Medibot-Health Care Chatbot

A Minor Project Synopsis Submitted to

Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal
Towards Partial Fulfillment for the Award of

Bachelor of Technology
(Computer Science and Engineering)

Under the Supervision of
Prof . Gajendra Chauhan

Submitted By
Jatin Jangid
(0827CS201105)
Joshi Bansari
(0827CS201110)
Kritarth Jain
(0827CS201118)
Meghesh Solanki
(0827CS201137)

Department of Computer Science and Engineering
Acropolis Institute of Technology & Research, Indore
Jan-June 2023

EXAMINER APPROVAL

The Project entitled ***"Medibot-Healthcare Chatbot"*** submitted by **Jatin jangid (0827CS201105), Joshi Bansari (0827CS201110), Kritarth jain (0827CS201118), Meghesh Solanki (0827CS201137)** has been examined and is hereby approved towards partial fulfillment for the award of ***Bachelor of Technology degree in Computer Science Engineering*** discipline, for which it has been submitted. It understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project only for the purpose for which it has been submitted.

(Internal Examiner)

Date:

GUIDE RECOMMENDATION

This is to certify that the work embodied in this project entitled **"*Medibot-Health care Chatbot*"** submitted by **Jatin jangid (0827CS201105), Joshi Bansari (0827CS201110), Kritarth jain (0827CS201118), Meghesh Solanki (0827CS201137) is** a satisfactory account of the bonafide work done under the supervision of ***Prof. Gajendra Chauhan***, is recommended towards partial fulfillment for the award of the Bachelor of Technology (Computer Science Engineering) degree by Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal.

(Project Guide) **(Project Coordinator)**

STUDENTS UNDERTAKING

This is to certify that project entitled ***"Medibot-Health Care Chatbot"*** has been developedby us under the supervision of ***Prof. Gajendra Chouhan***. The whole responsibility of work done in this project is ours. The sole intension of thiswork is only for practical learning and research.

We further declare that to the best of our knowledge; this report does not contain any part of any work which has been submitted for the award of any degree either in this University or in any other University / Deemed University without proper citation and if the same work found then we are liable for explanation to this.

Jatin Jangid(0827CS201105)
Joshi Bansari (0827CS201110)
Kritarth jain(0827CS201118)
Meghesh Solanki (0827CS201137)

Acknowledgement

We thank the almighty Lord for giving us the strength and courage to sail out through the tough and reach on shore safely.

There are number of people without whom this projects work would not have been feasible. Their high academic standards and personal integrity provided us with continuous guidance and support.

We owe a debt of sincere gratitude, deep sense of reverence and respect to our guide and mentor **Dr. Kamal Kumar Sethi,** Professor, AITR, Indore for his motivation, sagacious guidance, constant encouragement, vigilant supervision and valuable critical appreciation throughout this project work, which helped us to successfully complete the project on time. We express profound gratitude and heartfelt thanks to **Dr Kamal Kumar Sethi**, Professor & Head CSE, AITR Indore for his support, suggestion, and inspiration for carrying out this project. We are very much thankful to other faculty and staff members of CS Dept, AITR Indore for providing us all support, help and advice during the project. We would be failing in our duty if do not acknowledge the support and guidance received from **Dr S C Sharma**, Director, AITR, Indore whenever needed. We take opportunity to convey our regards to the management of Acropolis Institute of Technology, Indore for extending academic and administrative support and providing us all necessary facilities for project to achieve our objectives.

We are grateful to **our parents** and **family members** who have always loved and supported us unconditionally. To all of them, we want to say "Thank you", for being the best family that one could ever have and without whom none of this would have been possible.

Jatin Jangid(0827CS201105)
Joshi Bansari (0827CS201110)
Kritarth jain(0827CS201118)
Meghesh Solanki (0827CS201137)

Executive Summary

Medibot-Health Care Chatbot

This project is submitted to Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal (MP), India for partial fulfillment of Bachelor of Engineering in Information Technology branch under the sagacious guidance and vigilant supervision of ***Prof. Gajendra Chouhan***.

Chatbots are intelligent conversational computer systems designed to mimic human conversation to enable automated online guidance and support. The increased benefits of chatbots led to their wide adoption by many industries in order to provide virtual assistance to customers. Chatbots utilize methods and algorithms from two Artificial Intelligence domains: Natural Language Processing and Machine Learning. However, there are many challenges and limitations in their application. In this survey we review recent advances on chatbots, where Artificial Intelligence and Natural Language processing are used. We highlight the main challenges and limitations of current work and make recommendations for future research investigation.

The project is based on Python programming language. Functions in python are a block of statements that return the specific task. Several functions are defined in backend through python to execute the Medibot project. Application Programming Interface also known as API is a communication layer that acts as an interface for different systems to talk to each other without understanding what each of them does separately. It describes everything a programmer needs to know about the code and how to use it. We have written code in API instead of implementation as it is more effective. Among large number of libraries in python, there is a library that is used for making API calls. It is written in pure Python and does not have any dependencies except for the Python Standard Library.

Key words: Python, API, Python Standard Library, chatbot; conversational agents; human-computer dialogue system; social chatbots; ChatScript; conversational modelling; conversation systems; conversational system; conversational entities; embodied conversational agents.

“Plans are only good intentions unless they immediately degenerate into hard work. ”

-Peter Drucker.”

List of Figures

Figure 1.1 14
Figure 2.1 17
Figure 3.1 22
Figure 3.2 23
Figure 3.3 23
Figure 3.4 24
Figure 3.5 25
Figure 3.6 25

List of Abbreviations

AI – Artificial Intelligence

AIML – Artificial Intelligence Modelling Language

NumPy – Numerical Python

API – Application Program Interface

NLP – Natural Language Processing

GUI – Graphical User Interface

Table of Contents

1) EXAMINER APPROVAL..... 2
2) GUIDE RECOMMENDATION..... 3
3) STUDENTS UNDERTAKING..... 4
4) ACKNOWLEDGEMENT..... 5
5) EXECUTIVE SUMMARY..... 6
6) LIST OF FIGURES..... 8
7) LIST OF ABBREVIATIONS..... 9

CHAPTER 1. INTRODUCTION 12

1.1 Overview **12**
1.2 Background and Motivation **12**
1.3 Problem Statement and Objectives **13**
1.4 Scope of the Project **13**
1.5 Team Organization **14**
1.6 Report Structure **14**

CHAPTER 2. REVIEW OF LITERATURE 16

2.1 Preliminary Investigation **17**
2.1.1 Current System **17**
2.2 Limitations of Current System **17**
2.3 Requirement Identification and Analysis for Project **18**
2.3.1 Conclusion **20**

