HELPI – A HEALTHCARE CHATBOT

A Project Report submitted in partial fulfillment of the requirements for the award of degree of

Bachelor of Technology in Computer Science and Engineering with Minor in Artificial Intelligence & Machine Learning
M. Ambika(20B81A0565)

Bachelor of Technology in Computer Science and Engineering with Minor in Artificial Intelligence & Machine Learning
S. Gnanika (20B81A0574)

Bachelor of Technology in Computer Science and Engineering with Minor in Artificial Intelligence & Machine Learning
K. Tejaswini(20B81A05B1)

Under the Guidance of
Dr. R. K. SelvaKumar
Professor

Department of Computer Science and Engineering

CVR COLLEGE OF ENGINEERING

(An UGC Autonomous Institution, Affiliated to JNTUH, Accredited by NBA, and NAAC)
Vastunagar, Mangalpalli (V), Ibrahimpatnam (M),
Ranga Reddy (Dist.) - 501510, Telangana State

2023-24

CVR COLLEGE OF ENGINEERING

(An UGC Autonomous Institution, Affiliated to JNTUH,
Accredited by NBA, and NAAC)
Vastunagar, Mangalpalli (V), Ibrahimpatnam (M),
Ranga Reddy (Dist.) - 501510, Telangana State.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**HELPI–A HEALTHCARE CHATBOT**" being submitted by **M. Ambika (20B81A0565)**, **S. Gnanika (20B81A0574)**, **K. Tejaswini (20B81A05B1)** in partial fulfilment for the award of Bachelor of Technology in Computer Science and Engineering to the CVR College of Engineering, during the academic year 2023-2024.

The results embodied in this project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Signature of Project guide

Dr. R. K. SelvaKumar
Professor, Department of CSE.

Signature of the HOD

Dr. A. Vani Vasthala
Department of CSE

Signature of the Professor-in-charge projects

Dr. M SwamiDas
Department of CSE

External Examiner

DECLARATION

I hereby declare that this Mini project report titled "**HELPI- A HealthCare ChatBot**" submitted to the Department of Computer Science and Engineering, CVR College of Engineering, is a record of original work done by me under the guidance of **Dr. R. K. SelvaKumar**. The information and data given in the report is authentic to the best of my knowledge. This Mini project report is not submitted to any other university or institution for the award of any degree or diploma or published at any time before.

M. Ambika (20B81A0565)
S. Gnanika (20B81A0574)
K. Tejaswini (20B81A05B1)

Date:

Place:

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people who made it possible and whose encouragement and guidance has been a source of inspiration throughout the course of the project.

It is a great pleasure to convey our profound sense of gratitude to our principal **Dr. K. Rammohan Reddy, Dr. A. Vani Vathsala**, Head of CSE Department, CVR College of Engineering, for having been kind enough to arrange necessary facilities for executing the project in the college.

We deem it a pleasure to acknowledge our sense of gratitude towards our project guide **Dr. R. K. Selvakumar** under whom we have carried out the project work. His inclusive and objective guidance and timely advice encouraged us with constant flow of energy to continue the work.

We wish a deep sense of gratitude and heartfelt thanks to management for providing excellent lab facilities and tools. Finally, we thank all those whose guidance helped us in this regard.

ABSTRACT

Due to technological advancements, the healthcare industry has witnessed the emergence of innovative solutions, and one such solution is the healthcare chatbot. The primary objective of this project is to create a healthcare chatbot capable of offering medical assistance to patients. The healthcare chatbot serves as an AI-based conversational program designed to assist both patients and healthcare providers. These chatbots have the potential to revolutionize virtual customer service and streamline planning and management in the healthcare sector.

The proposed chatbot, named "HELPI," functions as a round-the-clock healthcare provider. It utilizes natural language processing (NLP) and machine learning (ML) algorithms such as decision trees to analyze user-provided symptoms and accurately detect specific illnesses or diseases. Subsequently, it offers appropriate healthcare recommendations and suggests relevant medications. The project implementation will involve leveraging modern technologies like Python. HELPI's training process involves utilizing diverse datasets, including the Disease Prediction Dataset, to enhance its knowledge base. This broadens HELPI's capability to address various healthcare-related concerns. In essence, HELPI aims to alleviate the burden on healthcare providers by providing an alternative platform for basic medical advice and support. The success of the HELPI chatbot lays the foundation for future enhancements. Additional features, such as appointment scheduling, guidance on lifestyle modifications, and medication reminders, could be incorporated to further enhance the chatbot's functionality.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATION

Abbreviation	**Full Form**
NLP	Natrual Language Processing
RE	Regular Expression
CART	Classification and Regression Tree

1. INTRODUCTION

Machine learning is an artificial intelligence technique that enables computers to learn from data and enhance their performance in specific tasks. Machine learning is a dynamic field within artificial intelligence that has revolutionized the way computers process and learn from data. It involves the development of algorithms and models that can automatically improve their performance on specific tasks as they are exposed to more data. This iterative learning process enables machines to make predictions, decisions, and take actions based on patterns and insights derived from large datasets. It encompasses various algorithms, including supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning is one of the fundamental approaches in machine learning. It involves training a model using labeled data, where the inputs are paired with corresponding desired outputs or labels. The goal is for the model to learn the underlying patterns in the data and be able to accurately predict the outputs for new, unseen inputs. This technique is widely used in applications such as image recognition, speech recognition, and sentiment analysis.

Unsupervised learning, on the other hand, deals with unlabeled data. The goal here is to discover hidden structures, patterns, and relationships within the data without any predefined labels. Clustering and dimensionality reduction are common unsupervised learning techniques. Clustering algorithms group similar data points together based on their inherent properties, while dimensionality reduction techniques aim to reduce the complexity of the data by extracting the most relevant features.

Reinforcement learning is a type of machine learning where an agent learns to interact with an environment in order to maximize a reward signal. The agent takes actions, receives feedback in the form of rewards or penalties, and adjusts its behavior accordingly. Through a process of trial and error, the agent learns the optimal policy or strategy to achieve the maximum cumulative reward. Reinforcement learning has been successfully applied in various domains, including game playing (e.g., AlphaGo), robotics, and autonomous vehicles.

Building a machine learning system typically involves several steps
Data preparation Gathering, cleaning, and preparing data for training the model.
Feature extraction Identifying relevant features crucial for the model to learn from.
Model selection Choosing the appropriate algorithm that suits the specific problem.
Training Using the data to train the model to learn from input data and labels.
Evaluation Testing the model on separate data to assess its performance and adjust parameters if necessary.

Deployment Integrating the model into the desired application or system.

Machine learning finds extensive applications in various fields such as computer vision, natural language processing, healthcare, finance, and more. It offers valuable tools for businesses and organizations aiming to enhance their operations and decision-making processes. Natural Language Processing (NLP) is a subfield of machine learning that focuses on human-computer interaction using natural language. Chatbots utilize NLP techniques to comprehend and interpret human language, enabling appropriate responses. Intent classification is essential for chatbots as it involves identifying the purpose or goal behind a user's message, aiding in understanding and responding accurately. Intent classification can employ machine learning techniques like supervised learning and deep learning. Entity recognition involves identifying specific entities like names, locations, dates, and organizations in a user's message, facilitating personalized and relevant responses. Machine learning techniques like named entity recognition (NER) can be used for entity recognition tasks.

1.1 MOTIVATION

Impact on lives: HELPI has the potential to positively impact countless lives by providing accessible, 24/7 medical assistance.

Revolutionizing healthcare: HELPI can contribute to a major evolution in healthcare. By easing the burden on healthcare providers and offering 24/7 support,

Learning and growth: Building HELPI presents a rich learning opportunity to deepen your understanding of AI, NLP, machine learning, and healthcare concepts. The process of gathering data, training the algorithms, and testing the chatbot will continuously challenge and expand your knowledge, giving you valuable technical and problem-solving skills.

1.2 PROBLEM STATEMENT

Millions of people lack easy access to basic healthcare information and guidance. Limited access to healthcare providers, especially in remote areas or during off-hours, creates a gap in timely and affordable medical assistance.

Current healthcare chatbots often lack accuracy, comprehensiveness, and user-friendliness. Many chatbots rely on simple keyword matching or limited symptom databases, resulting in inaccurate diagnoses or irrelevant recommendations. Therefore, we need a highly accurate, accessible, and user-friendly healthcare chatbot that can bridge the gap in basic healthcare access.

