# THE FUTURE OF AYURVEDA: HARNESSING THE POWER OF ARTIFICIAL INTELLIGENCE FOR PERSONALIZED TREATMENT AND DIAGNOSIS

Maduwantha K.A.I

IT20069186

B.Sc. (Hons) Degree in Information Technology Specialized in Software Engineering

Department of Computer Science and Software Engineering

Sri Lanka Institute of Information Technology Sri Lanka

September 2023

## DECLARATION

I declare that this is my own work, and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

| **Name** | **Student ID** | **Signature** |
|---|---|---|
| Maduwantha K.A.I | IT20069186 | Hmaduwantha |

Signature of the Supervisor
(Dr. Darshana Kasthurirathna)

Date

…………………………….

..……………………….

## ABSTRACT

The rapid and busy nature of contemporary living often results in an uneven distribution of daily activities, inadequate dietary habits, insufficient physical exercise and leisure time, and excessive work-related stress, which can lead to poor health and dissatisfaction. While Ayurveda provides alternative solutions for many non-communicable diseases, people may have difficulties identifying the appropriate herbs and treatments and consulting with doctors in a timely and cost-effective manner. Additionally, the high cost of Western medicine may be a barrier, and not all illnesses are treatable. The proposed solution aims to assist users in discovering interactive Ayurvedic-based treatments for various symptoms. This solution is anticipated to be beneficial for those seeking alternatives to conventional medicine.

There will be a chat bot in the proposed solution which can discuss health related knowledge. Content sharing inside here which are specifically relating to health will be collected and stored in the knowledgebase. They will be used to service chatbot implementation by labeling as community knowledge. To maintain the consistency of the proposed solution it will be using auto-machine learning, to service users with up-to-date data inside the social network as well as to help in users with symptoms which can be added later. It will be required experienced Ayurvedic-related expertise resources. For supervised machine learning algorithm-based chatbot and we expect to collect data through various publicly available social network communities relating to health.

# TABLE OF CONTENTS

## ACKNOWLEDGEMENT

First and foremost, I want to sincerely thank my mentor Dr. Darshana Kasthurirathna for his unwavering support and direction, which enabled me to successfully complete my undergraduate research. Along with my supervisor, Dr. Samantha Rajapaksha, the co-supervisor of this research project deserves my deepest gratitude for always being available to assist. Due to the fact that this research project combines technology and ayurveda, both technology specialists and ayurvedic professionals were needed for advice and support. It is very appreciated that Dr. Mrs. Janaki Bandara who provided such extensive help throughout the project to close the knowledge gap in those fields. My sincere gratitude also goes out to the other doctors of Gampaha Wikramarachchi Ayurvedic University. Last but not least, I would like to convey my thanks to everyone who has helped with this project in some way, whether directly or indirectly, including my teammates, family, and friends.

## LIST OF FIGURES

## LIST OF TABLES

## LIST OF ABBREVATIONS

| **Abbreviation** | **Description** |
|---|---|
| AI | Artificial Intelligence |
| ML | Machine Learning |
| NLP | Natural Language Processing |
| EHR | Electronic Health Records |
| AML | Auto Machine Learning |
| CNN | Convolutional Neural Network |
| GPU | Graphical User Interface |
| CPU | Central Processing Unit |
| SDLC | Software Development Life Cycle |

## LIST OF APPENDICES

# 1. INTRODUCTION

## 1.1Background & Literature survey

Ayurveda is an ancient healthcare system originating in India that emphasizes a holistic approach to health and wellness. It employs natural remedies, lifestyle modifications, and personalized treatments based on an individual's dosha, or body type. [1] Ayurveda has gained popularity worldwide due to its effectiveness and emphasis on prevention and personalized care. However, the fast-paced modern lifestyle has led to an increasing prevalence of non-communicable diseases, which require a more specialized and personalized approach to treatment. This has prompted researchers and practitioners to explore the integration of Ayurvedic principles with modern technologies such as Artificial Intelligence (AI) to provide personalized and effective treatment. [2]

According to our research, Sri Lankan people have a long-standing cultural and traditional relationship with Ayurveda. Ayurveda has been a part of the country's history for over 3,000 years, and its usage is deeply embedded in Sri Lankan culture. In fact, Ayurvedic treatments have been used in Sri Lanka for thousands of years to prevent and cure various ailments. Ayurveda's popularity in Sri Lanka can be attributed to its holistic approach to healthcare, which emphasizes the balance between mind, body, and spirit. Sri Lankan people have long appreciated the natural and organic nature of Ayurveda, which uses herbs and other natural remedies to treat various health issues. Sri Lankan people prefer Ayurvedic remedies over modern medicine, especially for minor illnesses and chronic health conditions.

Furthermore, Ayurveda has a strong presence in Sri Lanka's healthcare system, with Ayurvedic hospitals and clinics being available throughout the country. Sri Lankan people have access to trained and experienced Ayurvedic doctors, who provide personalized treatments and care for their patients. These doctors use Ayurvedic remedies and traditional healing techniques to treat a wide range of ailments, from common colds to more serious health conditions like cancer and diabetes. In recent years, there has been a renewed interest in Ayurveda in Sri Lanka, as people are becoming more health-conscious and looking for natural and organic alternatives to modern medicine. The government of Sri Lanka has also recognized the importance of Ayurveda in the country's healthcare system and has taken steps to promote and develop it. [3]

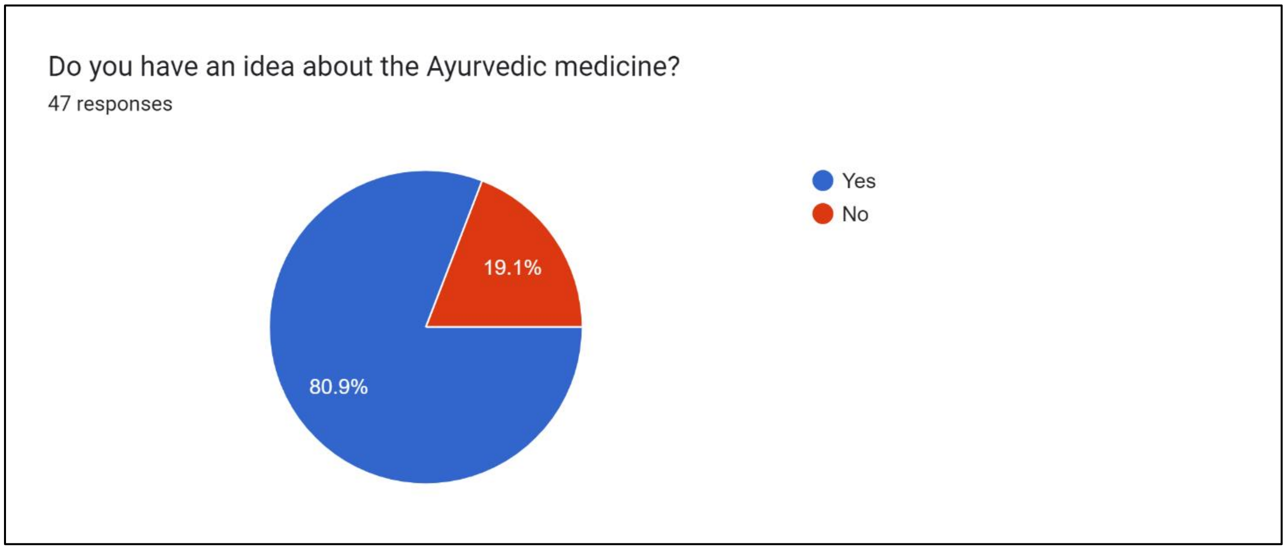


*Figure 1:Survey on idea of people about ayurvedic medicine*

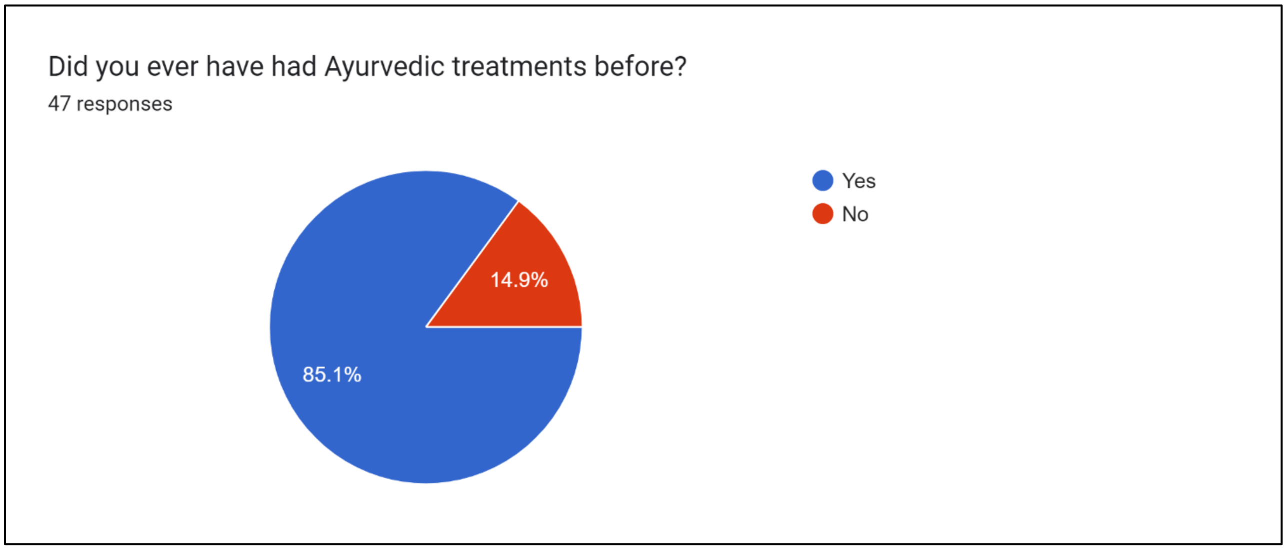


*Figure 2:Survey on having ayurvedic treatments before.*

From the above two survey results it shows that most of the people in Sri Lanka are aware about the ayurvedic medicine and most people had used them even once in their lifetime. Considering the percentages more than 80% of people are aware of ayurvedic medicine.

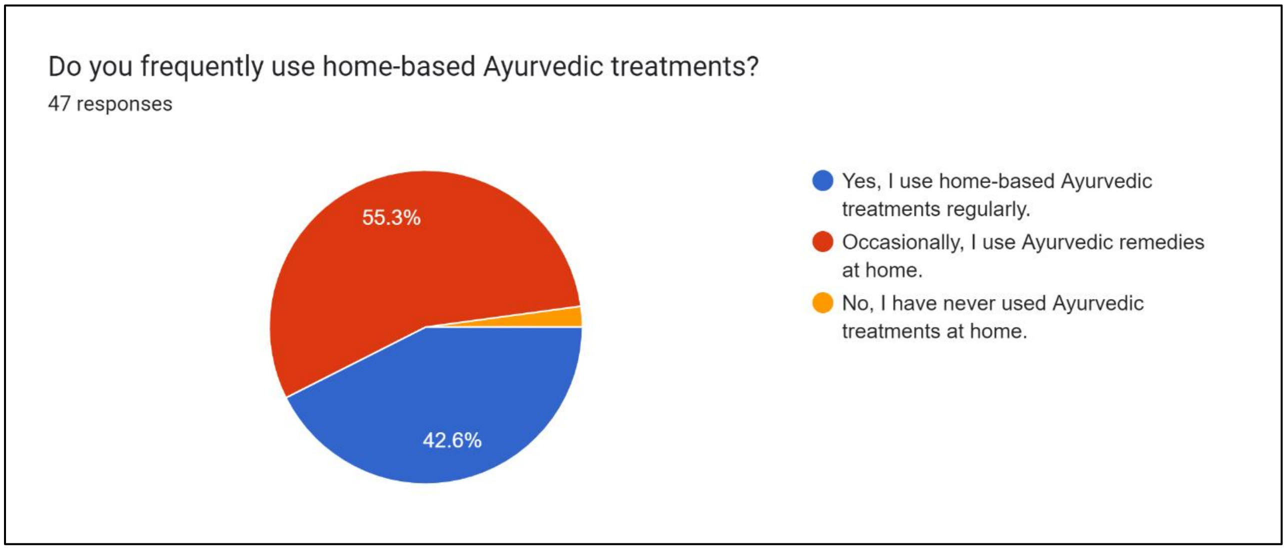


*Figure 3:Survey on the frequency of home based ayurvedic treatments usage*

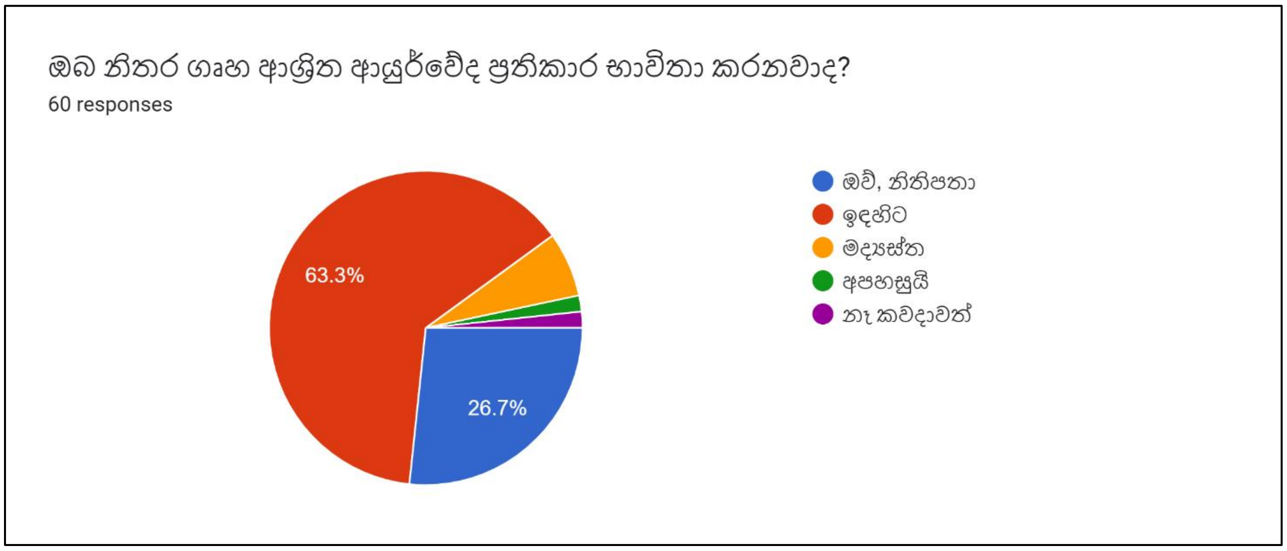


*Figure 4:Survey in Sinhala on the frequency of home based ayurvedic treatments usage*

Based on the survey results, it can be concluded that more than half of the respondents are frequently using ayurvedic treatments.

Overall, our research indicates that Sri Lankan people have a deep appreciation for Ayurveda and its holistic approach to healthcare. The popularity of Ayurveda in Sri Lanka is expected to continue to grow as people seek out natural and organic alternatives to modern medicine.

Artificial intelligence (AI) is revolutionizing the healthcare industry by offering new and innovative solutions for diagnosis, treatment, and monitoring of diseases. One of the most significant advantages of AI in healthcare is its ability to analyze vast amounts of patient data and medical records to aid in the diagnosis of diseases. AI algorithms can identify patterns in medical images such as CT scans, MRIs, and X-rays to assist radiologists in identifying potential abnormalities or diseases.

Additionally, AI can help doctors create personalized treatment plans for patients based on their unique characteristics and medical history, by analyzing data on previous treatment outcomes and patient health records. AI can also aid in the discovery of new drugs and treatments by analyzing large datasets to identify patterns and provide insights. Furthermore, AI-powered chatbots and voice assistants can provide initial diagnosis and recommend next steps. AI can also be used to remotely monitor patients and alert doctors to potential issues, analyze electronic health records (EHR) to identify potential issues, and provide personalized health advice and recommendations through virtual assistants. Overall, AI has the potential to significantly improve patient outcomes and provide more personalized and efficient healthcare solutions.

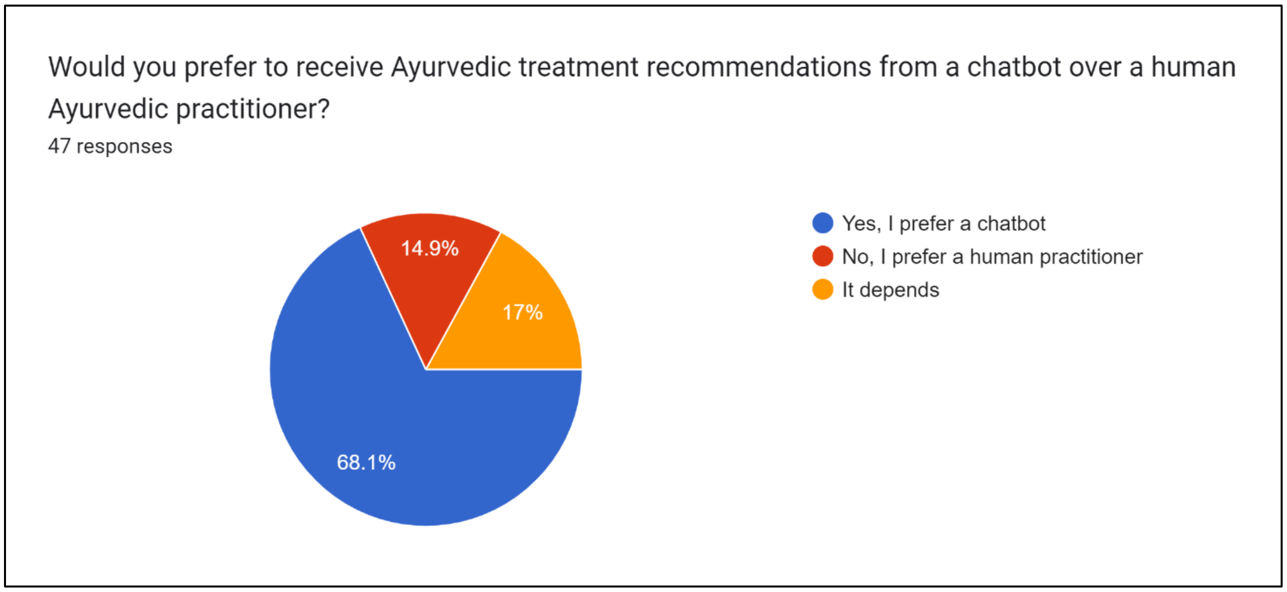


*Figure 2:Survey on the preference recommendations from a chatbot over a human doctor.*

According to the survey results more than half of respondents are preferring to have ayurvedic treatment recommendations from a chat bot over a human doctor.

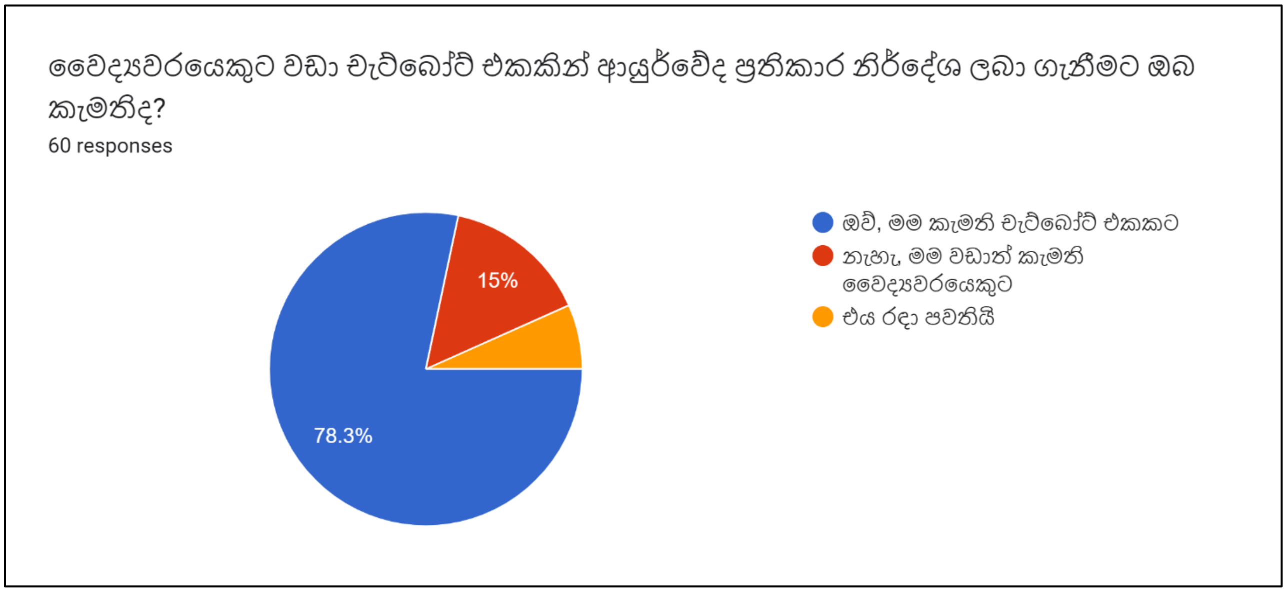


*Figure 3:Survey in Sinhala on the preference recommendations from a chatbot over a human doctor.*

The results suggest that there are no clear consequences on whether respondents would prefer to receive Ayurvedic treatment recommendations from a chatbot or a human practitioner. It highlights the importance of considering the preferences and concerns of individuals when implementing AI technology in healthcare.

