CHAT-BOT FOR TELEMEDICINE

DINITHI FERNANDO

A project report submitted in conformity with the requirements
for the degree of Master's of Science in Information Technology

Department of Mathematical and Physical Sciences
Faculty of Graduate Studies
Concordia University of Edmonton

CHAT-BOT FOR TELEMEDICINE

DINITHI FERNANDO

Approved:

Rossitza Marinova, Ph.D.

Supervisor Date

Committee Member Name, Ph.D.

Committee Member Date

Patrick Kamau, Ph.D.

Dean of Graduate Studies Date

Abstract

Chat-bots could be considered as one of widely used technologies since it increases the business efficiency and throughput. Proposed project is to develop a web based chat-bot that is capable of booking appointments via the website Telecare Plus. Machine learning techniques are used to provide an interactive and personalized learning experience for users. Users details are be retrieved by the bot to book an appointment, if the user gives permission. Various tools and resources such as python libraries are integrated with the chat-bot to help users to practice and provide answers. It is not easy for many users to book an online appointment because of the technical knowledge and skill levels they have. One of the main advantages of this chat-bot is the users are able to provide details to book an appointment in a similar way they do when a patient interacts with a live agent. User studies are used to evaluate the effectiveness of the chat-bot as well as to assess its usability, engagement, and outcomes. The medical field could have a significant impact because of this project by providing an accessible and efficient way to book appointments.

Keywords: Machine Learning; Chat-bot; Artificial Intelligence; Natural Language Processing; Telemedicine; TelecarePLUS; Electronic Medical Records; Digital Healthcare.

Acknowledgement

The project could not have been successfully completed without the careful assistance of Dr. Rossitza Marinova, her willingness to motivate the researcher by providing all the subject related detail contributed tremendously in developing the application. Besides the researcher would like to take this as an opportunity to, express gratitude towards Concordia University of Edmonton for offering this subject. It gave the researcher a great opportunity to participate and learn IT Project module. Then the students who have participated in the evaluation process of the application are gratefully acknowledged for offering their time and supporting to develop the application. Finally to family members and friends for their immense support and understanding in one way or another to complete the project, thank you.

Contents

List of Tables

List of Figures

Listings

1 Introduction

The way people interact with technology has changed dramatically with the development of artificial intelligence (AI) and natural language processing (NLP). Due to their capacity to engage consumers in discussions that are natural and human-like, chat-bots in particular have drawn a lot of interest. By automating the appointment scheduling procedure, the Appointment Booking Chat-bot bridges the gap between businesses and customers and gets rid of the need for manual form submissions and more time-consuming, traditional methods like text massages and email.

Chat-bot is a conversational agent that communicates with the user with the aid of natural language sentences. The process of the chat-bot is to produce a meaningful answer for the user entered question with the aid of the knowledge base and output the answer through a graphical user interface.[1].

Healthcare professionals may now communicate with patients thanks to technological improvements, especially for people with mobility issues who live in rural or distant areas. Enhancing patient care is telemedicine's main objective.

Enabling two-way, real-time contact between patients and providers at a distance to improve a patient's health[7]. Virtual reality and robotics components have also been incorporated into certain pre-existing platforms [6]. It is a quickly developing service that aims to expand accessibility to high-quality healthcare while making it simple and affordable, especially in the wake of the COVID-19 pandemic.

The project is to develop a chat-bot that can effectively book appointments by retrieving it from the users while having conversations. The chat-bot is designed using natural language processing (NLP) and machine learning (ML) techniques in order to provide interactive and personalized learning experience for users. Programming languages such as Python, HTML and Javascript are used to implement the chat-bot application, offer feedback and chat facilities [2].

The researcher has carefully selected prototyping methodology to develop the application after taking account about the requirements and evaluating the pros and cones of each existing methodologies. Prototyping methodology allows the users to test the application and provide feedback, so it helps for the researcher to let users to test a prototype of the application and do changes until the users are fully satisfied. **Figure 1** shows a clear idea of how the prototyping methodology works. In the figure it shows that the analysis, design and implementation phases are iterative and continued until the final system. Completion of the analysis and design phrases a prototype system could be generated so that the user feedback is obtained to do further changes.

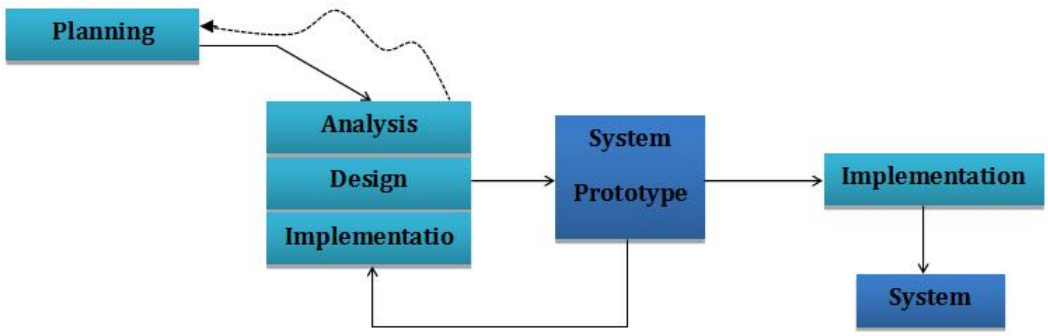

Figure 1: Prototyping Methodology [17]