CHAPTER 3. PROPOSED SYSTEM 21

3.1 The Proposal **21**
3.2 Benefits of the Proposed System..... **21**
3.3 Block Diagram **22**
3.4 Feasibility Study **22**

3.4.1 Technical 22
3.4.2 Economical..... 22
3.4.3 Operational 23
3.5 Design Representation..... 23
3.5.1 Use Case Diagram 23
3.5.2 Data Flow Diagram 24
3.5.3 Activity Diagram.... 25
3.6Requirement Specifications 26
3.6.1 Hardware 26
3.6.2 Software..... 26

CHAPTER 4. IMPLEMENTATION 27

4.1 Language Used 27
4.2 Libraries Used 28

CHAPTER 5. CONCLUSION..... 29

5.1 Conclusion 29
5.2 Limitations of the Work 30
5.3 Suggestion and Recommendations for Future Work 30

BIBLIOGRAPHY 31

SOURCE CODE 33

Chapter 1. INTRODUCTION

Introduction

CHATBOTS are artificially created virtual entities that interacts with users using interactive textual or speech skills. CHATBOTS directly chats with the people using artificial intelligence and Machine Learning concepts. This paper reviews the technique, terminology, and different platforms used to design and develop the MEDIBOT CHATBOT. It also presents some actual practical life typical applications and examples of CHATBOTS. Chatbot is Artificial Intelligence (AI) based conversational system that is able to process human language through various techniques including Natural Language Processing (NLP) and Neural Network (NN). Chatbots are also known as Artificial conversation entities, interactive agents, smart bots, and digital assistants.

1.1 Overview

This Project is based on Chatbots, also known as smart bots, interactive agents, digital assistants, or artificial conversation entities. Chatbots can mimic human conversation and entertain users but they are not built only for this. They are useful in applications such as education, information retrieval, business, and e-commerce

1.2 Background and Motivation

Productivity is the most important motivation for chatbot users, although different motivations include entertainment, social factors, and novelty interaction. Moreover, in business, chatbots have become so common because they reduce service costs and can handle many customers simultaneously.

Chatbots can automate tasks performed frequently and at specific times. This gives employees time to focus on more important tasks and prevents customers from waiting to receive responses. Proactive customer interaction.

A chatbot can motivate employees to complete digital learning content, and with the right AI-technology they can find new content that is relevant to the specific learner.

1.3 Problem Statement and Objectives

Chatbots helped in decreasing of human labor. Answered the questions all by itself **just like a human.** Chatbots can improve lead generation, qualification and nurturing. Used to further enhance human capabilities and free humans to be more creative and innovative, spending more of their time on strategic rather than tactical activities. This project aims in creating a chatbot that would answer to the questions asked in field of health and help save time and effort.

Our objective is to construct a Chatbot such that it will provide responses for a person instead of people needing to attend the problem themselves and it will answer based on the data provided by the user to avoid disturbance in many matters. An untrained instance of Chatterbot starts off with no knowledge of how to communicate. Each time a user enters a statement, the library saves the text that they entered and the text that the statement was in response to. As Chatterbot receives more input the number of responses that it can reply and the accuracy of each response in relation to the input statement increase. The program selects the closest matching response by searching for the closest matching known statement that matches the input, it then chooses a response from the selection of known responses to that statement.

1.4 Scope of the Project

The world's knowledge is vast. Bigger than the collections of all libraries. Larger than the works of all publishing companies combined. Greater than the Internet. Our chatbot summarizes knowledge, rather than try to contain all of it. Slogan "the sum of all human knowledge" is valid for our chatbot cause its goal is to provides introductions and overviews of notable subjects, to help alleviate the need to sift through all knowledge to understand the essentials. Once readers are familiar with the basics of a subject, and its jargon, they are better prepared to study and understand the rest of it. As its purpose is to benefit readers by acting as a widely accessible and free resource for users. The goal is to present a neutrally written summary of existing mainstream knowledge in a fair and accurate manner with a straightforward, "just-the-facts style". Interactive media appears to enhance student interest in developing and applying language skills. With young generation being involved on internet, it is a secure website as interactive stories are fun, engaging and appropriate for children.

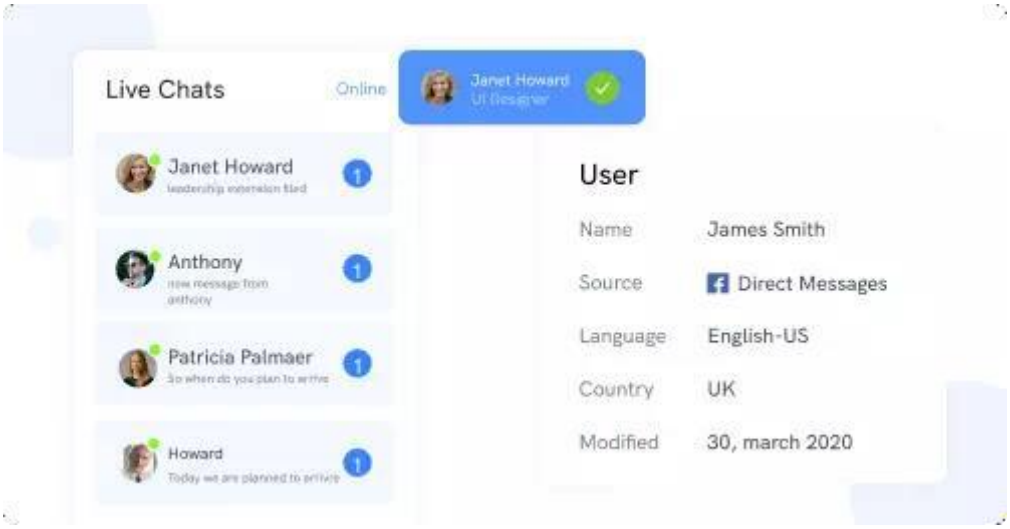

Figure 1.1 Live Chatbot

1.5 Team Organisation

Jatin Jangid: - Working in a team is a fantastic opportunity, and I am grateful that college provided me the opportunity to work on a project. I analyzed the project, and as a front-end developer, it's my responsibility to make the website eye-catching and highly interactive so users don't have any problems.

Joshi Bansari: -I am very excited and enthusiastic to work on it. I overview the topic, and as a backend developer, I have to create all the logic correctly and link the frontend, backend, and database successfully.

Kritarth Jain: - My role is to create a database to collect data and handle the documentation portion of the project while working with a fantastic team to get tremendous knowledge and expertise.

Meghesh Solanki: - I oversee the team and manage the backend and database integration for the project. I can state with some confidence that this experience will be useful to me in the future.

