Table data. Represents a data cellwithin a table.

**<th> :-**Table header cell.

**<title>:-** The title of an HTML document.

**<tr> :-**A row within a table.

**<textarea>:-**For multiline text entry.

**<u>:-** Useful for annotating text for various reasons.no longer specifically for underlining.

**<ul>**:- Unordered list. Use with <li> to make unordered list.

**CSS:-**CSS stands for Cascading Style Sheet.CSS describes how Html elements are to be displayed on screen ,paper ,or in other media.

**Font Properties :-**

**Font-Family** :-Changes the font family of certain words, sentences, paragraphs, etc.

**Font-Style** :- Changes text: normal, oblique, and italics.

**Font-Weight** :-Used to specify the weight of the font.

**Font-Size** :-Used to modify the size of the displayed font.

**Font**:- Used to combine all properties of fonts.

**Color and Background Properties :-**

**Color** :-Changes the color of text.

**Background-Color** :-Sets the background color of an element.

**Background-Image** :-Sets the background image of an element.

**Background-Repeat** :-Determines how a specified background image is repeated. The repeat-x value will repeat the image horizontally while the repeat-y value will repeat the image vertically.

**Background-Attachment** :-Determines if a specified background image will scroll with the content or be fixed with regard to the canvas.

**Background** :-Used to combine all properties of background.

**CHAPTER 4**

**SYSTEM ANALYSIS**

**4.1 Existing system**

Many of the existing systems have live chats through texts and some limitation such as there is no

instant response given to the patients they have to wait for experts acknowledgement for a long

time. Some of the processes may charge amount to live chat or telephony communication. However, the issue of these technologies are cost effective in clinical practice remains a consideration for future research studies.

Existing chatbots have limited availability of data and require some time for their self update.

This process leads to slower response times and expensive solutions.

Disadvantages in Existing system

* It takes more time to response to the user question
* Pay some charges to perform live chat

**4.2 Proposed System**

The chatbots are conversational virtual assistants which automate interactions with the users. Chatbots are powered by artificial intelligence using machine learning techniques to understand natural language. The main motive of the paper is to help the users with minor health information. Initially when the user visits the website first register themselves and later can ask the bot their queries. The system uses an expert system to answer the queries if the answer is not present in the database. Here the domain experts also should register themselves by giving various details. The data of the chatbot is stored in the database in the form of a pattern template. Here SQL is used for handling the database.

**CHAPTER 5**

**MODULE DESCRIPTION**

USER VALIDATION AND EXTRACTION OF SYMPTOMS:

The validation of the user login details occurs here. Then Symptoms are extracted using String Searching Algorithm where substring representing the symptoms is identified in the natural language text input. When users give directly the symptom name such as(e.g. ―I have cough, fever, and nausea‖), the system will easily identify it. But however, the system should also be able to handle input like, ―When I read, I‘m okay at first, but over time, my eyes seem to get tired, and I start to see double.‖ In this case, the system should extract substrings like ―eyes tired‖ and ―see double‖ (and not substrings like ―read‖ or ―okay‖).

MAPPING EXTRACTED SYMPTOMS WITH TRAINED DATASETS:

Given some extracted substring from the user’s input, we generate a list of suggested closest symptoms .We then ask the user to confirm if they have any of the suggested symptoms. Based on their reply few diseases are being shortlisted. Then further symptom clarification and symptom suggestions are being done by asking the users a series of questions and the mapping of the symptoms to the exact disease is done.

SPECIFYING THE DISEASE AND REFERRING A DOCTOR:

This process carries the list of diseases in the database and each symptom being entered is compared to the symptoms of the common diseases. Next symptom is checked until a matching one is found. The diseases are shortlisted based on the end users input on the question evaluation. The accurate disease is identified and specified to the end user by the chatbot. The chatbot checks whether the identified disease is a major issue or minor issue based on the conditions built in the chatbot. If it is a major issue the chatbot refers a specialist

**CHAPTER 6**

**IMPLEMENTATION/METHODOLOGY**

Natural language processing: Natural language processing (NLP) refers to a branch of AI -concerned with giving the computers the ability to understand text and spoken words in much same way human beings can. This includes, for example, the automatic translation of one language into another, but also spoken word recognition, or the automatic answering of questions. Computers often have trouble in understanding such tasks, because they usually try to understand the meaning of each individual word, rather than the sentence or phrase as a whole. So for a translation program, it can be difficult to understand the linguistic nuance in the word Greek when it comes to the examples My wife is Greek and It’s all Greek to me, for example through natural language processing, computers learn to accurately manage and apply overall linguistic meaning to text excerpts like phrases or sentences. But this isn’t just useful for translation or customer service chat bots: computers can also use it to process spoken commands or even generate audible responses that can be used in communication with the blind, for example, Summarizing long texts or targeting and extracting specific keywords and information within a large body of text also requires a deeper understanding of linguistic syntax than computers had previously been able to achieve.

How does natural language processing work?

It does not matter whether its processing an automatic translation or a conversation with a chat bot all natural language processing methods are the same in that they all involve understanding the hierarchies that dictate interplay between individual words. But this isn‘t easy – many words have different meaning ,Pass for example can mean a physical handover of something, a decision not to partake in something, and a measure of success in an exam or another test format. It also operates in the same conjugation as both a verb and a noun. The difference in meaning comes from the words that surround pass within the sentence or phrase (I passed the butter/on the opportunity/the exam).

These difficulties are the main reason that natural language processing is seen as one of the most complicated topics in computer science. Language is often littered with different meanings, so understanding the differences requires an extensive knowledge of the content in which the different meanings are used. Many users have first-hand experience of failed communication with chat bots due to their continued use as replacements for live chat support in customer service. But despite these difficulties, computers are improving their understanding of human language and its intricacies. To help speed this process up, computer linguists rely on the knowledge of various traditional linguistic fields: The term morphology is concerned with the interplay between words and their relationship with other words.

Syntax defines how words and sentences are put together. Semantics is the study of the meaning of words and groups of words. Pragmatics is used to explain the content of spoken expressions And lastly, phonology covers the acoustic structure of spoken language and is essential for language recognition.

We are using three algorithms to implement making health care chatbot using NLP technique

1.n-gram algorithm.

2.TF-IDF (term frequency-inverse data frequency).

3.Cosine similarity algorithm.

**N-gram algorithm**

In the fields of computational linguistics and probability, an n-gram is a contiguous sequence of n items from a given sample of text or speech. The item scan be phonemes, syllables, letters, words or base pairs according to the application. The n-grams typically are collected from a text or speech corpus. When the items are words, n-grams may also be called shingles Using Latin numerical prefixes, an n-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram" (or, less commonly, a "diagram"); size 3 is a "trigram". English cardinal numbers are sometimes used, e.g., "four-gram", "five-gram", and so on. In computational biology, a polymer or oligomer of a known size is called a k-mer instead of an n-gram, with specific names using Greek numerical prefixes such as "monomer", "dimer", "trimer", "tetramer", "pentamer", etc., or English cardinal numbers, "one-mer", "two-mer", "three-mer" etc.

**TF-IDF (term frequency-inverse data frequency):**

**Term frequency (tf ):**

Suppose we have a set of English text documents and wish to rank which document is most relevant to the query, "the brown cow". A simple way to start out is by eliminating documents that do not contain all three words "the", "brown", and "cow", but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document; the number of times a term occurs in a document is called its term frequency. However, in the case where the length of documents varies greatly, adjustments are often made (see definition below). The weight of a term that occurs in a document is simply proportional to the term frequency.

