DocBot: Medical Diagnosis Chatbot using *Scala 3*

Shady Ali[1], **Ahmed Sameh**[2], **Laila Khaled**[3] **and Nour Hany**[4]

[1]*Faculty of Computer and Information Sciences, Egypt University of Informatics*
Email: 22-101195@students.eui.edu.eg
[2]*Faculty of Computer and Information Sciences, Egypt University of Informatics*
Email: 22-101198@students.eui.edu.eg
[3]*Faculty of Computer and Information Sciences, Egypt University of Informatics*
Email: 22-101078@students.eui.edu.eg
[4]*Faculty of Computer and Information Sciences, Egypt University of Informatics*
Email: 22-101068@students.eui.edu.eg
[†]*These authors contributed equally to this work*

Abstract

The development and implementation of chatbots have been revolutionizing the way interactions and transactions occur within various domains, particularly in healthcare. Our project introduces "DocBot," a rule-based medical expert chatbot developed using Scala 3, designed to assist users in identifying potential health issues based on their symptoms. Utilizing a three-layer design, the chatbot operates with a sophisticated database system and employs a decision tree to process user inputs and offer preliminary diagnostic guidance. This system integrates various data tables including patient history, disease descriptions, and precautions to provide informed responses. Through a combination of rule-based logic and database interactions, DocBot is able to deliver initial health assessments, aiming to enhance user experience by increasing accessibility and efficiency in medical consultations.

Keywords: Chatbot, Decision Tree, Tokenization, Disease, Diagnosis, Symptoms, Database, Scala 3

1. INTRODUCTION

What is a chatbot?
A chatbot is a software application or web interface that is designed to mimic human conversation through text or voice interactions.

1.1. Historic Outline

While the concept of conversational agents has ancient roots in literature and folklore, the development of digital chatbots began in the 1950s. Over the decades, chatbots have evolved significantly, starting from a rule-based chatbots to advancements like natural language processing (NLP), machine learning, and artificial intelligence (AI).

1.2. Importance/Advantages of Chatbot

- Availability
- Accessibility
- Efficiency
- Scalability
- Reduce costs

1.3. How chatbots work?

A user starts a conversation by typing in a message or speaking to a chatbot through a user interface. The chatbot uses NLP to analyze the words and phrases in the message to understand the user's intent. The chatbot searches its database of pre-programmed responses for a relevant answer.

1.4. Types of Chatbots

- **Rule-Based Chatbots:** operate on predefined rules and responses
- **AI-Powered Chatbots:** uses machine learning and learn from data to improve overtime
- **Transactional Chatbots:** designed for specific tasks such as making reservations, placing orders, or providing customer support

2. Our chatbot's domain

In our project, we have chosen to develop a medical expert chatbot—a virtual assistant designed to assist users in identifying potential health issues based on their symptoms. This chatbot will serve as a valuable tool for individuals seeking preliminary guidance and information about their health concerns.
In this project, our main goal is to make an initial diagnosis for our user as they describe any symptoms or pains they feel. Our diagnosis consists of two *main* parts as following:

1. Detailed description of the disease provided by the bot
2. Multiple precautions to be taken by the patient

3. Project Description

3.1. Program Design

This project is using a Three-Layer Design:

1. **Presentation** Layer, designed to be the main graphical user interface (GUI).
2. **Business Logic** Layer, designed to hold the main logic behind the chatbot and all decision making process.
3. **Data Access** Layer, designed specifically to deal and communicate with the database and utilise it.

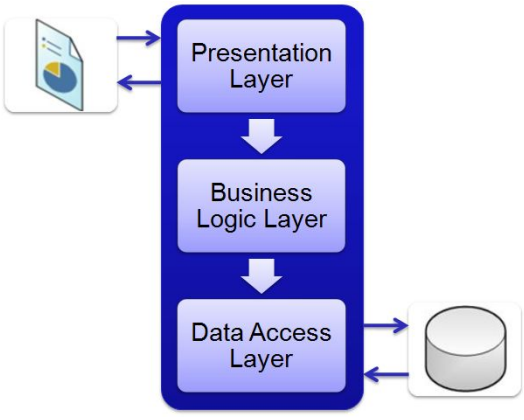

Figure 1. Three-tier program design

3.2. Chatbot Ruling

Our chatbot is mainly rule-based, which is further discussed and illustrated in the 4 Although it is a rule based chatbot, using a database has given us more flexibility and understanding of the user's different queries and responses, as we utilised many datasets for different applications in our bot ranging from understanding the user's impressions and overall feelings (**Sentiment Analysis**) to extracting the user's personal data or core topic words such as symptoms or illnesses he inquire on.