1.3 PROJECT OBJECTIVES

Healthcare plays a crucial role in people's lives, but it can be difficult to access in certain situations, such as remote locations, emergencies, or when individuals are unable to physically visit healthcare facilities. To address this challenge, chatbots have emerged as a popular solution, leveraging technology to provide round-the-clock medical assistance. HELPI is an intelligent healthcare chatbot that aims to meet this need by delivering personalized healthcare support to patients.

The HELPI chatbot is designed to handle a broad spectrum of medical inquiries, ranging from minor concerns to serious conditions. Powered by artificial intelligence (AI) and natural language processing (NLP), it efficiently and effectively provides medical assistance to patients. Its user-friendly interface enables patients to interact with the chatbot using natural language, allowing them to easily access the desired healthcare support. Whether through smartphones, laptops, or desktops, HELPI ensures seamless access to healthcare services for patients.

HELPI offers personalized healthcare assistance to patients as one of its key advantages. By harnessing the power of AI and NLP, the chatbot can comprehend and interpret patients' medical inquiries, providing them with accurate and relevant medical advice. Furthermore, HELPI can guide patients on appropriate steps to take based on their symptoms, medical history, and other pertinent factors. This tailored approach empowers patients to make wellinformed decisions regarding their health and overall well-being.

Another significant benefit of HELPI is its ability to alleviate the burden on healthcare providers. With the rising demand for healthcare services, healthcare providers often struggle to deliver timely medical assistance to patients. By offering a readily accessible 24/7 medical assistant, HELPI lightens the load on healthcare providers, enabling them to prioritize patients in need of immediate medical attention.

Privacy and confidentiality are paramount in HELPI's design. All medical queries and patient data shared within the chatbot remain confidential and secure. Rigorous testing ensures the chatbot's ability to provide accurate and reliable medical assistance to patients.

To provide comprehensive guidance to users, documentation is available to facilitate interaction with HELPI. This documentation outlines the chatbot's features, operations, and troubleshooting steps, aiming to equip users with a thorough understanding of HELPI and maximize their experience with this innovative healthcare chatbot.

When building a healthcare chatbot, there are several potential objectives that you can consider to enhance its functionality and impact

Enhancing Patient Engagement A key objective of a healthcare chatbot is to actively engage patients in their healthcare journey. By providing personalized support and advice, the chatbot can empower patients to take a more proactive role in managing their health. It can offer convenient access to healthcare services and encourage patients to stay involved in their wellbeing.

Providing 24/7 Support A healthcare chatbot can offer round-the-clock assistance to patients, addressing their queries and concerns even outside of regular office hours. By being available anytime, the chatbot helps alleviate the pressure on healthcare providers and contributes to improved patient satisfaction.

Reducing Healthcare Costs Timely advice and support from a healthcare chatbot can prevent minor health issues from escalating and requiring costly treatments. By promoting preventive care and early intervention, the chatbot can significantly reduce healthcare costs for both patients and providers.

Improving Health Outcomes The chatbot can play a crucial role in helping patients manage chronic conditions effectively. It can remind patients to take medications, monitor symptoms, and guide them toward making healthy lifestyle choices. By doing so, the chatbot contributes to better health outcomes and an improved quality of life for patients.

Enhancing Data Collection By collecting and analyzing patient data, a healthcare chatbot can provide valuable insights to healthcare providers. This data can offer a deeper understanding of patient behavior, preferences, and patterns, enabling providers to improve healthcare services and develop personalized treatment plans. It's important to note that the specific objectives of a healthcare chatbot will depend on the unique needs of your target audience and healthcare organization. By defining clear objectives and aligning the design and functionality of your chatbot accordingly, you can ensure that it delivers tangible value to both patients and healthcare providers.

1.4 PROJECT REPORT ORGANIZATION

The HELPI healthcare chatbot offers several features and capabilities

The "Project Report" provides a comprehensive overview of the design, methodology, and implementation of an application that aims to solve a real-world problem. The report breaks down the application's functionalities into modules, which are explained using use case and class diagrams. Screenshots of the working model of the application are also included in the report.

The report is organized into six chapters.

Chapter 1: It presents the Introduction,Motivation behind the project ,Problem Statement and report organisation

Chapter 2: It presents a literature survey, highlighting the characteristics and design challenges of the existing system and proposing a solution.

Chapter 3: It outlines the software requirements, including functional and non-functional requirements, system architecture, and specifications.

Chapter 4: It delves into the UML diagrams, including use case diagrams, class diagrams, activity diagrams, and architecture diagrams.

Chapter 5: It covers the implementation and testing using various tools, and includes detailed screenshots of the process.

Chapter 6: It concludes the report and provides insights into the future scope of the application being developed.

Overall, the report offers a detailed analysis of the application's development and aims to provide a solution to a real-world problem

2 LITERATURE SURVEY

2.1 EXISTING WORK

A Systematic Review of Chatbot Applications in Mental Healthcare [1] S. Smith reviews different types of conversational agents used in health care for chronic conditions, examining their underlying communication technology, evaluation measures, and AI methods. A systematic search was performed in February 2021 on PubMed Medline, EMBASE, PsycINFO, CINAHL, Web of Science, and ACM Digital Library. Out of 26 conversational agents (CAs), 16 were chatbots, seven were embodied conversational agents (ECA), one was a conversational agent in a robot, and another was a relational agent.

An Empirical Study on the Effectiveness of Chatbots in Patient Education [2]by J. Brown. [2] explores the potential use of AI systems and chatbots in the academic field and their impact on research and education from an ethical perspective. Through a qualitative methodology, the researcher performs exploratory research and data collection based on expert analysis and interpretation. [2] highlights the necessity of adaptation to the new reality of AI systems and chatbots. Co-living, sustainability and continuous adaptation to the development of these systems will become a matter of emergency. Raising awareness, adopting appropriate legislations and solidifying ethical values will strengthen research and protect educational systems. The presence of AI systems and chatbots in education needs to be considered as an opportunity for development rather than a threat.

Exploring the Acceptance and Usability of Chatbots in Elderly Care Settings [3] by M. Johnson.[3] explored perceptions of chatbots with varying identities for health information seeking in a diary and interview study with 30 older adults. Findings suggest that while racial and age likeness influence feelings of trust and comfort with chatbots, constructs such as professionalism and likeability and overall familiarity also influence reception. Based on these findings, we provide implications for designing text-based chatbots that consider older adults.

Chatbots for Chronic Disease Management: A Review of Current Applications and Challenges [3] by A. Martinez. [3] aims to report on the recent advances and current trends in chatbot technology in medicine. A brief historical overview, along with the developmental progress and design characteristics, is first introduced. The focus will be on cancer therapy, with in-depth discussions and examples of diagnosis, treatment, monitoring, patient support, workflow efficiency, and health promotion. A search of the literature published in the past 20 years was conducted using the IEEE Xplore, PubMed, Web of Science, Scopus, and OVID databases. The screening of chatbots was guided by the open-access Botlist directory for health care components and further divided according to the following criteria diagnosis, treatment, monitoring, support, workflow, and health promotion.

Improving Medication Adherence through Chatbot-based Reminders: A Randomized Controlled Trial [4] by L. Thompson. [4] aimed to review the current applications, gaps, and challenges in the literature on conversational agents in health care and provide recommendations for their future research, design, and application. A broad literature search was performed in MEDLINE (Medical Literature Analysis and Retrieval System Online; Ovid), EMBASE (Excerpta Medica database; Ovid), PubMed, Scopus, and Cochrane Central with the search terms “conversational agents,” “conversational AI,” “chatbots,” and associated synonyms. We also searched the gray literature using sources such as the OCLC (Online Computer Library Center) WorldCat database and ResearchGate.

2.2 LIMITATIONS OF EXISTING WORK

The review may lack specificity in terms of the types of mental health conditions addressed. Different mental health issues may require tailored approaches, and a generalized overview may not capture these nuances.
The study focuses on common diseases, potentially overlooking the accuracy of chatbots in assessing symptoms of less prevalent or emerging health conditions.
The study primarily focuses on user perceptions and lacks objective measurements of the impact of chatbot interventions on elderly care outcomes. It may not provide concrete evidence of the effectiveness of chatbots in improving health outcomes for the elderly**.**
The review does not extensively cover the potential privacy and security concerns associated with chatbot interactions in managing sensitive health information.