1.6 Report Structure

The project ***Medibot - Chatbot*** is primarily concerned with the **Artificial Intelligence (AI)** andwhole project report is categorized into five chapters.

Chapter 1: Introduction- introduces the background of the problem followed by rationale for the project undertaken. The chapter describes the objectives, scope and applications of the project. Further, the chapter gives the details of team members and their contribution in development of project which is then subsequently ended with report outline.

Chapter 2: Review of Literature- explores the work done in the area of Project undertaken and discusses the limitations of existing system and highlights the issues and challenges of project area. The chapter finally ends up with the requirement identification for present project work based on findings drawn from reviewed literature and end user interactions.

Chapter 3: Proposed System - starts with the project proposal based on requirement identified, followed by benefits of the project. The chapter also illustrate software engineering paradigm used along with different design representation. The chapter also includes block diagram and details of major modules of the project. Chapter also gives insights of different type of feasibility study carried out for the project undertaken. Later it gives details of the different deployment requirements for the developed project.

Chapter 4: Implementation - includes the details of different Technology/ Techniques/ Tools/ Programming Languages used in developing the Project. The chapter also includes the different user interface designed in project along with their functionality. Further it discusses the experiment results along with testing of the project. The chapter ends with evaluation of project on different parameters like accuracy and efficiency.

Chapter 5: Conclusion - Concludes with objective wise analysis of results and limitation of present work which is then followed by suggestions and recommendations for further improvement.

Chapter 2: REVIEW OF LITERATURE

Review of Literature

• Chatbots' History and Evolution: this aspect encompasses all papers that presented a detailed description of chatbots' evolution over time. This category is fundamental since it helped us understand the trends and technologies that ascended or discarded over time, indicating the evolution of the chatbot. It also helped us discover how and why chatbots emerged and how their applications and purposes changed over time.

• Chatbots' Implementation: this aspect includes papers that present examples of chatbots architectural design and implementation. This category allowed us to identify the commonly used algorithms for chatbots and the specific algorithms that are used for diverse types of chatbots based on the purpose of chatbot application. This also allowed to identify the industry standards in terms of chatbots' models and algorithms, as well as their shortcomings and limitations.

• Chatbots' Evaluation: For this aspect, some articles focused on the evaluation methods and metrics used for measuring chatbots performance. It was important to identify these papers in order to understand the way chatbots are evaluated and the evaluation metrics and methods used.

• Chatbots' Applications: this aspect encompasses all examples of chatbots applied to a specific domain, such as education, finance, customer support and psychology. Papers pertaining to this category helped us tie information from previous categories and get a better understanding of what models and what features are used for which applications in order to serve different purposes.

• Dataset: this category was used to classify chatbots depending on the dataset used to train machine learning algorithms for the development of language model.

These categories emerged from the literature review. The articles we reviewed covered one or more of these categories, often stated in the title or the abstract. It could be argued that these categories are widespread in the literature because they are strictly connected with chatbots development. Chatbot's application development requires, in fact, a study of different implementations and other applications (which are the result of chatbots' evolution

over time), as well as a dataset and an evaluation model. Any other aspect can be classified as a subcategory of one of these main categories.

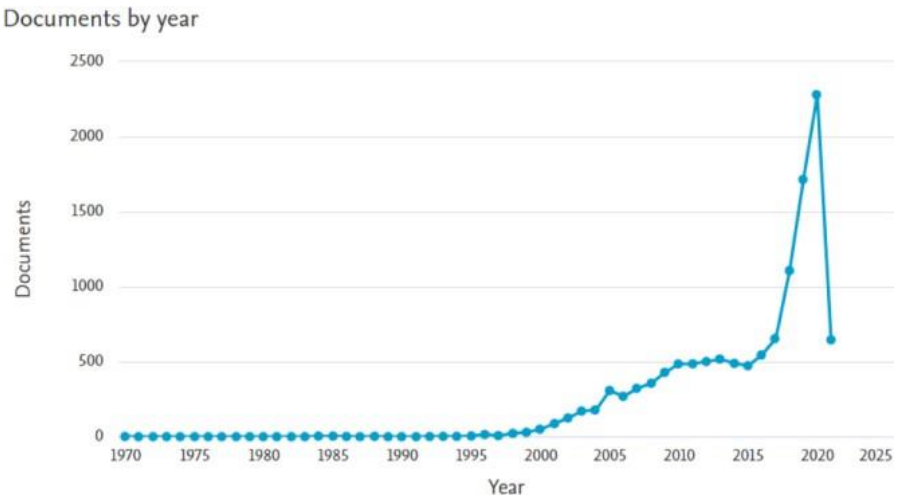

Figure 2.1 Search Results from Scopus, from 1970 to 2021

2.1 Preliminary Investigation

2.1.1 Current System

Information retrieval (IR) in computing and information science is the process of obtaining information system resources that are relevant to an information need from a collection of those resources. Searches can be based on full-text or other content-based indexing. Information retrieval is the science of searching for information in a document, searching for documents themselves, and also searching for the metadata that describes data, and for databases of texts, images or sounds.

2.2 Limitations of Current System

The limitations are as follows: -

- Currently many people depend on other humans to learn and gain knowledge about some particular topics or services. This process involves lot of time of others and ourselves as well and sometimes may become a costly source to have access to good and filtered knowledge.
- Communicating with boring plain receptionist is not fun for users making it a drawback of current system.

- Chatbots directly give response to problem without wasting any time of the client therefore most chatbots are used as personal assistants and secretaries instead of current slow system.
- Since chatbot is a software, it is scalable and can be easily updated automatically and keeps improving according to the updates of the system thus eliminating the constant role of a human and updating.
- Currently, chatbots are designed and developed for mobile messaging applications and mobiles keep updating frequently so scalable software.
- Chatbot help people explore online content and services giving knowledge more than any book or magazine.
- A human can understand a limited number of languages while electronic search engines can understand various languages.
- Current system is slow and languages and human dependent.

2.3 Requirement Identification and Analysis for Project

In this section, we will give an overview of chatbots' implementation methods. We will distinguish between Rule-based chatbots, and Artificial Intelligence (AI) based chatbots. Within AI-based chatbots, we will further distinguish among Information-Retrieval chatbots and Generative Chatbots.

Rule-Based Chatbots: - The very first attempts at chatbots' implementation were rule-based. Rule-based models are usually easier to design and to implement, but are limited in terms of capabilities, since they have difficulties answering complex queries. Rule-based chatbots answer users' queries by looking for patterns matches; hence, they are likely to produce inaccurate answers when they come across a sentence that does not contain any known pattern. Furthermore, manually encoding pattern matching rules can be difficult and time consuming. Furthermore, pattern matching rules are brittle, highly domain specific, and do not transfer well from one problem to the other.