Often times, when building a model with the goal of understanding text, you‘ll see all of stop words being removed. Another strategy is to score the relative importance of words using TF-IDF.

The number of times a word appears in a document divided by the total number of words in the document. Every document has its own term frequency.



* Example 1: I am suffering from fever.
* Example 2: I am suffering from headache.
* Tf = number of occurrences/number of words in document

Example1: Example2:

* Tf(I)= 1/5=0.2 Tf(I)= 1/5=0.2
* Tf(am)=1/5=0.2 Tf(am)=1/5=0.2
* Tf(suffering)=1/5=0.2 Tf(suffering)=1/5=0.2
* Tf(from)=1/5=0.2 Tf(from)=1/5=0.2
* Tf(fever)=1/5=0.2 Tf(headache)=1/5=0.2

**Inverse Data Frequency (IDF):**

Because the term "the" is so common, term frequency will tend to incorrectly emphasize documents which happen to use the word "the" more frequently, without giving enough weight to the more meaningful terms "brown" and "cow". The term "the" is not a good keyword to distinguish relevant and non-relevant documents and terms, unlike the less-common words "brown" and "cow". Hence an *inverse document frequency* factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely.

The log of the number of documents divided by the number of documents that contain the word ***w***. Inverse data frequency determines the weight of rare words across all documents in the corpus.



Exmaple1: I am suffering from fever.

Example2: I am suffering from headache.

Idf=log(number of documents/number of documents that containing a word)

Idf(I) = log(2/2) = 0

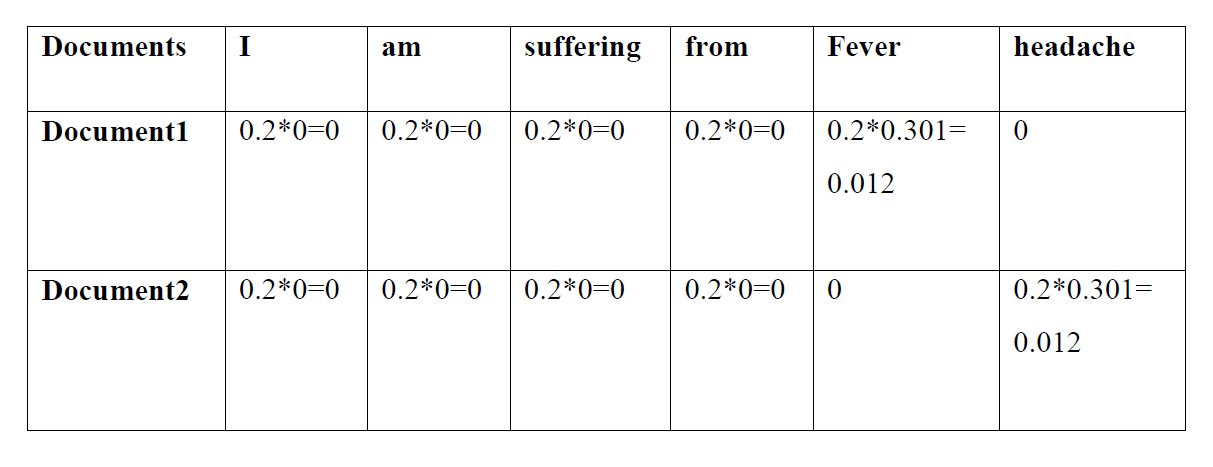
Idf(am) = log(2/2) = 0

Idf(suffering) = log(2/2) = 0

Idf(from) = log(2/2) = 0

Idf(fever) = log(2/1) = 0.301

Idf(headache)= log(2/1) = 0.301

**Tf -idf = Tf\*idf**

**Cosine similarity algorithm:**

Cosine similarity is been used to check the similarity between two sentences. The similarity between the query and document is directly proportional to the number of query weights. The similarity calculation result of the two documents ranges from 0 to 1 since the term frequency cannot be negative.it calculates similarity between two sentences using cosine formula The technique is also used to measure cohesion within clusters in the field of data mining. Cosine similarity=AB/|A||B|. Cosine distance is nothing but getting distance between two vectors in n dimension space. Distance represent how words are related to each other