2 Objectives / Research Questions

The main objectives of this project are:

- To develop a chat-bot application to retrieve user input during the conversation.
- To create interactive and natural texting experience for users by using machine learning tools and techniques.
- To create a user interface for the chat-bot to interact with users.
- To integrate the chat-bot with Telemedicare website to pass data and book appointments.
- To conduct user studies in order to rate the effectiveness in engagement, usability, and learning outcomes.
- To make it accessible for a broader audience to help with users who has various levels of technical knowledge.
- To contribute to the field of healthcare by providing an innovative and accessible approach to book appointments.

These objectives lead to provide an innovative and effective solution for booking appointments for patients. The project is hugely contributing to the field of health as well as promote inclusively in telemedicine.

Below is the list of research questions for TelecarePLUS platform:

- Compared to traditional measures, how effective is telemedicine in contributing to improve the overall outcome?
- What are the obstacles that could prevent telemedicine from being adopted by both parties?
- What are the main elements that make telemedicine successful?

- What are the legal and ethical concerns related to telemedicine, and how may they be resolved?
- What are the ideal technology and infrastructure requirements for the adoption of telemedicine?
- Can telemedicine help people in remote areas have better access to healthcare services?
- What long-term effects are connected to the adoption of telemedicine, and how can they be guaranteed for ongoing model optimization?

There are many benefits of telemedicine that contribute to patient and service provider's satisfaction since it is easy to use, able to provide improved outcomes, cost saving and more patients prefer telemedicine since they experience safety and trust.

3 Literature review

This chapter provides a detailed literature review about content which are relevant to web based chat-bot for students. It provides information about artificial neural networks and the ways of cat-bots communicate with humans.

3.1 AI in chat-bots

Over the years, telemedicine has become more and more popular. After the COVID-19 epidemic, it particularly saw an exponential surge. Due to recent developments in health technology and policy reforms, telemedicine has made it possible for providers of healthcare services to deliver remote medical care via electronic means [8].

The topic of the project is Web based chat-bot, which have a business domain of a chat-bot which retrieve details of patents who needs to book appointments. The technology undertaken is Machine Learning, Natural Language Processing, Python, HTML, JavaScript and CSS.

The literature review was carried out under advancement areas, as shown in **Table 1**. Areas such as artificial intelligence, information technology and website are subjects of this research.

Table 1: Areas of the project

Areas	**Similar Terms**	**Category**
Artificial Intelligence	Machine Learning, Deep Neural Network, Recurrent Neural Network, Natural Language Processing, Semantic analysis, Syntactic analysis, Word embedding	Chatbot
Graphical User Interface	User Interface	Technology
Website	Web page, Web design, Header, Link, Chat page, Send button	Chat

AI takes a special place when developing a chat-bot. AI is in its early 60's but it still has a long way to go. When it comes to AI people think that machines are taking the place where humans are and they are going to defeat humans. But actually what happens is that AI helps humans to do their work more efficiently.

Machine Learning is a sub-path of AI. Machine Learning includes vivid deep learning concepts which enable computers to think and do whatever it comes to humans naturally. Deep learning gains attraction towards technology by making things possible which people who lived before 20th century thought unbelievable.

Under deep learning techniques there exist many sub neural networks, among them most important neural networks are DNN, CNN and RNN. As illustrated in **Figure 2** deep neural network (DNN) includes an input layer, an output layer and at least one hidden layer in between those two layers. Each layer was assigned with a specific task which includes ordering and sorting. Most of all DNN (Figure 2) are similar to the activity of human brain.

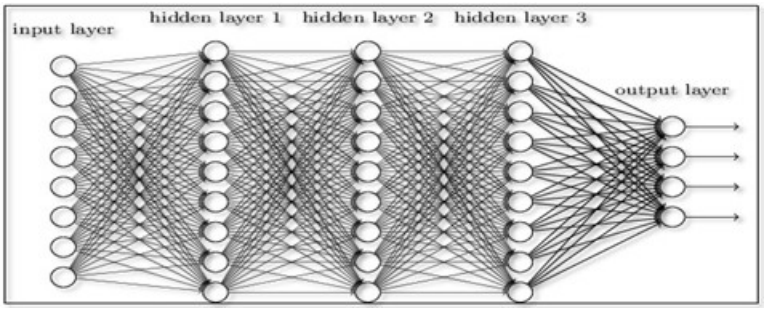

Figure 2: Deep Neural Network [18]

When it comes to convolution neural network (CNN) it is enable to learn input data and users. CNN is able to process 2D data like images by using 2D convolutional layers as illustrated in **Figure 3**. In order to interpret words in a sentence recurrent neural networks (RNN) are used. In RNN output of one layer is added to the next input. See **Figure 4** for an example of RNN.