Artificial Intelligence Chatbots: - AI models, contrary to Rule-based models, are based on Machine Learning algorithms that allow them to learn from an existing database of human conversations. In order to do so, they need to be trained through Machine Learning algorithms that can train the model using a training dataset. Through the use of Machine Learning algorithms, there is no longer the need to manually define and code new pattern matching rules, which allows chatbots to be more flexible and no longer dependent on domain specific knowledge. As stated, AI models can be further categorised into Information Retrieval based models and Generative models.

Information Retrieval Models: - Information Retrieval based models are designed so that given a dataset of textual information, the algorithm will be capable of retrieving the information needed based on the user's input. The algorithm used is usually a Shallow Learning algorithm; nonetheless, there are also cases of Information Retrieval models that use Rule-based algorithms and Deep Learning ones. Information Retrieval based models include a pre-defined set of possible answers; the chatbot processes the user query and based on this input it picks one of the answers available in its set. The knowledge base for this kind of model is usually formed by a database of question-answer pairs. A chat index is constructed from this database, in order to list all the possible answers based on the message that prompted them. When the user provides the chatbot with an input, the chatbot treats that input as a query, and an Information Retrieval model akin to those used for web queries is used to match the user's input to similar ones in the chat index. The output returned to the user is thus the answer paired with the selected question among those present in the chat index. The main advantage of this model is that it ensures the quality of the responses since they are not automatically generated. These models have seen a surge in popularity with the advent of the Web 2.0 and the increase in available textual information that could be retrieved on social media platforms, forums, and chats. One of the main downsides of this approach is that creating the necessary knowledge base can be costly, time-consuming, and tedious. Furthermore, if the great volume of data available provides for a greater training set and a wider knowledge base, it also implies it will be all the more challenging to match a user's input to the correct answer; a significant amount of time and resources must be deployed to train the system to select one of the correct answers available.

Generative Models: - Generative based models, as the name suggests, generate new responses word by word, based on the input of the user. These models are thus able to create entirely new sentences to respond to users' queries; however, they need to be trained in order to learn sentence structure and syntax, and the outputs can somewhat lack in quality or consistency. Generative models are usually trained on a large dataset of natural phrases issued from a conversation. The model learns sentence structure, syntax, and vocabulary through the data that it has been fed. The overall aim is for the algorithm to be able to generate an appropriate, linguistically correct response based on the input sentence. This approach is usually based on a Deep Learning Algorithm composed of an Encoder-Decoder Neural Network model with Long-Short-Term-Memory mechanisms to counterbalance the vanishing gradient effect present in vanilla Recurrent Neural Networks.

This model does offer some interesting advantages. First, it does not involve domain specific knowledge, but is rather an end-to-end solution that can be trained using different datasets, thus on different domains. Furthermore, although the model does not need domain-specific knowledge to provide valuable results, it can be adapted to work with other algorithms if

further analysis on domain-specific knowledge is needed. It is thus a simple yet widely general and flexible model that can be used to solve different NLP tasks. For these reasons, the Sequence-to-Sequence model seems to have become the industry standard choice for dialogue generation and many NLP tasks in recent years.

Nonetheless, it has a considerable limit: the entirety of the information contained in the input sentence must be encoded in a fixed length vector, the context vector, and thus, the longer the sentence, the more information gets lost in the process. That is why Sequence to Sequence models do not perform well when they must respond to longer sentences and tend to give vague answers. Furthermore, when generating an answer, these models tend to focus on a single response, which creates a lack of coherence in the turns of a conversation.

2.3.1 Conclusion

This chapter reviews the literature surveys that have been done during the research work. The related work that has been proposed by many researchers has been discussed. The research papers related to object detection and recognition of objects from 1970 to 2021 have been shown which discussed about different methods and algorithm to identify objects.

Chapter 3: PROPOSED SYSTEM

Proposed System

3.1 The Proposal

The proposal is to construct a Chatbot such that it will provide responses for a person instead of people needing to attend the problem themselves and it will answer based on data provided by the user to avoid disturbance in so many matters. A Chatterbot has no knowledge of how to communicate. Every time a user enters a statement, library saves the text that they entered and the text that the statement was in response to. As a Chatterbot receives more input, the number of responses that it can reply and the accuracy of each response in relation to the input statement, increases. The program selects the closest matched response by searching for the closest matching known statement which matches the input, it then chooses the response from the selection of known responses to that statement.

3.2 Benefits of the Proposed System

- **Can hold multiple conversations at once**. Chatbots can converse simultaneously with thousands of buyers. This increases business productivity and eliminates wait times.
- **Cost effective**. A chatbot is a faster and cheaper one-time investing and creating a dedicated, cross-platform app or hiring additional employees. In addition, chatbots can costly problems caused by human error. User acquisition costs also decrease with a chatbot's ability to responds within seconds.
- **Saves time.** Chatbots can automate tasks performed frequently and at specific times. This gives employees time to focus on more important tasks and prevents customers from waiting to receive responses.
- **Proactive customer interaction.** In the past, organizations relied on passive customer interaction and waited for buyers to reach out first. With chatbots, organizations can interact proactively, as bots can initiate conversations and monitor how customers use the websites and landing pages.
- **Eases scalability and to local markets**. Chatbots can solve customer concerns and queries in multiple languages. Their 24/7 access enables customers to use them regardless of time or time zone.

- **Monitors and analyzes consumer data**. Chatbots collect feedback from each interaction to help business improve their services and products or optimize their websites. Bots can also record user data to track behavior and purchasing patterns.

3.3 Block Diagram

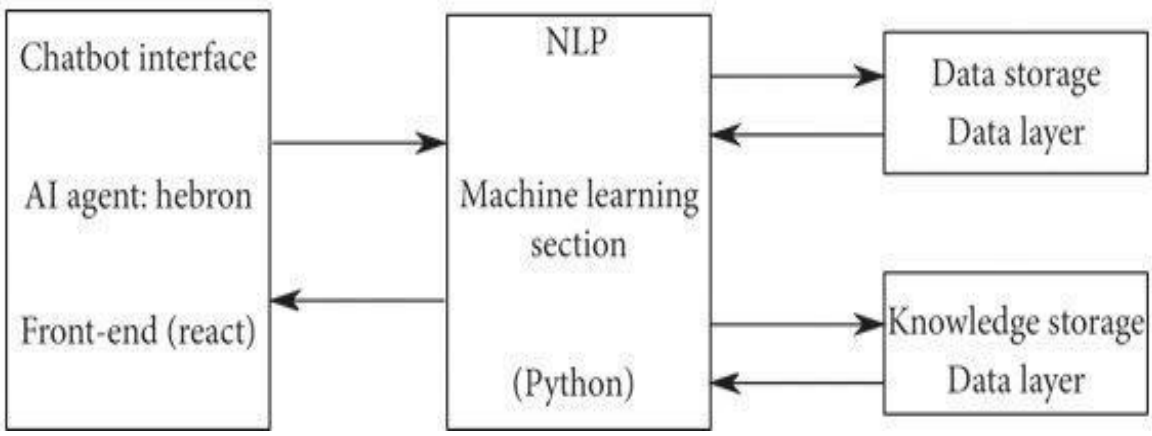

Figure 3.1

3.4 Feasibility Study

A feasibility study is an analysis of how successfully a system can be implemented, accounting for factors that affect it such as economic, technical and operational factors to determine its potential positive and negative outcomes before investing a considerable amount of time and money into it.