**CHAPTER 7**

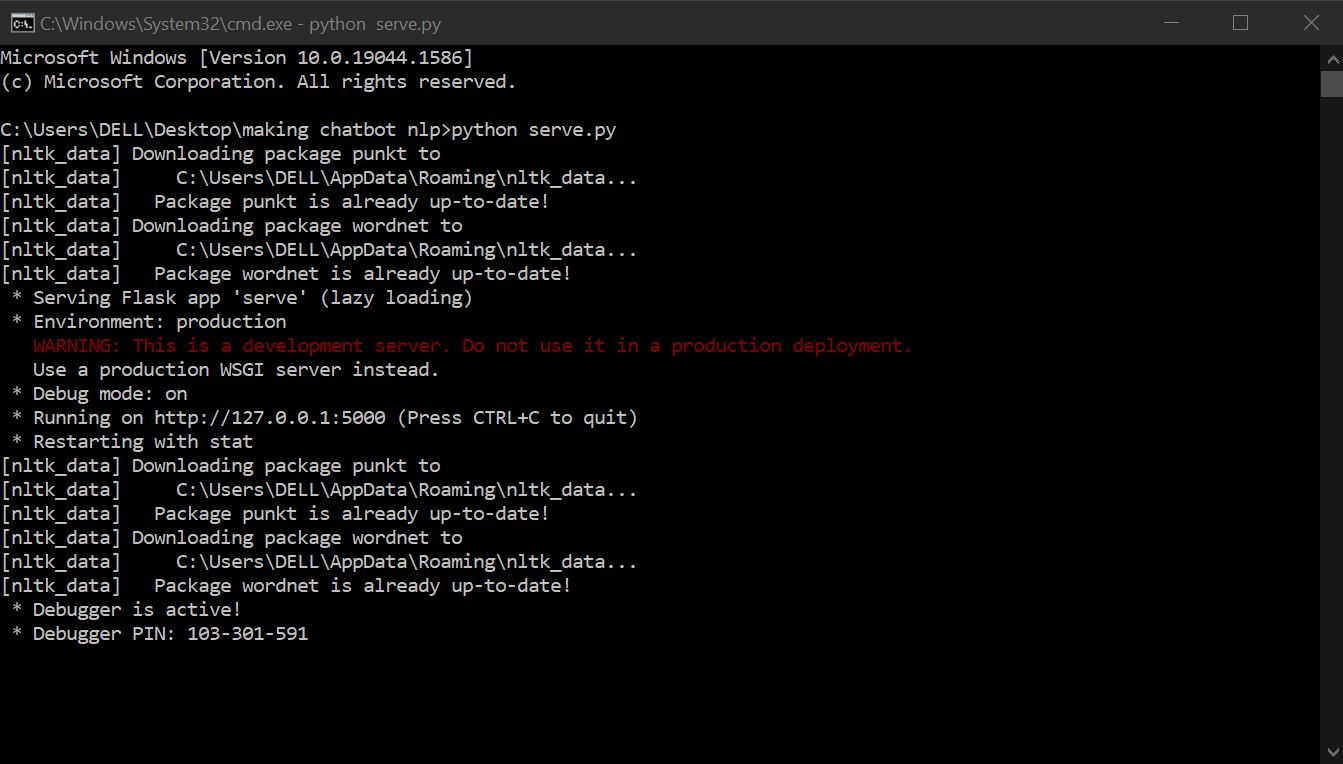
 **SOFTWARE TESTING**

Fig. Opening the file in command prompt

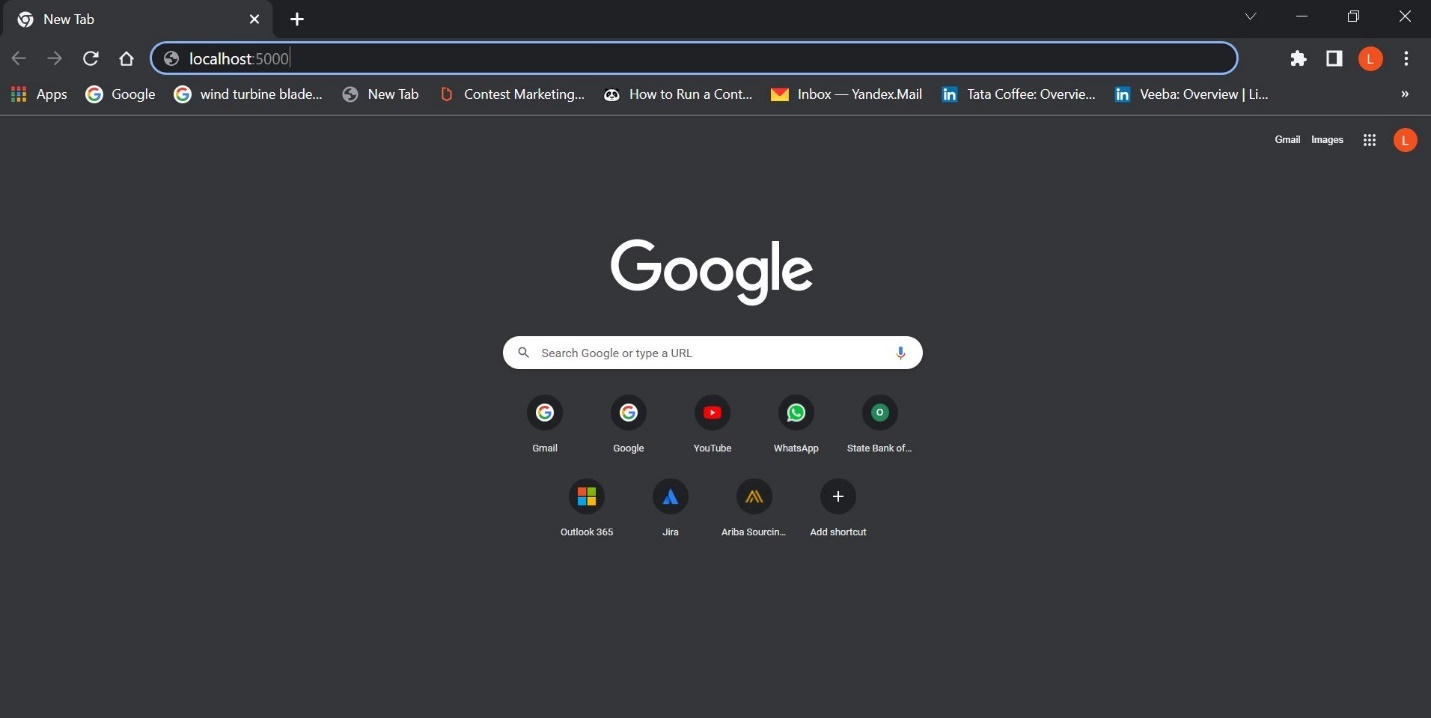


Fig. Running on chrome browser - localhost:5000

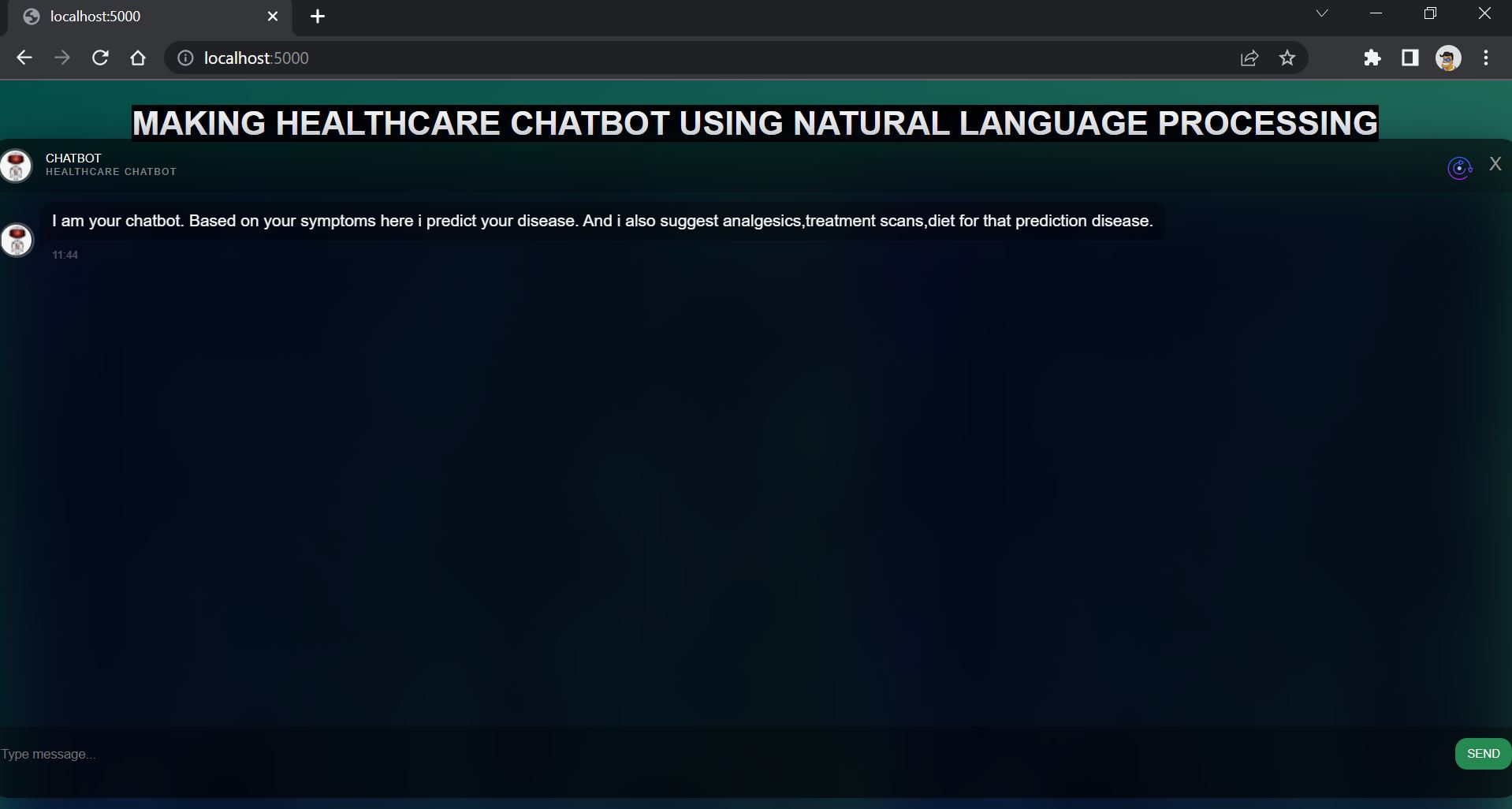


Fig. Chatbot User Interface

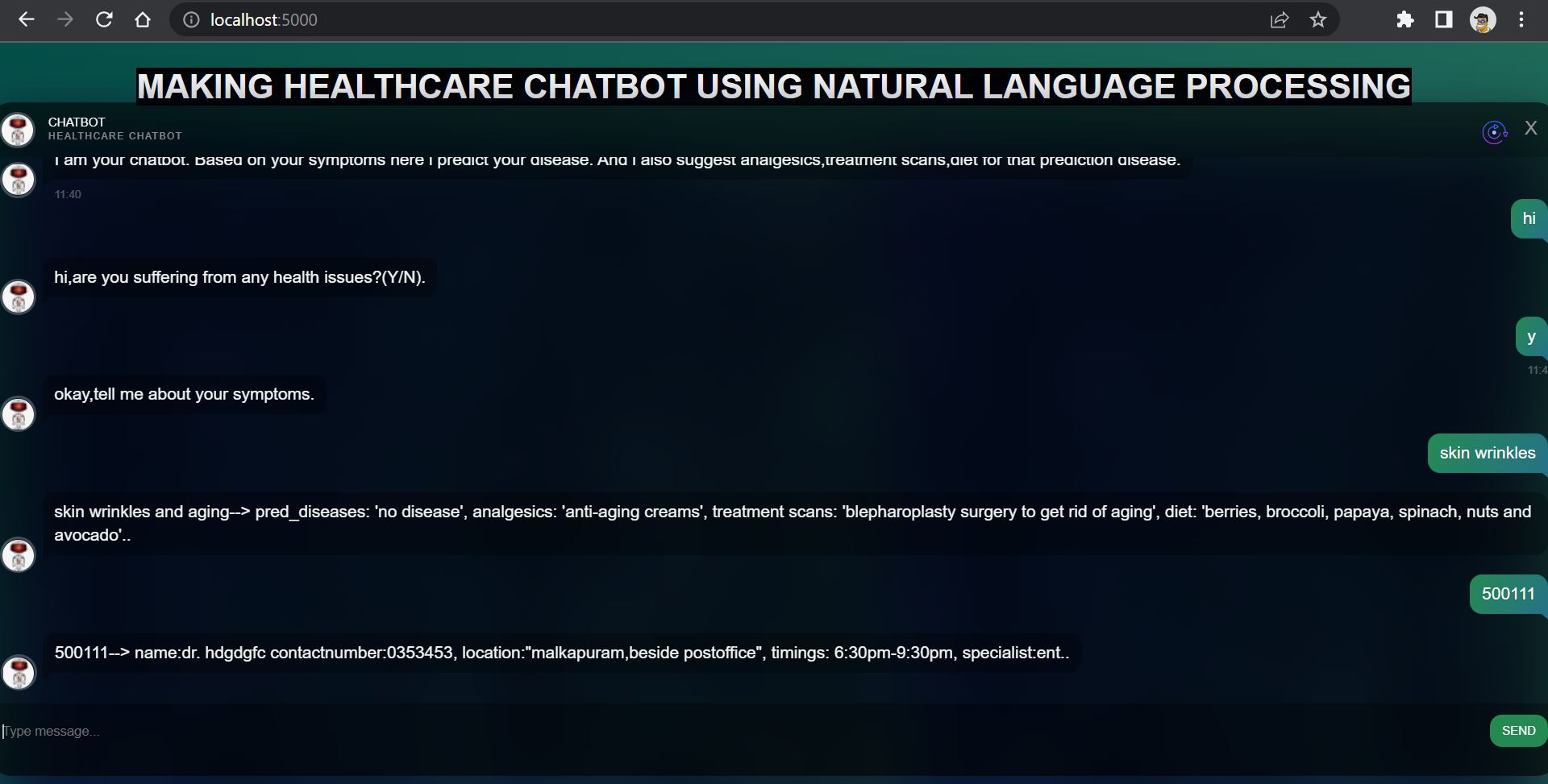


Fig. Chatbot Communication Page

**CHAPTER 8**

**CODE**

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

from bot2 import chat

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

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

def indexpage():

if request.method == "POST":

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

return render\_template("index2.html")

return render\_template("index2.html")

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

def entry():

req = request.get\_json()

print(req)

res = make\_response(jsonify({"name":"{}.".format(chat(req)),"message":"OK"}), 200)

return res

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

app.run(debug=True)

import nltk

import warnings

warnings.filterwarnings("ignore")

# nltk.download() # for downloading packages

#import tensorflow as tf

import numpy as np

import random

import string # to process standard python strings

f=open('symptom.txt','r',errors = 'ignore')

m=open('pincodes.txt','r',errors = 'ignore')

checkpoint = "./chatbot\_weights.ckpt"

#session = tf.InteractiveSession()

#session.run(tf.global\_variables\_initializer())

#saver = tf.train.Saver()

#saver.restore(session, checkpoint)

raw=f.read()

rawone=m.read()

raw=raw.lower()# converts to lowercase