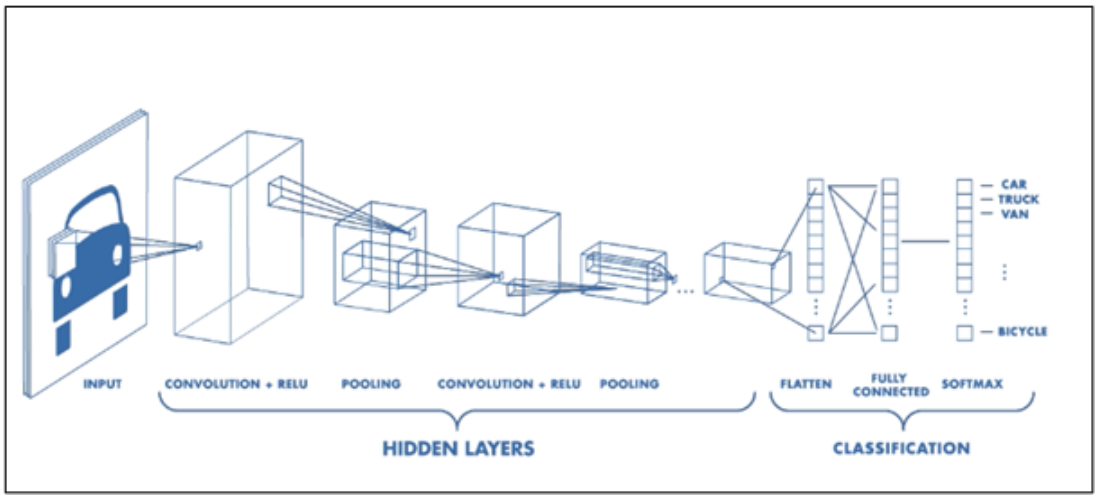

Figure 3: Convolution Neural Network [19]

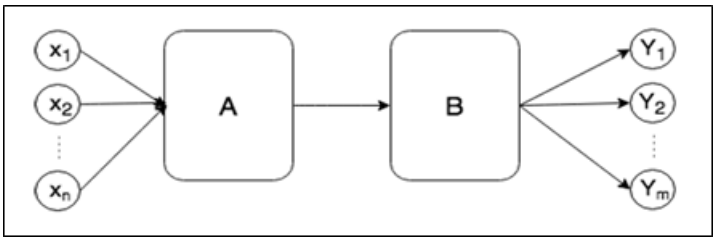

Figure 4: RNN with two hidden layers [20]

3.2 Chat-Bot Systems

Many people are interested in using chat-bots so why not integrate it with health care sector to schedule appointments? According to studies by R. Sanjaya et al. (2022) [9], a chat-bot can be successfully integrated with a healthcare appointment system to greatly lessen the workload of administrative employees and give patients a convenient way to schedule appointments.

Using user surveys, Chao et al. (2018) [2] developed a chat-bot that was integrated with a platform for learning programming. The outcomes demonstrated that the chat-bot was successful in assisting students in learning programming concepts and in giving them performance-based feedback.

Similar to this, Salehi [15] created a smart chat-bot that employs machine learning

to customise students' learning experiences. It was determined through user research that the chat-bot considerably increased pupils' knowledge and skill in programming.

A chat-bot that offers tailored programming learning routes was created by Wang et al. (2020) using reinforcement learning [3].

3.3 Technologies Used In Chat-Bot Systems

A recent study [10] has stated technologies and programming of cloud-based chat-bots. Initially it has considered about how the chat-bot works. First the user should initiate the communication by adding user names in their instant messaging platform account. Then the signal including the chat-bot user ID is delivered over HTTP to the bot server. Finally the bot output a welcome or out-of-service message in order to continue the communication. **Figure 5** shows a use case on how the chat-bot works.

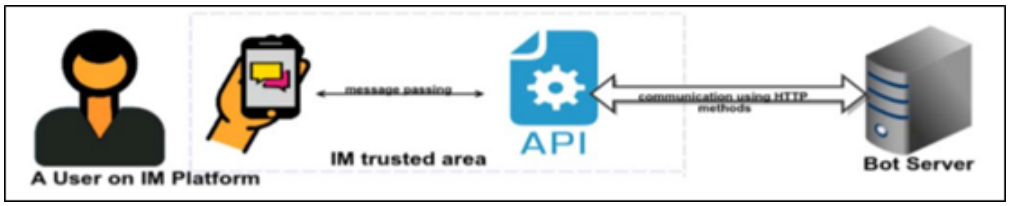

Figure 5: How chat-bot works [10]

Conversational dialog engine enables to generate responses based on previous conversations, which was developed in Python. This provides the base for developing most chat-bots.