3.4.1 Technical

Artificial intelligence chatbots employ **AI and natural language processing (NLP) technology** to recognize sentence structure, interpret the knowledge, and improve their ability to answer questions.

3.4.2 Economical

The chatbot is primarily an automation tool that can accomplish a large number of tasks alone if it is connected to the internal tools of the company. The cost savings from reducing the time spent on inquiries are therefore huge!

In customer support or sales tools, the chatbot has the ability to integrate perfectly into the life of the company and to give time to resources usually mobilized on these tasks, to focus on more qualitative and complex missions.

3.4.3 Operational

The main motto of our system is to reduce the manual efforts and time of people by automating it. The system is able to do that accurately and efficiently making the system operationally feasible.

3.5 Design Representation

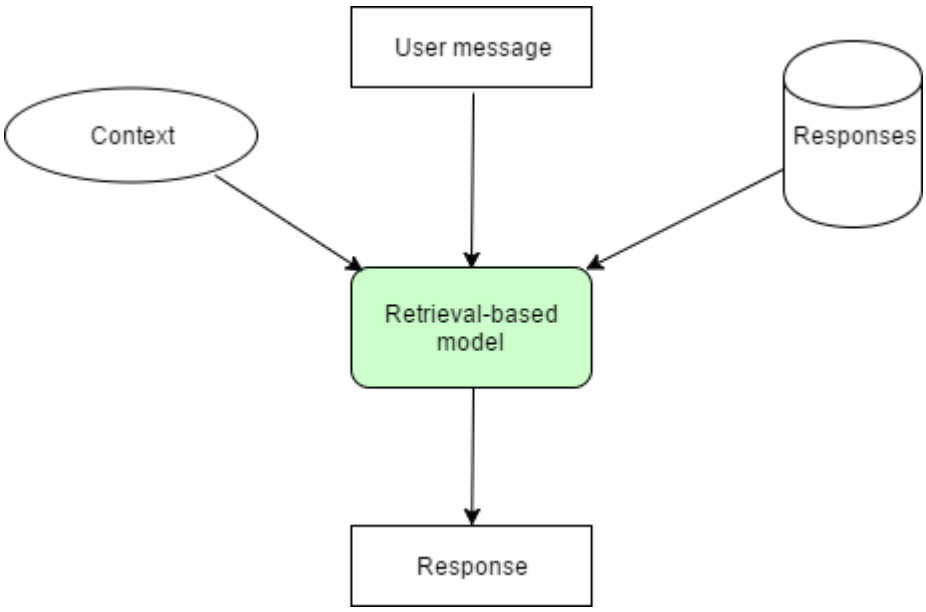

Figure 3.2

3.5.1 Use Case Diagram

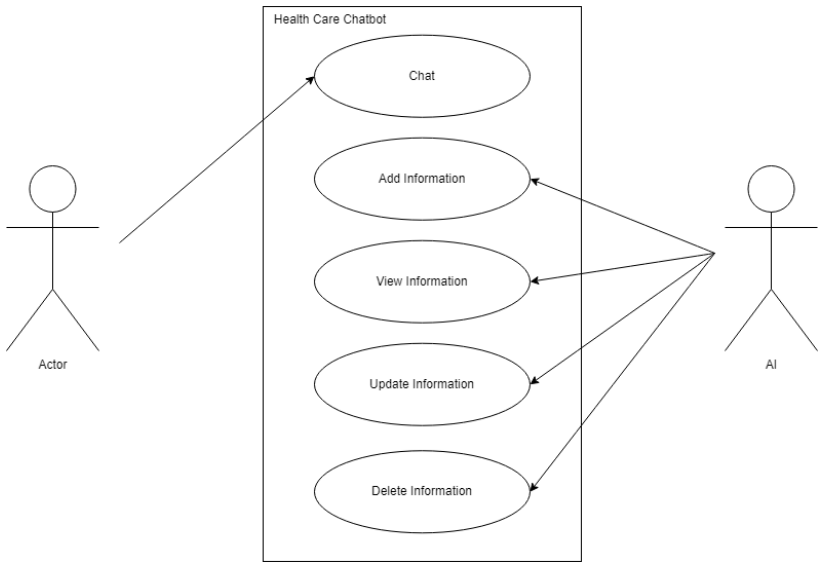

Figure 3.3

Actor	Description
User of Medibot Chatbot	User asks question to the chatbot or plays interactive story game with it.
Medibo t Chatbot	Chatbot asks user his name and and fetches result from Wikipedia through API of the question asked and plays story game with the user.

Table 1: actor table for MEDIBOT CHATBOT

#	Use Case	Description
UC1	Asks Question	User asks question on the topic he needs information.
UC2	Plays Interactive Game	User plays interactive games with the chatbot.
UC3	Asks Name	Initially, chatbot asks the name of the user.
UC4	Asks for Selection	Chatbot asks user to make selection whether user wants to asks a question or play game.
UC5	Gives Result	Chatbot fetches information and gives result.