rawone=rawone.lower()# converts to lowercase

nltk.download('punkt') # first-time use only

nltk.download('wordnet') # first-time use only

sent\_tokens = nltk.sent\_tokenize(raw)# converts to list of sentences

word\_tokens = nltk.word\_tokenize(raw)# converts to list of words

sent\_tokensone = nltk.sent\_tokenize(rawone)# converts to list of sentences

word\_tokensone = nltk.word\_tokenize(rawone)# converts to list of words

sent\_tokens[:2]

sent\_tokensone[:2]

word\_tokens[:5]

word\_tokensone[:5]

lemmer = nltk.stem.WordNetLemmatizer()

def LemTokens(tokens):

return [lemmer.lemmatize(token) for token in tokens]

remove\_punct\_dict = dict((ord(punct), None) for punct in string.punctuation)

def LemNormalize(text):

return LemTokens(nltk.word\_tokenize(text.lower().translate(remove\_punct\_dict)))

Introduce\_Ans = [" "]

GREETING\_INPUTS = ("hello", "hi","hiii","hii","hiiii","hiiii", "greetings", "sup", "what's up","hey",)

GREETING\_RESPONSES = ["hi,are you suffering from any health issues?(Y/N)", "hey,are you having any health issues?(Y/N)", "hii there,are you having any health issues?(Y/N)", "hi there,are you having any health issues?(Y/N)", "hello,are you having any health issues?(Y/N)", "I am glad! You are talking to me,are you having any health issues?(Y/N)"]

Basic\_Q = ("yes","y")

Basic\_Ans = "okay,tell me about your symptoms"

Basic\_Om = ("no","n")

Basic\_AnsM = "thank you visit again"

fev=("iam suffering from fever", "i affected with fever","i have fever","fever")

feve\_r=("which type of fever you have? and please mention your symptoms then we try to calculate your disease.")