3 SOFTWARE AND HARDWARE SPECIFICATIONS

3.1 USER REQUIREMENTS

3.1.1 Functional Requirements

1. Functional requirements describe the intended behaviour and functionality of the system. They specify what the system should do in terms of inputs, processing, and outputs. Functional requirements typically answer questions such as "What tasks should the system perform?" and "What are the desired system features?"
2. **Use Case Scenarios** Descriptions of system interactions with users or other systems, capturing the flow of events and expected outcomes.
3. **User Stories** Descriptions of specific user tasks or features in a narrative format, often used in Agile development methodologies.
4. **Functional Specifications** The system should describe its functionality, including input requirements, processing rules, and output formats.
5. **Business Rules** Statements that define the constraints, regulations, or policies that govern the system's behaviour.

3.1.2 Non-Functional Requirements

Non-functional requirements describe the qualities and constraints that govern the system's behaviour and performance. They focus on aspects such as system reliability, usability, performance, security, and maintainability. Non-functional requirements typically answer questions such as "How well should the system perform?" and "What are the quality attributes of the system?"

1. **Performance** The system should be capable of processing and analyzing the user input in near real-time to provide timely detection and response.
2. **Usability** The system should have an intutive interface, making it easy for users to use the chatbot.
3. **Reliability** The system should be able to perform its intended functions accurately and consistently over time.
4. **Security** The system should be able to take the measures and controls needed to protect the system and its data from unauthorized access or threats.
5. **Maintainability** The system should designes in such a way that it can be modified, extended, or repaired easily without causing significant disruptions.
6. **Scalability** The system should be able to handle increasing workloads and accommodate future growth.
7. **Compatibility** The system should be able to operate and interact with other systems, platforms, or environments.

3.2 SOFTWARE REQUIREMENTS

1. **Programming Language:** Python
2. **Platform:** Google Colab
3. **Other Libraries:** Pandas, sklearn, NumPy, csv
4. **Front-end:** Python-Streamlit

3.3 HARDWARE REQUIREMENTS

1. **Processor:** Intel Core i3, Recommended i5 or i7, clock speed of at least 2.4 GHz
2. **RAM:** 8GB is recommended
3. **Storage**: As necessitated by the model (250 GB is recommended)
4. **Graphic Card:** Min. NVIDIA T4
5. **Operating System:** Windows/ Mac OS/ Linux

4 SYSTEM DESIGN

4.1 PROPOSED METHOD

HELPI was developed using the decision tree algorithm, a machine learning technique that employs a tree-like model to make decisions and determine their potential outcomes. To train the algorithm, a dataset called Disease Prediction was acquired from the online platform Kaggle. The dataset was carefully processed to eliminate irrelevant information and transform textual data into a suitable format for the algorithm. The decision tree algorithm was chosen due to its ability to handle both categorical and numerical data and its interpretability. The implementation of the algorithm was carried out using the Python programming language.

The following steps were undertaken during the development process using the decision tree algorithm

1. **Data Collection** The initial step involved gathering data that would be utilized to train the algorithm. This data encompassed patient inquiries, responses from healthcare providers, and medical records. It was crucial for the data to be representative of the types of inquiries the chatbot was expected to handle.
2. **Data Preprocessing** Once the data was collected, it underwent preprocessing to eliminate irrelevant information and convert the textual data into a suitable format for the algorithm. This preprocessing stage involved tasks such as removing stop words, converting text to lowercase, and tokenization.
3. **Feature Extraction** The subsequent step was to extract pertinent features from the preprocessed data. These features could encompass symptoms, medical history, and preferences. The decision tree algorithm would utilize these features to determine appropriate responses to patient inquiries.
4. **Algorithm Selection** The decision tree algorithm was selected as it is widely used for developing healthcare chatbots. Its versatility in handling both categorical and numerical data and its capability to produce interpretable outcomes were among the reasons for its choice. Nonetheless, other machine learning algorithms like neural networks, support vector machines, and random forests could also be employed.
5. **Model Training** Training the decision tree algorithm involved utilizing the preprocessed data and the extracted features. The algorithm was trained to understand the relationships between the features and the appropriate responses. This training phase could be conducted through supervised learning, where labeled data is provided, or unsupervised learning, allowing the algorithm to discern patterns in the data without labeled examples.

6. **Model Testing and Validation** After the algorithm was trained, it underwent testing and validation to ensure its accuracy and reliability. This process typically involved using a sample dataset of patient inquiries and responses. The algorithm's accuracy could be evaluated by comparing its responses with those provided by healthcare providers.
7. **Chatbot Integration** Once the algorithm was successfully tested and validated, it was integrated into the chatbot platform. The chatbot was designed to offer personalized support and advice to patients based on the decision tree algorithm's responses.

Additionally, the chatbot platform collected data from patient inquiries and responses, which could be used to enhance and refine the algorithm over time.
The decision tree algorithm employed by HELPI provides several advantages over other machine learning algorithms. It offers interpretability, enabling healthcare providers to comprehend the decision-making process. Furthermore, the algorithm's ability to handle both categorical and numerical data enhances its versatility. However, it is important to note that the decision tree algorithm utilized by HELPI may not be suitable for all types of patient inquiries, especially those involving complex cases that require more nuanced decision-making. Additionally, the accuracy of the algorithm may vary based on the quality of the training data and the complexity of the decision tree structure.

Decision Tree Algorithm Selection and Configuration (CART Algorithm)

Choosing and configuring the decision tree algorithm is a critical aspect of developing the HELPI healthcare chatbot. The CART algorithm, which is capable of handling both classification and regression tasks, has been selected for this project. It partitions the data recursively based on chosen features to create subsets that are as pure as possible in relation to the target variable. The resulting decision tree consists of internal nodes representing feature tests and leaf nodes representing predicted responses.

The decision to use the CART algorithm is driven by its flexibility in dealing with mixed data types, encompassing categorical and numerical variables. This adaptability is particularly advantageous in the healthcare field where patient inquiries often involve a combination of symptoms, medical history, and preferences, necessitating a versatile algorithm for effective decision-making. Configuring the CART algorithm involves adjusting essential hyperparameters to optimize its performance. Key considerations include the maximum depth of the tree, the minimum number of samples required for node splitting, and the minimum number of samples needed at a leaf node. The maximum depth governs the tree's complexity, with deeper trees capturing more intricate relationships but also risking overfitting. It is crucial to experiment with different depth values and evaluate the model's performance to strike the right balance between complexity and generalization.

The minimum number of samples required for node splitting determines when the algorithm stops creating splits based on available samples. Properly setting this parameter prevents the algorithm from generating insignificant splits that introduce noise. Similarly, the minimum number of samples for a leaf node ensures each leaf represents a sufficient number of samples for reliable predictions. Adjusting this parameter controls the granularity of the decision tree, guaranteeing reliable predictions for each leaf.

Optimal configuration of the CART algorithm is typically achieved through experimentation and evaluation using performance metrics like accuracy, precision, recall, or F1 score. Techniques such as cross-validation can be employed to assess performance on different subsets of data and mitigate issues of overfitting or underfitting. Additionally, post-pruning techniques can be applied to trim the fully grown decision tree, reducing complexity by removing unnecessary branches or nodes. Pruning enhances generalization and prevents overfitting. Common methods include cost complexity pruning (alpha pruning) and reduced error pruning.

By selecting and configuring the CART algorithm, the HELPI healthcare chatbot can effectively analyze patient inquiries, make accurate predictions, and provide personalized support and advice. Its ability to handle mixed data types and the flexibility of its hyperparameters make it a suitable choice for addressing the unique challenges of the healthcare domain.