Table 2: Use cases for MEDIBOT CHATBOT

3.5.2 Data - Flow Diagrams

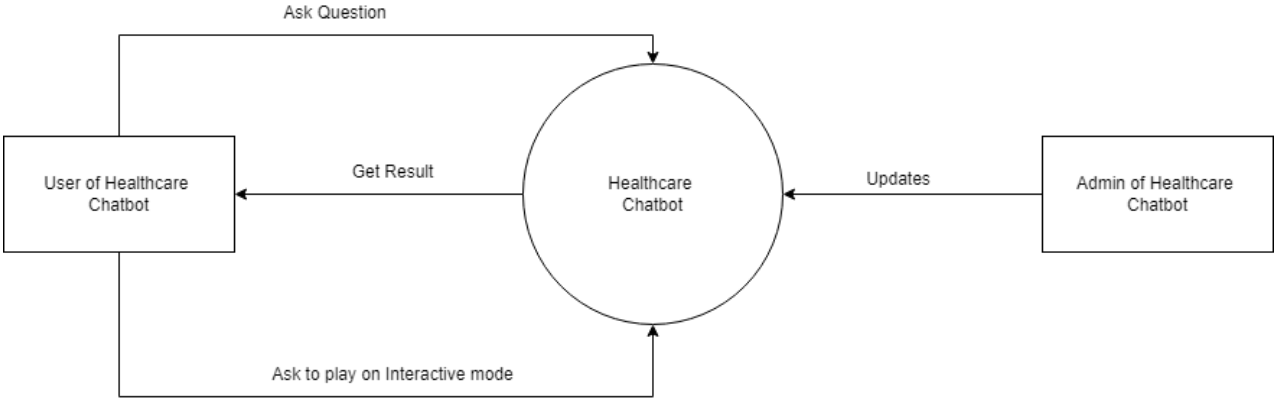

Figure 3.4

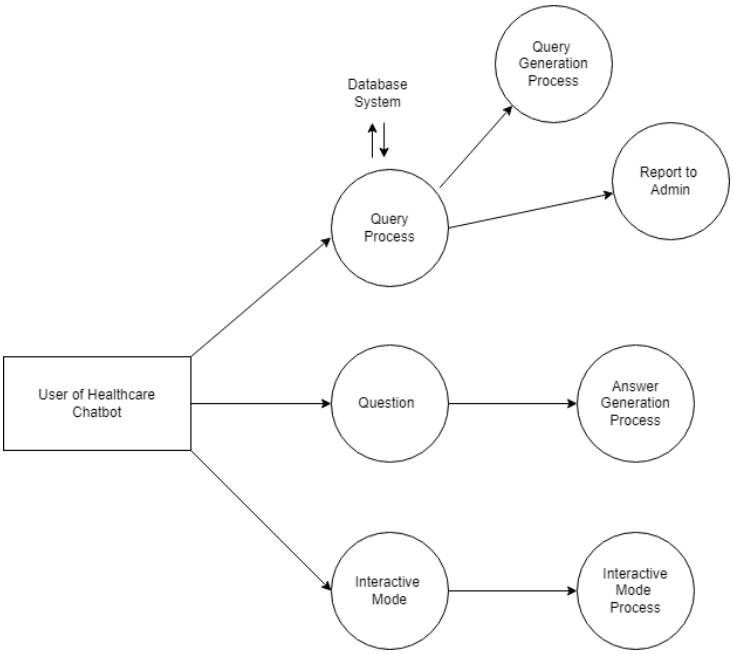

Figure 3.5

3.5.3 Sequence Diagram

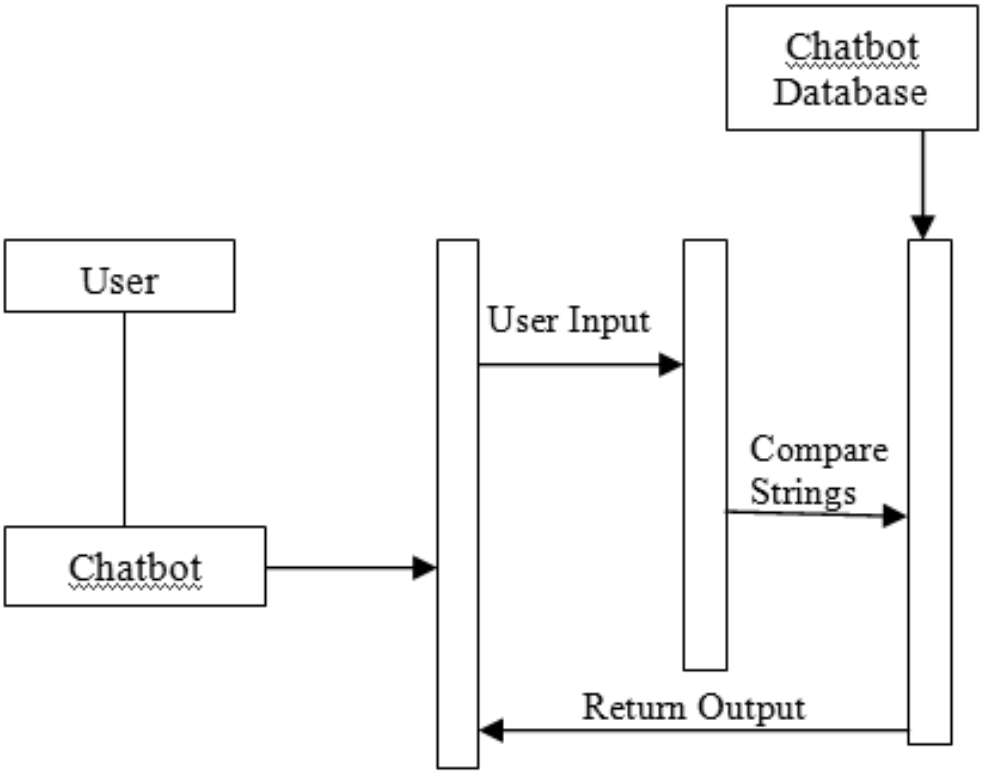

Figure 3.6

3.6 Requirement Specifications

3.6.1 Hardware Specifications: -

- Processor: Intel core i5
- RAM: 2GBs
- Hard Disk: 500GB (Minimum 80 GBs)

3.6.2 Software Specifications: -

- Operating System: Windows 11
- Front – End: Atom
- Back – End: VS Code

Chapter 4: Implementation

Implementation

MEDIBOT CHATBOT is an interactive chatbot that gives output of a Wikipedia page when asked about some information or question. It has basic functionalities that are possessed by almost every chatbot like, 'Hello, what's your name?', 'Goodbye', 'Thank You' etc. It has a unique and very cool feature of interactive story where there are both male and female versions of story and user can select the option of male or female. It initially starts by asking 1,2as to whathas to done where 1 is Wikipedia general question and 2 is interactive story mode. And redirectsthe user accordingly. For example, below are some code screenshots that tell a lot about our MEDIBOT CHATBOT.

4.1 Language Used

Python language is used in the system due to the following Characteristics:

Simple:

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English (but very strict English!). This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the syntax i.e., the language itself.

Free and Open Source:

Python is an example of a FLOSS (Free/Libre and Open-Source Software). In simple terms, you can freely distribute copies of this software, read the software's source code, make changes to it, use pieces of it in new free programs, and that you know you can do these things. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and improved by a community who just want to see a better Python.

Object Oriented:

Python supports procedure-oriented programming as well as object-oriented programming. In procedure-oriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and functionality. Python has a very powerful but simple

way of doing object-oriented programming, especially, when compared to languages like C++ or Java.