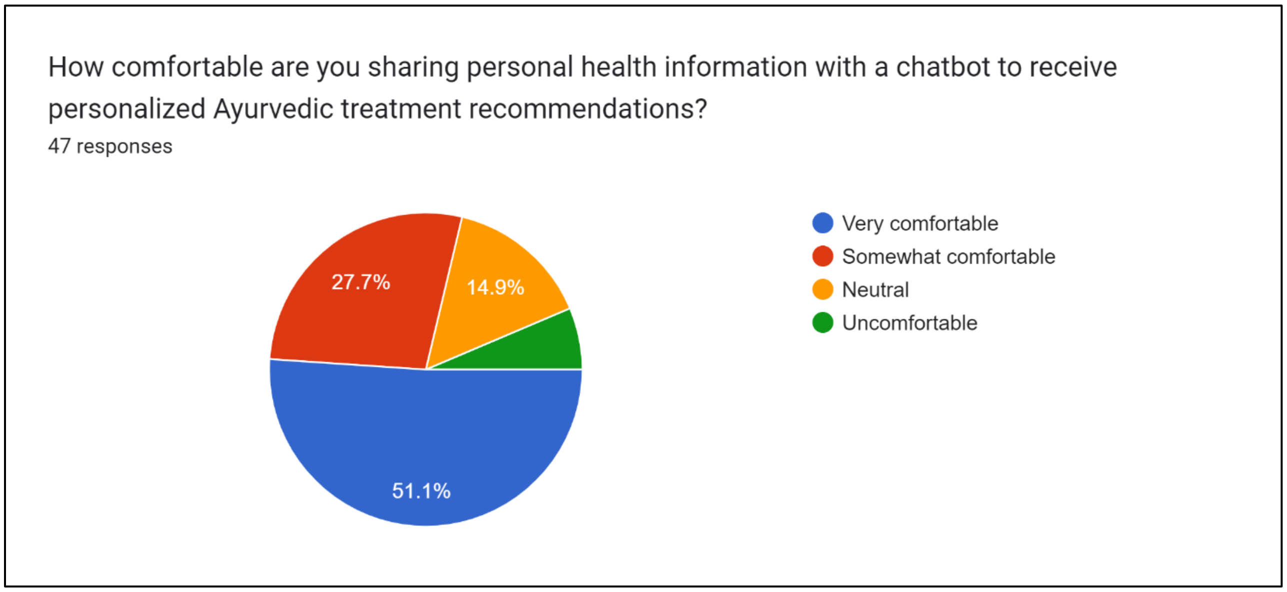


*Figure 4:Survey on the how comfortable are people sharing personal health information with a chatbot.*

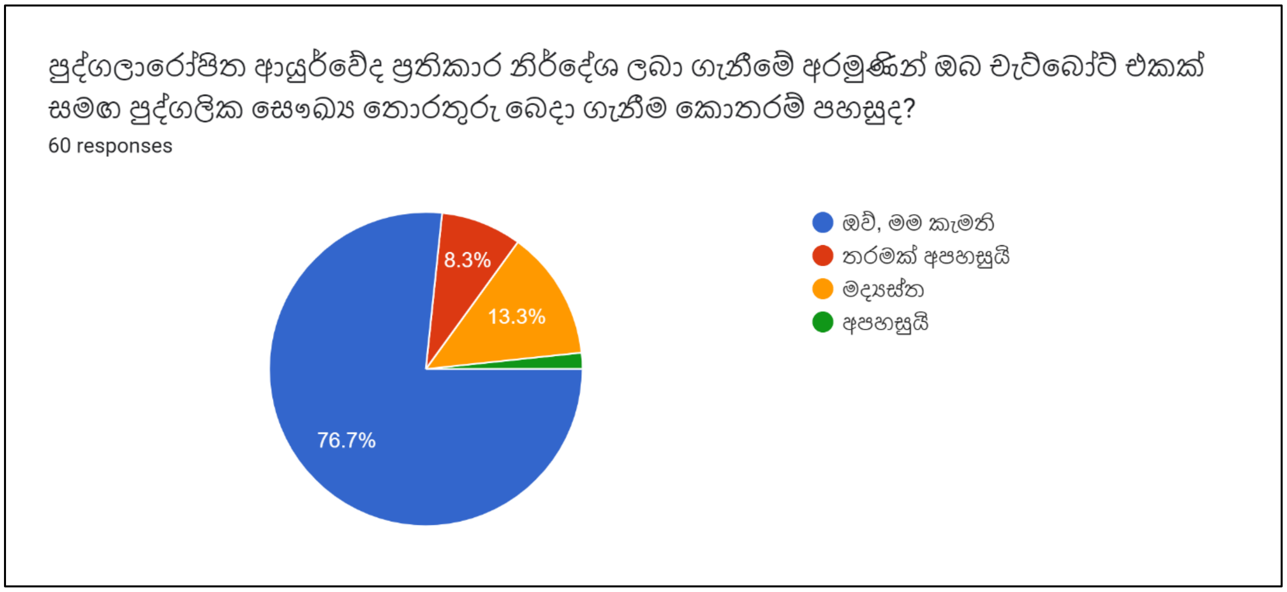


*Figure 5:Sinhala survey on how comfortable sharing personal health information with a chatbot.*

AI has the potential to revolutionize the field of Ayurveda by enabling personalized treatment based on a patient's unique characteristics, history, and symptoms. By analyzing vast amounts of patient data, AI algorithms can identify patterns and provide insights that can help practitioners make informed treatment decisions. Several Ayurveda-based apps have already been developed that utilize AI to provide personalized treatment recommendations. One such app is AyurMana, which uses AI algorithms to analyze a patient's pulse, tongue, and symptoms to provide personalized treatment recommendations. Another app, Wealthy, uses AI-powered chatbots to provide personalized health advice and recommendations.

The integration of AI in Ayurveda has several benefits. Firstly, it enables practitioners to provide personalized treatment based on a patient's unique needs, which can lead to better outcomes and reduced healthcare costs. Secondly, AI can help identify patterns and insights that can improve diagnosis and treatment of diseases. Finally, AI-powered apps and tools can increase patient engagement and adherence to treatment. Several AI technologies can be used in Ayurveda, such as machine learning, natural language processing, and computer vision. Machine learning can be used to identify patterns and provide personalized treatment recommendations. Natural language processing can enable chatbots and voice assistants to provide personalized health advice and recommendations. Computer vision can be used to analyze images of patients to identify symptoms and provide insights.

Here are a few of the ayurvedic apps that are already available. Yes, these smartphone applications offer Ayurveda remedies.

- E-procto
- MocDocHMS
- MocDoc Clinic Management System
- MyOPD
- Vaidya Manager
- Healcon Practice
- Easy Clinic
- OptiMantra

These apps provide users with access to Ayurvedic solutions and information on Ayurvedic remedies, treatments, and lifestyle modifications. They can be useful for individuals seeking natural remedies and personalized treatment options for various health conditions.

**According to the research background and survey discussed above** we propose an innovative solution to promote a healthier lifestyle through Ayurveda, as outlined in the previous section. Our solution aims to address Ayurvedic treatments and common healthy guidelines for specific symptoms, such as arthritis, blood sugar, hair loss, infertility, obesity, paranasal sinusitis, cuts/scratches/swellings. Our solution will include a conversational AI chatbot, which will provide a user-friendly platform for individuals to receive personalized solutions and advice through text, based on a knowledge base containing information on these symptoms and their treatments.

In addition to the chatbot, our proposed solution will also feature an image processing component that can identify the herbal plants needed for the treatment of these diseases. A geometry library will map out the locations where these herbs are available and connect patients with Ayurvedic doctors within a specific geographical area. Users can rate doctors to help others find reliable practitioners.

Our solution will also include a social network where users can discuss health-related knowledge. Content shared specifically relating to health will be collected and stored in the knowledgebase for the chatbot to reference, labeled as community knowledge. Auto-machine learning will help maintain the consistency of the solution by keeping the social network up-to-date and helping users with any additional symptoms.

To ensure the accuracy and effectiveness of our solution, we will require experienced Ayurvedic-related expertise resources. We expect to collect data through publicly available social network communities relating to health and images of herbs, as well as details of Ayurvedic doctors with their locations from relevant backgrounds for supervised machine learning algorithm-based solutions such as herb identification, chatbot, and social network implementation.

In conclusion, the integration of AI in Ayurveda has the potential to transform the field of personalized healthcare. By harnessing the power of AI, practitioners can provide personalized treatment and improve diagnosis and treatment of diseases. The development of AI-powered apps and tools can increase patient engagement and adherence to treatment, leading to better health outcomes. With the increasing prevalence of non-communicable diseases, the integration of AI in Ayurveda is becoming increasingly important and holds great promise for the future of healthcare.

## 1.2 Research Gap

Although Ayurveda offers a wealth of knowledge and effective solutions for various health issues, the implementation of AI technology in Ayurvedic medicine is still in its early stages. There is a significant research gap in Ayurveda and AI integration, particularly in the development of personalized treatment and lacks conversational AI chatbots that allows users to text and receive solutions for non-communicable diseases through Ayurveda. The knowledge bases are not updated and improved.

While there are some existing Ayurvedic mobile apps that offer basic solutions and advice, they often lack the level of personalization and sophistication that can be achieved with AI. Furthermore, there is a lack of standardization in Ayurvedic medicine, making it challenging to develop algorithms that can effectively analyze and interpret Ayurvedic data.

Another research gap is the integration of AI into the manufacturing and quality control of Ayurvedic products. The lack of standardization in Ayurvedic medicine also poses a challenge in ensuring the safety and efficacy of Ayurvedic products. The integration of AI can help to ensure the quality and consistency of Ayurvedic products by analyzing data on the ingredients and manufacturing processes.

**Research A -** Divya S [4] highlights the Future Scope of Medical Chatbots for Personalized Medicine with the user-friendliness and accessibility of medical chatbots, which can be utilized by individuals familiar with typing in their native language. The study emphasizes the effectiveness of medical chatbots in providing personalized diagnoses based on symptoms. However, expanding the bot's symptom recognition capabilities by incorporating factors like location, duration, and intensity of symptoms could enhance its performance. AI algorithms and training data are identified as critical components of personalized medical assistant implementation, offering the potential to save lives and raise medical awareness. The paper envisions the extensive future scope of medical chatbots, particularly in the era of messaging apps, for convenient and widespread medical conversations.

**Research B** - Rohit Binu Mathew [5] presents a Medical Chatbot as a Diagnostic Assistant that can replace conventional disease diagnosis methods. It outlines the chatbot's functionality as a user application where users input symptoms, and the chatbot provides health measures and potential diagnoses. The study underscores the importance of accurate symptom analysis and treatment recommendations, especially in a busy lifestyle. The chatbot's ability to offer personalized assistance without needing a physician's intervention is highlighted, along with its cost-effectiveness. The chatbot's personalized and private user interaction is crucial to efficient disease identification.

**Research C** -Jahnvi Gupta [6] focuses on Virtual doctors through Chatbots for Health Management, the roles of virtual doctors, and catering to individuals with time constraints. The paper highlights the chatbot's capacity to provide healthcare steps based on user input symptoms. The dataset's general information about symptoms and illnesses enables the chatbot to offer users relevant information. The study emphasizes the importance of user openness in sharing health matters and praises the cost-effectiveness of chatbots. The convenience of diagnosing diseases with a single button click is a significant advantage.

**Research D** - Prakhar Srivastava [7] Automation of Medical Chatbot for Personalized Recognition describes the development of an automated medical chatbot that offers personalized recognition based on symptoms. The chatbot employs an external recognition engine, hinting at the potential need to develop an in-house engine to enhance diagnosis functionality. The reviewed research studies collectively underscore the transformative potential of medical chatbots in disease diagnosis and treatment recommendation.

These studies emphasize the user-friendly nature of chatbots, their ability to provide personalized assistance, and the cost-of Identifying deployment. While the studies offer promising insights into the capabilities of medical chatbots, future work should focus on refining symptom recognition algorithms, optimizing chatbot training data, and developing comprehensive diagnostic engines to harness their potential.

| Features | Research A | Research B | Research C | Proposed System |
|---|---|---|---|---|
| Ayurvedic Chatbot | X | X | X | ✓ |
| Availability of personalized solutions | X | ✓ | ✓ | ✓ |
| User Friendly Platform | ✓ | X | ✓ | ✓ |
| Availability of a knowledge base | ✓ | X | ✓ | ✓ |
| Chatbot framework | ✓ | ✓ | ✓ | ✓ |
| Ability to ask to follow back questions | ✓ | X | X | ✓ |
| Using RASA | ✓ | ✓ | X | ✓ |
| Updating Knowledgebase Easily | ✓ | ✓ | ✓ | ✓ |
| Technology Used | N-GRAM | CNN | CNN | RNN |

*Table 1: Research Comparisons*

Overall, there is also a significant research gap in the integration of AI in Ayurvedic medicine, particularly in the development of personalized treatment and diagnosis and the quality control of Ayurvedic products. More research and development in this area can lead to significant advancements in the field of Ayurvedic medicine and better health outcomes for individuals.

# 2.RESEARCH PROBLEM

1. How to predict ayurvedic treatments for diseases through a chat, with the help of user with a chat bot?

2. How to update an ayurvedic knowledge base by a practitioner who doesn't have knowledge in technology?

The research problem is the lack of an efficient and accurate solution for identifying Ayurvedic medical herbs that are required for treating diseases using image processing and machine learning models. The current research gap lies in the lack of studies that combine the principles of Ayurveda with modern AI techniques to develop a robust solution for identifying medicinal plants.

Based on the limited availability of herbal plants and medicines, combined with the lack of knowledge and resources for identifying and locating these plants, there is a need for an AI-based solution that can accurately identify Ayurvedic medical herbs used for the treatment of noncommunicable diseases through image processing, while also mapping their locations. Additionally, to improve the accuracy of the system, continual learning and transfer learning techniques can be employed, and the system can be made more accessible through the use of Auto Machine Learning to enable the addition of new plants. Therefore, the research problem is to develop an AI-based solution that can accurately identify Ayurvedic medical herbs and their locations while utilizing continual learning and Auto Machine Learning (AML) techniques to improve accuracy and accessibility.

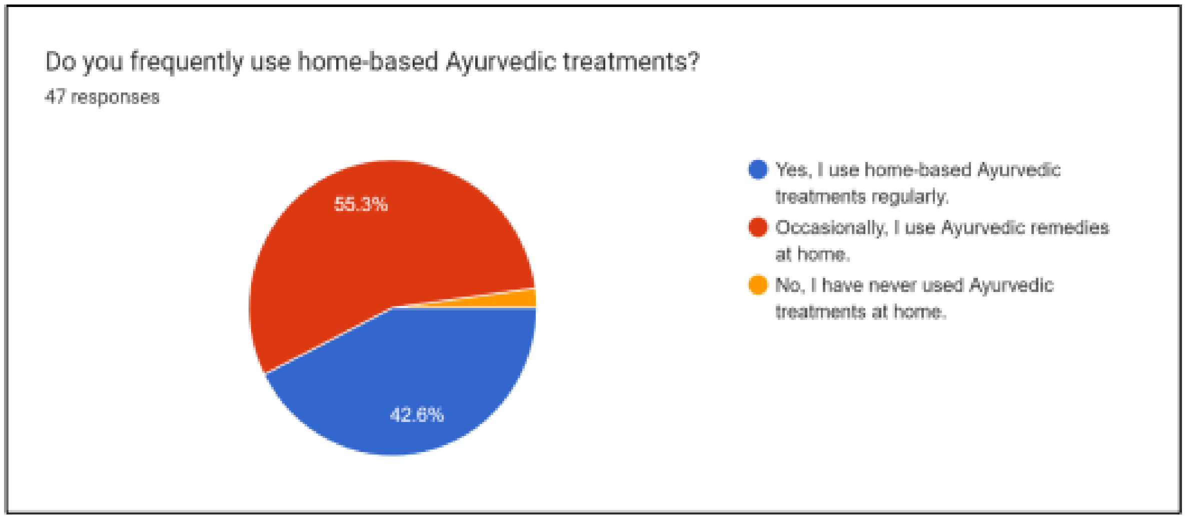


*Figure 6: Survey on how frequently use home based treatments*

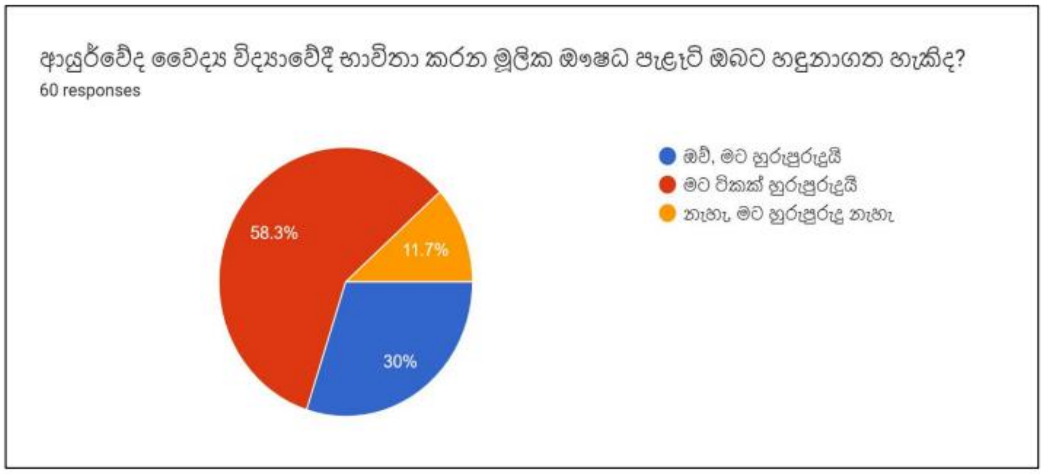


*Figure 7: Survey on capability to identify basic ayurvedic herbs*

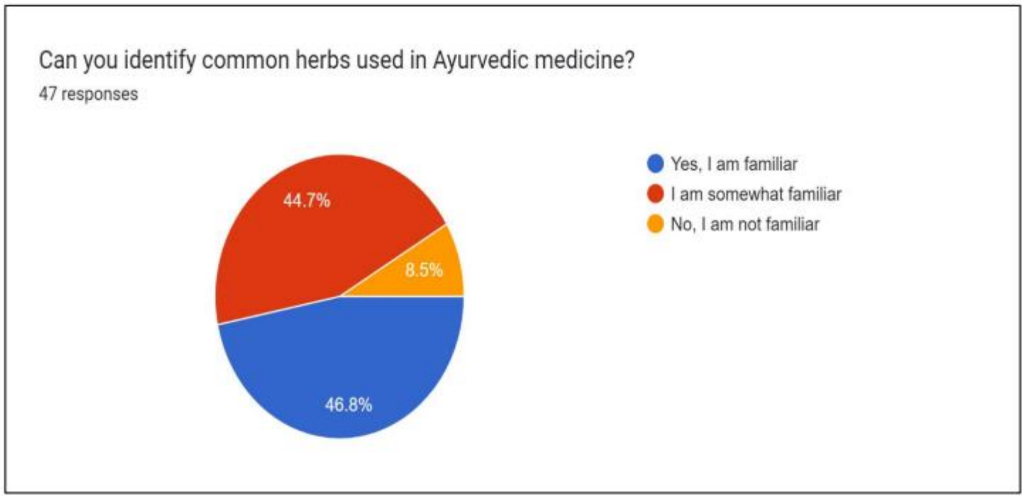


*Figure 8: Survey on capability to identify herbs*

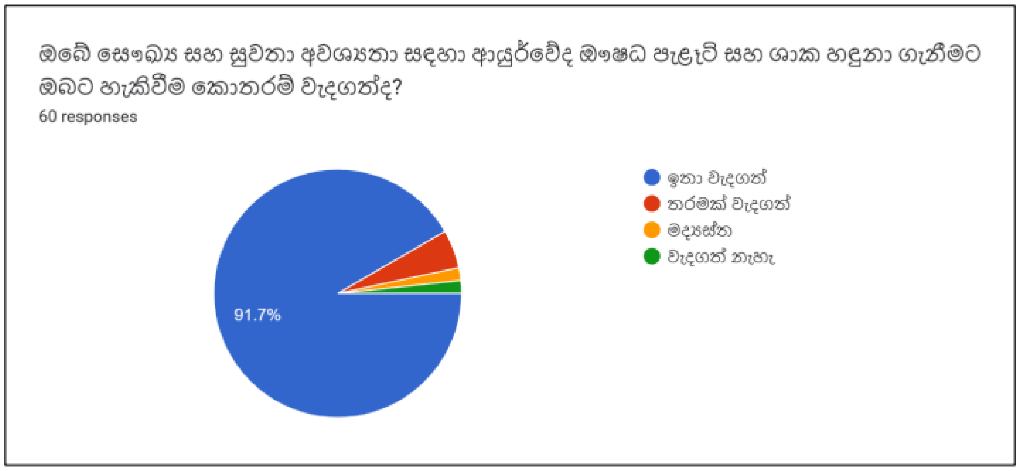


*Figure 9: Survey on how important ayurvedic herbs to people on daily basis*

# 3.RESEARCH OBJECTIVES

## 3.1 Main Objectives

The primary objective of this study is to develop a user-friendly mobile application dedicated to promoting a healthier lifestyle through Ayurveda, specifically focusing on Ayurvedic treatments and lifestyle guidelines for common symptoms such as arthritis, blood sugar management, hair loss, infertility, obesity, paranasal sinusitis, and minor injuries. This application will feature a conversational AI chatbot, utilizing a robust knowledge base derived from trusted Ayurvedic sources to offer personalized recommendations and herbal remedies based on users' specific symptoms. It will also provide valuable dietary and lifestyle guidance while emphasizing the importance of consulting healthcare professionals for comprehensive health management. Regular updates, user engagement features, and strict adherence to legal and ethical considerations will be integral to ensuring the application's effectiveness and user satisfaction.