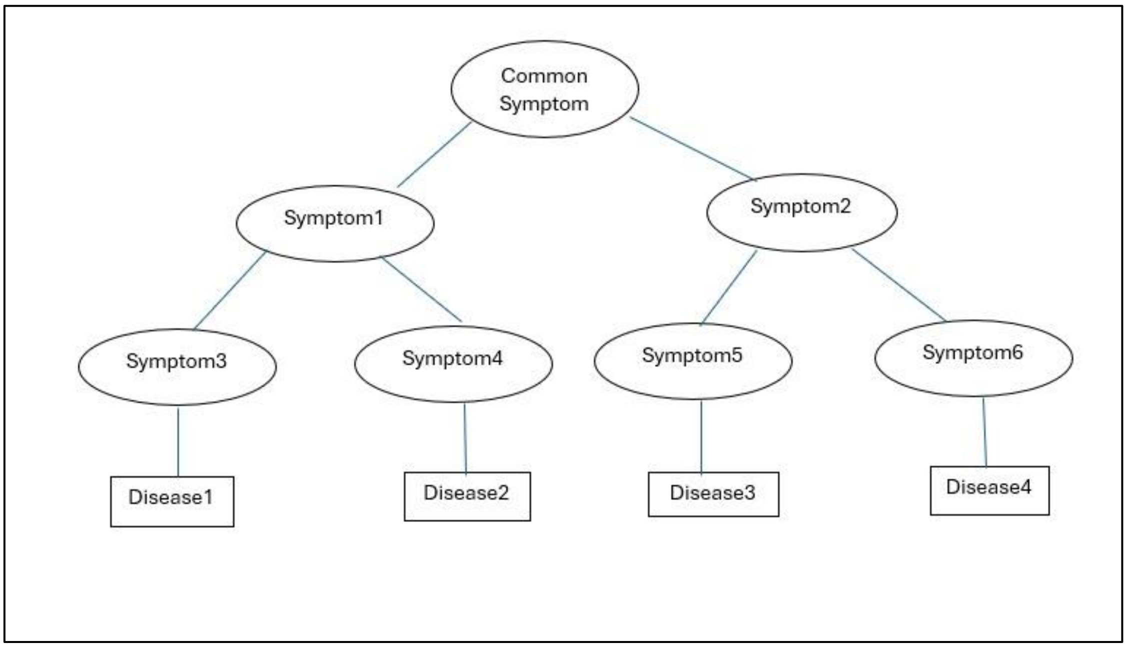


Fig 4.1 Decision Tree

4.2 USE CASE DIAGRAM

The use case diagram (fig 4.1) represents the main actions or functionalities that the healthcare chatbot provides to the user. The central box represents the three use cases associated with the chatbot.

Symptoms This use case represents the user entering symptoms into the chatbot for disease prediction.

Predict Disease This use case represents the chatbot processing the symptoms and predicting the likely disease.

Provide Measures This use case represents the chatbot providing recommended measures or actions to be taken based on the predicted disease.

Each use case is connected to the chatbot box with a line.

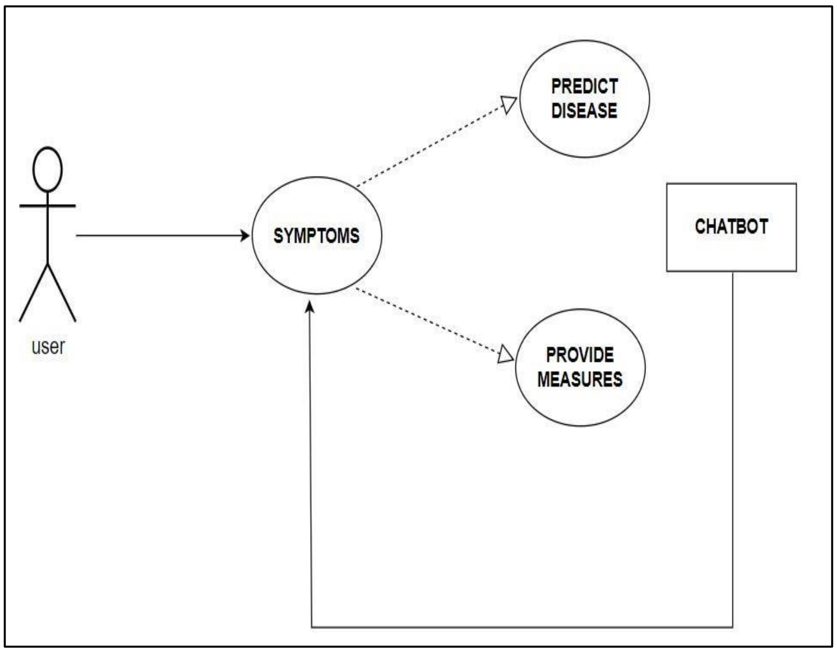


Fig.4.1 Use Case Diagram for HAC

4.3 ACTIVITY DIAGRAM

An activity diagram (fig 4.2) represents the flow of activities or actions within a system The diagram starts with the Start node and proceeds to the Enter Symptoms activity.

In the Enter Symptoms activity, the user provides the symptoms to the chatbot.

Once the symptoms are entered, the chatbot compares the input string with the dataset and it asks few questions to the user based on the symptoms entered by the user.

Now the chatbot moves to the Predict Disease activity, where it processes the symptoms and predicts the disease.
From the Predict Disease activity, the chatbot proceeds to the Provide Measures activity, where it determines the recommended measures based on the predicted disease.
After providing the measures, the chatbot moves to the Display Results activity to show the results to the user.

Finally, the flow ends at the Finish node.

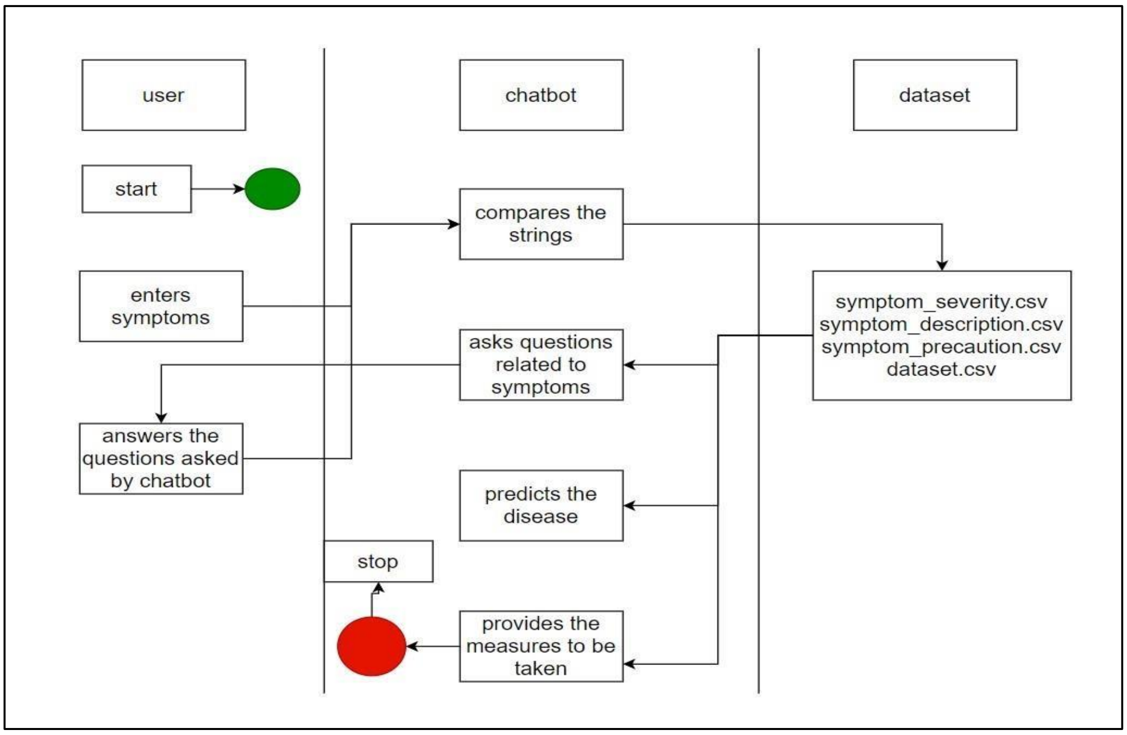


Fig.4.2 Activity Diagram for HAC

4.4 SEQUENCE DIAGRAM

The sequence diagram starts with the user interacting with the chatbot by entering symptoms.

The user input is received by the Chatbot. The chatbot then compares the input string entered by the user as input with the dataset.

After string comparison, the chatbot will predict the disease.

After predicting the disease, the chatbot returns the measures to be taken by the user.

Finally, the chatbot displays the results to the user, completing the interaction.