One of special libraries in Python is ChatterBot [16]. This enables the programmers to create chat-bots easily since ChatterBot is responsible of generating automatic responses using different machine learning algorithms. In order to install the user should run 'pip install chatterbot' command. **Figure 6** includes the basic configuration codes of the ChatterBot.

```
from chatterbot import ChatBot

chatbot = ChatBot(
'Chat bot,
trainer='chatterbot.trainers.ChatterBotCorpusTrainer'
)

# english corpus
chatbot.train("chatterbot.corpus.english")

# Get a response to an input statement
chatbot.get_response("Hi, how are you today?")
```

Figure 6: Basic configuration codes of Python ChatterBot

There are two types of responses that a chat-bot could generate, they are static and dynamic. Static responses are generated using the templates which the chat-bot already has. For instance if the chat-bot response "The train time is <ft> hours", <ft> is a variable the chat-bot is now computing. Dynamic responses are generated by selecting a better response out of its knowledge base. In order to do so the chat-bot should learn millions of responses.

There are three major categories of chat-bot platforms, they are non-programming chat-bots, conversation-oriented chat-bots and platforms by tech giants chat-bots.

Let's consider each platform briefly.

- *Non-programming chat-bot* includes one of the basic types of non-technically oriented platforms. As the name implies in order to develop such type of chat-bots the developer doesn't need any knowledge regarding machine learning nor natural language processing. Examples for such platforms are Chatfuel, ManyChat and Motion.ai.
- *Conversation-Oriented chat-bots* use platforms which includes specification languages like AIML. User interactions are modeled by AIML. To get a better knowledge on AIML lets consider about few tags used in AIML.

  The <star> tag captures a particular text out of user entered sentence. In the codes of **Figure 7** the chatterbot is able to answer that it likes anything the user also likes.

```
<category>
   <pattern> I LIKE * </pattern>
   <template>
      I like <star/> too.
   </template>
</category>

<category>
   <pattern> A * IS A * </pattern>
   <template>
      When a <star index="1"/> is not a <star index="2"/>?
   </template>
</category>
```

Figure 7: star tag

The <srai> tag enables AIML interpreter to search answers efficiently from different user inputs. In **Figure 8**, separate <category> tags are created to talk about Alan Turing and Albert Sabin. But users might not ask about those researchers in the same way. In order to chat-bot to identify the different questions about those same researchers a new <category> tag was created including the <srai> tag.

```
1  <category>
2     <pattern> WHO IS ALAN TURING? </pattern>
3     <template>
         Alan  Turing  was  a  British  mathematician,  cryptographer,
4        and computer scientist often credited as
         the founder of modern Computer Science.
5     </template>
6  </category>
7
8  <category>
9     <pattern> WHO IS ALBERT SABIN? </pattern>
10    <template>
11       Albert Sabin was the researcher who developed
         the vaccine that is the main defense against polio.
12    </template>
13 </category>
14
15 <category>
16    <pattern> DO YOU KNOW WHO * IS? </pattern>
17    <template>
18       <srai> WHO IS <star/> </srai>
19    </template>
20 </category>
```

Figure 8: category tag

In line 16 of **Figure 8** the wildcard "*" identifies the name of the researcher that the user enters.

The <set> tag is used in order to define variables and the <get> tag is used to return answers stored by the set tag. **Figures 9 and 10** indicate <set> tag and <get> tag respectively.

```
<category>
        <pattern> MY NAME IS * </pattern>
        <template>
                Hello <set name="nameUser">
                <star/></set>
        </template>
</category>
```

Figure 9: set tag

```
<category>
        <pattern> GOOD NIGHT </pattern>
        <template>
        Good night <get name="nameUser"/>
        </template>
</category>
```

Figure 10: get tag

- *Chat-bots by tech giants* could be considered as easily developable chat-bots. Tech giants such as Facebook developers Wit.ai, Google develop Api.ai, Microsoft develops LUIS, IBM develops Waatson and Amazon develops Lex. Let's consider some of them.

Contexts and intents are considered as key concepts of the Api.ai model. Intents are used to create links including actions the chat-bot should perform for the user inputs. Context includes the string values which differentiate the requests. Within the help of contexts and intents Api.ai are able to handle large and complex systems. One of main features of Api.ai is that it allows integrating with several platforms like Facebook and Twitter from one click. Not only that but also these chat-bots are able to handle the code proactively in order to decrease server side coding. **Figure 11** includes an Api.ai chat-bot.