Extensive Libraries:

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, ftp, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces) using Tk, and also other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the "batteries included" philosophy of Python.

4.2 Libraries Used

- pandas
- pyttax3
- sklearn
- sklearn.tree
- numpy
- sklearn.model_selection
- train_test_split
- cross_val_score
- csv
- warning

Chapter 5: CONCLUSION

Conclusion

5.1 Conclusion

In this paper we have provided a survey of relevant works of literature on the subject, and we have analyzed the state of the art in terms of language models, applications, datasets used, and evaluation frameworks. We have also underlined current challenges and limitations, as well as gaps in the literature. Despite technological advancements, AI chatbots are still unable to simulate human speech. This is due to a faulty approach to dialogue modeling and a lack of domain-specific data with open access. For Information Retrieval chatbots, there is also a lack of a learnt AI model. A model like this might be used in a variety of sectors. There is still a gap to be closed in terms of applications between industry models and current advancements in the sector. Large models necessitate a lot of computing power and a lot of training data. There is no universal framework for evaluating chatbots. Several models depend on human evaluation, yet human evaluation is expensive, time-consuming, difficult to scale, biased, and lacks coherence. A new, reliable automatic evaluation approach should be provided to overcome these restrictions. Furthermore, recent studies have revealed a scarcity of data on the most recent developments in language models that may be used to chatbots like Transformers. As a result, it's critical to examine and analyze the data used to train the various models. This type of study provides for a more accurate comparison of different models and their results. In fact, the distinction between chatbots' applications and social or companion chatbots appears to be hazy. Chatbot modeling is a fascinating challenge that mixes Deep Learning and Natural Language Processing. Despite the fact that the first chatbots were created sixty years ago, the area has continued to grow and provide new and exciting problems. To bridge these gaps, smaller, flexible, less domain dependent models would be beneficial. Improved, scalable, and flexible language models for industry specific applications, more human-like model architectures, and improved evaluation frameworks would surely represent great steps forward in the field.

In this paper, MEDIBOT CHATBOT is reviewed. Chatbots are a new paradigm on how people interactvirtually with a software to get their problems solved. Chatbots are a necessary part of our lives asthey make it easy (as discussed above). Some of the already existing chatbot systems are discussed in the paper. It is discussed about the functionalities of MEDIBOT CHATBOT like what kind of libraries are used. API and socket programming are also discussed. Design of MEDIBOT is also given

through level0 and level1 DFD (data flow diagram) and Activity diagram. Working and unique personality of MEDIBOT is well explained in the paper. Since Artificial Intelligence concepts are used in CHATBOT, it is the best alternative to solve query processing problems faced by the user on daily basis with colourful and fun interface. This project may be helpful to generate new insightsand guide future chatbot that are to be designed, implemented, developed.

5.2 Limitations of the Work

- Chatbots Don't Understand Human Context.
- They Don't Do Customer Retention.
- They Can't Make Decisions.
- Exorbitant Installation.
- Chatbots Have the Same Answer for a Query.

5.3 Suggestion and Recommendations for Future Work

- We also tried to add here tracking of the current weather but had difficulties executing it.
- Also, audio could've been enabled in our chatbot.
- Linguistic and conversational abilities must be improved.
- System might fail to recognize similar questions phrased in different ways, even within the same conversation.

BIBLIOGRAPHY

1. Jia, J. The Study of the Application of a Keywords-based Chatbot System on the Teaching of Foreign Languages. arXiv 2003, arXiv:cs/0310018.

2. Sojasingarayar, A. Seq2Seq AI Chatbot with Attention Mechanism. Master's Thesis, Department of Artificial Intelligence, IA School/University-GEMA Group, Boulogne-Billancourt, France, 2020.

3. Bala, K.; Kumar, M.; Hulawale, S.; Pandita, S. Chat-Bot For College Management System Using A.I. Int. Res. J. Eng. Technol. (IRJET) 2017, 4, 4.

4. Ayanouz, S.; Abdelhakim, B.A.; Benhmed, M. A Smart Chatbot Architecture based NLP and Machine Learning for Health Care Assistance. In Proceedings of the 3rd International Conference on Networking, Information Systems & Security, Marrakech, Morocco, 31 March–2 April 2020; pp. 1–6. [CrossRef]

5. Kumar, R.; Ali, M.M. A Review on Chatbot Design and Implementation Techniques. Int. J. Eng. Technol. 2020, 7, 11. [CrossRef]

6. Cahn, J. CHATBOT: Architecture, Design, & Development. Ph.D. Thesis, University of Pennsylvania, School of Engineering and Applied Science, Philadelphia, PA, USA, 2017.

7. Okuda, T.; Shoda, S. AI-based Chatbot Service for Financial Industry. FUJITSU Sci. Tech. J. 2018, 54, 5.

8. Brandtzaeg, P.B.; Følstad, A. Why People Use Chatbots. In Internet Science; Kompatsiaris, I., Cave, J., Satsiou, A., Carle, G., Passani, A., Kontopoulos, E., Diplaris, S., McMillan, D., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Berlin/Heidelberg, Germany, 2017; Volume 10673, pp. 377–392.

9. Costa, P. Conversing with personal digital assistants: On gender and artificial intelligence. J. Sci. Technol. Arts 2018, 10, 59–72. [CrossRef]

10. Go, E.; Sundar, S.S. Humanizing chatbots: The effects of visual, identity and conversational cues on humanness perceptions. Comput. Hum. Behav. 2019, 97, 304–316. [CrossRef]

11. https://replika.ai/

12. https://mobilemonkey.com/

13. https://chatfuel.com/

14. https://www.proprofschat.com/

15. https://www.aivo.co/

16. https://home.pandorabots.com/home.html

17. https://www.meya.ai/

11. https://dev.botframework.com/
12. https://www.userlike.com/en/
13. https://wordpress.org/plugins/wp-chatbot/