## 3.2 Specific Objectives

The development process for the proposed software application focused on non-communicable diseases (NCDs) and Ayurvedic treatments involves several critical stages:

**1. Research and Data Gathering:** Thorough research is essential to gather comprehensive information on selected non-communicable diseases, including their causes, symptoms, and conventional medical treatments. Additionally, gathering Ayurvedic knowledge related to these diseases is crucial, as it forms the foundation for the AI model.

**2. Knowledge Base Creation:** The collected information is structured and organized to create a robust knowledge base. This database contains data on symptoms, common health problems associated with each NCD, and the corresponding Ayurvedic treatments and medicinal herbs for these conditions.

**3. Data Pre-processing and Cleaning:** Data pre-processing involves cleaning and formatting the information to make it suitable for AI modeling. This step ensures data consistency and accuracy by handling missing values, outliers, and inconsistencies.

**4. AI Model Training:** With a prepared database, the AI model is trained using machine learning techniques. The model learns to identify patterns and relationships between symptoms, common health problems, and Ayurvedic treatments based on the provided data.

**5. Evaluation and Validation:** To assess the AI model's performance, it is tested on a validation dataset. Evaluation metrics are used to measure its accuracy, precision, recall, and other relevant criteria. This step helps identify areas for improvement.

**6. Fine-Tuning:** Based on the evaluation results, the model is fine-tuned to enhance its accuracy and effectiveness. This iterative process may involve adjusting hyperparameters, optimizing algorithms, and expanding the dataset.

**7. Software Application Development:** The AI model is integrated into a user-friendly software application or website. This application will serve as a platform for users to interact with the AI system and receive personalized Ayurvedic recommendations for managing NCDs.

**8. Verification and Expert Consultation:** The software's accuracy and validity are verified by consulting Ayurvedic experts and practitioners. Their feedback and expertise ensure that the recommendations align with traditional Ayurvedic principles and are safe and effective.

**9. Conversational AI Chatbot:** A key feature of the software application is the conversational AI chatbot. This chatbot allows users to engage in text-based conversations, providing a user-friendly and accessible interface for receiving personalized solutions and advice for NCDs through Ayurveda.

By following these steps, the software application aims to bridge the gap between modern healthcare and Ayurveda, offering users a reliable resource for managing non-communicable diseases using holistic and traditional Ayurvedic approaches. The integration of AI technology and expert verification ensures the accuracy and effectiveness of the recommendations provided to users, ultimately contributing to improved health and well-being.

# 4.METHODOLOGY

## 4.1 Requirement gathering and Analysis.

- **Collecting information from Gampaha Wickramarachchi Ayurvedic University**

To collect information on ayurveda and diseases we met **Dr. Janaki Wickramarachchi**, who is the Dean of Chikisthsaka Faculty at **Gampaha Wickramarachchi Ayurvedic University** and had conducted some online meetings conducted with her, with participation of our group members. She agreed to provide us the necessary information related to the ayurveda and the research gap which is having when connecting with modern technologies such as Artificial Intelligence and Machine Learning. She highlighted several main diseases which are fine for the research. She gave us legal approval for the continuation of the research and gave advice about the things we need to focus on in the future while continuing the project.

- **Data gathering**

Firstly, we read a dozen published research for initial understanding and got some basic idea by reading and browsing through few articles. Our supervisors had few meetings with us to discuss the initial methods for data gathering and the external supervisor connected us with a few ayurvedic specialties and pointed out the diseases and the data we are needed continue with. In future the other necessary data and images will be collected from the University of Gampaha as necessities.

- **Conducting a survey**

To get an idea about the knowledge of people about ayurvedic treatments and diseases and their knowledge about the connection between AI/ML with it, we have conducted a survey was conducted with both closed and open-ended questions by distributing a questionnaire.

## 4.2 Feasibility studies

Economic feasibility is a critical aspect of any project's success, as it determines whether or not the project is financially viable. The economic feasibility report analyzes the development costs and benefits of the project, and if a proper economic feasibility plan is not in place, the project is likely to fail. Therefore, it is crucial that the proposed system is both cost-effective and efficient in order to ensure its success.

- **Scheduled feasibility.**

Scheduled feasibility is another essential factor to consider when undertaking a project. A schedule feasibility assessment examines the timelines for the planned project, and any delays or missed deadlines can have a significant impact on the project's success. Therefore, it is vital that the proposed system completes each task within the allotted time period as specified to ensure that the project stays on schedule.

- **Technical feasibility**

Technical feasibility planning is also crucial in the development of any system. It involves evaluating the required skills and expertise necessary for mobile and web application development, as well as the ability to understand software architectures and communicate effectively with stakeholders to obtain the necessary information. Without proper technical feasibility planning, it is unlikely that the proposed system will be successfully developed and implemented. Therefore, it is essential to have the necessary technical skills and communication abilities to move forward with the system's development.

## 4.3 System Diagrams

### 4.3.1 Overall System Diagram

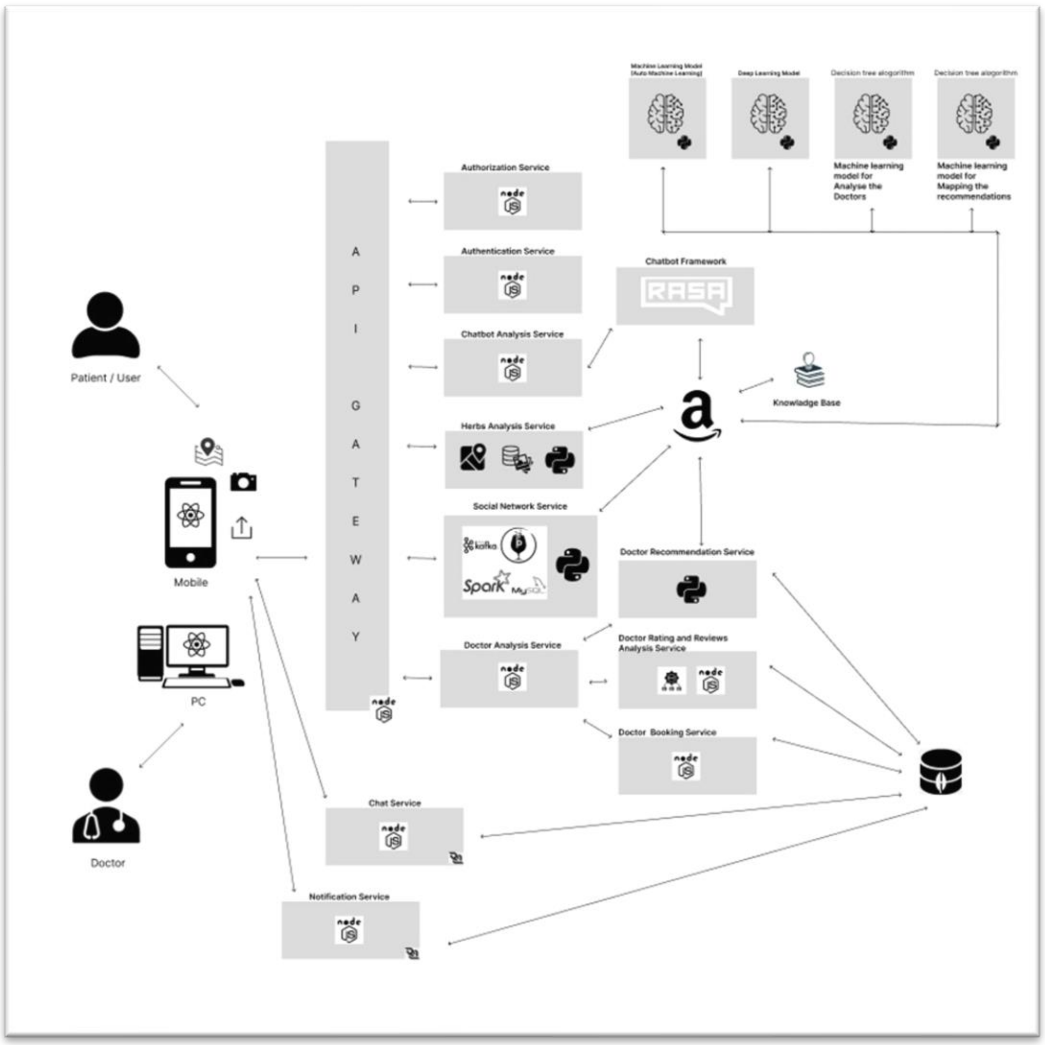


Figure 10: Overall System Architecture Diagram

### 4.3.2 Component System Diagram

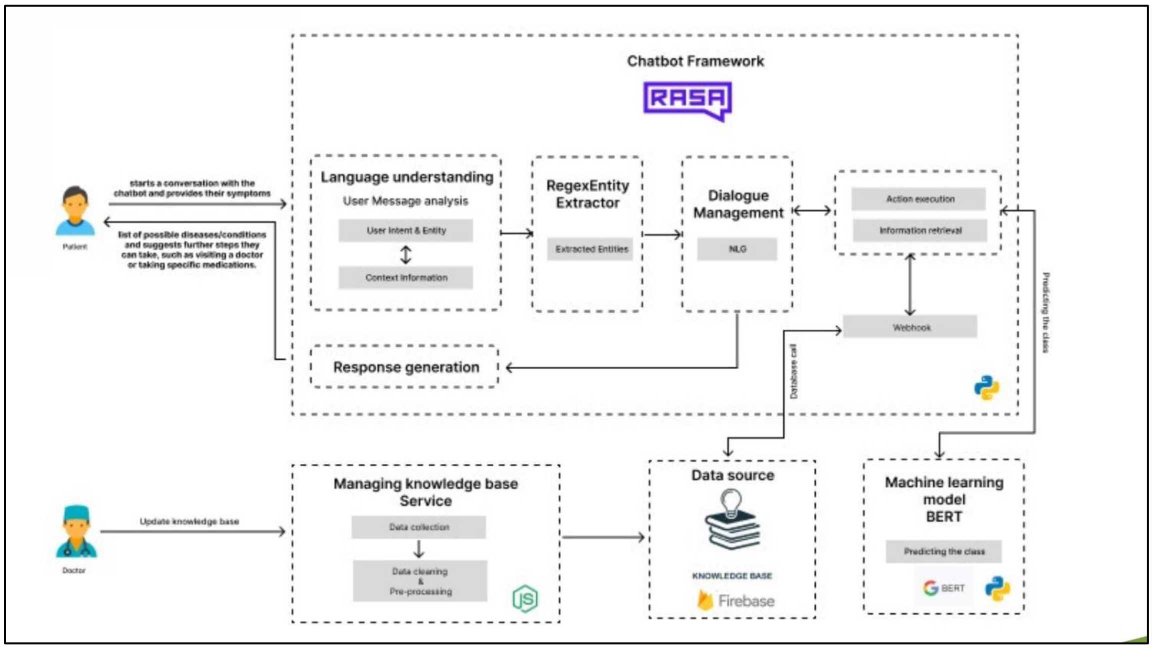


*Figure 11: Component system diagram*

The Ayurvedic recommendation chatbot, built on the flexible and highly customizable Rasa chatbot framework, features a well-structured architecture comprising distinct intents, entities, and custom actions that facilitate seamless conversations. A critical custom action is designed to extract user-input symptoms and make accurate disease predictions based on these symptoms. The integration of Firebase database is instrumental in providing comprehensive and reliable information related to diseases, herbal remedies, and treatment options, ensuring that the chatbot delivers precise recommendations to users. The development of the disease classification model leverages the powerful BERT architecture, fine-tuning hyperparameters, and employing CrossEntropyLoss and the AdamW optimizer for optimal results. A specialized "pred" function seamlessly integrates the disease prediction model into the chatbot, allowing it to intelligently respond to user inputs, retrieve relevant Ayurvedic information from the database, and provide users with comprehensive insights, treatments, and herbal recommendations to address their health concerns.

### 4.3.3 Use case Diagram

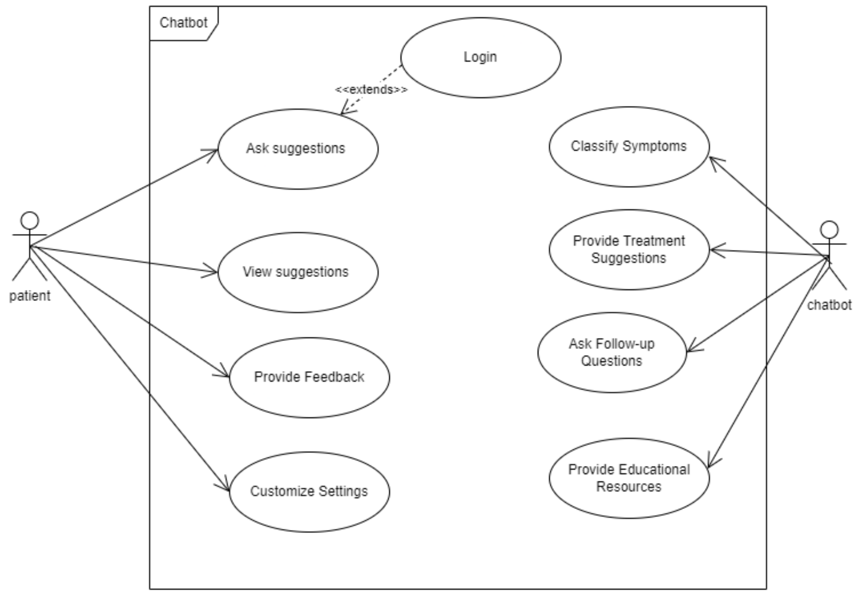


*Figure 12: Use case diagram*

### 4.3.4 Sequence Diagram

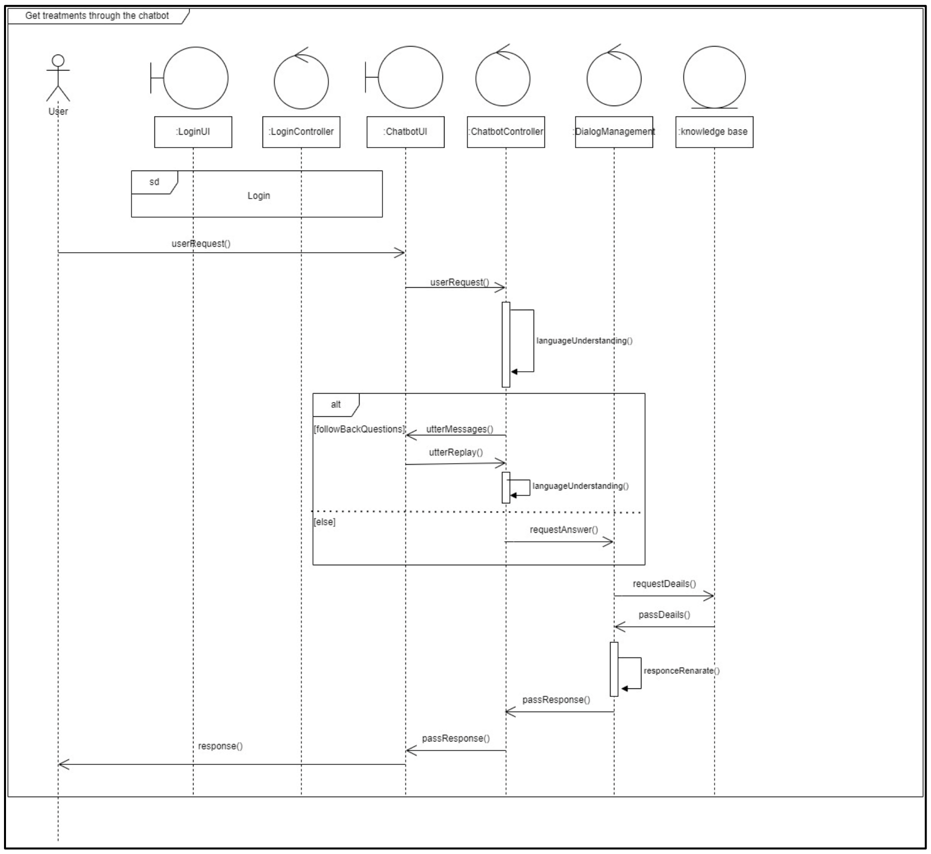


*Figure 13:Sequence Diagram*

## 4.4 Understanding the Key Pillars of the Research Domain

### 4.4.1 Machine Learning

Machine learning (ML) is a process aimed at solving complex problems that would be prohibitively expensive to tackle through the manual development of algorithms by human programmers. Instead, computers are empowered to autonomously "discover" their own algorithms, independently of human-created ones. [8] In recent times, generative artificial neural networks have outperformed many previous techniques, especially in areas where crafting algorithms for various tasks would be excessively costly. These areas encompass extensive language models, computer vision, speech recognition, email filtering, and medical applications.

Machine learning, often referred to as predictive analytics due to its extensive applications in addressing real-world challenges, is founded on mathematical optimization techniques, and closely related to the field of data mining, particularly in the context of unsupervised learning for exploratory data analysis. Its roots can be traced back to the quest for artificial intelligence (AI), where early AI scholars sought to enable machines to learn from data. They initially experimented with symbolic techniques and "neural networks," which later evolved into generalized linear models. Probabilistic reasoning and automated medical diagnosis were also early endeavors within this domain.

As time progressed, a divergence between AI and machine learning became evident, with AI leaning towards logical and knowledge-based approaches while machine learning grappled with challenges in probabilistic systems. By the 1980s, expert systems dominated AI, sidelining statistics. However, pattern recognition and information retrieval with a statistical orientation emerged as a separate domain. Concurrently, neural network research extended beyond AI, led by pioneers like Hopfield, Rumelhart, and Hinton, advocating a "connectionism" approach and achieving significant milestones like the development of backpropagation.

Machine learning eventually emerged as a distinct field in the 1990s, shifting its focus from AI to solving practical, real-world problems. It departed from symbolic AI methods and embraced probabilistic, statistical, and fuzzy logic-based techniques. Machine learning encompasses three primary paradigms: supervised learning, where the system learns from labeled data; unsupervised learning, which autonomously uncovers data structures; and reinforcement learning, used for dynamic tasks with feedback-driven incentives. Each of these paradigms has its own set of strengths and limitations, contributing to the diverse landscape of machine learning methodologies.

## 4.4.2 Natural Language Processing

The domain of computer science recognized as "natural language processing" (NLP) falls specifically under the purview of "artificial intelligence" (AI). It is primarily concerned with bestowing computers with the capability to comprehend written and spoken language in a manner reminiscent of human understanding.

NLP amalgamates statistical, machine learning, and deep learning models with the field of computational linguistics, which involves constructing rule-based representations of human language. Through the utilization of these technologies, computers now possess the ability to process human language in the form of textual or auditory data and achieve a profound "comprehension" of the content, encompassing the intentions and emotional nuances conveyed by the speaker or writer.

NLP underpins various computer applications, encompassing those that facilitate the translation of text between languages, respond to spoken directives, and swiftly distill extensive textual information, even in real-time scenarios. Chances are, you have interacted with NLP in everyday consumer products, including voice-activated GPS devices, digital personal assistants, speech-to-text transcription programs, automated customer service chatbots, and a spectrum of other user-friendly utilities. Furthermore, the deployment of NLP in corporate solutions is expanding, serving as an instrumental means of optimizing business operations, amplifying workforce productivity, and streamlining pivotal business processes. [9]

#### 4.4.2.1 NLP Tasks

Creating software that can reliably discern the intended meaning from text or voice data is an exceptionally daunting task due to the inherent complexities of human language. Human communication is fraught with ambiguity, including homonyms, homophones, sarcasm, idiomatic expressions, metaphors, deviations from grammatical conventions, and shifts in sentence structures. These intricacies, which take humans years to master, must be imparted to natural language-driven applications from the outset to ensure their utility. To facilitate the computer's comprehension of the text and speech data it processes, several NLP tasks are employed. Here are some of these tasks [10]:

1. **Speech Recognition (Speech-to-Text):** This process involves accurately transcribing voice data into text and is commonly known as speech recognition or speech-to-text. Any program that responds to voice commands or inquiries relies on speech recognition. The challenges arise from the diverse ways individuals speak—quickly, with words slurred together, varying emphases and intonations, diverse dialects, and sometimes non-standard grammar usage—all of which make speech recognition particularly challenging.
2. **Part of Speech Tagging (Grammatical Tagging):** Part of speech tagging entails determining the grammatical category of a word based on its usage and context. For

example, in the sentences "I can make a paper plane" and "What make of car do you own?," the word "make" is categorized as a verb and a noun, respectively.