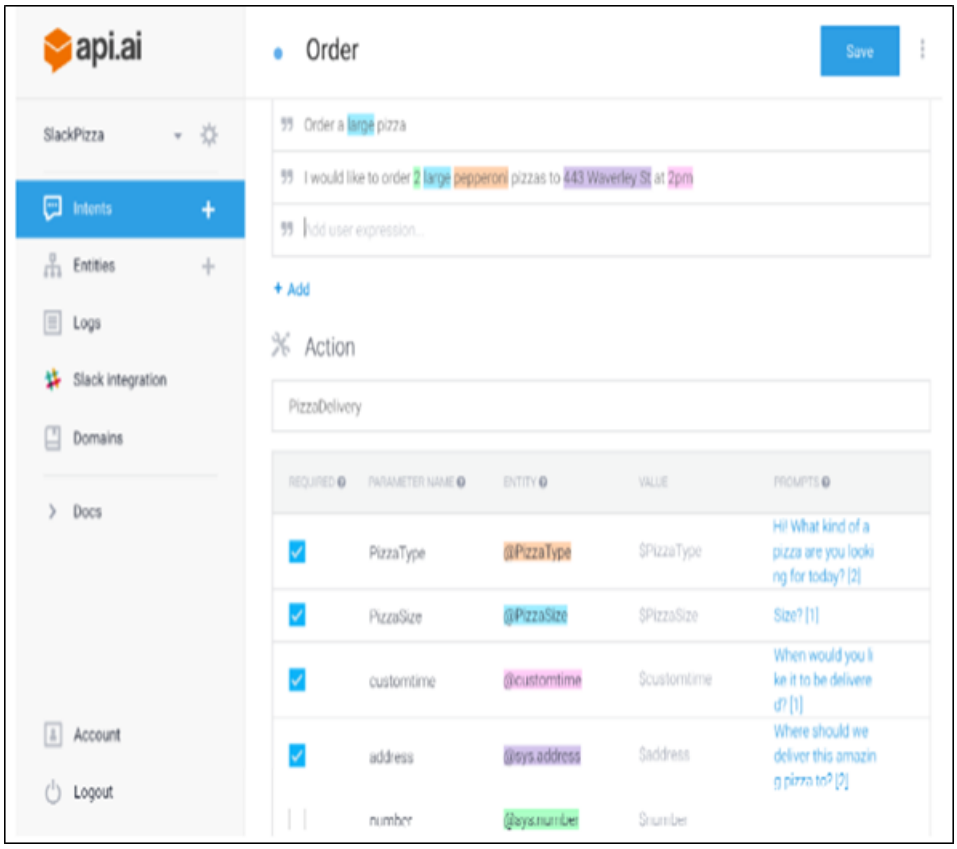

Figure 11: Api.ai chatbot [10]

The Wit.ai chat-bots are based on the stories which enable to model the behavior. Usually the developers use examples to teach the Wit.ai. The chat-bot extract entities by processing requests when a user input information about a similar item. Not only that but also these chat-bots can have branches to trigger when it comes to specific conditions in order to control the conversation. Below **Figure 12** includes a Wit.ai chat-bot [10], [11], [12].