4. Analysis and Design

As explained in 3, we use a three-tier design, we will discuss the business logic layer further.
As a *rule-based* chatbot, we need to follow some kind of logic in a sequential fashion throughout the dialogue with the user. A decision tree model was implemented to make the dialogue flow more abstract, precise, and clear.

4.1. Business Layer

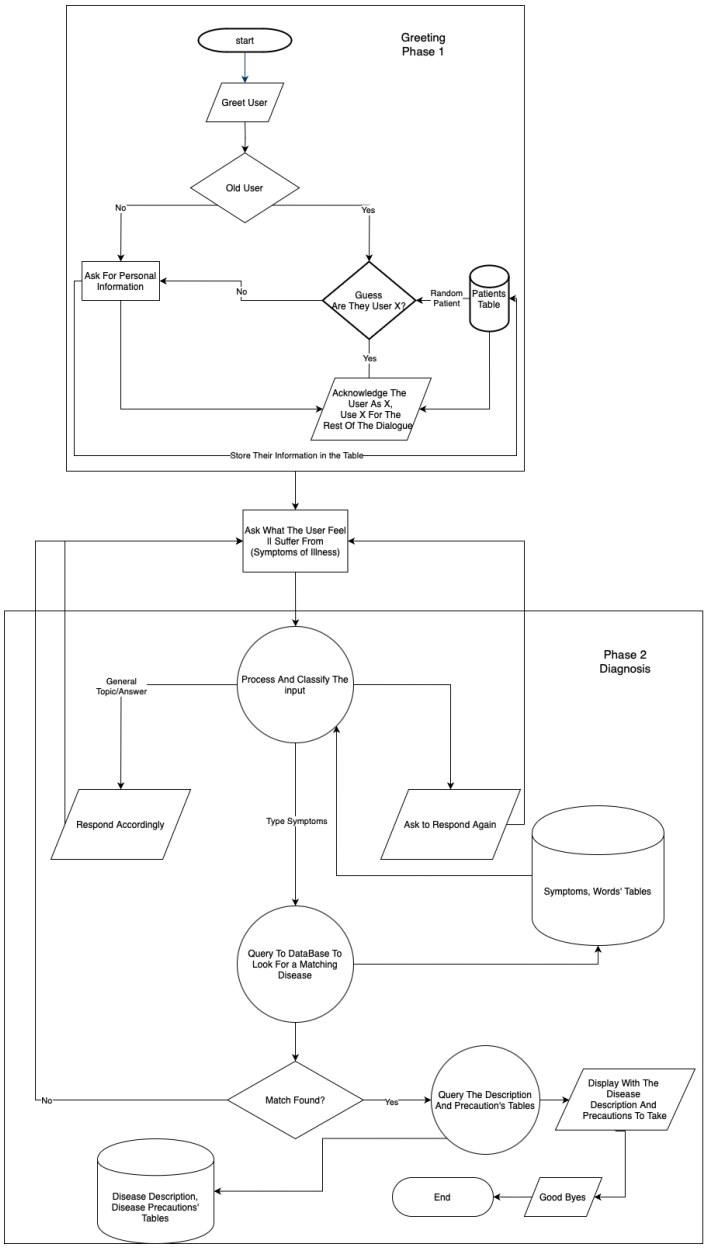

Figure 2. DocBot's Decision Tree Visualization

As 2 illustrates, there are two main phases in the code and dialogue:

4.1.1. Phase 1: Greetings

The Greetings are where the conversation starts. Initially, the user is greeted by the bot. The user if then asked whether he interacted with the bot before or not, if he did, the bot will try to guess his name through the Patients' Table in the database. If he was not able to guess correctly or the user responded with negation from the beginning, he will be asked to input his information such as first and last names, age, and gender.

4.1.2. Phase 2: Diagnosis

The Diagnosis phase is the core part of this project, as in it we try to process, analyze the user's input when asked how they feel or if there are certain symptoms of illness showing to find three potential cases:

1. They responded in a general manner, such as "I love potatoes". In this case, the bot can engage in this conversation briefly but try to direct back to asking about symptoms again.
2. They responded in an incomprehensible manner. In such case, the bots asks the user to clarify his words and describe the symptoms again.
3. Lastly, the user describes the symptoms directly. The bot then access the database to match the given symptoms with a known disease, if successful, the bot confirms the possibility of having such disease, with describing the disease through the Description Table and directing the user to proceed certain precautions to help them until a diagnosis from a practicing doctor is available to them.

4.2. Data Access Layer

In this layer, we used 15 unique tables in our database to utilise:

- Two Tables (Patients, Patient History) were used to store the patients data from names, age, gender, to disease history.
- Three Tables (Disease Description, Symptoms, Precautions) were used to diagnose the patient, give them details about their illness, and introduce some precautions to take.
- Ten Tables (Noun, Verb, Adjective, Adverb, Words), with each table having two versions, Positive and Negative, were used to categorize and breakdown the user's queries for sentiment analysis and intention identification.