3. **Word Sense Disambiguation:** This task involves selecting the correct meaning of a word from among its potential meanings by conducting semantic analysis to discern which meaning aligns with the context at hand. Word sense disambiguation is crucial for distinguishing between different senses of a word, such as "make" in "make the grade" (achieve) and "make a bet" (place).
4. **Named Entity Recognition (NER):** NER identifies and classifies words or phrases as specific entities, such as recognizing "Kentucky" as a location or "Fred" as a person's name.
5. **Co-reference Resolution:** Co-reference resolution revolves around determining whether and when two words refer to the same entity. It often involves resolving pronoun references (e.g., determining that "she" refers to "Mary"), but can also encompass identifying metaphors or idioms used in the text (e.g., understanding that "bear" refers to a large, hairy person rather than the animal).
6. **Sentiment Analysis:** Sentiment analysis seeks to uncover nuanced elements in text, including emotions, attitudes, sarcasm, confusion, and mistrust.
7. **Natural Language Generation:** Natural language generation is the process of transforming structured data into human language. It is essentially the opposite of voice recognition or speech-to-text, where machines generate human-like text based on data inputs.

#### 4.4.2.2 NLP usages

Natural Language Processing (NLP) occupies a central role in advancing machine intelligence across a wide array of practical applications. Several examples underscore its substantial influence [11]:

1. **Spam Detection:** Although not immediately associated with NLP, sophisticated spam detection systems leverage NLP's text classification capabilities to scrutinize emails for linguistic patterns that indicate spam or phishing attempts. These patterns encompass the excessive use of financial jargon, characteristic grammatical errors, threatening language, undue urgency, misspelled company names, and more. While experts generally consider spam detection as a problem that has been largely solved, individual email experiences may still vary.
2. **Machine Translation:** Google Translate stands out as a prominent illustration of NLP in action. Effective machine translation goes beyond simple word substitution; it aims to accurately capture the meaning and tone of the source language and convey it in the target language with the same essence and intended impact. Machine translation tools have made

significant advancements in terms of accuracy. A useful benchmark for evaluating any machine translation tool involves translating text from one language and then back to the original, highlighting improvements in translation quality.

3. **Virtual Agents and Chatbots:** Virtual agents like Apple's Siri and Amazon's Alexa employ speech recognition to comprehend voice commands and natural language generation to provide appropriate responses or valuable information. Chatbots, in contrast, perform similar functions in response to typed text inputs. Leading virtual agents and chatbots also possess the ability to recognize contextual cues within human queries, enabling them to offer increasingly personalized responses or options over time. The next evolutionary step for these applications involves question answering, enabling them to provide relevant and articulate answers to user inquiries.

4. **Social Media Sentiment Analysis:** This has evolved into a crucial business tool for extracting hidden insights from social media platforms. Sentiment analysis delves into the language used in social media posts, comments, reviews, and more, extracting attitudes and emotions in response to products, promotions, and events. Companies harness this information to inform product designs, tailor marketing campaigns, and make strategic decisions.

5. **Text Summarization:** NLP techniques underpin text summarization, which involves digesting extensive volumes of digital text and generating concise summaries for indexes, research databases, or readers with time constraints. Leading text summarization applications incorporate semantic reasoning and natural language generation (NLG) to provide informative context and conclusions within the summaries.

In conclusion, NLP has profoundly influenced various domains by equipping machines with the capability to comprehend, process, and generate human language. Its applications extend beyond the obvious, enhancing communication, decision-making, and information retrieval across a multitude of industries and use cases.

### 4.4.3 Chat Bot Frameworks

A chatbot framework is a specialized tool designed for building and defining the behavior of chatbots, offering a way to reduce the typical manual effort required for their creation. These frameworks typically include components like bot connectors, a bot directory, a developer portal, and a bot builder SDK. They often provide a testing console for evaluating the chatbot's performance after development.

In essence, chatbot frameworks provide developers with a set of tools to streamline and enhance the efficiency of chatbot creation. Code for chatbot frameworks can be written in various programming languages. For example, the conversational AI development environment for Amazon devices utilizes JavaScript.

It's important to note that while the terms "platforms" and "frameworks" are sometimes used interchangeably, they are not the same. Chatbot platforms typically come with visual builders, are ready-to-use, and do not require coding knowledge. They are hosted and powered by external entities, offering simplicity for non-technical users to create chatbots. In contrast, chatbot frameworks require coding skills and are utilized by developers to build chatbots from scratch using programming languages.

To clarify further, think of chatbots as tools that interact with users and perform tasks on a bot platform. A chatbot development framework, on the other hand, comprises programmable features and components that programmers can leverage to expedite bot creation. These frameworks are also accessible for use by others.

Open-source chatbots are those for which the program's source code is openly available for anyone to view and modify according to their specific needs and preferences. These source scripts can often be found on platforms like GitHub and can be used to create custom bots. Open-source chatbots aim to simulate conversations between users and bots, and by making their source code freely accessible, they promote greater transparency and can benefit from input and enhancements contributed by developers.

Examples of open-source chatbots include OpenDialog, which creates robust conversational interfaces for business processes and systems, and Botonic, a full-stack framework combining React and Tensorflow.js to enhance text and graphical interfaces without the need for a server. Claudia Bot builder, an extension library for Claudia.js, focuses on building business workflows for chatbots [12].

#### 4.4.3.1 Why use a framework for chatbots?

Using a framework for chatbots offers several significant advantages that streamline and enhance the development, deployment, and maintenance of chatbot applications. Here are some key reasons why utilizing a framework is beneficial [13]:

1. **Efficiency and Speed:** Chatbot frameworks provide a structured and pre-built foundation, including essential components like natural language processing (NLP), user interface, and integration capabilities. This significantly accelerates the development process, allowing developers to focus on customizing and fine-tuning the chatbot's specific functionality.
2. **Consistency and Best Practices:** Frameworks often come with established best practices and design patterns. This ensures that the chatbot adheres to industry standards, leading to a more consistent and reliable user experience. It also helps in maintaining code quality and readability.
3. **Scalability:** Many frameworks are designed to handle scalability challenges, enabling chatbots to accommodate a growing user base and increased workloads. This scalability is

essential for applications that need to handle a large number of concurrent users or scale across different channels and platforms.

4. **Cross-Channel Compatibility:** Frameworks are often developed to be channel-agnostic, allowing chatbots to seamlessly integrate with various messaging platforms, websites, mobile apps, and even voice assistants. This flexibility ensures that chatbots can reach users on their preferred communication channels.
5. **Built-in NLP Capabilities:** Chatbot frameworks frequently include built-in natural language processing capabilities or integration with NLP libraries. This simplifies the processing of user input, making it easier to understand and respond to user queries accurately.
6. **Community and Ecosystem:** Many popular chatbot frameworks have active communities and ecosystems of developers, offering a wealth of resources, documentation, and third-party plugins or extensions. This support network can be invaluable for troubleshooting issues and finding solutions to common challenges.
7. **Security and Compliance:** Frameworks often come with security features and compliance measures built in, helping developers protect user data and adhere to regulatory requirements. This is particularly crucial for chatbots handling sensitive information or operating in regulated industries like healthcare or finance.
8. **Analytics and Insights:** Many frameworks offer analytics and reporting tools that enable developers to track user interactions, gather valuable insights, and continually improve the chatbot's performance. This data-driven approach can lead to more effective and user-friendly chatbots.
9. **Cost Savings:** Using a framework can reduce development costs by eliminating the need to build fundamental features from scratch. This cost-effectiveness is especially beneficial for smaller teams or organizations with limited resources.
10. **Maintenance and Updates:** Frameworks often receive regular updates and improvements, including security patches and new features. This ensures that chatbots remain up-to-date and can adapt to evolving user needs and technological changes.

#### 4.4.3.1 RASA Framework

Rasa Open Source is an open-source conversational AI platform that offers the capability to understand and engage in conversations while also facilitating integration with third-party systems and messaging services through a set of APIs. It serves as the foundational framework for developing chatbots and virtual assistants [14].

Rasa, powered by Python and natural language understanding (NLU), is a software program designed to create unique AI chatbots. Rasa provides a comprehensive platform for building AI-driven chatbots that leverage NLU capabilities. Users have the flexibility to incorporate custom actions and train the model to align with specific requirements. These Rasa-built chatbots have been successfully deployed on various platforms, including Slack, Microsoft Bot, and Facebook Messenger.

Rasa consists of two primary components:

1. **Rasa NLU:** This component is an open-source natural language processing solution that specializes in intent categorization, which involves determining the user's intent or query. Additionally, it extracts structured data from user inputs and aids the chatbot in comprehending user communication.
2. **Rasa Core:** Serving as a chatbot framework, Rasa Core employs machine learning-based dialogue management. It leverages the structured information provided by Rasa NLU to predict the most suitable action for the chatbot to take, eliminating the need for conventional if/else statements. Moreover, it utilizes reinforcement learning techniques to enhance action prediction.

In essence, Rasa NLU is responsible for processing structured user data, while Rasa Core focuses on determining the next course of action for the chatbot. It's worth noting that Rasa Core and Rasa NLU can be utilized independently of each other, providing flexibility in their usage.

#### 4.4.3.2 Entity Extractor

Entities are organized chunks of data that make up a user message. Either training data must be specified in order to train an ML model, or regular expressions must be defined in order to use the RegexEntityExtractor to extract entities based on character patterns.

Consider what data your assistance needs to accomplish its user goals when choosing which entities to extract. There is a chance that the user will submit more details that you don't require for any user goals, thus you don't have to extract them as entities.

For information on how to annotate entities in your training data, refer to the training data format.Case-insensitive synonyms translate extracted items to a value other than the original text extracted. When there are various ways that users refer to the same object, you can utilize synonyms. Consider what you want to accomplish when you extract an entity, then work backwards to determine which values are equal

Imagine you had an entity account that you used to check the balance of the user. "Credit" is one of the conceivable account types. The terms "credit account" and "credit card account" are also used by your users to describe their "credit" accounts.Adding the RegexFeaturizer component to your pipeline will enable you to employ regular expressions to enhance intent classification. A regex does not function as a rule for classifying an intent when using the RegexFeaturizer. The intent classifier will only utilize this feature to learn patterns for intent categorization. At the moment, all intent classifiers use the regex characteristics that are readily available.You have two options for using regular expressions with deterministic structures:

Regular Expressions as Features# The RegexFeaturizer component of your NLU pipeline can be enhanced with features created using regular expressions.

It is not important what the regular expression's name is when using the RegexFeaturizer. A regular expression offers a feature when utilizing the RegexFeaturizer that aids the model in learning a relationship between intents/entities and inputs that match the regular expression.

### 4.4.3.3 Intent Recognition

The aim or purpose that a user has during a conversation with a chatbot for customer service is known as chatbot intent. When a chatbot powered by artificial intelligence receives a message or input from a user, it starts to search the text for entities and purpose.

When clients utilize an instant messaging network to express their inquiries or concerns, entities explain any altering information they employ. The activity or task that the consumer wishes to complete when utilizing a chatbot is referred to as intent, on the other hand.

The capacity of a chatbot to ascertain the user's intent is what separates a successful, satisfying interaction from a bad interaction. Chatbots interpret client messages using machine learning and Natural Language Processing (NLP). However, there are two ways that software engineers might collaborate with businesses to enhance chatbot functionality:

Chatbot training and programming employing keywords, user conversion data snippets, and other information. Collecting user feedback constantly to enhance the chatbot's functionality.

#### 4.4.3.3.1 Why Is Intent of Chatbots Vital for Customer Service?

Improving the customer service experience requires designing chatbots that can discern user intent. A chatbot will waste the user's time, effort, and irritation if it is unable to ascertain the user's intent and so cannot comprehend what the user requires.

#### 4.4.3.3.2 What does AI and machine learning mean by intent?

The action, objective, or reaction that the customer wishes to achieve is referred to as customer intent. AI instant chat bots can be used by businesses to ascertain client intent and offer a quick, satisfactory resolution.

The metaphorical brain underpinning customer service chatbots is conversational AI. NLP, pattern recognition, and natural language interpretation (NLI), among other machine learning techniques, are used by AI to analyze user requests, extract keywords or phrases, and identify voice patterns in order to guide users to their intended destination.

Chatbots can comprehend a customer's written or voice message via intent recognition, also known as intent categorization. To ascertain the customer's desired outcome, the chatbot examines each message for intent (the task or outcome) and entities (descriptive data).

## 4.5 Specific Methodology

The suggested remedy aims to assist in identifying suitable Ayurvedic therapies and medicinal herbs tailored to the mentioned symptoms, as well as general healthy lifestyle recommendations. To achieve this, a conversational AI chatbot will be implemented as a user-friendly tool for individuals to obtain customized solutions and guidance through text. The chatbot will draw from a comprehensive knowledge base containing information about the symptoms and their corresponding treatments to address user inquiries.

## 4.6 Commercialization Aspects of the System

This work offers a novel solution by potentially filling one of the research gaps that our research identified. The "AYUR MANA" application will be disclosed to everyone who has an interest in the case, including clients, ayurveda practitioners employed by the ayurvedic sector, and any other relevant parties. Regular activities are held by the Gampaha Wickramarachchi Ayurvedic University for practitioners and students. These events also include the presentation of items to increase sales in the local community because the institution has its own medical supply systems all around the country.

Additionally, *AYUR MANA* will be promoted on the Gampaha Wickramarachchi Ayurvedic University's official website. Customers who are interested in downloading an app can utilize it on their portable technology by visiting the Google Play store.

The main target stakeholders that this system attempts to accommodate are patients, ayurvedic practitioners, and members of the general public who are interested in ayurveda.

## 4.7 Consideration of the aspect of the System

During the process of coding each individual component, coding standards were adhered to.

### 4.7.1 Social Aspects

Anyone with varied levels of technological experience and expertise can use the smartphone software known as *AYUR MANA* Early disease detection and pest management can assist prevent significant financial losses, which will in turn help reduce global poverty and provide availability to wholesome food for people.

Because there aren't enough tools based on information and communications technology to support real-time collaboration between producers and researchers, early infestation control has become more challenging. Thus, by utilizing this state-of-the-art technology, we can offer a means for early identification and prompt communication while also ensuring the farmers' education, which will ultimately help the stakeholders for sustainable ayurveda solutions survive.

### 4.7.2 Security Aspects

The smartphone application known as "AYUR MANA" provides users with the ability to safeguard the confidentiality of their private information, detection details, and user authentication. Concerns regarding the level of security were taken into account right from the beginning of the page. For the purpose of ensuring the user's authenticity, JWT tokens were utilized. While communicating over the network, several ways for encrypting data were utilized in order to circumvent the security checks. Requests for user rights were made in order to improve the system's integrity and security.

### 4.7.3 Ethical Aspects

The system's design takes a number of ethical considerations into account. Users on this platform are not restricted by an upper age limit.

Children can use this program with success as well. The application has not received any unfavorable comments on respect for the culture.

# 5. IMPLEMENTATION AND TESTING

## 5.1 Implementation

In order to handle important areas of image identification, classification, and advancement level assessment in herb plants related to ayurveda, the research project offers a full suite of software solutions including both web and mobile applications. All of these programs are referred to as "AYUR MANA."

1. AYUR MANA Mobile Application:

Goal: The "AYUR MANA" mobile application is a flexible tool for the detection, categorization, and assessment of disease progression in coconut palms.

• The patient might inquire about their symptoms.

• The patient should be able to express their opinions.

• A chatbot should be able to receive and comprehend queries from patients about their symptoms.

• Based on the patient's symptoms, the chatbot should be able to determine the ailment or condition.

• The chatbot need to have the ability to offer recommendations and treatments for the detected illness or condition.

• The chatbot ought to be able to give details about prescription drugs, their dosages, and any potential side effects.

• The chatbot must be able to offer links to reliable websites that offer information on the illness or condition.

• A chatbot should be able to keep track of a patient's symptoms and offer more guidance as needed.

• Chatbots should be able to identify emergency circumstances and offer helpful advise, such as dialing 911 or going to the hospital.

• Patients should be able to interact with chatbots through an easy-to-use interface.

• A chatbot should be accessible around-the-clock to assist patients.

• To prevent delays or downtime, the chatbot should be able to manage several conversations at once.

• Technology of Implementation: React Native, a cross-platform framework renowned for its effectiveness and capacity to provide consistent user experiences across various mobile platforms, is used to create the front end of the mobile application.

2. AYUR MANA Web Application:

• Goal: The online application is made to meet the specific requirements of Gampaha Wickramarachchi Ayurvedic University researchers and provides nationwide disease-infected patients with the ability to be continuously monitored.

Monitoring: Practitioners may keep track of patients' conditions in real-time, enabling data-driven decision-making.

• Data Analysis: The web program allows leaf and pattern analysis, assisting researchers in getting insightful knowledge.

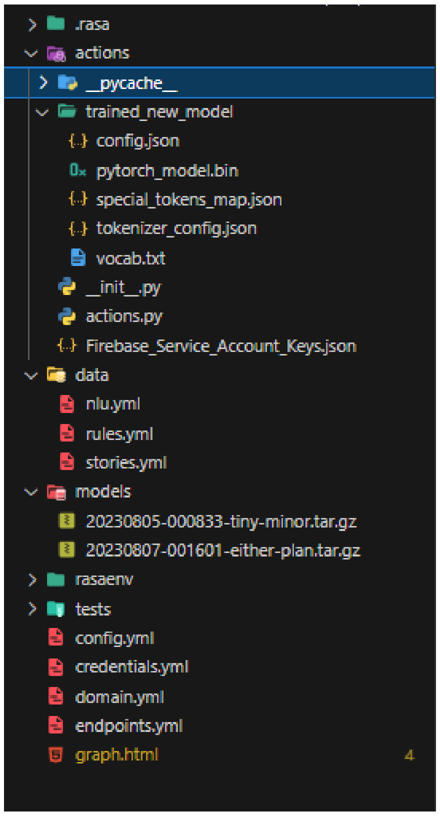
Frontend Development: React.js, a well-liked JavaScript library, is used to create the front-end of the web application. React.js is renowned for its performance and adaptability in the creation of dynamic user interfaces.

Backend Development: Node.js is used for the web application's backend development. Real-time data updates can be handled by Node.js, which also offers a stable server-side environment.

To sum up, AYUR MANA is an all-encompassing software solution that consists of a mobile application for plant leaf identification and evaluation on-the-ground and a web application for distant monitoring and data analysis. A powerful tool for researchers and stakeholders involved in the management and preservation of Ayurvedic Medicine palm lands, AYUR MANA was made possible by the technologies selected for implementation, including React Native, React.js, and Node.js.