# Checking for greetings

def greeting(sentence):

"""If user's input is a greeting, return a greeting response"""

for word in sentence.split():

if word.lower() in GREETING\_INPUTS:

return random.choice(GREETING\_RESPONSES)

# Checking for Basic\_Q

def basic(sentence):

for word in Basic\_Q:

if sentence.lower() == word:

return Basic\_Ans

def fever(sentence):

for word in fev:

if sentence.lower() == word:

return feve\_r

# Checking for Basic\_QM

def basicM(sentence):

"""If user's input is a greeting, return a greeting response"""

for word in Basic\_Om:

if sentence.lower() == word:

return Basic\_AnsM

# Checking for Introduce

def IntroduceMe(sentence):

return random.choice(Introduce\_Ans)

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.metrics.pairwise import cosine\_similarity

# Generating response

def response(user\_response):

robo\_response=''

sent\_tokens.append(user\_response)

TfidfVec = TfidfVectorizer(tokenizer=LemNormalize, stop\_words='english')

tfidf = TfidfVec.fit\_transform(sent\_tokens)

vals = cosine\_similarity(tfidf[-1], tfidf)

idx=vals.argsort()[0][-2]

flat = vals.flatten()

flat.sort()

req\_tfidf = flat[-2]

if(req\_tfidf==0):

robo\_response=robo\_response+"I am sorry! I don't understand you"

return robo\_response

else:

robo\_response = robo\_response+sent\_tokens[idx]

return robo\_response

# Generating response

# Generating response

def responseone(user\_response):

robo\_response=''

sent\_tokensone.append(user\_response)

TfidfVec = TfidfVectorizer(tokenizer=LemNormalize, stop\_words='english')

tfidf = TfidfVec.fit\_transform(sent\_tokensone)

vals = cosine\_similarity(tfidf[-1], tfidf)

idx=vals.argsort()[0][-2]

flat = vals.flatten()

flat.sort()

req\_tfidf = flat[-2]

if(req\_tfidf==0):

robo\_response=robo\_response+"I am sorry! I don't understand you"

return robo\_response

else:

robo\_response = robo\_response+sent\_tokensone[idx]

return robo\_response

def chat(user\_response):

user\_response=user\_response.lower()

keyword = " module "

keywordone = " module"

keywordsecond = "module "

if(user\_response!='bye'):

if(user\_response=='thanks' or user\_response=='thank you' ):

flag=False

#print("ROBO: You are welcome..")

return "You are welcome.."

elif(basicM(user\_response)!=None):

return basicM(user\_response)

else:

if(user\_response.find(keyword) != -1 or user\_response.find(keywordone) != -1 or user\_response.find(keywordsecond) != -1):

#print("ROBO: ",end="")

#print(responseone(user\_response))

return responseone(user\_response)

sent\_tokensone.remove(user\_response)

elif(greeting(user\_response)!=None):

#print("ROBO: "+greeting(user\_response))

return greeting(user\_response)

elif(user\_response.find("your name") != -1 or user\_response.find(" your name") != -1 or user\_response.find("your name ") != -1 or user\_response.find(" your name ") != -1):

return IntroduceMe(user\_response)

elif(basic(user\_response)!=None):

return basic(user\_response)

elif(fever(user\_response)!=None):

return fever(user\_response)

else:

#print("ROBO: ",end="")

#print(response(user\_response))

return response(user\_response)

sent\_tokens.remove(user\_response)

else:

flag=False

#print("ROBO: Bye! take care..")

return "Bye! take care.."

**SOUND OF THE INTERFACE:**

var synth = window.speechSynthesis;

// get all voices that browser offers

var available\_voices = window.speechSynthesis.getVoices();

// this will hold an english voice

var english\_voice = '';

// find voice by language locale "en-US"

// if not then select the first voice

for(var i=0; i<available\_voices.length; i++) {

if(available\_voices[i].lang === 'en-US') {

english\_voice = available\_voices[i];

break;

}

}

if(english\_voice === '')

english\_voice = available\_voices[0];

function textToSpeech(message) {

// new SpeechSynthesisUtterance object

var utter = new SpeechSynthesisUtterance();

utter.rate = 1;

utter.pitch = 0.5;

utter.text = message;

utter.voice = english\_voice;

// event after text has been spoken

///utter.onend = function() {

//alert('Speech has finished');

// }

// speak

window.speechSynthesis.speak(utter);

}

var $messages = $('.messages-content'),

d, h, m,

i = 0;

$(window).load(function() {

$messages.mCustomScrollbar();

setTimeout(function() {

$('<div class="message new"><figure class="avatar"><img src="/static/robo1.jpg" /></figure>' + 'I am your chatbot. Based on your symptoms here i predict your disease. And i also suggest analgesics,treatment scans,diet for that prediction disease.' + '</div>').appendTo($('.mCSB\_container')).addClass('new');

setDate();

updateScrollbar();

textToSpeech('welcome..' );

textToSpeech('I am your chatbot. Based on your symptoms here i predict your disease. And i also suggest analgesics,treatment scans,diet for that prediction disease.' );

}, 100);

});

function updateScrollbar() {

$messages.mCustomScrollbar("update").mCustomScrollbar('scrollTo', 'bottom', {

scrollInertia: 10,

timeout: 0

});

}

function setDate(){

d = new Date()

if (m != d.getMinutes()) {

m = d.getMinutes();

$('<div class="timestamp">' + d.getHours() + ':' + m + '</div>').appendTo($('.message:last'));

}

}

function insertMessage() {

msg = $('.message-input').val();

if ($.trim(msg) == '') {

return false;

}

$('<div class="message message-personal">' + msg + '</div>').appendTo($('.mCSB\_container')).addClass('new');

setDate();

$('.message-input').val(null);

updateScrollbar();

setTimeout(function() {

$('<div class="message loading new"><figure class="avatar"><img src="/static/robo1.jpg" /></figure><span></span></div>').appendTo($('.mCSB\_container'));

updateScrollbar();

fetch(`${window.origin}/entry`, {

method: "POST",

credentials: "include",

body: JSON.stringify(msg),

cache: "no-cache",

headers: new Headers({

"content-type": "application/json"

})

})

.then(function(response) {

if (response.status !== 200) {

console.log(`Looks like there was a problem. Status code: ${response.status}`);

return;

}

response.json().then(function(data) {

console.log(data);

$('.message.loading').remove();

$('<div class="message new"><figure class="avatar"><img src="/static/robo1.jpg" /> </figure>' + data.name + '</div>').appendTo($('.mCSB\_container')).addClass('new');

setDate();

updateScrollbar();

textToSpeech(data.name);

});

})

.catch(function(error) {

console.log("Fetch error: " + error);

});

}, 1000 + (Math.random() \* 20) \* 100);

}

$('.message-submit').click(function() {

insertMessage();

});

$(window).on('keydown', function(e) {

if (e.which == 13) {

insertMessage();

return false;

}

})

var Fake = [

'Hi im your chatbot ',

'please enter your name ',

'Please Enter Your age',

'good.....What is your comfortable level for investment loss (in %) <input type="range" value="50" min="0" max="100" step="10" />',

'we are Predicting... <div class="loading-img"><img src="5.png" alt=""/></div>',

'great.. do you want to predict another? <button class="buttonx sound-on-click">Yes</button> <button class="buttony sound-on-click">No</button> ',

'Bye',

':)'

]

function fakeMessage() {

msg = $('.message-input').val()

if (msg != '') {

return false;

}

$('<div class="message loading new"><figure class="avatar"><img src="/static/robo1.jpg" /></figure><span></span></div>').appendTo($('.mCSB\_container'));

updateScrollbar();

setTimeout(function() {

$('.message.loading').remove();

fetch(`${window.origin}/entry`, {

method: "POST",

credentials: "include",

body: JSON.stringify(msg),

cache: "no-cache",

headers: new Headers({

"content-type": "application/json"