The figure below illustrates the tables further:

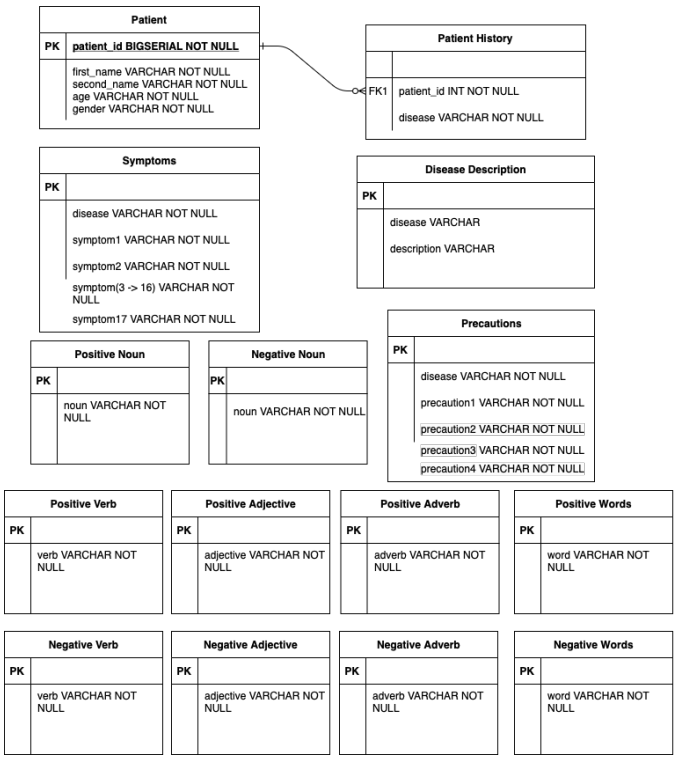

Figure 3. Entity Relation Diagram illustrates the tables and their relations

5. Code Functionality

This project was mainly made with Scala 3 as the core coding language, as well as implementing this project in a functional programming methodology, we utilised declarative programming, higher-order functions, and much more concepts.

- We used PostgreSQl Relational DataBase as our database and also docker. The database is accessed through Slick Library in Scala. Slick allows the developer to make queries to a database using Scala's functions such as Filter, Reduce, Map instead of querying with SQL directly. Tototoshi Library is used to read CSV files to import the datasets' content into their respective tables in the database.
- No certain libraries was used in the Business Logic Layer except trivial Scala libraries such as scala.io, scala.util, scala.concurrent.
- In the Presentation Layer, ScalaFX and CSS programming language was used to implement a proper interface for the user to interact through with the bot.

6. Contributions

Shady Ali:

- Database and Data Access Layer
- Greeting Phase
- Report

Nour Hany:

- Input Handling
- Demo Video

Laila Khaled:

- Diagnosis Phase
- Report

Ahmed Sameh:

- Presentation Layer
- Linking the Business Layer with the Presentation Layer

7. Discussion And Future Directions

The rule based direction to develop the chatbot by proved quite useful unlike initially expected in comparison with other approaches like NLP and Machine Learning. While it is still limited in many ways in its interactions and responses, combined with a database, we were able to break free from many restrictions and limitations in development phase to make the bot more interactive and responsive.
We faced difficulties in the process of importing the datasets into the database, as we planned initially to import them directly through SQL, but we couldn't overcome an error related to the database client (***PSQL***) as he was not able to locate the datasets in the system, so we imported them using Scala's CSV library *Tototoshi*, which we wanted to avoid but deemed inevitable.
We plan to imporve on this project further in the future by integrating Machine Learning techniques such as Naive Bayes to make it a hybrid chatbot using both Machine Learning and Decision Trees. We might also directly transition to NLP as a new challenge to practice tokenization methods and advanced sentiment analysis.

8. Conclusion

The DocBot project demonstrates the effective use of rule-based systems combined with a robust database to create a responsive and reliable medical diagnostic chatbot. Throughout its development, the project faced several challenges, particularly in dataset integration, which were overcome by adapting the implementation strategy to include Scala's CSV handling libraries. Despite its current limitations in handling complex user interactions, the chatbot has shown significant potential in improving user engagement and diagnostic accuracy. Future improvements will focus on integrating machine learning techniques to create a hybrid system, enhancing the chatbot's capabilities to handle more nuanced interactions and to provide more accurate diagnoses. The transition towards incorporating natural language processing stands as a promising frontier to further enhance the chatbot's effectiveness and reach in medical assistance.