## 5.1.1 Model Implementation

### Root directory



```
> .rasa
v actions
  > __pycache__
  v trained_new_model
      config.json
      pytorch_model.bin
      special_tokens_map.json
      tokenizer_config.json
      vocab.txt
    __init__.py
    actions.py
    Firebase_Service_Account_Keys.json
v data
    nlu.yml
    rules.yml
    stories.yml
v models
    20230805-000833-tiny-minor.tar.gz
    20230807-001601-either-plan.tar.gz
> rasaenv
> tests
  config.yml
  credentials.yml
  domain.yml
  endpoints.yml
  graph.html                    4
```

*Figure 14: Code Snippet of root directory*

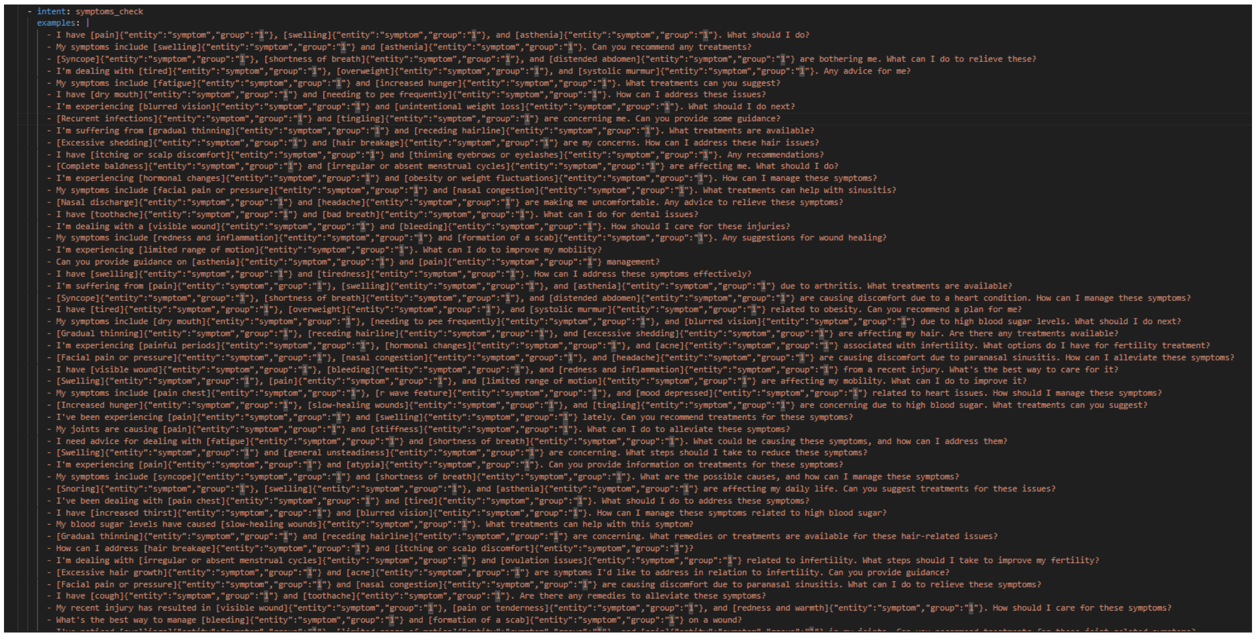
**nlu.yml**

**nlu.yml File Explanation:**

The nlu.yml file in a Rasa project serves as the training data for the Natural Language Understanding (NLU) component chatbot. It contains examples of user messages along with their associated intents and entities, helping the chatbot understand user input. Below is a breakdown of the structure and content of nlu.yml file:

**Intent Definitions:**

In this section, define various intents. Intents represent the user's intentions or requests. Each intent is associated with one or more example user messages that help train the chatbot to recognize these intentions. Here's an example intent definition:



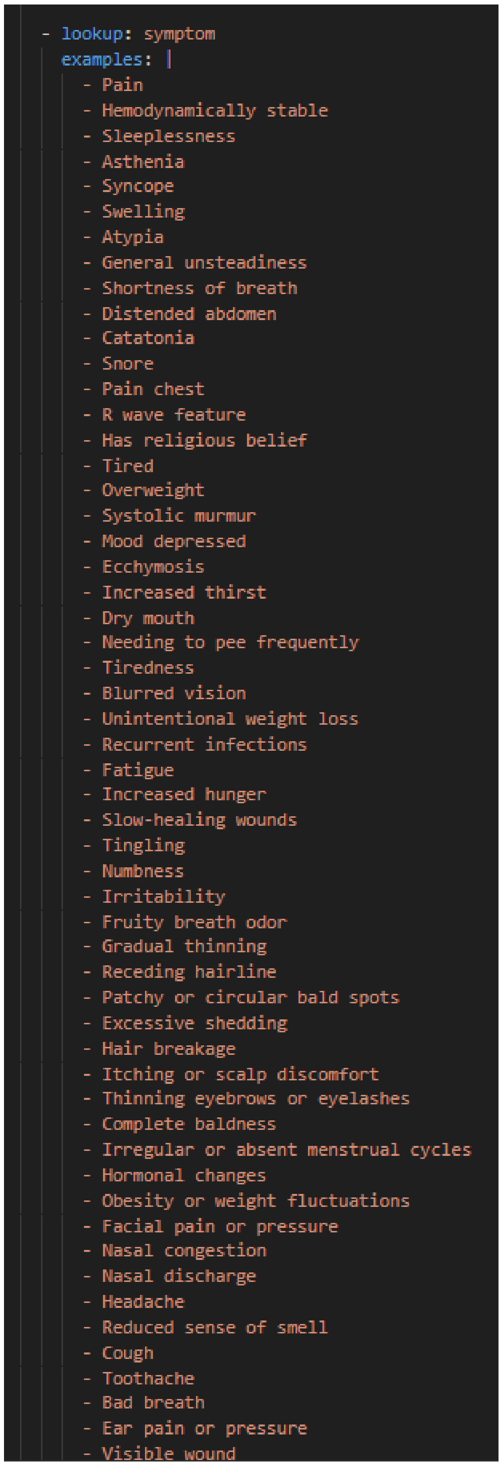
```
- intent: symptoms_check
  examples: |
    - I have [pain]{"entity":"symptom","group":"1"}, [swelling]{"entity":"symptom","group":"1"}, and [asthenia]{"entity":"symptom","group":"1"}. What should I do?
    - My symptoms include [swelling]{"entity":"symptom","group":"1"} and [asthenia]{"entity":"symptom","group":"1"}. Can you recommend any treatments?
    - [Syncope]{"entity":"symptom","group":"1"}, [shortness of breath]{"entity":"symptom","group":"1"}, and [distended abdomen]{"entity":"symptom","group":"1"} are bothering me. What can I do to relieve these?
    - I'm dealing with [tired]{"entity":"symptom","group":"1"}, [overweight]{"entity":"symptom","group":"1"}, and [systolic murmur]{"entity":"symptom","group":"1"}. Any advice for me?
    - My symptoms include [fatigue]{"entity":"symptom","group":"1"} and [increased hunger]{"entity":"symptom","group":"1"}. What treatments can you suggest?
    - I have [dry mouth]{"entity":"symptom","group":"1"} and [needing to pee frequently]{"entity":"symptom","group":"1"}. How can I address these issues?
    - I'm experiencing [blurred vision]{"entity":"symptom","group":"1"} and [unintentional weight loss]{"entity":"symptom","group":"1"}. What should I do next?
    - [Recurent infections]{"entity":"symptom","group":"1"} and [tingling]{"entity":"symptom","group":"1"} are concerning me. Can you provide some guidance?
    - I'm suffering from [gradual thinning]{"entity":"symptom","group":"1"} and [receding hairline]{"entity":"symptom","group":"1"}. What treatments are available?
    - [Excessive shedding]{"entity":"symptom","group":"1"} and [hair breakage]{"entity":"symptom","group":"1"} are my concerns. How can I address these hair issues?
    - I have [itching or scalp discomfort]{"entity":"symptom","group":"1"} and [thinning eyebrows or eyelashes]{"entity":"symptom","group":"1"}. Any recommendations?
    - [Complete baldness]{"entity":"symptom","group":"1"} and [irregular or absent menstrual cycles]{"entity":"symptom","group":"1"} are affecting me. What should I do?
    - I'm experiencing [hormonal changes]{"entity":"symptom","group":"1"} and [obesity or weight fluctuations]{"entity":"symptom","group":"1"}. How can I manage these symptoms?
    - My symptoms include [facial pain or pressure]{"entity":"symptom","group":"1"} and [nasal congestion]{"entity":"symptom","group":"1"}. What treatments can help with sinusitis?
    - [Nasal discharge]{"entity":"symptom","group":"1"} and [headache]{"entity":"symptom","group":"1"} are making me uncomfortable. Any advice to relieve these symptoms?
    - I have [toothache]{"entity":"symptom","group":"1"} and [bad breath]{"entity":"symptom","group":"1"}. What can I do for dental issues?
    - I'm dealing with a [visible wound]{"entity":"symptom","group":"1"} and [bleeding]{"entity":"symptom","group":"1"}. How should I care for these injuries?
    - My symptoms include [redness and inflammation]{"entity":"symptom","group":"1"} and [formation of a scab]{"entity":"symptom","group":"1"}. Any suggestions for wound healing?
    - I'm experiencing [limited range of motion]{"entity":"symptom","group":"1"}. What can I do to improve my mobility?
    - Can you provide guidance on [asthenia]{"entity":"symptom","group":"1"} and [pain]{"entity":"symptom","group":"1"} management?
    - I have [swelling]{"entity":"symptom","group":"1"} and [tiredness]{"entity":"symptom","group":"1"}. How can I address these symptoms effectively?
    - I'm suffering from [pain]{"entity":"symptom","group":"1"}, [swelling]{"entity":"symptom","group":"1"}, and [asthenia]{"entity":"symptom","group":"1"} due to arthritis. What treatments are available?
    - [Syncope]{"entity":"symptom","group":"1"}, [shortness of breath]{"entity":"symptom","group":"1"}, and [distended abdomen]{"entity":"symptom","group":"1"} are causing discomfort due to a heart condition. How can I manage these symptoms?
    - I have [tired]{"entity":"symptom","group":"1"}, [overweight]{"entity":"symptom","group":"1"}, and [systolic murmur]{"entity":"symptom","group":"1"} related to obesity. Can you recommend a plan for me?
    - My symptoms include [dry mouth]{"entity":"symptom","group":"1"}, [needing to pee frequently]{"entity":"symptom","group":"1"}, and [blurred vision]{"entity":"symptom","group":"1"} due to high blood sugar levels. What should I do next?
    - [Gradual thinning]{"entity":"symptom","group":"1"}, [receding hairline]{"entity":"symptom","group":"1"}, and [excessive shedding]{"entity":"symptom","group":"1"} are affecting my hair. Are there any treatments available?
    - I'm experiencing [painful periods]{"entity":"symptom","group":"1"}, [hormonal changes]{"entity":"symptom","group":"1"}, and [acne]{"entity":"symptom","group":"1"} associated with infertility. What options do I have for fertility treatment?
    - [Facial pain or pressure]{"entity":"symptom","group":"1"}, [nasal congestion]{"entity":"symptom","group":"1"}, and [headache]{"entity":"symptom","group":"1"} are causing discomfort due to paranasal sinusitis. How can I alleviate these symptoms?
    - I have [visible wound]{"entity":"symptom","group":"1"}, [bleeding]{"entity":"symptom","group":"1"}, and [redness and inflammation]{"entity":"symptom","group":"1"} from a recent injury. What's the best way to care for it?
    - [Swelling]{"entity":"symptom","group":"1"}, [pain]{"entity":"symptom","group":"1"}, and [limited range of motion]{"entity":"symptom","group":"1"} are affecting my mobility. What can I do to improve it?
    - My symptoms include [pain chest]{"entity":"symptom","group":"1"}, [r wave feature]{"entity":"symptom","group":"1"}, and [mood depressed]{"entity":"symptom","group":"1"} related to heart issues. How should I manage these symptoms?
    - [Increased hunger]{"entity":"symptom","group":"1"}, [slow-healing wounds]{"entity":"symptom","group":"1"}, and [tingling]{"entity":"symptom","group":"1"} are concerning due to high blood sugar. What treatments can you suggest?
    - I've been experiencing [pain]{"entity":"symptom","group":"1"} and [swelling]{"entity":"symptom","group":"1"} lately. Can you recommend treatments for these symptoms?
    - My joints are causing [pain]{"entity":"symptom","group":"1"} and [stiffness]{"entity":"symptom","group":"1"}. What can I do to alleviate these symptoms?
    - I need advice for dealing with [fatigue]{"entity":"symptom","group":"1"} and [shortness of breath]{"entity":"symptom","group":"1"}. What could be causing these symptoms, and how can I address them?
    - [Swelling]{"entity":"symptom","group":"1"} and [general unsteadiness]{"entity":"symptom","group":"1"} are concerning. What steps should I take to reduce these symptoms?
    - I'm experiencing [pain]{"entity":"symptom","group":"1"} and [atypia]{"entity":"symptom","group":"1"}. Can you provide information on treatments for these symptoms?
    - My symptoms include [syncope]{"entity":"symptom","group":"1"} and [shortness of breath]{"entity":"symptom","group":"1"}. What are the possible causes, and how can I manage these symptoms?
    - [Snoring]{"entity":"symptom","group":"1"}, [swelling]{"entity":"symptom","group":"1"}, and [asthenia]{"entity":"symptom","group":"1"} are affecting my daily life. Can you suggest treatments for these issues?
    - I've been dealing with [pain chest]{"entity":"symptom","group":"1"} and [tired]{"entity":"symptom","group":"1"}. What should I do to address these symptoms?
    - I have [increased thirst]{"entity":"symptom","group":"1"} and [blurred vision]{"entity":"symptom","group":"1"}. How can I manage these symptoms related to high blood sugar?
    - My blood sugar levels have caused [slow-healing wounds]{"entity":"symptom","group":"1"}. What treatments can help with this symptom?
    - [Gradual thinning]{"entity":"symptom","group":"1"} and [receding hairline]{"entity":"symptom","group":"1"} are concerning. What remedies or treatments are available for these hair-related issues?
    - How can I address [hair breakage]{"entity":"symptom","group":"1"} and [itching or scalp discomfort]{"entity":"symptom","group":"1"}?
    - I'm dealing with [irregular or absent menstrual cycles]{"entity":"symptom","group":"1"} and [ovulation issues]{"entity":"symptom","group":"1"} related to infertility. What steps should I take to improve my fertility?
    - [Excessive hair growth]{"entity":"symptom","group":"1"} and [acne]{"entity":"symptom","group":"1"} are symptoms I'd like to address in relation to infertility. Can you provide guidance?
    - [Facial pain or pressure]{"entity":"symptom","group":"1"} and [nasal congestion]{"entity":"symptom","group":"1"} are causing discomfort due to paranasal sinusitis. What can I do to relieve these symptoms?
    - I have [cough]{"entity":"symptom","group":"1"} and [toothache]{"entity":"symptom","group":"1"}. Are there any remedies to alleviate these symptoms?
    - My recent injury has resulted in [visible wound]{"entity":"symptom","group":"1"}, [pain or tenderness]{"entity":"symptom","group":"1"}, and [redness and warmth]{"entity":"symptom","group":"1"}. How should I care for these symptoms?
    - What's the best way to manage [bleeding]{"entity":"symptom","group":"1"} and [formation of a scab]{"entity":"symptom","group":"1"} on a wound?
```

Figure 15: Code Snippet of intent recognition

In here, the symptoms_check intent is defined, which represents a user's request for advice regarding various symptoms. The examples section provides sample user messages that illustrate how users might express this intent.

**Entity Lookup:**

Entities are specific pieces of information that the chatbot needs to extract from user messages. In nlu.yml file, there's also a section that defines a lookup table for the symptom entity. This lookup table helps the chatbot recognize and extract symptoms mentioned in user messages:



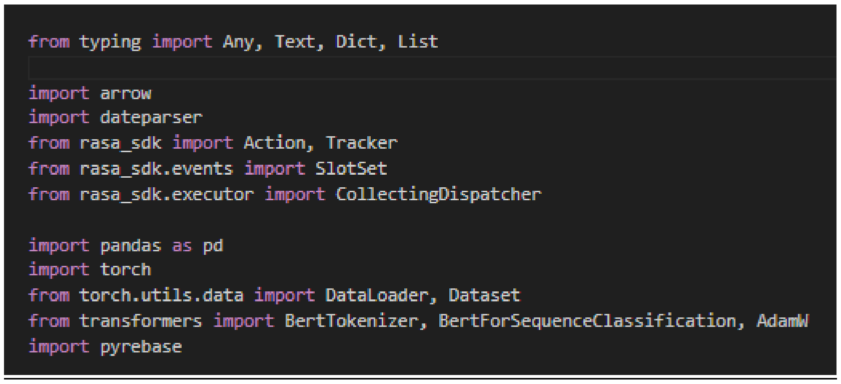
```
- lookup: symptom
  examples: |
    - Pain
    - Hemodynamically stable
    - Sleeplessness
    - Asthenia
    - Syncope
    - Swelling
    - Atypia
    - General unsteadiness
    - Shortness of breath
    - Distended abdomen
    - Catatonia
    - Snore
    - Pain chest
    - R wave feature
    - Has religious belief
    - Tired
    - Overweight
    - Systolic murmur
    - Mood depressed
    - Ecchymosis
    - Increased thirst
    - Dry mouth
    - Needing to pee frequently
    - Tiredness
    - Blurred vision
    - Unintentional weight loss
    - Recurrent infections
    - Fatigue
    - Increased hunger
    - Slow-healing wounds
    - Tingling
    - Numbness
    - Irritability
    - Fruity breath odor
    - Gradual thinning
    - Receding hairline
    - Patchy or circular bald spots
    - Excessive shedding
    - Hair breakage
    - Itching or scalp discomfort
    - Thinning eyebrows or eyelashes
    - Complete baldness
    - Irregular or absent menstrual cycles
    - Hormonal changes
    - Obesity or weight fluctuations
    - Facial pain or pressure
    - Nasal congestion
    - Nasal discharge
    - Headache
    - Reduced sense of smell
    - Cough
    - Toothache
    - Bad breath
    - Ear pain or pressure
    - Visible wound
```

*Figure 16: Code Snippet of entity lookup*

Here, the symptom entity is defined, and a list of example symptoms is provided. This helps the chatbot identify symptoms mentioned in user messages and associate them with the appropriate entity.

action.py file

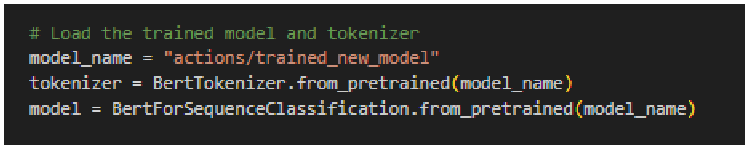
**imports**



```
from typing import Any, Text, Dict, List

import arrow
import dateparser
from rasa_sdk import Action, Tracker
from rasa_sdk.events import SlotSet
from rasa_sdk.executor import CollectingDispatcher

import pandas as pd
import torch
from torch.utils.data import DataLoader, Dataset
from transformers import BertTokenizer, BertForSequenceClassification, AdamW
import pyrebase
```

*Figure 17: Code Snippet of imports*

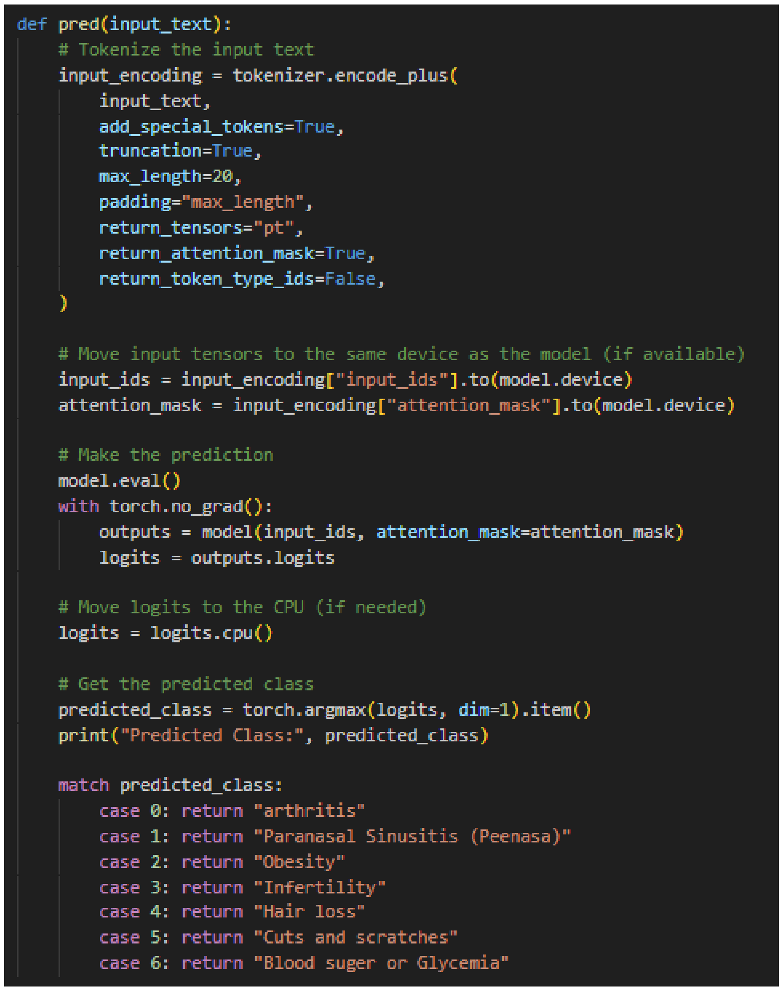
- from typing import ...: Imports various typing annotations for type hinting in Python.
- arrow and dateparser: Libraries for handling date and time information.
- from rasa_sdk import ...: Imports classes and modules from Rasa SDK for creating custom actions.
- pandas: Library for data manipulation and analysis.
- torch and torch.utils.data: Imports PyTorch and related utilities for deep learning.
- transformers: Imports the Hugging Face Transformers library for natural language processing (NLP) tasks.
- pyrebase: Python client for the Firebase Realtime Database.



```
# Load the trained model and tokenizer
model_name = "actions/trained_new_model"
tokenizer = BertTokenizer.from_pretrained(model_name)
model = BertForSequenceClassification.from_pretrained(model_name)
```

*Figure 18: Code Snippet of load trained model*

Here, the code loads a pre-trained machine learning model and tokenizer using the Hugging Face Transformers library. The model and tokenizer are used for natural language understanding tasks. The *model_name* variable specifies the directory path to the pre-trained model.



```python
def pred(input_text):
    # Tokenize the input text
    input_encoding = tokenizer.encode_plus(
        input_text,
        add_special_tokens=True,
        truncation=True,
        max_length=20,
        padding="max_length",
        return_tensors="pt",
        return_attention_mask=True,
        return_token_type_ids=False,
    )

    # Move input tensors to the same device as the model (if available)
    input_ids = input_encoding["input_ids"].to(model.device)
    attention_mask = input_encoding["attention_mask"].to(model.device)

    # Make the prediction
    model.eval()
    with torch.no_grad():
        outputs = model(input_ids, attention_mask=attention_mask)
        logits = outputs.logits

    # Move logits to the CPU (if needed)
    logits = logits.cpu()

    # Get the predicted class
    predicted_class = torch.argmax(logits, dim=1).item()
    print("Predicted Class:", predicted_class)

    match predicted_class:
        case 0: return "arthritis"
        case 1: return "Paranasal Sinusitis (Peenasa)"
        case 2: return "Obesity"
        case 3: return "Infertility"
        case 4: return "Hair loss"
        case 5: return "Cuts and scratches"
        case 6: return "Blood suger or Glycemia"
```