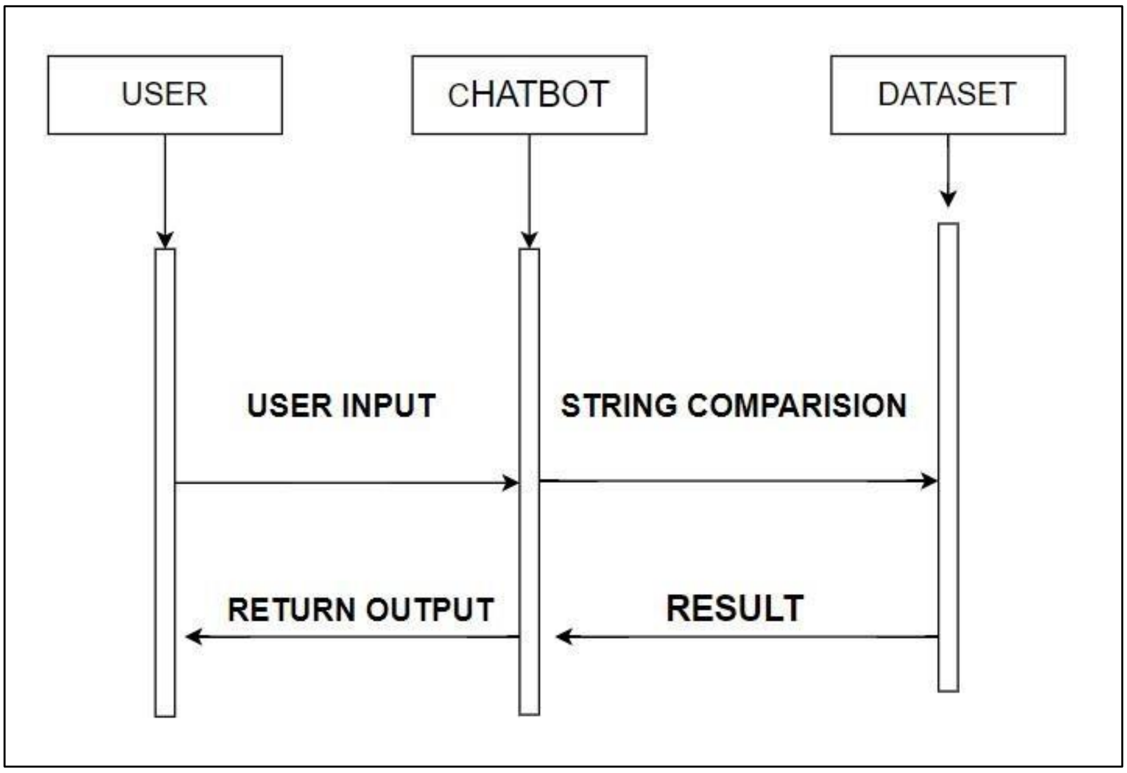


Fig.4.3 Sequence Diagram for the system.

4.5 SYSTEM ARCHITECTURE

The HELPI healthcare chatbot is built with an architecture that enables smooth communication and intelligent decision-making. It adopts a client-server model, where the client refers to the user interface or chatbot application, while the server hosts the backend components responsible for handling user input and generating responses. This design ensures efficient interaction between the user and the chatbot, allowing for seamless communication and effective processing of information.

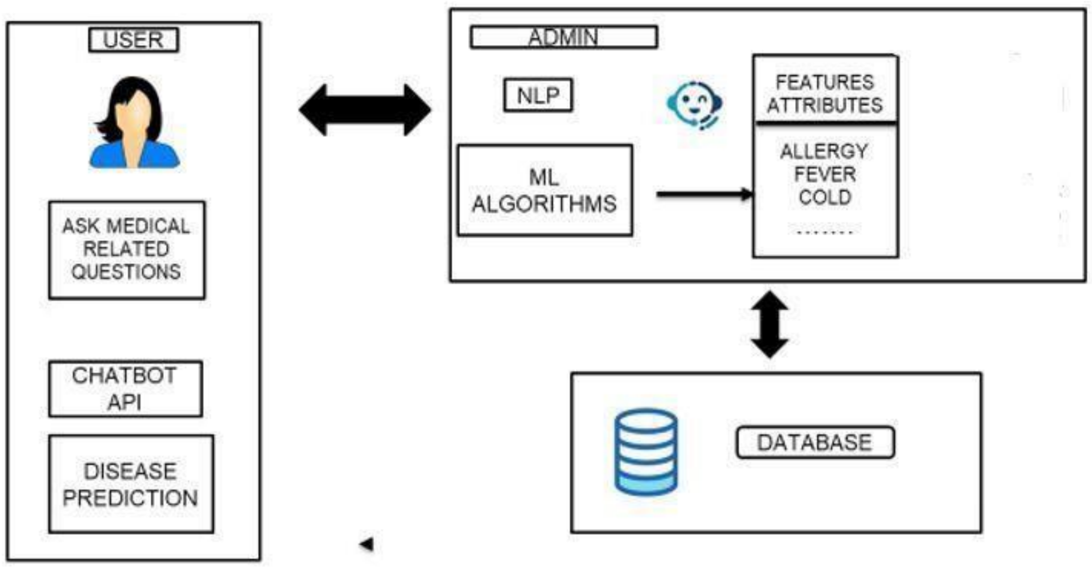


Fig.4.5. System Architecture Diagram

4.6 TECHNOLOGY DESCRIPTION

Natural Language Processing (NLP) Module

The NLP module is responsible for processing and understanding user inquiries and responses. It involves the implementation of algorithms and techniques such as text preprocessing, tokenization, part-of-speech tagging, named entity recognition, and sentiment analysis to extract meaningful information from user input.

The NLP module is a critical component of your healthcare chatbot system that is responsible for processing and understanding user input. It incorporates various natural language processing techniques to analyse and extract meaningful information from the input. Here are some key aspects of the NLP module

Text Pre-processing The NLP module performs text pre-processing tasks to clean and normalize the user input. It removes irrelevant characters, punctuation, or special symbols. It may also handle common pre-processing tasks such as lowercase conversion, removing stop words, and handling abbreviations or acronyms.

Tokenization The NLP module breaks down the user input into individual tokens or words. It splits the input into meaningful units, allowing further analysis at the word or phrase level. Tokenization helps in understanding the structure of the input and identifying key elements for analysis.

Part-of-Speech (POS) Tagging The NLP module performs part-of-speech tagging to assign grammatical tags to each word in the user input. It identifies the role of each word in the sentence, such as nouns, verbs, adjectives, etc. POS tagging helps in understanding the syntactic structure of the input and can aid in extracting relevant information.

Named Entity Recognition (NER) The NLP module incorporates named entity recognition techniques to identify and classify named entities in the user input. It can identify entities such as names of people, organizations, locations, medical terms, or other specific entities relevant to the healthcare domain. NER helps in extracting important entities that may be used for further analysis or to retrieve relevant information from the knowledge base.

Sentiment Analysis The NLP module may perform sentiment analysis to understand the sentiment or emotion expressed in the user input. It analyses the tone or polarity of the text, classifying it as positive, negative, or neutral. Sentiment analysis can provide insights into the user's emotional state or sentiment, which can be considered in generating appropriate responses.

Intent Recognition The NLP module identifies the user's intent or purpose behind the input. It aims to understand the user's goal or desired outcome from the conversation. Intent recognition helps in directing the conversation flow and determining the appropriate response strategy.

Contextual Understanding The NLP module considers the context of the conversation to enhance understanding and generate context-aware responses. It maintains the dialogue history and takes into account previous interactions to provide more relevant and coherent responses. Contextual understanding allows the chatbot to maintain a natural and continuous conversation with the user.

The NLP module plays a crucial role in deciphering user input, extracting meaningful information, and transforming it into a structured format that can be further processed by other modules. It incorporates various techniques to understand the syntax, semantics, and context of the input, enabling the chatbot to provide accurate and contextually relevant responses to user inquiries in the healthcare domain.

Decision Tree Module

The decision tree module forms the core of the healthcare chatbot's decision-making process. It includes the implementation of the decision tree algorithm, specifically the CART algorithm, to analyse patient inquiries and generate appropriate responses. This module involves constructing the

decision tree, training it using relevant data, and configuring hyperparameters to optimize its performance.
The Decision Tree module forms the core of your chatbot's decision-making process. It utilizes the CART (Classification and Regression Trees) algorithm, which is a decision treebased machine learning algorithm, to analyse the processed data from the NLP module and make informed decisions.
Here are some key aspects of the Decision Tree module

Feature Selection The Decision Tree module takes the processed data from the NLP module, which typically includes various features extracted from the user input. It selects the most relevant features to consider for decision-making. The selection of features can be based on their importance, relevance to the healthcare domain, or their ability to contribute to accurate decision-making.