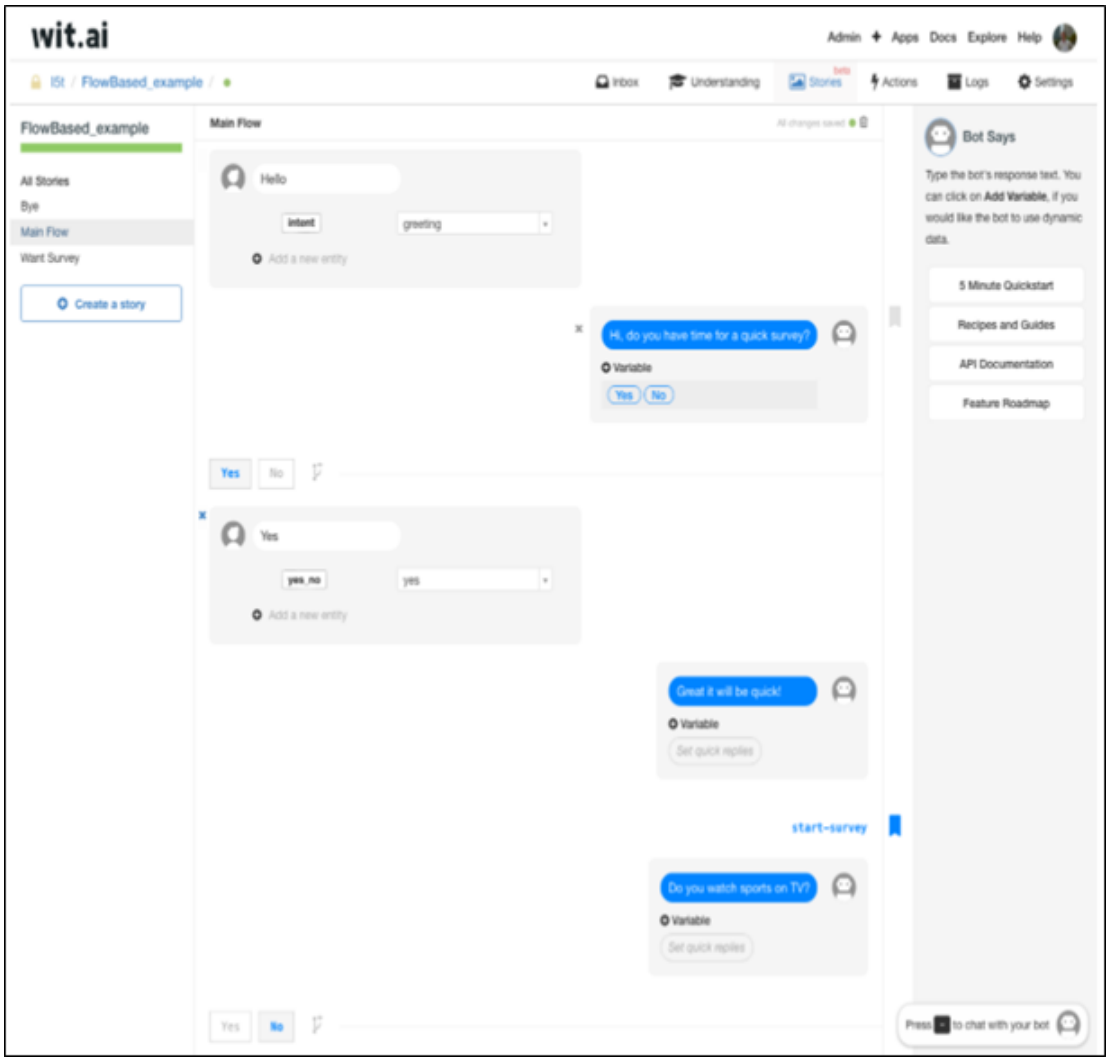

Figure 12: Wit.ai chat-bot [10]

4 Project Design

This phrase of the software development life cycle addresses the diagrams, wire-frame designs and answering algorithm of the system. It was difficult to find a proper application to design the wire-frames of the system. Many applications that were tested by the researcher include the watermark of the application within the wire-frame, which make changes of the final picture of the wire-frame designs. Because of this the wire-frames are simply designed using Mockflow website.

The researcher has also created a Gantt chart to break down the project tasks. By creating a Gantt chart the researcher was able to plan work around deadlines by managing the time and allocating resources efficiently. **Figure 13** states the Gantt chart for the chatbot.

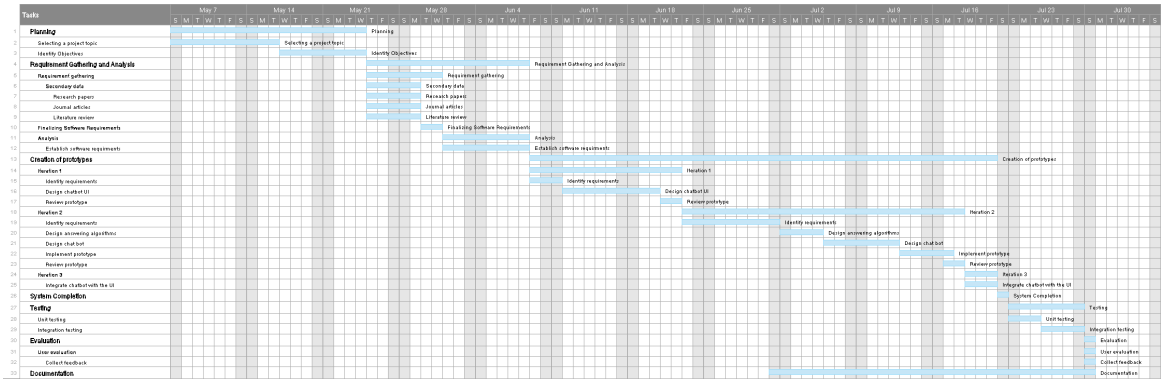

Figure 13: Gantt Chart

5 Project Implementation and Results

This section indicates about the overall system, technologies used and it's functionalities.

5.1 Technology Used

The chat-bot of the proposed system was implemented using Python programming language. The researcher has used Flask web framework that helps to build web applications quickly and efficiently. Flask framework helps to handle the chatbot's logic and interactions with users. The main reason it was selected than other Python frameworks is because Flask enables integration with various messaging platforms and web interfaces.

Below points indicate how Python Flask [14] can be utilized for building chat-bot applications:

- **Web Server:** Flask can act as a web server, receiving incoming HTTP requests from users interacting with the chatbot.

- **Routing and Endpoints:** One of the specialties of Flask is it enables to define routs and endpoints to handle different types of user interactions.
- **NLP Integration:** In order to understand user messages it could be even combined with Natural Language Processing libraries.
- **Dialog Management:** This feature helps to maintain context during conversations and guide users through multy-turn interactions.
- **Integration with External Services:** This feature enables the facility to retrive appoinment schedules or confirm bookings.
- **Continuous Learning and AI Integration:** To improve chatbots response it can use reinforcement learning capabilities based on user feedback and interactions.
- **Authentication and Security:** The security measures helps to protect sensitive user information.
- **Response Rendering:** Depending on the messaging platform or the web interface used Flask can handle chatbot responses in different formats like text, buttons and etc...
- **Scalability and Deployment:** Easy to deploy and can be hosted on various platforms.
- **Logging and Analytics:** To improve chatbot's performance Flask provides logging capabilities, allowing developers to monitor and analyze chatbot interactions.

In order to design a responsive and a user friendly UI the researcher has used HTML and CSS along with JavaScript. With the help of these tools an attractive web design was developed.