})

})

.then(function(response) {

if (response.status !== 200) {

console.log(`Looks like there was a problem. Status code: ${response.status}`);

return;

}

response.json().then(function(data) {

console.log(data);

$('<div class="message new"><figure class="avatar"><img src="/static/robo1.jpg" /></figure>' + data.name + '</div>').appendTo($('.mCSB\_container')).addClass('new');

setDate();

updateScrollbar();

});

})

.catch(function(error) {

console.log("Fetch error: " + error);

});

i++;

}, 1000 + (Math.random() \* 20) \* 100);

}

**FONT STYLE:**

/\*--------------------

Mixins

--------------------\*/

/\*--------------------

Body

--------------------\*/

\*,

\*::before,

\*::after {

box-sizing: border-box;

}

html,

body {

height: 100%;

}

body {

background: -webkit-linear-gradient(315deg, #044f48, #2a7561);

background: linear-gradient(135deg, #044f48, #2a7561);

background-size: cover;

font-family: 'Open Sans', sans-serif;

font-size: 12px;

line-height: 1.3;

overflow: hidden;

}

.r-nav{ position: absolute;

right: 9px;

width: 100px;

top: 0px;

height: 45px;

}

.r-nav li {

list-style: none;

float: right;

display: inline-block;

width: 30px;

text-align: center;

}

.r-nav li a {

margin: 0;

padding: 0;

line-height: 18px;

font-size: 16px;

color: #979797;

}

.bg {

width: 100%;

height: 100%;

top: 0;

left: 0;

z-index: 1;

background: url(ba.jpg);

-webkit-filter: blur(30px);

filter: blur(30px);

}

.loading-img{

width: 100px;

display: inline;

margin: 10px 80px;

height: 100px;}

.buttonx {

background: #0070ff;;

padding: 5px 30px;

margin: 5px 5px 5px 0px;

border: 1px solid #0070ff;

color: #fff;

border-radius: 10px;

}

.buttony { background: transparent;

padding: 5px 30px;

margin: 5px 5px 5px 0px;

border: 1px solid #0070ff;

color: #0070ff;;

border-radius: 10px;

}

.oracle-search{width: 100%;

margin: 7px auto;

height: 29px;

padding: 5px;

font-size: 12px;

border-radius: 15px;

border: 1px solid rgba(255, 255, 255, 0.6);

color: #fff;

background: rgba(68, 68, 68, 0.47);}

/\*--------------------

Chat

--------------------\*/

input[type="range"] {

-webkit-appearance: none;

width: 340px;

height: 2px;

background: #0070ff;

background-position: center;

background-repeat: no-repeat;

position: absolute;

top: 0px;

bottoM: 0px;

left: 0px;

right: 0px;

margin: auto;

display: block;

margin-top: 60px;

}

input[type="range"]::-webkit-slider-thumb {

-webkit-appearance: none;

width: 20px;

height: 20px;

border-radius: 100%;

background: #434343;

position: relative;

border: 3px solid #0070ff;

z-index: 3;

cursor: pointer;

content: counter(3)

}

.chat {

position: absolute;

top: 53%;

left: 50%;

-webkit-transform: translate(-50%, -50%);

transform: translate(-50%, -50%);

width: 1300px;

height: 90vh;

max-height: 640px;

z-index: 2;

overflow: hidden;

box-shadow: 0 5px 30px rgba(0, 0, 0, 0.2);

background: rgba(0, 0, 0, 0.5);

border-radius: 20px;

display: -webkit-box;

display: -ms-flexbox;

display: flex;

-webkit-box-pack: justify;

-ms-flex-pack: justify;

justify-content: space-between;

-webkit-box-orient: vertical;

-webkit-box-direction: normal;

-ms-flex-direction: column;

flex-direction: column;

}

/\*--------------------

Chat Title

--------------------\*/

.chat-title {

-webkit-box-flex: 0;

-ms-flex: 0 1 45px;

flex: 0 1 45px;

position: relative;

z-index: 2;

background: rgba(0, 0, 0, 0.2);

color: #fff;

text-transform: uppercase;

text-align: left;

padding: 10px 10px 10px 50px;

}

.chat-title h1, .chat-title h2 {

font-weight: normal;

font-size: 10px;

margin: 0;

padding: 0;

}

.chat-title h2 {

color: rgba(255, 255, 255, 0.5);

font-size: 8px;

letter-spacing: 1px;

}

.chat-title .avatar {

position: absolute;

z-index: 1;

top: 8px;

left: 9px;

border-radius: 30px;

width: 30px;

height: 30px;

overflow: hidden;

margin: 0;

padding: 0;

border: 2px solid rgba(255, 255, 255, 0.24);

}

.chat-title .avatar img {

width: 100%;

height: auto;

}

/\*--------------------

Messages

--------------------\*/

.messages {

-webkit-box-flex: 1;

-ms-flex: 1 1 auto;

flex: 1 1 auto;

color: rgb(239, 245, 243);

overflow: hidden;

position: relative;

width: 100%;

}

.messages .messages-content {

position: absolute;

top: 0;

left: 0;

height: 101%;

width: 100%;

}

.messages .message {

clear: both;

float: left;

padding: 6px 10px 7px;

border-radius: 10px 10px 10px 0;

background: rgba(0, 0, 0, 0.3);

margin: 8px 0;

font-size: 14px;

line-height: 1.4;

margin-left: 35px;

position: relative;

text-shadow: 0 1px 1px rgba(0, 0, 0, 0.2);

}

.messages .message .timestamp {

position: absolute;

bottom: -19px;

font-size: 9px;

color: rgba(255, 255, 255, 0.3);

}

.messages .message::before {

content: '';

position: absolute;

bottom: -6px;

border-top: 6px solid rgba(0, 0, 0, 0.3);

left: 0;

border-right: 7px solid transparent;

}

.messages .message .avatar {

position: absolute;

z-index: 1;

bottom: -15px;

left: -35px;

border-radius: 30px;

width: 30px;

height: 30px;

overflow: hidden;

margin: 0;

padding: 0;

border: 2px solid rgba(255, 255, 255, 0.24);

}

.messages .message .avatar img {

width: 100%;

height: auto;

}

.messages .message.message-personal {

float: right;

color: #fff;

text-align: right;

background: -webkit-linear-gradient(330deg, #248A52, #257287);

background: linear-gradient(120deg, #248A52, #257287);

border-radius: 10px 10px 0 10px;

}

.messages .message.message-personal::before {

left: auto;

right: 0;

border-right: none;

border-left: 5px solid transparent;

border-top: 4px solid #257287;

bottom: -4px;

}

.messages .message:last-child {

margin-bottom: 30px;

}

.messages .message.new {

-webkit-transform: scale(0);

transform: scale(0);

-webkit-transform-origin: 0 0;

transform-origin: 0 0;

-webkit-animation: bounce 500ms linear both;

animation: bounce 500ms linear both;

}

.messages .message.loading::before {

position: absolute;

top: 50%;

left: 50%;

-webkit-transform: translate(-50%, -50%);

transform: translate(-50%, -50%);

content: '';

display: block;

width: 3px;

height: 3px;

border-radius: 50%;

background: rgba(255, 255, 255, 0.5);

z-index: 2;

margin-top: 4px;