Tree Construction The Decision Tree module constructs a decision tree based on the selected features and the corresponding target variable. It utilizes the CART algorithm, which recursively partitions the data based on different feature values, creating nodes and branches in the decision tree. The tree construction process involves determining the optimal splitting criteria for each node, such as Gini impurity or information gain, to maximize the predictive power of the tree.

Decision Making The Decision Tree module uses the constructed decision tree to make decisions based on the features of the user input. It follows the paths in the tree based on the values of different features, leading to a specific decision or action. The decisionmaking process involves evaluating the feature values at each node and traversing the tree until reaching a leaf node that represents a specific decision or recommendation. **Classification and Prediction** The Decision Tree module can perform both classification and prediction tasks. In classification, it assigns a predefined label or category to the user input based on the decision tree's paths. In prediction, it estimates a numerical value or outcome based on the feature values and their corresponding decision tree paths. **Model Evaluation and Optimization** The Decision Tree module can be evaluated and optimized to improve its performance and accuracy. Evaluation techniques, such as crossvalidation or performance metrics like accuracy, precision, recall, or F1 score, can be used to assess the effectiveness of the decision tree model. Optimization techniques, such as pruning or tuning hyperparameters, can be applied to enhance the decision tree's generalization ability and avoid overfitting.

Explainability The Decision Tree module provides explainability for the decisions made by the chatbot. As decision trees are interpretable models, the module can trace the decision-making process back to the specific feature values and paths in the tree. This transparency allows users to understand how the chatbot arrived at a particular decision or recommendation, enhancing trust and confidence in the chatbot's responses.

The Decision Tree module leverages the CART algorithm to analyse the processed data, construct a decision tree, and make decisions based on the features of the user input. It offers flexibility, interpretability, and the ability to handle both classification and prediction tasks. By employing model evaluation and optimization techniques, the Decision Tree module ensures accurate decision-making and enhances the overall performance of your healthcare chatbot.

5 IMPLEMENTATION AND TESTING

5.1 IMPLEMENTATION

Here's the algorithm for the given code

1. Import the necessary libraries, including pandas and DecisionTreeClassifier from scikit-learn.
2. Read the training data from a CSV file into a pandas Data Frame.
3. Split the data into input features (x) and the target variable (y).
4. Create an instance of the DecisionTreeClassifier algorithm.
5. Split the data into training and testing sets using the train_test_split function.
6. Train the classifier on the training data by calling the fit method on the classifier object, passing the training features (x_train) and target variable (y_train) as arguments.
7. Evaluate the classifier's performance on the testing data by calling the score method on the classifier object, passing the testing features (x_test) and target variable (y_test) as arguments. Store the accuracy score in a variable.
8. Print the accuracy score to the console.

5.1.1 CODE

Installing required libraries and Train the classifier on the training data by calling the fit method on the classifier object, passing the training features (x_train) and target variable (y_train) as arguments.

```
!pip install pyttsx3

# connecting google drive
from google.colab import drive
drive.mount('/content/drive')

##installing streamlit

!pip install streamlit


####main.py

%%writefile main.py
```

```
import streamlit as st
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)


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


def getDescription():
    global description_list
    with
open("/content/drive/MyDrive/minor/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("/content/drive/MyDrive/minor/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("/content/drive/MyDrive/minor/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():
    st.write("-----------------------------------HealthCare ChatBot---------------------
-------------")
    name = st.text_input("Your Name?")
    st.write("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("/content/drive/MyDrive/minor/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:
        disease_input = st.text_input("Enter the symptom you are experiencing")
        conf, cnf_dis = check_pattern(chk_dis, disease_input)
        if conf == 1:
            if len(cnf_dis) > 0:
                conf_inp = st.selectbox("Select the symptom you meant", cnf_dis)
                disease_input = conf_inp
                break
        else:
            st.write("Enter a valid symptom.")
    while True:
        try:
            num_days = int(st.text_input("From how many days have you been
experiencing it?", key="hello"))
            break
        except ValueError:
            st.write("Enter a 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])
            red_cols = reduced_data.columns
            symptoms_given =
red_cols[reduced_data.loc[present_disease].values[0].nonzero()]

            st.write("Are you experiencing any of the following symptoms?")
            symptoms_exp = []
            for syms in list(symptoms_given):
                inp = st.selectbox(syms, ["Yes", "No"])
                if inp == "Yes":
                    symptoms_exp.append(syms)
                elif inp == "No":
                    symptoms_exp.append("No " + syms)

            second_prediction = sec_predict(symptoms_exp)
            calc_condition(symptoms_exp, num_days)
            if present_disease[0] == second_prediction[0]:
                st.write("You may have " + present_disease[0])
                st.write(description_list[present_disease[0]])
            else:
                st.write("You may have " + present_disease[0] + " or " +
second_prediction[0])
                st.write(description_list[present_disease[0]])
                st.write(description_list[second_prediction[0]])

            precution_list = precautionDictionary[present_disease[0]]
            st.write("Take following measures:")
            for i, j in enumerate(precution_list):
```

```
            st.write(i + 1, ")", j)

    recurse(0, 1)


def main(severityDictionary, description_list, precautionDictionary, clf, cols,
le, training, reduced_data):
    st.sidebar.title("Options")
    menu = st.sidebar.selectbox("Select an option", ["Home", "ChatBot"])

    if menu == "Home":
        st.title("HELPI")
        getInfo()
    elif menu == "ChatBot":
        st.title("HealthCare ChatBot")
        st.write("----------------------------------HealthCare ChatBot-------------------
----------------")
        st.write("Enter your symptoms and get possible diagnoses.")
        st.write("Please make sure to enter symptoms correctly and answer the
questions accurately.")
        st.write("Let's get started!")

        tree_to_code(clf, cols)

    st.write("--------------------------------------------------------------------------------
-------")


if __name__ == "__main__":
    severityDictionary = dict()
    description_list = dict()
    precautionDictionary = dict()
    le = preprocessing.LabelEncoder()
    training = pd.read_csv("/content/drive/MyDrive/minor/Data/Training.csv")
    reduced_data = training.groupby(training['prognosis']).max()
    cols = training.columns[:-1]
    x = training[cols]
    y = training['prognosis']
    le.fit(y)
    y = le.transform(y)
    clf1 = DecisionTreeClassifier()
    clf = clf1.fit(x, y)
    getDescription()
    getSeverityDict()
    getprecautionDict()
    main(severityDictionary, description_list, precautionDictionary, clf, cols, le,
training, reduced_data)


##for connecting streamlit
print("############")
print("IPv4")
!curl https://ipv4.icanhazip.com/
print("############")

##running main.py

!streamlit run main.py & npx localtunnel --port 8501
```

5.2 RESULTS

The proposed project delivered the following results

1. Displays the predicted disease.
2. Displays the measures to be taken.

The proposed healthcare chatbot- HELPI takes symptoms provided by user as input and displays predicted disease and measures to be taken as the output.