5.2 Back-end Development

Back-end of the chat-bot is like the engine of a vehicle. It includes all the functionalities that require to process the outcome developer requires, and in this chat-bot program it is mainly focused to retrieve responses from the user to pass to Telemedicare website. The researcher has created a file name 'App.py' to code all the main Python functionalities and then has imported the Flask framework. **Listing 1** shows how to import Flask.

```
from flask import Flask, render_template, request, session

app = Flask(__name__)
app.secret_key = 'your_secret_key'
```

Listing 1: Import Flask

Below are the code that helps to store user entered data shown in the **Listing 2**. Here the chat-bot initiates the conversation by introducing itself and requesting consent from the user to share details. If only user type 'yes' as the response the chat-bot navigates to ask the name of the user.

The chat-bot captures the user name in a variable called 'name' and stores it in a session. So when it's next response the bot retrieves the stored variable data in 'name' and output it to ask the next question by calling the user by her/his name. This method is also enables user to validate the data that they entered previously and could be implemented in a later version to add changes to the data entered while the conversation is flowing without waiting till the end of the conversation.

After retrieving the first name the chat-bot then greets the user by name and asks for the last name of the user. Bot always check whether the data user entered is already stored in the session and if not then the bot take action so store the new data. This avoids the user entering duplicate data. After checking if the last name is not stored the chat-bot then will store users last name in a variable named as 'last_name'.

Then the chat-bot navigates to retrieve age of the user by outputting the chat dialog 'Thank you. Enter your age ? Please specify as X years Y months format.' as shown in line 11 of **Listing 2**. Here the user is asked to specify age as in number of years and months. This action is taken by the developer because the patient could be less that one year old infant. After user reply for the question chat-bot will store the number of years and number of months in two separate variables 'age_ in_ years' and 'age_ in_ months'.

The next 'elif' statement navigates the question for the user to enter their gender. The developer hopes to develop this functionality to implement a drop-down list so that the user can use select their gender out of the options in the drop-down list in the future development.

After that it was asked by the user to enter their phone number. Here the developer has used a special built-in string method 'isdigit()'. This allows to validate user input whether it is a positive integer digit or else the function returns false and it'll stop the conversational flow. At this point the developer is hoping to develop a function in future developments that allow the user to re-enter the phone number if he/she got it wrong in the first time.

Next the chat-bot asks about the symptoms that user have. The chat-bot will output the chat dialog ' What are your symptoms ? Please specify at least three symptoms separated by a comma'. The list of symptoms that the user entered are separated by comma and stored in the session one by one by the bot.

Then the chat-bot asks the user whether the user have any medical files to upload. If yes the bot also asks to enter the keyword 'upload'. If use enter 'upload' then the chat-bot will finally output the file picker in the chat. See line 33 of **Listing 2**.

```
1   stored_data = session['stored_data']
```

```
    if query == "yes":
        response = "What is your first name?"
    elif 'first_name' not in stored_data:
        name = query.strip()
        response = f"Hi {name}. Good to meet you. What is your last name?"
        stored_data['first_name'] = name
    elif 'first_name' in stored_data and 'last_name' not in stored_data:
        last_name = query.strip()
        response = "Thank you. Enter your age? Please specify as X years Y months format."
        stored_data['last_name'] = last_name
    elif "years" in query.lower() or "months" in query.lower():
        split_data = query.split(" ")
        age_in_years = split_data[0]
        age_in_months = split_data[2]
        stored_data['age_in_years'] = age_in_years
        stored_data['age_in_months'] = age_in_months
        response = "What is your gender? Please specify as Male/Female/Others?"
    elif "male" in query.lower() or "female" in query.lower() or "others" in query.lower():
        gender = query.strip().capitalize()
        stored_data['gender'] = gender
        response = "Enter phone number"
        stored_data.pop('phone_number', None)
    elif query.isdigit():
        pn = query.strip()
        response = "What are your symptoms? Please specify at least three symptoms separated by a comma."
        stored_data['phone_number'] = pn
    elif "," in query.lower():
        response = "Great! We now have all the information we need. If you have medical documents to send type 'upload'"
        stored_data['symptoms'] = query
    elif "upload" in query.lower():
        response = open_file_upload()
        stored_data['file_upload'] = True
    else:
        response = "It's nice to chat with you! Goodbye"
        # Clear stored data in the session
        stored_data.pop('response', None)
        stored_data.pop('first_name', None)
        stored_data.pop('last_name', None)
        stored_data.pop('phone_number', None)
        stored_data.pop('gender', None)
```

Listing 2: Python Function to Store Data

Then to retrieve stored data the below function is used shown in **Listing 3**.

```
stored_data = session.get('stored_data')
```

```
    if stored_data:
        # Access stored first_name and last_name
        first_name = stored_data.get('first_name')
        last_name = stored_data.get('last_name')
        if first_name and last_name:
            print("Stored First Name:", first_name)
            print("Stored Last Name:", last_name)
            print("Stored Data:", stored_data)

    return response
```

Listing 3: Retrive stored data

The function to upload the medical files are stated in the below set of codes shown in **Listing4**. This function is called in line 33 of **Listing2**.

```
def open_file_upload():
    html_code = """
    <form>
        <input type="file" id="filePicker"  accept=".pdf,.png,.jpg,.
    jpeg">
    </form>
    <script>
        // Function to trigger the FilePicker dialog
        function openFilePicker() {
            document.getElementById('filePicker').click();
        }
    </script>
    """
    return html_code
```

Listing 4: File upload function

Finally at the end of storing all user entered data the chat-bot then log them in to the back-end as shown in **Figure 14**. In order to log data to back-end the developer has implemented below set of codes exhibited in **Listing 5**.