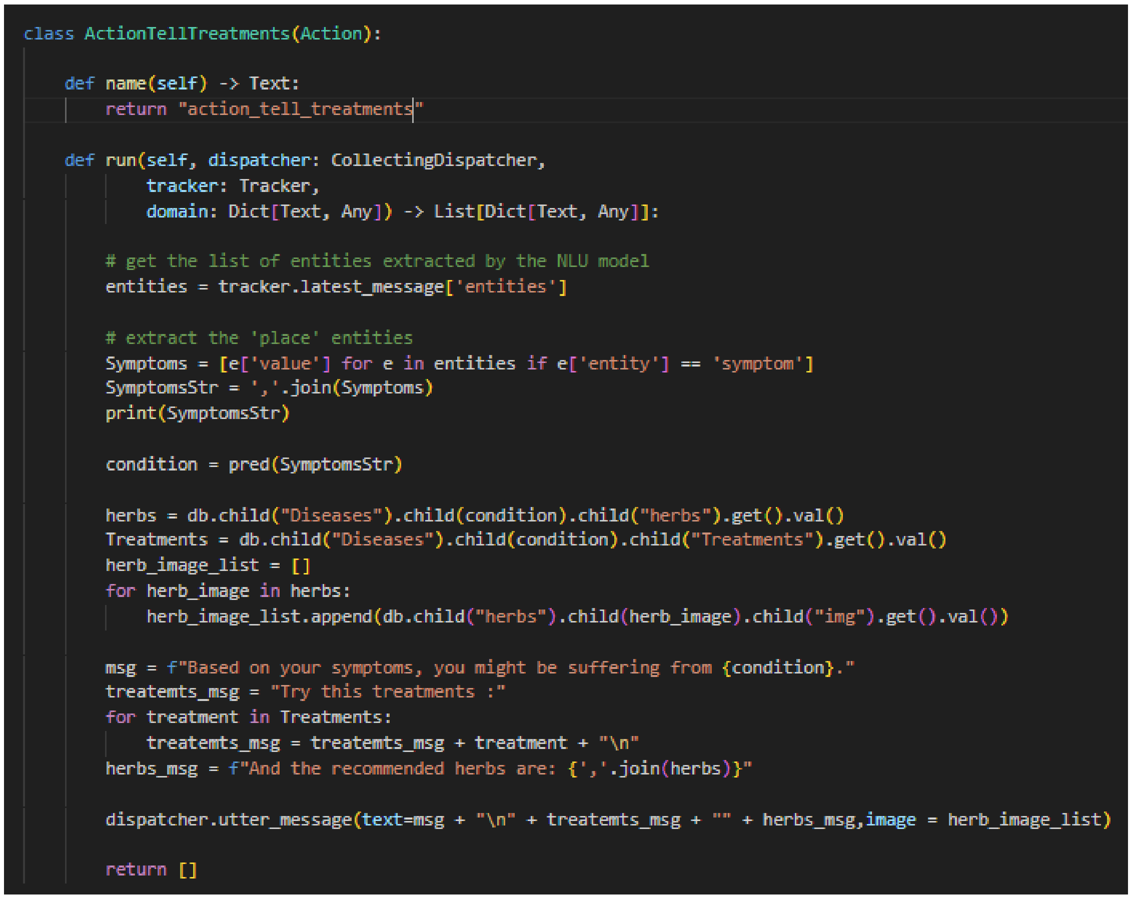
*Figure 19:: Code Snippet of named pred to take input texts*

This section defines function named pred that takes an input text (user symptoms) and predicts a medical condition based on the text. Here's a step-by-step explanation of what the function does:

- **Tokenization**: It uses the BertTokenizer to tokenize the input text. The tokenizer adds special tokens, truncates or pads the text to a maximum length of 20 tokens, and returns tensors suitable for model input.

- **Model Prediction**: The code moves the input tensors to the same device as the model (e.g., GPU) and makes a prediction using the loaded BERT-based model. It retrieves the logits (raw model outputs) from the model.

- **Predicted Class**: The code identifies the predicted class (medical condition) by finding the class with the highest logit score. It then maps the class index to a human-readable condition using a match statement.



```python
class ActionTellTreatments(Action):

    def name(self) -> Text:
        return "action_tell_treatments"

    def run(self, dispatcher: CollectingDispatcher,
            tracker: Tracker,
            domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:

        # get the list of entities extracted by the NLU model
        entities = tracker.latest_message['entities']

        # extract the 'place' entities
        Symptoms = [e['value'] for e in entities if e['entity'] == 'symptom']
        SymptomsStr = ','.join(Symptoms)
        print(SymptomsStr)

        condition = pred(SymptomsStr)

        herbs = db.child("Diseases").child(condition).child("herbs").get().val()
        Treatments = db.child("Diseases").child(condition).child("Treatments").get().val()
        herb_image_list = []
        for herb_image in herbs:
            herb_image_list.append(db.child("herbs").child(herb_image).child("img").get().val())

        msg = f"Based on your symptoms, you might be suffering from {condition}."
        treatemts_msg = "Try this treatments :"
        for treatment in Treatments:
            treatemts_msg = treatemts_msg + treatment + "\n"
        herbs_msg = f"And the recommended herbs are: {','.join(herbs)}"

        dispatcher.utter_message(text=msg + "\n" + treatemts_msg + "" + herbs_msg,image = herb_image_list)

        return []
```

*Figure 20: Code Snippet of action tell treatments*

This section defines a custom action class named ActionTellTreatments. Here's what this action does:

`name` Method: It defines the name of the action, which is "action_tell_treatments."

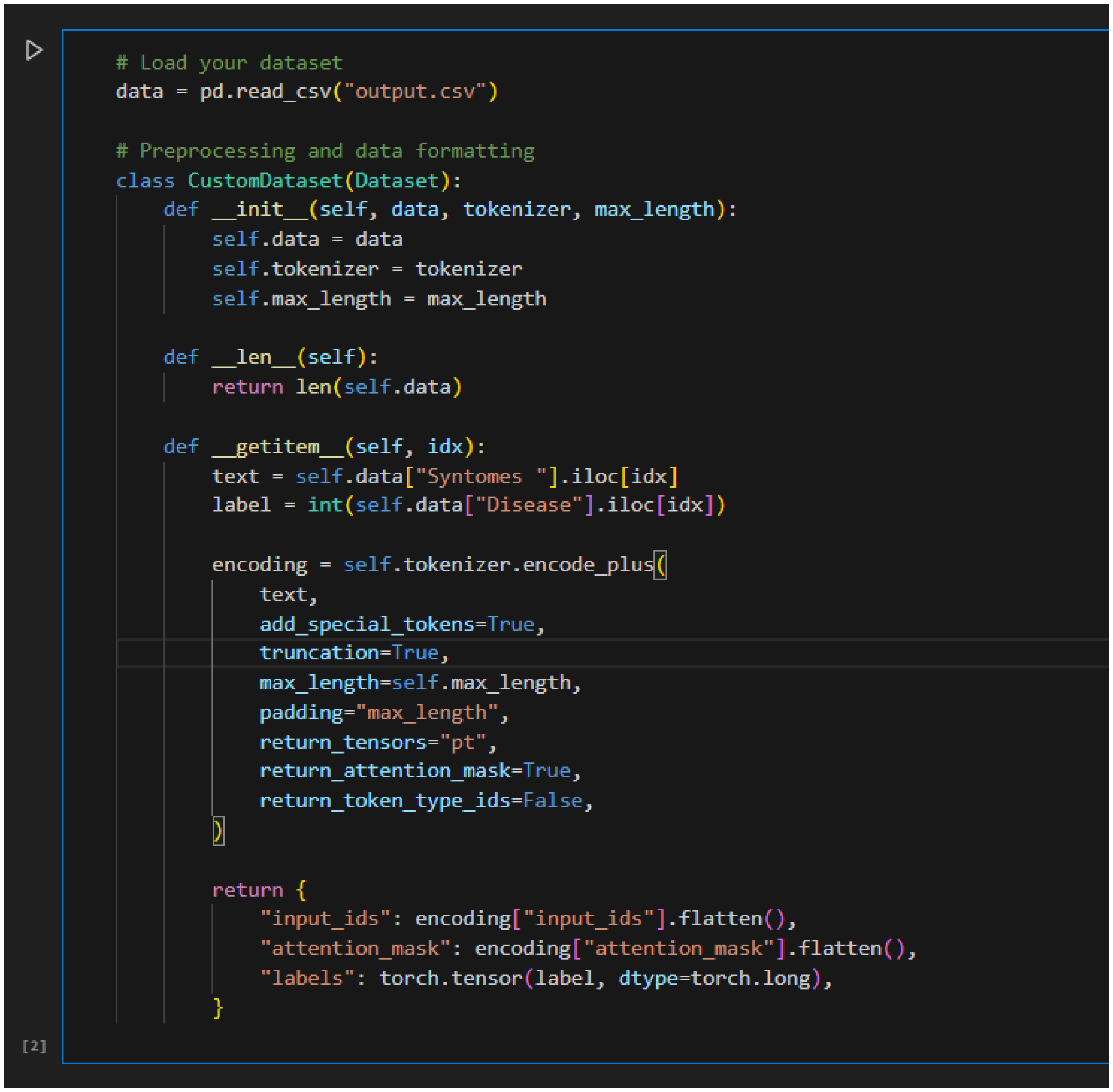
`run` Method: This method is called when the action is triggered. It performs the following steps:

- Extracts entities (user symptoms) from the user's message.
- Joins the extracted symptoms into a single string.
- Calls the *pred* function to predict the medical condition based on the symptoms.
- Retrieves information about treatments and herbs related to the predicted condition from the Firebase database.
- Creates a list of herb images.

- Constructs response messages including the predicted condition, treatment recommendations, and herb images
- Sends the response messages to the user using the Rasa dispatcher.

This action essentially uses machine learning to identify a potential health condition based on user-provided symptoms and provides relevant information to the user.

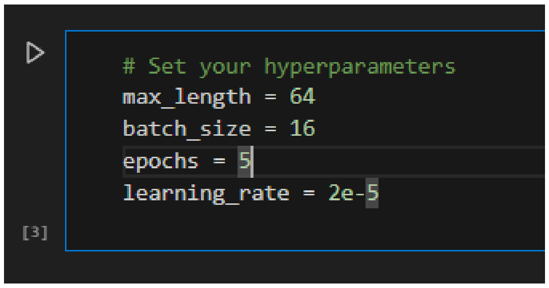
**BERT MODEL CODE SNIPPET**



```
# Load your dataset
data = pd.read_csv("output.csv")

# Preprocessing and data formatting
class CustomDataset(Dataset):
    def __init__(self, data, tokenizer, max_length):
        self.data = data
        self.tokenizer = tokenizer
        self.max_length = max_length

    def __len__(self):
        return len(self.data)

    def __getitem__(self, idx):
        text = self.data["Syntomes "].iloc[idx]
        label = int(self.data["Disease"].iloc[idx])

        encoding = self.tokenizer.encode_plus(
            text,
            add_special_tokens=True,
            truncation=True,
            max_length=self.max_length,
            padding="max_length",
            return_tensors="pt",
            return_attention_mask=True,
            return_token_type_ids=False,
        )

        return {
            "input_ids": encoding["input_ids"].flatten(),
            "attention_mask": encoding["attention_mask"].flatten(),
            "labels": torch.tensor(label, dtype=torch.long),
        }
[2]
```

*Figure 21: Code Snippet of BERT Model*

This code snippet begins by loading a dataset from a CSV file named "output.csv" using the Pandas library. This dataset likely contains valuable information about symptoms and their associated disease labels. Following this, the code defines a custom dataset class called **CustomDataset** to prepare the data for training a BERT-based model. Inside this class, there are essential methods such as **__init__**, **__len__**, and **__getitem__**. The **__init__** method initializes the dataset with the dataset itself, a BERT tokenizer, and a maximum sequence length. The **__len__** method specifies the length of the dataset based on the number of rows in the DataFrame. The most crucial part, the **__getitem__** method, retrieves an item from the dataset at a given index. It extracts both the text (representing symptoms) and the corresponding disease label from the DataFrame. The BERT tokenizer is then applied to encode the text, incorporating special tokens, managing truncation, and padding. Finally, the method returns a dictionary containing the tokenized input text, an attention mask indicating padded tokens, and the disease label as a torch tensor. Overall, this code prepares and formats the data for training a BERT-based sequence classification model, facilitating the utilization of the dataset in machine learning tasks related to disease prediction based on symptom descriptions.



```
# Set your hyperparameters
max_length = 64
batch_size = 16
epochs = 5
learning_rate = 2e-5
```

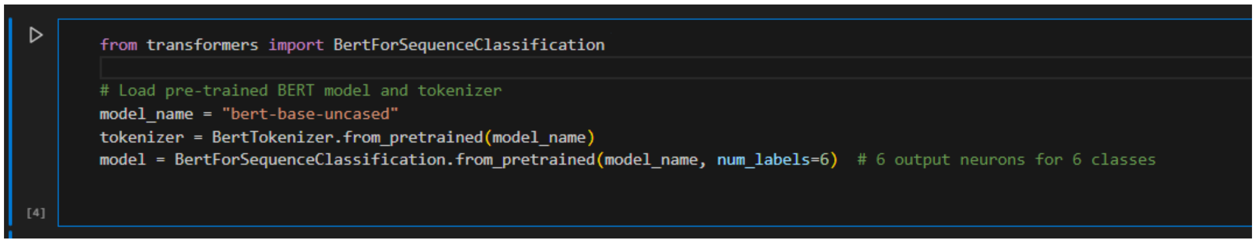
*Figure 22: Code Snippet of set hyperparameters*

1. **max_length = 64**:
   - This hyperparameter specifies the maximum length of input sequences that will be processed by the BERT model. Input sequences longer than this length will be truncated or padded to match this length. In this case, the maximum allowed sequence length is set to 64 tokens.
2. **batch_size = 16**:
   - The batch size determines the number of data samples that are processed together in each iteration during training. A batch size of 16 means that 16 data samples will be used in each forward and backward pass through the neural network during training. The choice of batch size affects training efficiency and memory usage.
3. **epochs = 5**:
   - The number of epochs defines how many times the entire training dataset is passed through the model during training. In this case, training is set to run for 5 epochs,

meaning that the model will learn from the dataset five times. The number of epochs is a crucial hyperparameter that influences model convergence.

4. **learning_rate = 2e-5**:

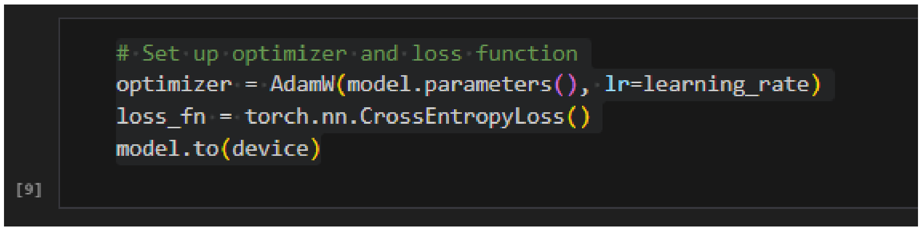
    - Learning rate is a hyperparameter that controls the step size taken during gradient descent optimization. A learning rate of 2e-5 (0.00002) is relatively small and is commonly used when fine-tuning pre-trained language models like BERT. It determines how quickly or slowly the model adapts its weights during training. The choice of learning rate can significantly impact training stability and convergence.



```
from transformers import BertForSequenceClassification

# Load pre-trained BERT model and tokenizer
model_name = "bert-base-uncased"
tokenizer = BertTokenizer.from_pretrained(model_name)
model = BertForSequenceClassification.from_pretrained(model_name, num_labels=6)  # 6 output neurons for 6 classes
```

*Figure 23: Code Snippet of load pretrained model*

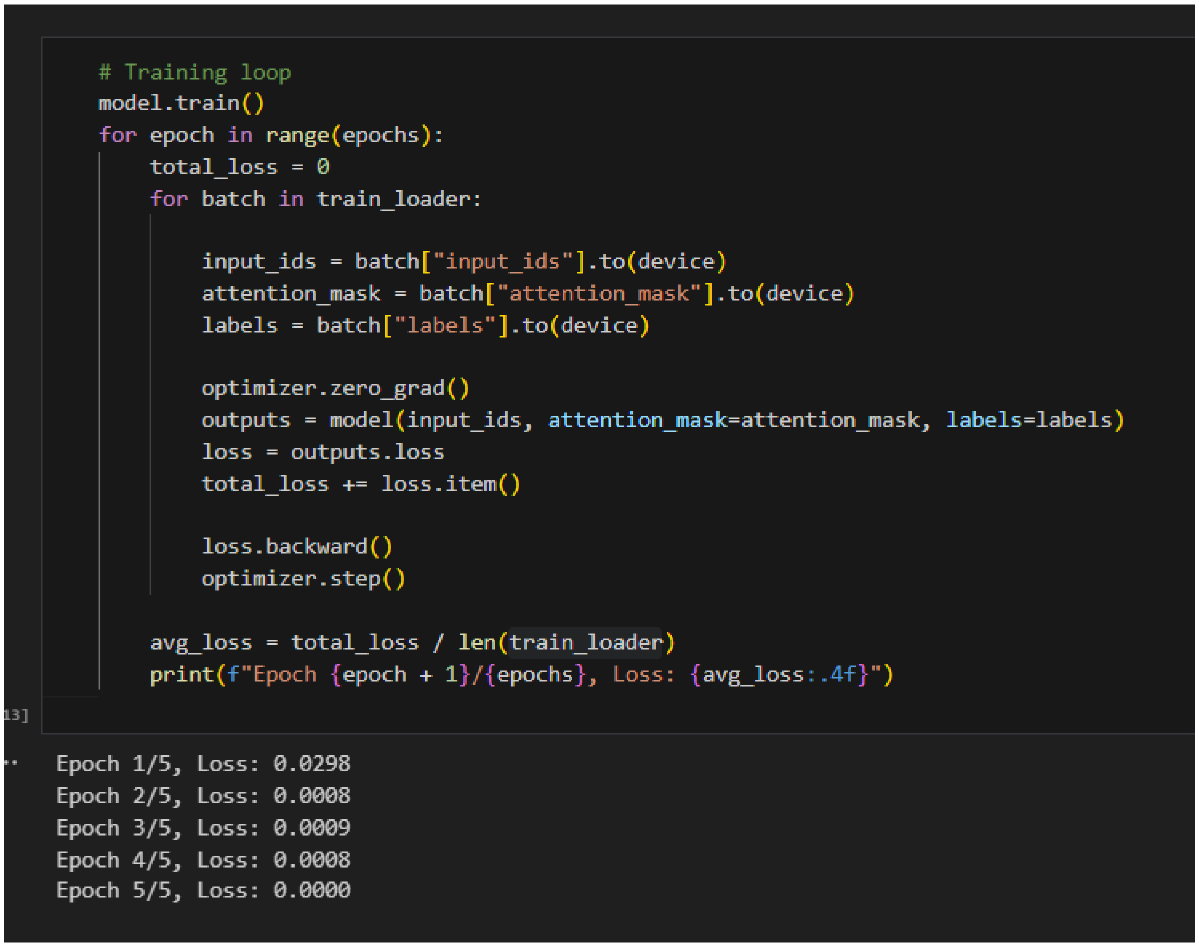
In here, a pre-trained BERT (Bidirectional Encoder Representations from Transformers) model and tokenizer are loaded for the purpose of sequence classification. The chosen pre-trained model, "bert-base-uncased," represents a widely used BERT base model with a lowercase tokenizer. The **BertTokenizer.from_pretrained()** method initializes the tokenizer, responsible for preprocessing text data into a format suitable for BERT input, handling tasks such as tokenization, adding special tokens, and padding or truncating sequences. Subsequently, the **BertForSequenceClassification.from_pretrained()** function loads the BERT model architecture along with its pre-trained weights. It is configured for sequence classification with **num_labels=6**, indicating that the model is tailored for a classification task involving six distinct classes. This process sets up a powerful foundation for leveraging BERT's contextual language understanding in various natural language processing tasks, particularly in text classification scenarios, where the model can be fine-tuned to excel in specific classification challenges.



```
# Set up optimizer and loss function
optimizer = AdamW(model.parameters(), lr=learning_rate)
loss_fn = torch.nn.CrossEntropyLoss()
model.to(device)
```