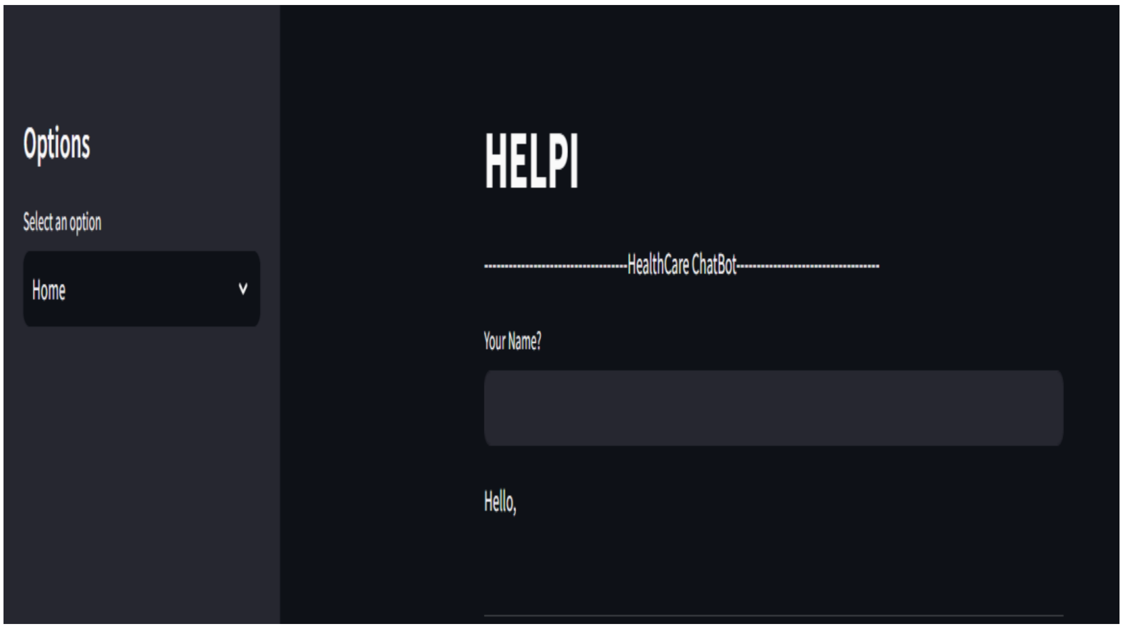
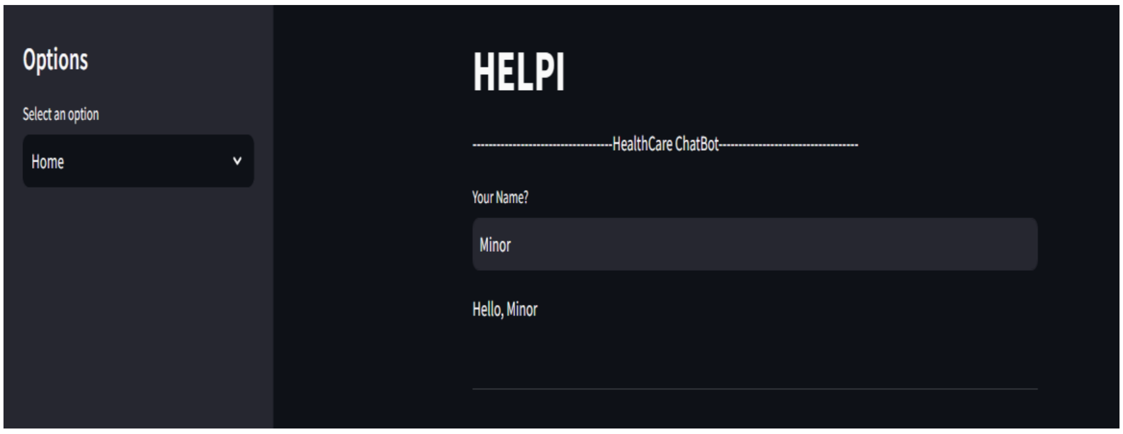

Options
Select an option
Home
HELPI
----------------------HealthCare ChatBot----------------------
Your Name?
Hello,

Fig. 5.1 User interface



HELPI

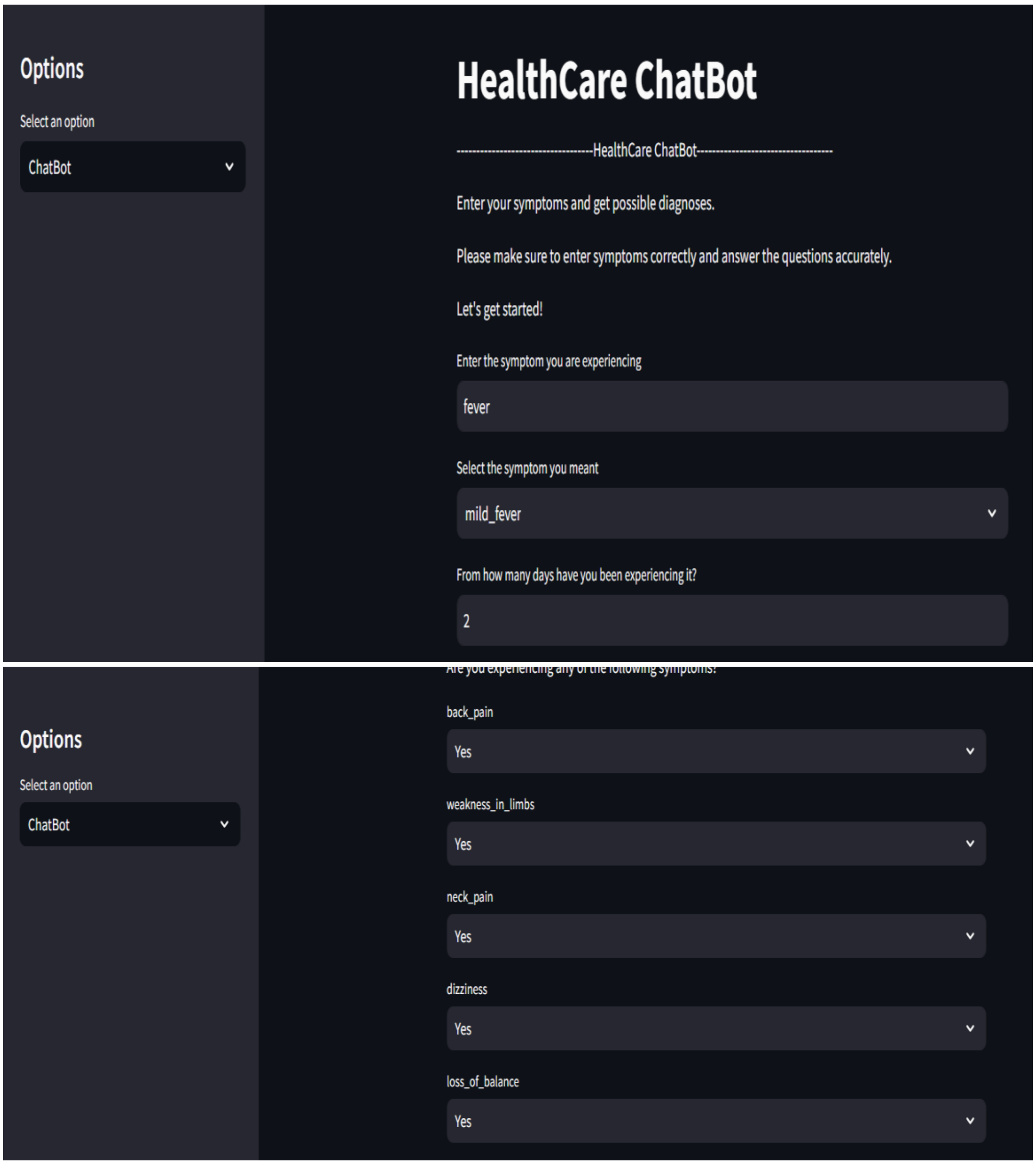
--------------------------------HealthCare ChatBot--------------------------------

Your Name?

Minor

Hello, Minor

Fig. 5.2 Welcome message



HealthCare ChatBot

--------------------------------HealthCare ChatBot--------------------------------

Enter your symptoms and get possible diagnoses.

Please make sure to enter symptoms correctly and answer the questions accurately.

Let's get started!

Enter the symptom you are experiencing

fever

Select the symptom you meant

mild_fever

From how many days have you been experiencing it?