SOURCE CODE

```
import re
import pandas as pd
import pyttsx3
from sklearn import preprocessing
from sklearn.tree import DecisionTreeClassifier,_tree
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.svm import SVC
import csv
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)


training = pd.read_csv('Data/Training.csv')
testing= pd.read_csv('Data/Testing.csv')
cols= training.columns
cols= cols[:-1]
x = training[cols]
y = training['prognosis']
y1= y


reduced_data = training.groupby(training['prognosis']).max()

#mapping strings to numbers
le = preprocessing.LabelEncoder()
le.fit(y)
y = le.transform(y)


x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.33,
random_state=42)
testx    = testing[cols]
testy    = testing['prognosis']
testy    = le.transform(testy)


clf1  = DecisionTreeClassifier()
clf = clf1.fit(x_train,y_train)
# print(clf.score(x_train,y_train))
# print ("cross result========")
scores = cross_val_score(clf, x_test, y_test, cv=3)
# print (scores)
print (scores.mean())
```

```
model=SVC()
model.fit(x_train,y_train)
print("for svm: ")
print(model.score(x_test,y_test))

importances = clf.feature_importances_
indices = np.argsort(importances)[::-1]
features = cols

def readn(nstr):
    engine = pyttsx3.init()

    engine.setProperty('voice', "english+f5")
    engine.setProperty('rate', 130)

    engine.say(nstr)
    engine.runAndWait()
    engine.stop()


severityDictionary=dict()
description_list = dict()
precautionDictionary=dict()

symptoms_dict = {}

for index, symptom in enumerate(x):
       symptoms_dict[symptom] = index
def calc_condition(exp,days):
    sum=0
    for item in exp:
         sum=sum+severityDictionary[item]
    if((sum*days)/(len(exp)+1)>13):
        print("You should take the consultation from doctor. ")
    else:
        print("It might not be that bad but you should take precautions.")


def getDescription():
    global description_list
    with open('MasterData/symptom_Description.csv') as csv_file:
        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        for row in csv_reader:
            _description={row[0]:row[1]}
            description_list.update(_description)




def getSeverityDict():
```

```
    global severityDictionary
    with open('MasterData/symptom_severity.csv') as csv_file:

        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        try:
            for row in csv_reader:
                _diction={row[0]:int(row[1])}
                severityDictionary.update(_diction)
        except:
            pass


def getprecautionDict():
    global precautionDictionary
    with open('MasterData/symptom_precaution.csv') as csv_file:

        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        for row in csv_reader:
            _prec={row[0]:[row[1],row[2],row[3],row[4]]}
            precautionDictionary.update(_prec)


def getInfo():
    print("-----------------------------------HealthCare ChatBot-----------------------
-----------")
    print("\nYour Name? \t\t\t\t",end="->")
    name=input("")
    print("Hello, ",name)

def check_pattern(dis_list,inp):
    pred_list=[]
    inp=inp.replace(' ','_')
    patt = f"{inp}"
    regexp = re.compile(patt)
    pred_list=[item for item in dis_list if regexp.search(item)]
    if(len(pred_list)>0):
        return 1,pred_list
    else:
        return 0,[]
def sec_predict(symptoms_exp):
    df = pd.read_csv('Data/Training.csv')
    X = df.iloc[:, :-1]
    y = df['prognosis']
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=20)
    rf_clf = DecisionTreeClassifier()
    rf_clf.fit(X_train, y_train)

    symptoms_dict = {symptom: index for index, symptom in enumerate(X)}
```

```
    input_vector = np.zeros(len(symptoms_dict))
    for item in symptoms_exp:
      input_vector[[symptoms_dict[item]]] = 1

    return rf_clf.predict([input_vector])


def print_disease(node):
    node = node[0]
    val  = node.nonzero()
    disease = le.inverse_transform(val[0])
    return list(map(lambda x:x.strip(),list(disease)))

def tree_to_code(tree, feature_names):
    tree_ = tree.tree_
    feature_name = [
        feature_names[i] if i != _tree.TREE_UNDEFINED else "undefined!"
        for i in tree_.feature
    ]

    chk_dis=",".join(feature_names).split(",")
    symptoms_present = []

    while True:

        print("\nEnter the symptom you are experiencing  \t\t",end="->")
        disease_input = input("")
        conf,cnf_dis=check_pattern(chk_dis,disease_input)
        if conf==1:
            print("searches related to input: ")
            for num,it in enumerate(cnf_dis):
                print(num,")",it)
            if num!=0:
                print(f"Select the one you meant (0 - {num}):  ", end="")
                conf_inp = int(input(""))
            else:
                conf_inp=0

            disease_input=cnf_dis[conf_inp]
            break
            # print("Did you mean: ",cnf_dis,"?(yes/no) :",end="")
            # conf_inp = input("")
            # if(conf_inp=="yes"):
            #     break
        else:
            print("Enter valid symptom.")

    while True:
        try:
            num_days=int(input("Okay. From how many days ? : "))
            break
```

```
        except:
            print("Enter valid input.")
    def recurse(node, depth):
        indent = "  " * depth
        if tree_.feature[node] != _tree.TREE_UNDEFINED:
            name = feature_name[node]
            threshold = tree_.threshold[node]

            if name == disease_input:
                val = 1
            else:
                val = 0
            if  val <= threshold:
                recurse(tree_.children_left[node], depth + 1)
            else:
                symptoms_present.append(name)
                recurse(tree_.children_right[node], depth + 1)
        else:
            present_disease = print_disease(tree_.value[node])
            # print( "You may have " +  present_disease )
            red_cols = reduced_data.columns
            symptoms_given =
red_cols[reduced_data.loc[present_disease].values[0].nonzero()]
            # dis_list=list(symptoms_present)
            # if len(dis_list)!=0:
            #     print("symptoms present  " + str(list(symptoms_present)))
            # print("symptoms given "  +  str(list(symptoms_given)) )
            print("Are you experiencing any ")
            symptoms_exp=[]
            for syms in list(symptoms_given):
                inp=""
                print(syms,"? : ",end='')
                while True:
                    inp=input("")
                    if(inp=="yes" or inp=="no"):
                        break
                    else:
                        print("provide proper answers i.e. (yes/no) : ",end="")
                if(inp=="yes"):
                    symptoms_exp.append(syms)

            second_prediction=sec_predict(symptoms_exp)
            # print(second_prediction)
            calc_condition(symptoms_exp,num_days)
            if(present_disease[0]==second_prediction[0]):
                print("You may have ", present_disease[0])
                print(description_list[present_disease[0]])

                # readn(f"You may have {present_disease[0]}")
                # readn(f"{description_list[present_disease[0]]}")
```

```
            else:
                print("You may have ", present_disease[0], "or ", second_prediction[0])
                print(description_list[present_disease[0]])
                print(description_list[second_prediction[0]])

            # print(description_list[present_disease[0]])
            precution_list=precautionDictionary[present_disease[0]]
            print("Take following measures : ")
            for  i,j in enumerate(precution_list):
                print(i+1,")",j)

            # confidence_level = (1.0*len(symptoms_present))/len(symptoms_given)
            # print("confidence level is " + str(confidence_level))

    recurse(0, 1)
getSeverityDict()
getDescription()
getprecautionDict()
getInfo()
tree_to_code(clf,cols)
print("----------------------------------------------------------------------------------------
-------")
```