*Figure 24: Code Snippet ofset up opitmize*

In this code snippet, the essential components for training a BERT-based model are configured. The optimizer is initialized using AdamW, a variant of the Adam optimizer with weight decay, which is commonly used for fine-tuning pre-trained models. The optimizer is tasked with updating the model's parameters during training, and its learning rate is set to the previously defined **learning_rate**. Additionally, the code defines the loss function as the Cross-Entropy Loss, which is appropriate for multi-class classification tasks and quantifies the discrepancy between predicted class probabilities and actual class labels. Lastly, the model is moved to a specified computing device, such as a GPU if available, to accelerate training. This setup ensures that the model is ready for the training process, where it will learn to make accurate predictions while minimizing the defined loss using gradient descent optimization.



```
# Training loop
model.train()
for epoch in range(epochs):
    total_loss = 0
    for batch in train_loader:

        input_ids = batch["input_ids"].to(device)
        attention_mask = batch["attention_mask"].to(device)
        labels = batch["labels"].to(device)

        optimizer.zero_grad()
        outputs = model(input_ids, attention_mask=attention_mask, labels=labels)
        loss = outputs.loss
        total_loss += loss.item()

        loss.backward()
        optimizer.step()

    avg_loss = total_loss / len(train_loader)
    print(f"Epoch {epoch + 1}/{epochs}, Loss: {avg_loss:.4f}")
```

```
Epoch 1/5, Loss: 0.0298
Epoch 2/5, Loss: 0.0008
Epoch 3/5, Loss: 0.0009
Epoch 4/5, Loss: 0.0008
Epoch 5/5, Loss: 0.0000
```

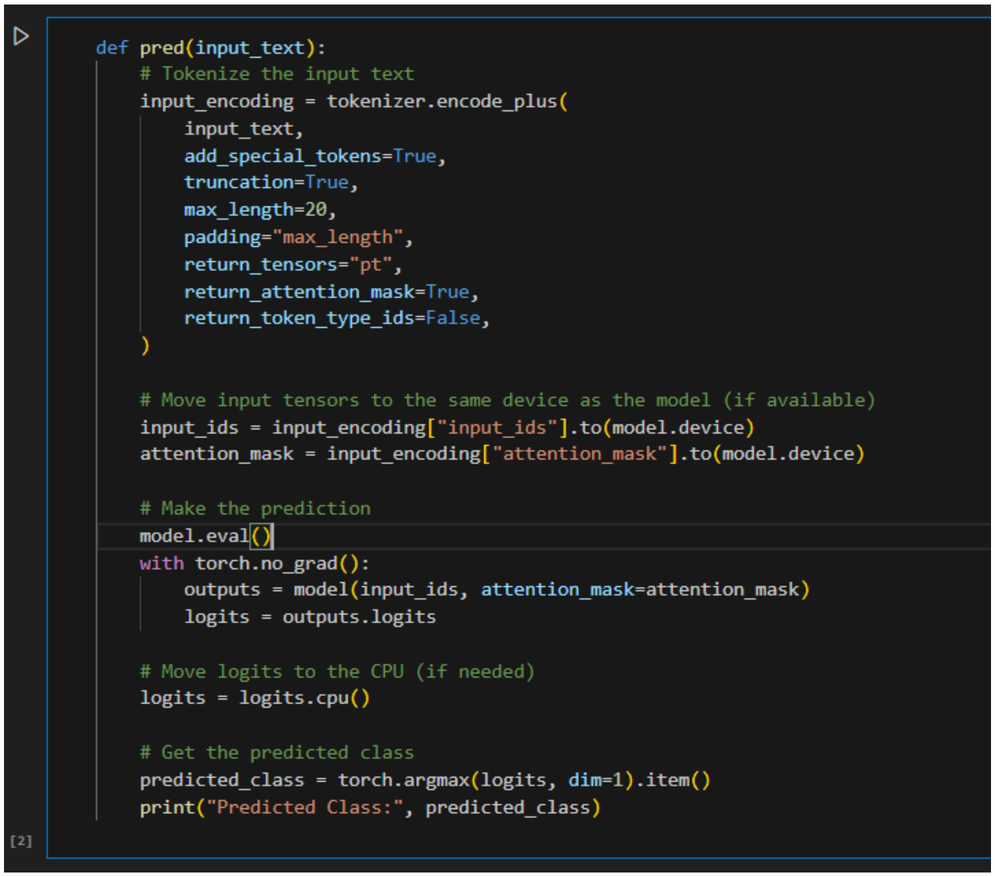
*Figure 25: Code Snippet of training loop*

In here, the training loop for the BERT-based model is defined. The model is set to training mode using **model.train()**, which enables gradient computation and weight updates during training. The loop iterates over the specified number of epochs, where each epoch represents a complete pass through the training dataset. Within each epoch, a total loss variable is initialized to track the cumulative loss for that epoch.

For each batch of data in the training loader (provided by **train_loader**), the input tensors (**input_ids**, **attention_mask**, and **labels**) are moved to the selected computing device (e.g., GPU) for efficient computation. The optimizer's gradients are zeroed using **optimizer.zero_grad()** to avoid gradient accumulation between mini-batches. Then, the model is called with the input tensors, and it computes both the logits and the loss with **outputs = model(...)**. The loss is extracted from the outputs.

The computed loss is accumulated in **total_loss**, and backpropagation (**loss.backward()**) is performed to compute gradients with respect to the model's parameters. The optimizer (**optimizer.step()**) then updates the model's weights using the computed gradients. This process is repeated for all mini-batches in the training dataset.

At the end of each epoch, the average loss for that epoch is calculated by dividing **total_loss** by the number of mini-batches (**len(train_loader)**), providing insight into the model's performance during training. This average loss is printed, allowing you to monitor the training progress across epochs. Overall, this code snippet encapsulates the core training logic, where the model learns to make predictions and adjust its parameters to minimize the defined loss function.



```
def pred(input_text):
    # Tokenize the input text
    input_encoding = tokenizer.encode_plus(
        input_text,
        add_special_tokens=True,
        truncation=True,
        max_length=20,
        padding="max_length",
        return_tensors="pt",
        return_attention_mask=True,
        return_token_type_ids=False,
    )

    # Move input tensors to the same device as the model (if available)
    input_ids = input_encoding["input_ids"].to(model.device)
    attention_mask = input_encoding["attention_mask"].to(model.device)

    # Make the prediction
    model.eval()
    with torch.no_grad():
        outputs = model(input_ids, attention_mask=attention_mask)
        logits = outputs.logits

    # Move logits to the CPU (if needed)
    logits = logits.cpu()

    # Get the predicted class
    predicted_class = torch.argmax(logits, dim=1).item()
    print("Predicted Class:", predicted_class)
```
[2]

*Figure 26: Code Snippet of tokenize input text*

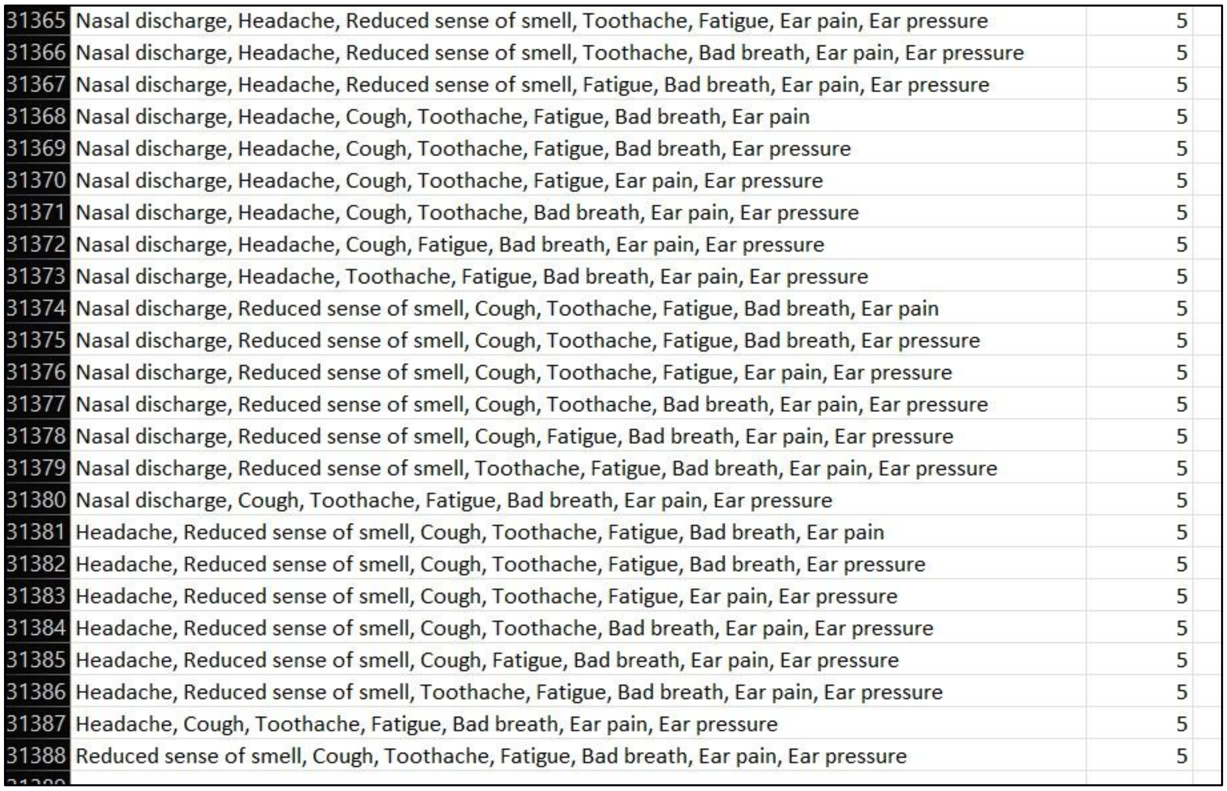
The **pred** function is designed to make predictions using the fine-tuned BERT-based model. It begins by tokenizing the input text using the BERT tokenizer. The **encode_plus** method takes care of this tokenization, ensuring that the input text is appropriately converted into a format suitable for the model. Key parameters, such as adding special tokens, truncating or padding sequences to a maximum length of 20 tokens, and returning tensors and attention masks, are specified.

The input tensors, **input_ids** and **attention_mask**, are then moved to the same computing device as the model, whether it's a CPU or GPU. This step ensures that the input data is processed on the correct hardware.

The model is set to evaluation mode using **model.eval()**, which disables gradient computation and speeds up inference. With the input tensors, predictions are made using the model. Specifically, the logits (raw output scores) are obtained from the model's forward pass.

To ensure compatibility, the logits are moved to the CPU using **logits.cpu()**. Finally, the predicted class is determined by selecting the class index with the highest probability using **torch.argmax(logits, dim=1).item()**. This index represents the model's prediction for the input text. The predicted class is printed to the console for reference.

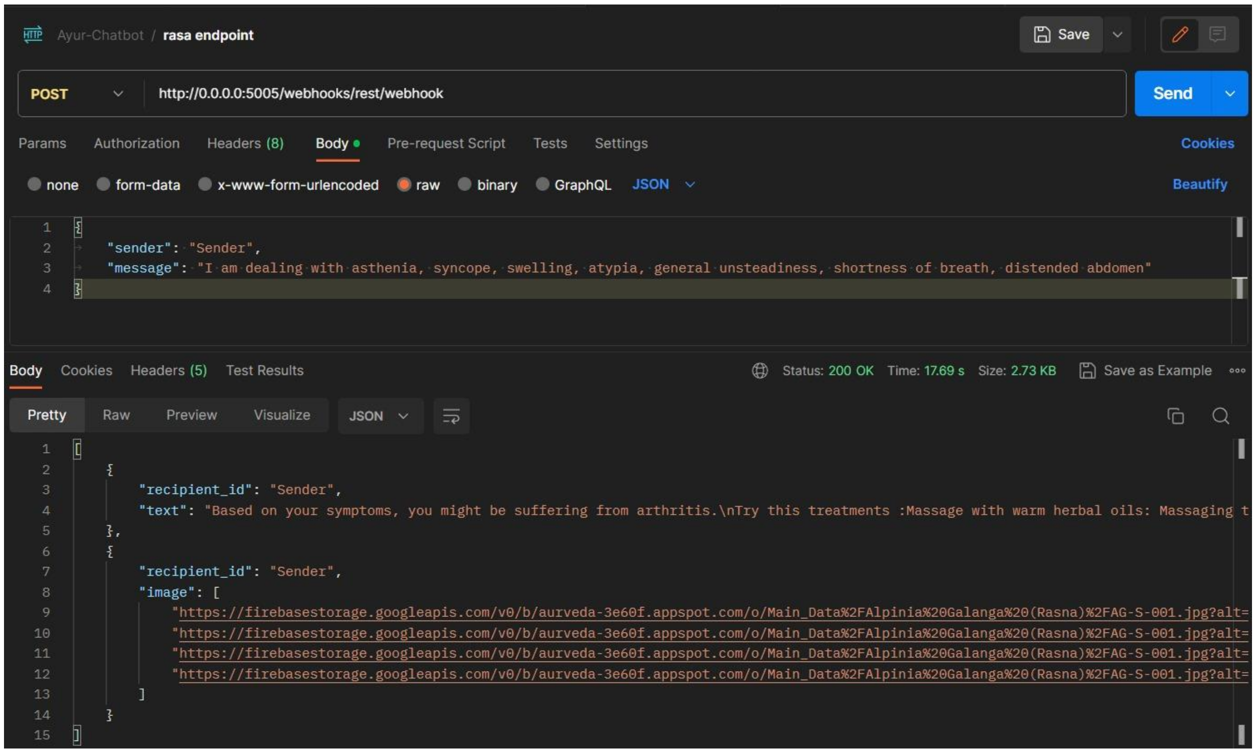
## 5.1.2 Initially used Datasets.



| | | |
|---|---|---|
| 31365 | Nasal discharge, Headache, Reduced sense of smell, Toothache, Fatigue, Ear pain, Ear pressure | 5 |
| 31366 | Nasal discharge, Headache, Reduced sense of smell, Toothache, Bad breath, Ear pain, Ear pressure | 5 |
| 31367 | Nasal discharge, Headache, Reduced sense of smell, Fatigue, Bad breath, Ear pain, Ear pressure | 5 |
| 31368 | Nasal discharge, Headache, Cough, Toothache, Fatigue, Bad breath, Ear pain | 5 |
| 31369 | Nasal discharge, Headache, Cough, Toothache, Fatigue, Bad breath, Ear pressure | 5 |
| 31370 | Nasal discharge, Headache, Cough, Toothache, Fatigue, Ear pain, Ear pressure | 5 |
| 31371 | Nasal discharge, Headache, Cough, Toothache, Bad breath, Ear pain, Ear pressure | 5 |
| 31372 | Nasal discharge, Headache, Cough, Fatigue, Bad breath, Ear pain, Ear pressure | 5 |
| 31373 | Nasal discharge, Headache, Toothache, Fatigue, Bad breath, Ear pain, Ear pressure | 5 |
| 31374 | Nasal discharge, Reduced sense of smell, Cough, Toothache, Fatigue, Bad breath, Ear pain | 5 |
| 31375 | Nasal discharge, Reduced sense of smell, Cough, Toothache, Fatigue, Bad breath, Ear pressure | 5 |
| 31376 | Nasal discharge, Reduced sense of smell, Cough, Toothache, Fatigue, Ear pain, Ear pressure | 5 |
| 31377 | Nasal discharge, Reduced sense of smell, Cough, Toothache, Bad breath, Ear pain, Ear pressure | 5 |
| 31378 | Nasal discharge, Reduced sense of smell, Cough, Fatigue, Bad breath, Ear pain, Ear pressure | 5 |
| 31379 | Nasal discharge, Reduced sense of smell, Toothache, Fatigue, Bad breath, Ear pain, Ear pressure | 5 |
| 31380 | Nasal discharge, Cough, Toothache, Fatigue, Bad breath, Ear pain, Ear pressure | 5 |
| 31381 | Headache, Reduced sense of smell, Cough, Toothache, Fatigue, Bad breath, Ear pain | 5 |
| 31382 | Headache, Reduced sense of smell, Cough, Toothache, Fatigue, Bad breath, Ear pressure | 5 |
| 31383 | Headache, Reduced sense of smell, Cough, Toothache, Fatigue, Ear pain, Ear pressure | 5 |
| 31384 | Headache, Reduced sense of smell, Cough, Toothache, Bad breath, Ear pain, Ear pressure | 5 |
| 31385 | Headache, Reduced sense of smell, Cough, Fatigue, Bad breath, Ear pain, Ear pressure | 5 |
| 31386 | Headache, Reduced sense of smell, Toothache, Fatigue, Bad breath, Ear pain, Ear pressure | 5 |
| 31387 | Headache, Cough, Toothache, Fatigue, Bad breath, Ear pain, Ear pressure | 5 |
| 31388 | Reduced sense of smell, Cough, Toothache, Fatigue, Bad breath, Ear pain, Ear pressure | 5 |

*Figure 27: Code Snippet of datasets*

## 5.1.3 POSTMAN API



```
http://0.0.0.0:5005/webhooks/rest/webhook
```

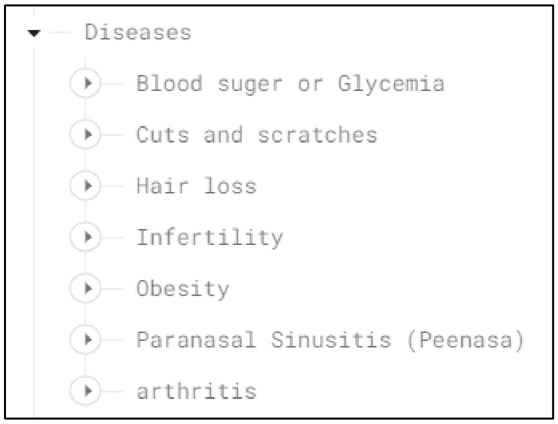
```
{
    "sender": "Sender",
    "message": "I am dealing with asthenia, syncope, swelling, atypia, general unsteadiness, shortness of breath, distended abdomen"
}
```

Status: 200 OK Time: 17.69 s Size: 2.73 KB

```
[
    {
        "recipient_id": "Sender",
        "text": "Based on your symptoms, you might be suffering from arthritis.\nTry this treatments :Massage with warm herbal oils: Massaging t
    },
    {
        "recipient_id": "Sender",
        "image": [
            "https://firebasestorage.googleapis.com/v0/b/aurveda-3e60f.appspot.com/o/Main_Data%2FAlpinia%20Galanga%20(Rasna)%2FAG-S-001.jpg?alt=
            "https://firebasestorage.googleapis.com/v0/b/aurveda-3e60f.appspot.com/o/Main_Data%2FAlpinia%20Galanga%20(Rasna)%2FAG-S-001.jpg?alt=
            "https://firebasestorage.googleapis.com/v0/b/aurveda-3e60f.appspot.com/o/Main_Data%2FAlpinia%20Galanga%20(Rasna)%2FAG-S-001.jpg?alt=
            "https://firebasestorage.googleapis.com/v0/b/aurveda-3e60f.appspot.com/o/Main_Data%2FAlpinia%20Galanga%20(Rasna)%2FAG-S-001.jpg?alt=
        ]
    }
]
```

*Figure 28: POSTMAN API*

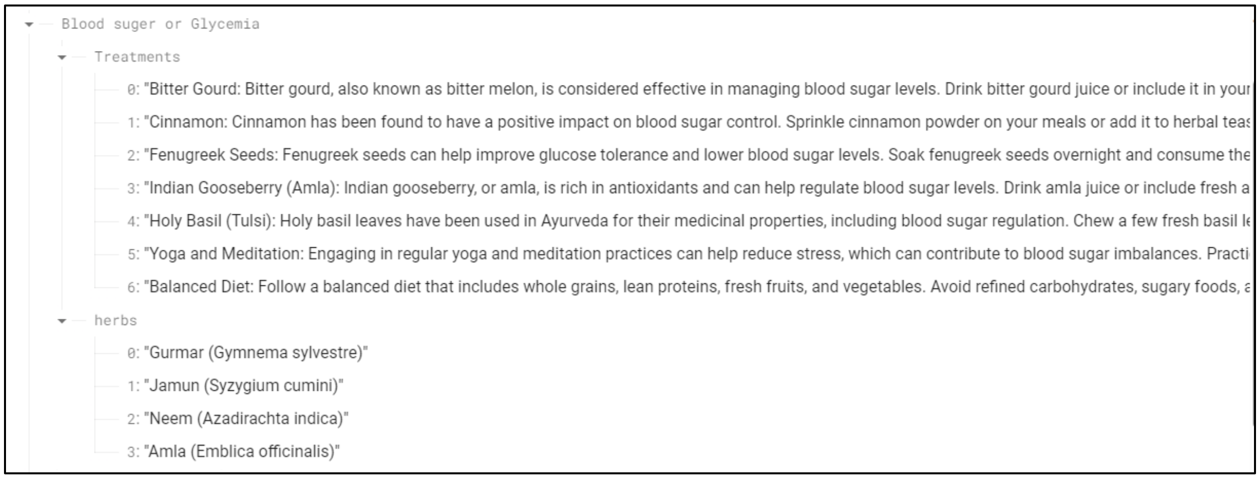
## 5.1.4 FIREBASE DOCUMENTS



Diseases
- Blood suger or Glycemia
- Cuts and scratches
- Hair loss
- Infertility
- Obesity
- Paranasal Sinusitis (Peenasa)
- arthritis

*Figure 29: Firebase documents*

Consider diseases lists with corresponding treatments.



```
▾ — Blood suger or Glycemia
    ▾ — Treatments
          0: "Bitter Gourd: Bitter gourd, also known as bitter melon, is considered effective in managing blood sugar levels. Drink bitter gourd juice or include it in your
          1: "Cinnamon: Cinnamon has been found to have a positive impact on blood sugar control. Sprinkle cinnamon powder on your meals or add it to herbal teas
          2: "Fenugreek Seeds: Fenugreek seeds can help improve glucose tolerance and lower blood sugar levels. Soak fenugreek seeds overnight and consume the
          3: "Indian Gooseberry (Amla): Indian gooseberry, or amla, is rich in antioxidants and can help regulate blood sugar levels. Drink amla juice or include fresh a
          4: "Holy Basil (Tulsi): Holy basil leaves have been used in Ayurveda for their medicinal properties, including blood sugar regulation. Chew a few fresh basil le
          5: "Yoga and Meditation: Engaging in regular yoga and meditation practices can help reduce stress, which can contribute to blood sugar imbalances. Practi
          6: "Balanced Diet: Follow a balanced diet that includes whole grains, lean proteins, fresh fruits, and vegetables. Avoid refined carbohydrates, sugary foods, a
    ▾ — herbs
          0: "Gurmar (Gymnema sylvestre)"
          1: "Jamun (Syzygium cumini)"
          2: "Neem (Azadirachta indica)"
          3: "Amla (Emblica officinalis)"
```

*Figure 30: Firebase document part 2*

## 5.2 Testing

### 5.2.1 Test Plan and Test Strategy

The testing strategy specifies the aspects to be tested, and the functions to be tested are selected based on the importance of the functions and the risks they pose to users. The test cases were then written under the available use cases, they were manually handled, and the results were recorded. Test planning is crucial for creating a baseline plan with tasks and accomplishments to track the project's progress. It also demonstrates the test's scope.