2

Are you experiencing any of the following symptoms?

back_pain

Yes

weakness_in_limbs

Yes

neck_pain

Yes

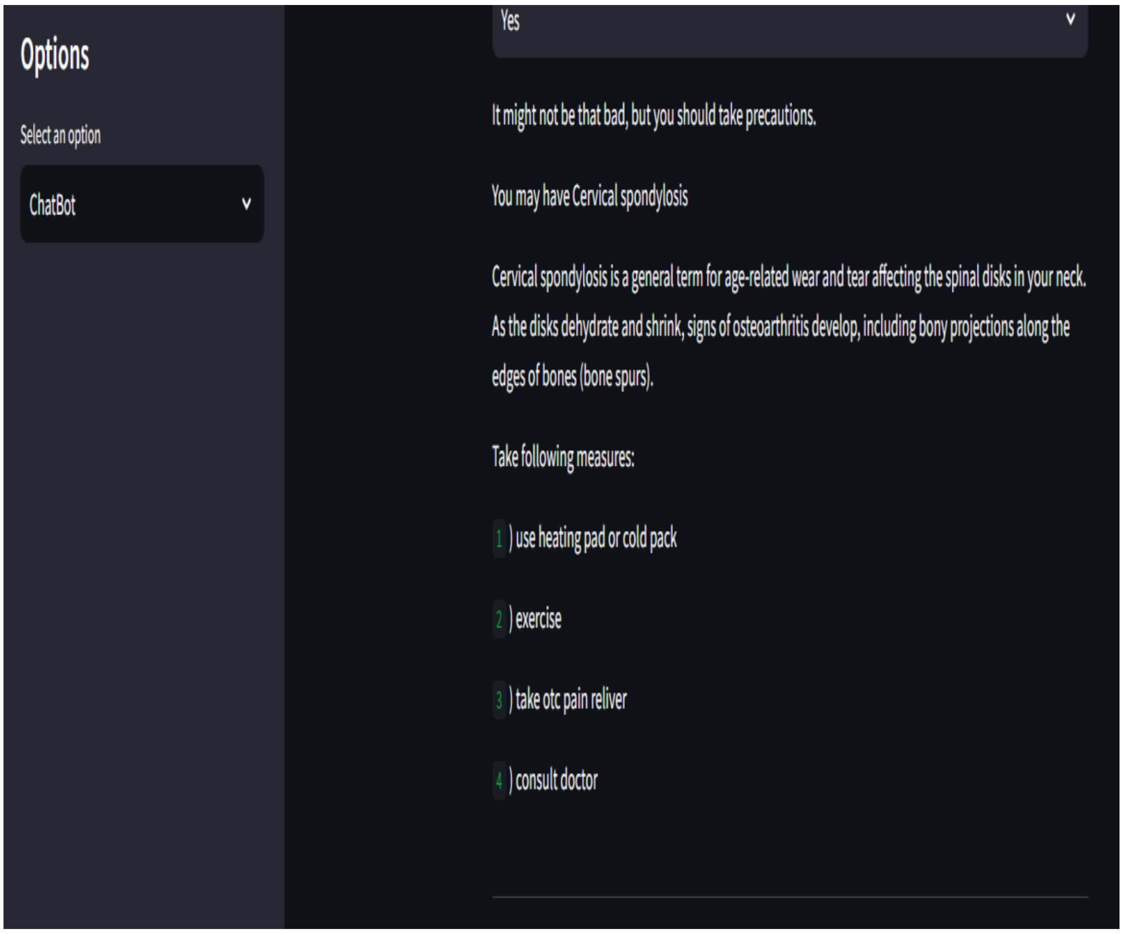
dizziness

Yes

loss_of_balance

Yes

Fig. 5.3 User input



It might not be that bad, but you should take precautions.

You may have Cervical spondylosis

Cervical spondylosis is a general term for age-related wear and tear affecting the spinal disks in your neck. As the disks dehydrate and shrink, signs of osteoarthritis develop, including bony projections along the edges of bones (bone spurs).

Take following measures:

1) use heating pad or cold pack

2) exercise

3) take otc pain reliver

4) consult doctor

Fig. 5.4 output

The healthcare chatbot is designed to assist users in identifying potential diseases based on the symptoms they provide. After analysing the user's input, the chatbot predicts the most likely disease and provides a set of recommended measures to be taken.

The output provided by the chatbot includes two main components the predicted disease and the recommended measures.

The predicted disease is the result of the chatbot's analysis of the symptoms provided by the user. It determines the disease that best matches the given symptoms based on a predefined database or algorithm. For example, if the user enters symptoms such as fever, cough, and sore throat, the chatbot may predict a possible diagnosis of 'Common Cold' or 'Influenza'.
In addition to the predicted disease, the chatbot also offers a set of recommended measures to be taken. These measures are tailored to the predicted disease and aim to guide the user in managing their condition effectively. The recommended measures may include general advice, specific actions, lifestyle modifications, or medications. For instance, if the chatbot predicts the user has 'Common Cold', it may suggest measures such as getting

plenty of rest, staying hydrated, taking over-the-counter cold medication, and practicing good hygiene to prevent the spread of the virus.

It is important to note that the output provided by the chatbot is for informational purposes only and should not substitute professional medical advice. Users are advised to consult healthcare professionals for accurate diagnosis and personalized treatment recommendations based on their specific medical history and condition.

6 CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

Our healthcare chatbot is an innovative and user-friendly tool designed to assist individuals in assessing their symptoms, predicting potential diseases, and providing relevant measures to be taken. By simply inputting their symptoms into the chatbot, users can receive valuable insights and recommendations to better understand their health condition. The chatbot utilizes machine learning algorithms and a comprehensive medical database to analyse the symptoms provided by the user. It employs a decision tree model, to predict the most probable disease based on the symptoms reported. This predictive capability allows for early detection and intervention, leading to improved health outcomes.

Once the disease is predicted, the chatbot offers a range of precautionary measures and recommendations to be taken. These measures may include lifestyle modifications, suggested treatments, self-care practices, and preventive strategies. The chatbot provides accurate and upto-date information about the recommended measures, ensuring users have access to reliable healthcare advice. Furthermore, our healthcare chatbot serves as an educational resource, offering users detailed information about various diseases. It also emphasizes the importance of proactive healthcare management, providing guidance on preventive care, healthy habits, and wellness tips.

The chatbot's user-friendly interface and conversational approach make it accessible to individuals of all technical backgrounds. It can be easily accessed, allowing users to seek medical guidance anytime, anywhere. With the ability to accurately predict diseases based on symptoms and provide relevant measures, our healthcare chatbot empowers individuals to make informed decisions about their health. It acts as a reliable virtual healthcare companion, offering personalized support and guidance throughout the user's healthcare journey.

In summary, our healthcare chatbot is a cutting-edge solution that combines machine learning techniques, medical expertise, and user-friendly interface to facilitate symptom assessment, disease prediction, and provision of necessary measures. It aims to improve healthcare accessibility, empower individuals, and promote proactive healthcare management.

6.2 FUTURE SCOPE

The future scope of healthcare chatbots is vast and holds great potential in transforming the healthcare industry.

Remote Monitoring and Telemedicine Chatbots can play a crucial role in remote patient monitoring, enabling healthcare providers to monitor patients' health conditions from a distance. Chatbots can collect and analyse patient data, provide real-time feedback, and assist in remote consultations, reducing the need for in-person visits and enhancing the efficiency of healthcare delivery.

Integration with IoT Devices With the growth of the Internet of Things (IoT), chatbots can be integrated with various IoT devices and sensors. This integration can enable chatbots to gather real-time health data, such as vital signs, glucose levels, or medication adherence, and provide personalized feedback and interventions.

Decision Support for Healthcare Professionals Healthcare chatbots can assist healthcare professionals in making informed decisions by providing them with quick access to relevant medical information, clinical guidelines, and treatment recommendations. Chatbots can also aid in triage and initial assessment, helping prioritize patient care based on symptom severity.

7 REFERENCES

[1] S. Smith ,"A Systematic Review of Chatbot Applications in Mental Healthcare", National Library of Medicine, doi: 10.3390/s22072625, 2022 Mar 29 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9003264/

[2] J. Brown, "An Empirical Study on the Effectiveness of Chatbots in Patient Education", doi 10.3390/su15075614 ,March 2023. https://www.mdpi.com/2071-1050/15/7/5614

[3] M. Johnson,"Exploring the Acceptance and Usability of Chatbots in Elderly Care Settings" , doi 10.1145/3544548.3580719,April 2023

https://dl.acm.org/doi/fullHtml/10.1145/3544548.3580719

[4] A. Martinez ,"Chatbots for Chronic Disease Management: A Review of Current Applications and Challenges" by, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8669585/

[5] Lekha Athota, Vinod Kumar Shukla, Nitin Pandey, Ajay Rana, "Chatbot for Healthcare System Using Artificial Intelligence", **ISBN:**978-1-7281-7017-615,September 2020

https://ieeexplore.ieee.org/document/9197833

[6] R. Garcia ,"Enhancing Patient Engagement through Intelligent Chatbots in Primary Care" , 3.4.2023 , Vol 25

https://www.jmir.org/2023/1/e43293/