-webkit-animation: ball 0.45s cubic-bezier(0, 0, 0.15, 1) alternate infinite;

animation: ball 0.45s cubic-bezier(0, 0, 0.15, 1) alternate infinite;

border: none;

-webkit-animation-delay: .15s;

animation-delay: .15s;

}

.messages .message.loading span {

display: block;

font-size: 0;

width: 20px;

height: 10px;

position: relative;

}

.messages .message.loading span::before {

position: absolute;

top: 50%;

left: 50%;

-webkit-transform: translate(-50%, -50%);

transform: translate(-50%, -50%);

content: '';

display: block;

width: 3px;

height: 3px;

border-radius: 50%;

background: rgba(255, 255, 255, 0.5);

z-index: 2;

margin-top: 4px;

-webkit-animation: ball 0.45s cubic-bezier(0, 0, 0.15, 1) alternate infinite;

animation: ball 0.45s cubic-bezier(0, 0, 0.15, 1) alternate infinite;

margin-left: -7px;

}

.messages .message.loading span::after {

position: absolute;

top: 50%;

left: 50%;

-webkit-transform: translate(-50%, -50%);

transform: translate(-50%, -50%);

content: '';

display: block;

width: 3px;

height: 3px;

border-radius: 50%;

background: rgba(255, 255, 255, 0.5);

z-index: 2;

margin-top: 4px;

-webkit-animation: ball 0.45s cubic-bezier(0, 0, 0.15, 1) alternate infinite;

animation: ball 0.45s cubic-bezier(0, 0, 0.15, 1) alternate infinite;

margin-left: 7px;

-webkit-animation-delay: .3s;

animation-delay: .3s;

}

/\*--------------------

Message Box

--------------------\*/

.message-box {

-webkit-box-flex: 0;

-ms-flex: 0 1 40px;

flex: 0 1 40px;

width: 100%;

background: rgba(0, 0, 0, 0.3);

padding: 10px;

position: relative;

}

.message-box .message-input {

background: none;

border: none;

outline: none !important;

resize: none;

color: rgba(255, 255, 255, 0.7);

font-size: 11px;

height: 40px;

margin: 0;

padding-right: 20px;

width: 1200px;

}

textarea {

font-family: sans-serif; /\* 1 \*/

font-size: 100%; /\* 1 \*/

line-height: 2; /\* 1 \*/

margin: 0; /\* 2 \*/

}

.message-box textarea:focus:-webkit-placeholder {

color: transparent;

}

.message-box .message-submit {

position: absolute;

z-index: 1;

top: 9px;

right: 10px;

color: #fff;

border: none;

background: #248A52;

font-size: 10px;

text-transform: uppercase;

line-height: 1.5;

padding: 6px 10px;

border-radius: 10px;

outline: none !important;

-webkit-transition: background .2s ease;

transition: background .2s ease;

}

.message-box .message-submit:hover {

background: #1D7745;

}

/\*--------------------

Custom Srollbar

--------------------\*/

.mCSB\_scrollTools {

margin: 1px -3px 1px 0;

opacity: 0;

}

.mCSB\_inside > .mCSB\_container {

margin-right: 0px;

padding: 0 10px;

}

.mCSB\_scrollTools .mCSB\_dragger .mCSB\_dragger\_bar {

background-color: rgba(0, 0, 0, 0.5) !important;

}

/\*--------------------

Bounce

--------------------\*/

@-webkit-keyframes bounce {

0% {

-webkit-transform: matrix3d(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

4.7% {

-webkit-transform: matrix3d(0.45, 0, 0, 0, 0, 0.45, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.45, 0, 0, 0, 0, 0.45, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

9.41% {

-webkit-transform: matrix3d(0.883, 0, 0, 0, 0, 0.883, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.883, 0, 0, 0, 0, 0.883, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

14.11% {

-webkit-transform: matrix3d(1.141, 0, 0, 0, 0, 1.141, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1.141, 0, 0, 0, 0, 1.141, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

18.72% {

-webkit-transform: matrix3d(1.212, 0, 0, 0, 0, 1.212, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1.212, 0, 0, 0, 0, 1.212, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

24.32% {

-webkit-transform: matrix3d(1.151, 0, 0, 0, 0, 1.151, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1.151, 0, 0, 0, 0, 1.151, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

29.93% {

-webkit-transform: matrix3d(1.048, 0, 0, 0, 0, 1.048, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1.048, 0, 0, 0, 0, 1.048, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

35.54% {

-webkit-transform: matrix3d(0.979, 0, 0, 0, 0, 0.979, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.979, 0, 0, 0, 0, 0.979, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

41.04% {

-webkit-transform: matrix3d(0.961, 0, 0, 0, 0, 0.961, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.961, 0, 0, 0, 0, 0.961, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

52.15% {

-webkit-transform: matrix3d(0.991, 0, 0, 0, 0, 0.991, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.991, 0, 0, 0, 0, 0.991, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

63.26% {

-webkit-transform: matrix3d(1.007, 0, 0, 0, 0, 1.007, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1.007, 0, 0, 0, 0, 1.007, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

85.49% {

-webkit-transform: matrix3d(0.999, 0, 0, 0, 0, 0.999, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.999, 0, 0, 0, 0, 0.999, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

100% {

-webkit-transform: matrix3d(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

}

@keyframes bounce {

0% {

-webkit-transform: matrix3d(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

4.7% {

-webkit-transform: matrix3d(0.45, 0, 0, 0, 0, 0.45, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.45, 0, 0, 0, 0, 0.45, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

9.41% {

-webkit-transform: matrix3d(0.883, 0, 0, 0, 0, 0.883, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.883, 0, 0, 0, 0, 0.883, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

14.11% {

-webkit-transform: matrix3d(1.141, 0, 0, 0, 0, 1.141, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1.141, 0, 0, 0, 0, 1.141, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

18.72% {

-webkit-transform: matrix3d(1.212, 0, 0, 0, 0, 1.212, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1.212, 0, 0, 0, 0, 1.212, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

24.32% {

-webkit-transform: matrix3d(1.151, 0, 0, 0, 0, 1.151, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1.151, 0, 0, 0, 0, 1.151, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

29.93% {

-webkit-transform: matrix3d(1.048, 0, 0, 0, 0, 1.048, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1.048, 0, 0, 0, 0, 1.048, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

35.54% {

-webkit-transform: matrix3d(0.979, 0, 0, 0, 0, 0.979, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.979, 0, 0, 0, 0, 0.979, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

41.04% {

-webkit-transform: matrix3d(0.961, 0, 0, 0, 0, 0.961, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.961, 0, 0, 0, 0, 0.961, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

52.15% {

-webkit-transform: matrix3d(0.991, 0, 0, 0, 0, 0.991, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.991, 0, 0, 0, 0, 0.991, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

63.26% {

-webkit-transform: matrix3d(1.007, 0, 0, 0, 0, 1.007, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1.007, 0, 0, 0, 0, 1.007, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

85.49% {

-webkit-transform: matrix3d(0.999, 0, 0, 0, 0, 0.999, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(0.999, 0, 0, 0, 0, 0.999, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

100% {

-webkit-transform: matrix3d(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

transform: matrix3d(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);

}

}

@-webkit-keyframes ball {

from {

-webkit-transform: translateY(0) scaleY(0.8);

transform: translateY(0) scaleY(0.8);

}

to {

-webkit-transform: translateY(-10px);

transform: translateY(-10px);