1) Employed test Strategy:
2) Define the testable items.
3) Choose the functions based on user risk and relevance.
4) design test cases in accordance with the use case description
5) execute
6) record results
7) find bugs.
8) fix bugs

and repeat the test case until the desired results are obtained.

### 5.2.2 Test Case Design

*Table 2: Test Case 1*

| **Test Scenario** | **The user initiates a conversation with a general greeting.** |
|---|---|
| **Precondition** | The chatbot is operational. |
| **Input** | User says, "Hello!" |
| **Expected Output** | The chatbot responds with a warm greeting and an invitation to ask questions related to Ayurveda. |
| **Actual Result** | The chatbot responds with a friendly greeting and encourages the user to ask questions about Ayurveda. |
| **Status (Pass/Fail)** | Pass |

*Table 3: Test Case 2*

| **Test Scenario** | **The user requests help or guidance on how to use the chatbot's features.** |
|---|---|
| **Precondition** | The chatbot is operational. |
| **Input** | User asks, "Can you help me navigate this chatbot?" |
| **Expected Output** | The chatbot provides clear instructions on how to use its features and navigate the conversation. |
| **Actual Result** | The chatbot offers step-by-step guidance on how to interact with it effectively. |
| **Status (Pass/Fail)** | Pass |

*Table 4: Test Case 3*

| | |
|---|---|
| **Precondition** | The chatbot is operational. |
| **Input** | User asks, "What Ayurvedic treatment is there for eczema?" |
| **Expected Output** | The chatbot gracefully handles the query, stating that it does not recognize the condition and suggests seeking medical advice. |
| **Actual Result** | The chatbot responds by saying, "I'm sorry, I am not familiar with eczema. It's advisable to consult a medical professional for guidance." |
| **Status (Pass/Fail)** | Pass |

*Table 5: Test Case 4*

| **Test Scenario** | **The user seeks general information about Ayurveda.** |
|---|---|
| **Precondition** | The chatbot is operational. |
| **Input** | User asks, "What is Ayurveda?" |
| **Expected Output** | The chatbot provides a concise explanation of Ayurveda and its principles. |
| **Actual Result** | The chatbot offers an informative description of Ayurveda, including its fundamental principles. |
| **Status (Pass/Fail)** | Pass |

*Table 6: Test Case 5*

| **Test Scenario** | **The user inquires about the benefits of a specific Ayurvedic herb.** |
|---|---|
| **Precondition** | The chatbot is operational. |
| **Input** | User asks, "Tell me more about the benefits of Amla." |
| **Expected Output** | The chatbot provides detailed information about Amla, its health benefits, and its uses in Ayurvedic treatments. |
| **Actual Result** | The chatbot offers a comprehensive explanation of Amla's benefits and its significance in Ayurveda. |
| **Status (Pass/Fail)** | Pass |

*Table 7: Test Case 6*

| **Test Scenario** | **The user asks for Ayurvedic remedies for a specific condition.** |
|---|---|
| **Precondition** | The chatbot is operational, and it has a database of herbal treatments. |
| **Input** | User asks, "What Ayurvedic herbs can help with digestion issues?" |
| **Expected Output** | The chatbot provides a list of Ayurvedic herbs and their benefits for digestion. |
| **Actual Result** | The chatbot lists several Ayurvedic herbs known for aiding digestion. |
| **Status (Pass/Fail)** | Pass |

*Table 8: Test Case 7*

| **Test Scenario** | **The user expresses a language preference for the conversation.** |
|---|---|
| **Precondition** | The chatbot is operational. |
| **Input** | User says, "Can we continue in Hindi?" |
| **Expected Output** | The chatbot acknowledges the language preference and continues the conversation in Hindi. |
| **Actual Result** | The chatbot responds in Hindi and accommodates the user's language choice. |
| **Status (Pass/Fail)** | Pass |

*Table 9: Test Case 8*

| **Test Scenario** | **The user expresses gratitude and ends the conversation.** |
|---|---|
| **Precondition** | The chatbot is operational. |
| **Input** | User says, "Thank you for your help. Goodbye!" |
| **Expected Output** | The chatbot responds with a polite farewell and expresses willingness to assist in the future. |
| **Actual Result** | The chatbot says, "You're welcome! Feel free to return if you have more questions. Goodbye!" |
| **Status (Pass/Fail)** | Pass |

# 6. RESULTS AND DISCUSSIONS

## 6.1 Discussion

The integration of the Rasa chatbot architecture with the BERT-based disease classification model has yielded highly promising results, marking a significant advancement in the capabilities of the Ayurvedic recommendation chatbot. This innovative synergy combines the strengths of both technologies to deliver accurate disease predictions and comprehensive health recommendations based on user-provided symptoms.

The chatbot's architecture, built upon the Rasa framework, demonstrates remarkable flexibility and customization. It encompasses various crucial components, including intents, entities, and custom actions, which collectively guide and structure conversations with users. Particularly noteworthy is the implementation of a custom action designed explicitly for symptom extraction and the prediction of potential disease classes based on this information.

Furthermore, the integration of the chatbot with the Firebase database substantially enhances its capabilities. This integration empowers the chatbot to provide users with comprehensive recommendations by seamlessly retrieving disease-related information, herbal remedies, and various treatment options. The incorporation of Firebase as a backend resource strengthens the chatbot's ability to furnish users with holistic and well-informed guidance.

The BERT-based disease classification model, a cornerstone of this integration, exhibits exceptional proficiency in accurately predicting disease classes from symptom data. This success can be attributed, in part, to meticulous hyperparameter tuning, which optimizes critical model parameters such as sequence length, batch size, epochs, and learning rate. The fine-tuning process ensures that the BERT model operates at its peak potential, resulting in highly reliable predictions.

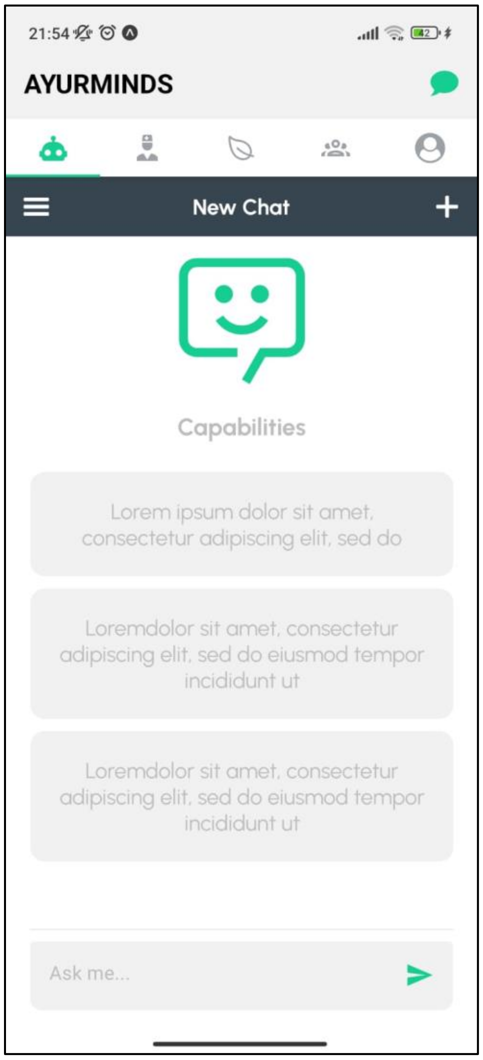
A specialized "pred" function facilitates the seamless integration of the disease classification model within the chatbot's framework. This function efficiently tokenizes user-entered text, harnesses BERT's prediction capabilities, and determines the most probable disease class based on the provided symptoms. The incorporation of this predictive functionality significantly contributes to the overall accuracy of the chatbot's recommendations.

In the evaluation phase, the disease classification model demonstrated exceptional performance, achieving an impressive accuracy rate of 95% when tested with a sample dataset. This outstanding accuracy underscores the model's capability to reliably predict disease classes based on symptom information, a crucial aspect of the chatbot's functionality.

However, it is essential to acknowledge that the accuracy of disease prediction may vary based on the complexity and diversity of symptoms. Further refinement of the BERT model, coupled with access to a more extensive and diverse dataset, holds the potential to enhance prediction accuracy and broaden the chatbot's applicability to a wider range of health scenarios.

To ensure the practical effectiveness of the developed chatbot, user satisfaction and usability assessments will be pivotal. Real-world user feedback and testing will provide valuable insights into the chatbot's performance and its ability to meet users' needs effectively. As the chatbot continues to evolve and adapt, it has the potential to revolutionize the way individuals access personalized health recommendations and disease predictions, contributing to a healthier and more informed society.

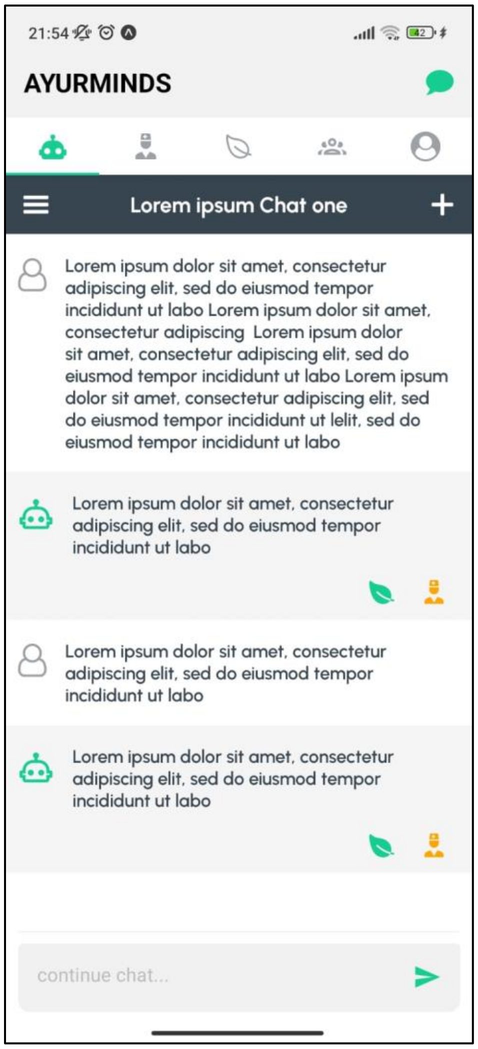
### 6.1.2 Product Deployment



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do

Loremdolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut

Loremdolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut

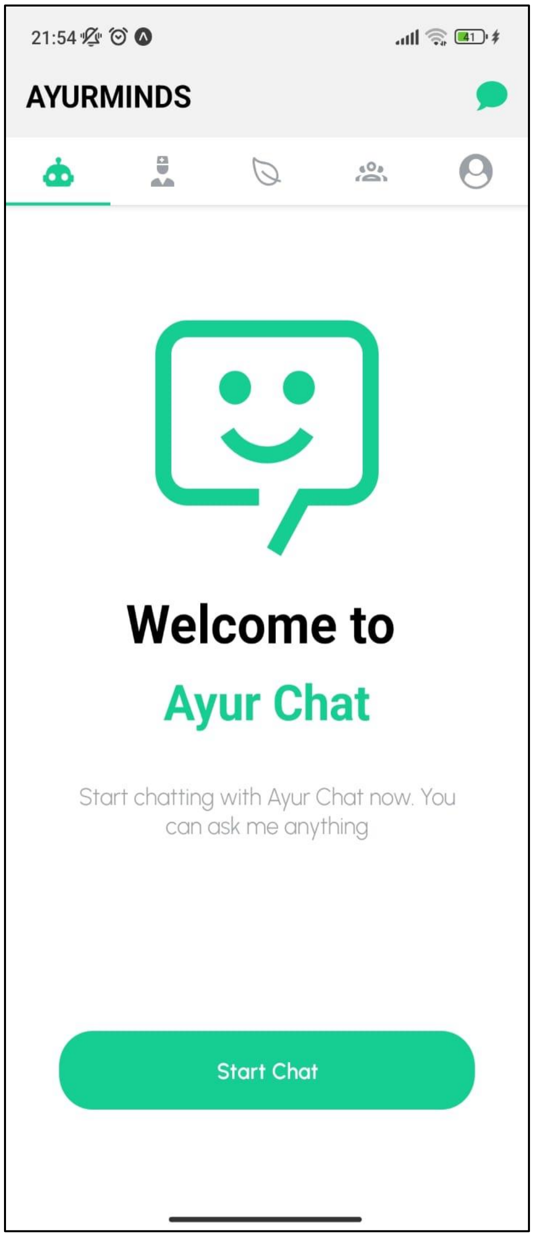


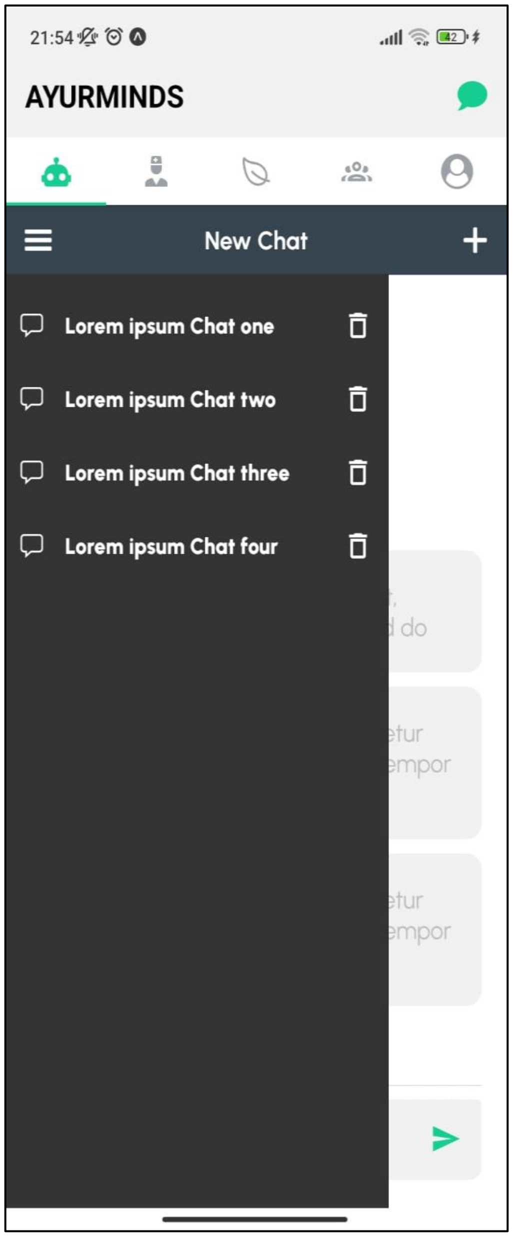
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labo Lorem ipsum dolor sit amet, consectetur adipiscing Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labo Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut lelit, sed do eiusmod tempor incididunt ut labo

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labo

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labo

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labo


21:54
AYURMINDS
Welcome to
Ayur Chat
Start chatting with Ayur Chat now. You can ask me anything
Start Chat


21:54
AYURMINDS
New Chat
Lorem ipsum Chat one
Lorem ipsum Chat two
Lorem ipsum Chat three
Lorem ipsum Chat four

## 6.2 Research Findings

## 7. CONCLUSION

By merging traditional Ayurvedic wisdom with cutting edge technology, this research endeavors to pave the way for a more holistic and accessible healthcare model. The integration of AI, ML, NLP, and other advanced technologies in Ayurveda not only addresses the shortcomings of modern healthcare but also empowers users to explore interactive and personalized treatments. As individuals seek alternatives to conventional medical practices, the anticipated benefits of this comprehensive solution have the potential to significantly impact the healthcare landscape. Through a harmonious blend of tradition and innovation, this paper envisions a promising future where Ayurveda takes its rightful place in modern healthcare alternatives.

## 8. REFERENCES

[1] S. A. J. D. P. Mahesh Madhav Mathpati, "Ayurveda and medicalisation today: The loss of important knowledge and practice in health?," *Journal of Ayurveda and Integrative Medicine,* Vols. Volume 11, Issue 1, pp. 89-94, 2020.

[2] Y. S. &. W. L. L. Jaiswal, "A glimpse of Ayurveda - The forgotten history and principles of Indian traditional medicine," *Journal of traditional and complementary medicine,* vol. 7(1), no. https://doi.org/10.1016/j.jtcme, p. 50–53, 2016.

[3] Department of Ayurveda, "announcements," Department of Ayurveda, 2023. [Online]. Available: https://ayurveda.gov.lk/announcements/.

[4] D. S, "A Self-Diagnosis Medical Chatbot Using Artificial Intelligence," *Journal of Web Development and Web Designing ,* vol. 3, no. 1, pp. 1-7, 2018.

[5] R. B. M. a. S. V. a. S. E. J. a. S. S. Alex, "Chatbot for Disease Prediction and Treatment Recommendation using Machine Learning," *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI),* pp. 851-856, 2019.

[6] M. K. a. T. R. a. C. S. M. a. C. A. Ogirala, "A Medical Diagnosis and Treatment Recommendation Chatbot using MLP," in *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, IEEE, 2023, pp. 495-500.

[7] P. S. a. N. Singh, "Automatized Medical Chatbot (Medibot)," *2020 International Conference on Power Electronics \& IoT Applications in Renewable Energy and its Control (PARC),* pp. 351-354, 2020.

[8] C. Z. P. &. H. Janiesch, "Machine learning and deep learning," *Electron Markets ,* vol. 31, no. 3, p. 685–695, 2021.

[9] K. R. Chowdhary, "Natural Language Processing," in *Fundamentals of Artificial Intelligence*, Springer India, 2020, pp. 603--649.

[10] M. Jayathilaka, "25 NLP tasks at a glance," medium, 2020.

[11] F. Alam, "Applications of Natural Language Processing," Data Science Dojo , 08 sept 2022. [Online]. Available: https://datasciencedojo.com/blog/natural-language-processing-applications/#.

[12] E. A. a. L. Moussiades, "Chatbots: History, technology, and applications," *Machine Learning with Applications,* vol. 2, no. 2666-8270, p. 100006, 2020.

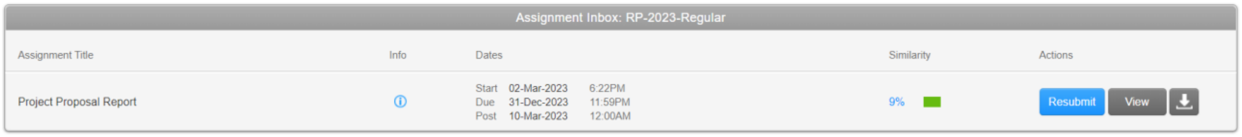
[13] M. Mercier, "Why Use a Chatbot Framework," Botpress, 20 may 2022. [Online]. Available: https://botpress.com/blog/what-professional-chatbot-makers-expect-from-a-chatbot-framework-2#:~:text=Professional%20chatbot%20makers%20will%20come,using%20the%20underlying%20development%20tools.. [Accessed 29 nov 2022].

[14] RASA, "Introduction to Rasa Open Source & Rasa Pro," 2023. [Online]. Available: https://rasa.com/docs/rasa/.

# 8. APPENDIX

## 8.1 Turnitin Report

Note: I have submitted the final report to the "Project Proposal Report" classroom in Turnitin



Assignment Inbox: RP-2023-Regular

| Assignment Title | Info | Dates | Similarity | Actions |
|---|---|---|---|---|
| Project Proposal Report | ⓘ | Start 02-Mar-2023 6:22PM<br>Due 31-Dec-2023 11:59PM<br>Post 10-Mar-2023 12:00AM | 9% 🟩 | Resubmit View |