}

}

@keyframes ball {

from {

-webkit-transform: translateY(0) scaleY(0.8);

transform: translateY(0) scaleY(0.8);

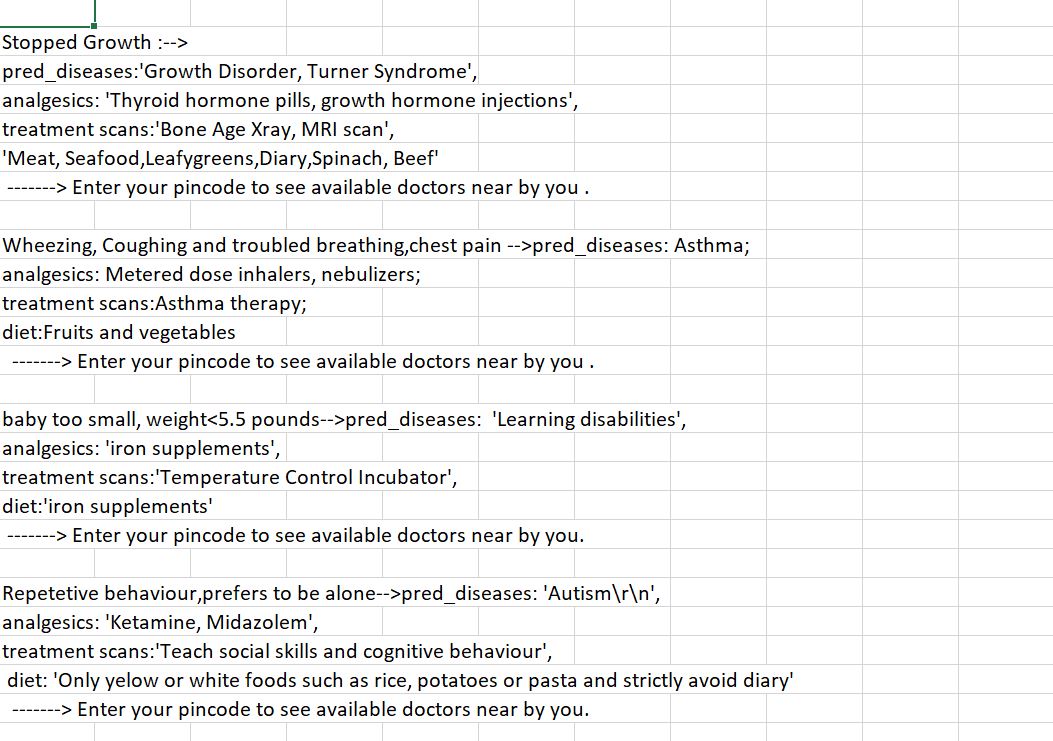
}

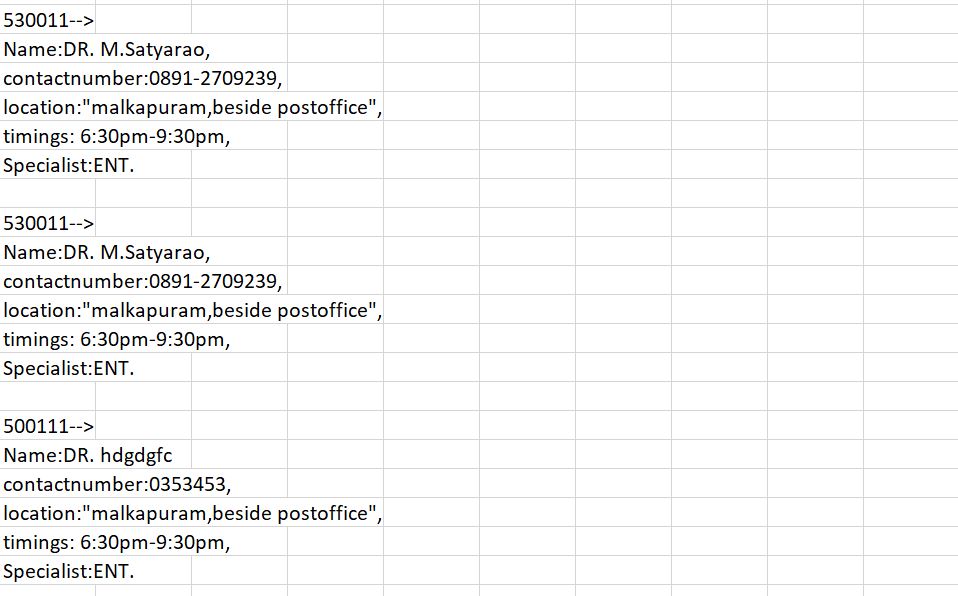
to {

-webkit-transform: translateY(-10px);

transform: translateY(-10px);

}}

**DATASETS:**



**CHAPTER 9**

**CONCLUSION**

Chatbot is great tool for conversation between human and machine. The application is developed for getting a quick response from the bot which means without any delay it gives the accurate result to the user.

It is concluded that, the usage of chatbot is user friendly and can be used by any person who knows how to type in their own language.Chatbot provides personalized diagnosis based on symptoms. From the review of various journals, it is concluded that, the usage of Chatbot is user friendly and can be used by any person who knows how to type in their own language in mobile app or desktop version. A medical chatbot provides personalized diagnoses based on symptoms. In the future, the bot‘s symptom recognition and diagnosis performance could be greatly improved by adding support for more medical features, such as location, duration, and intensity of symptoms, and more detailed symptom description. The implementation of Personalized Medical assistant heavily relies on AI algorithms as well as the training data. At last, the implementation of personalized medicine would successfully save many lives and create a medical awareness among the people. As said before, the future era is the era of messaging app because people going to spend more time in messaging app than any other apps. A text-to-text diagnosis Bot engages patients in conversation about their medical issues and provides a personalized diagnosis based on their symptoms. Hence, people will have an idea about their health and have the right protection.

Thus medical chatbot has wide and vast future scope. No matter how far people are, they can have this medical conversation. The only requirement they need is a simple desktop or smartphone with internet connection. The efficient of the chatbot can be improved by adding more combination of words and increasing the use of database so that of the medical chabot could handle all type of diseases. Even voice conversation can be added in the system to make it more easy to use.

**CHAPTER 10**

**REFERENCES**

1. Saurav Kumar Mishra, DhirendraBharti, Nidhi Mishra,‖ Dr.Vdoc: A Medical Chatbot that Acts as a Virtual Doctor‖, Journal of Medical Science and Technology ,Volume: 6, Issue 3,2017.
2. Hameedullah Kazi, B.S. Chowdhry,Zeesha Memon, ‖ MedChatBot: An UMLS based Chatbot for Medical Students‖, International Journal of Computer Applications (0975 – 8887)Volume 55– No.17, October 2016.
3. BenildaEleonor V. Comendador, Bien Michael B. Francisco, Jefferson S. Medenilla, Sharleen Mae T. Nacion, and Timothy Bryle E. Serac, ―Pharmabot: A 90 Pediatric Generic Medicine Consultant Chatbot ―,Journal of Automation and Control Engineering Vol. 3, No. 2, April 2015.
4. Abbas Saliimi Lokman, JasniMohamadZain,FakultiSistemKom puter, KejuruteraanPerisian,‖ Designing a Chatbot for Diabetic Patients‖,ACM Transactions on Management Information Systems (TMIS), Volume 4, Issue 2, August 2015 .