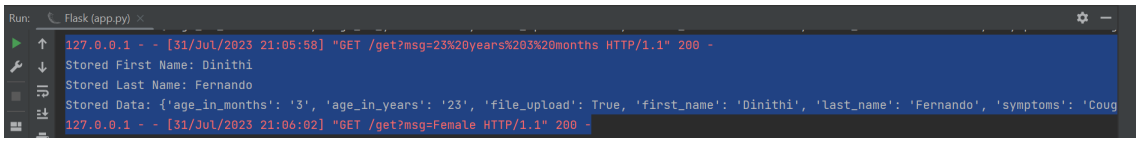

Figure 14: User input

```
stored_data = session.get('stored_data')
    if stored_data:
        # Access stored first_name and last_name
        first_name = stored_data.get('first_name')
        last_name = stored_data.get('last_name')
        if first_name and last_name:
            print("Stored First Name:", first_name)
            print("Stored Last Name:", last_name)
```

```
9             print("Stored Data:", stored_data)
```

Listing 5: Log data to backend

5.3 Front-end Development

The chat interface of the chat-bot is developed using the popular programming language 'HTML' by the developer. The whole chat GUI is developed inside a 'section' tag. **Listing 6** shows the code snippets of the chat GUI.

As soon as a user navigates to chat interface the chat-bot initiates the conversation using the dialog 'My name is TelecarePlus. I will help you book an online appointment, but I will need your details for that. Can you provide me with your details? Type Yes/No'. Here the chat-bot introduces it self and ask for the user permission to get data to book an appointment. See line 8 to 12 in **Listing 6**.

In order to type replies and enter it to the chat for the user, the developer has develop a user input area in a 'div' tag which has an id called 'userInput' (line 14 of the **Listing 6**). Inside the 'div' tag there is a text input field for the user to type text replies. See line 14 of the **Listing 6**. Then a submit button was created by the developer to send user replies (line 15 of the **Listing 6**).

```
1  <section class="msger">
2          <header class="msger-header">
3        <div class="msger-header-title">
4          <i class="fa fa-hospital-o"></i> Telecare Plus Chat
5        </div>
6      </header>
7        <div id="chatbox">
8          <p class="botText"><span>My name is TelecarePlus.
9              <br>I will help you book an online appointment,
10             <br>but I will need your details for that.
11             <br>Can you provide me with your details?
12             <br>Type Yes/No</span></p>
13       </div>
14       <div id="userInput" class="msger-inputarea">
15         <input id="textInput" class="msger-input"  type="text" name=
       "msg" placeholder="Type a message">
16         <input id="buttonInput" class="msger-send-btn" type="submit"
        value="Send">
17         </div>
18 </section>
```

Listing 6: Chat interface

In order to style the the Graphical User Interface (GUI), the developer used Cascading Style Sheets (CSS). The display of text and the chat bubble formatting of the chat dialogues as well as the background of the chat interface is controlled by CSS. **Listing 7** shows some of the codes used to change web display of the chat-bot. The background image of the chat is inserted using CSS (line 6 of **Listing 7**) [21].

```
#chatbox {
    flex: 1;
    overflow-y: auto;
    padding: 10px;
    background-image: url(https://static.vecteezy.com/system/resources/previews/004/696/041/non_2x/medical-outline-seamless-pattern-with-pill-syringe-thermometer-and-stethoscope-illustration-on-green-background-vector.jpg);
    padding: 10px;
}

#userInput {
    margin-left: auto;
    margin-right: auto;
    /*width: 40%;*/
    /*margin-top: 60px;*/
}
#textInput {
    /*width: 87%;*/
    border: none;
    border-bottom: 3px solid #009688;
    font-family: monospace;
    font-size: 17px;
}
#buttonInput {
    padding: 3px;
    font-family: monospace;
    font-size: 17px;
}
.userText {
    color: white;
    font-family: monospace;
    font-size: 14px;
    text-align: right;
    line-height: 30px;
}
.userText span {
    background-color: #009688;
    padding: 15px;
    border-radius: 15px;
    display: initial;
    align-items: flex-end;
    margin-bottom: 10px;
    max-width: 343px;
    border-bottom-right-radius: 0px;
}
.botText {
    color: white;
    font-family: monospace;
    font-size: 14px;
    text-align: left;
    line-height: 30px;
```

```
}
.botText span {
    background-color: #3e79b4;
    padding: 15px;
    border-radius: 15px;
    display: flex;
    align-items: flex-end;
    margin-bottom: 10px;
    border-bottom-left-radius: 0;
    max-width: 343px;
}
```

Listing 7: CSS for the GUI

The design of the text input area and for the send button is also done using CSS. See **Listing 8**. When ever a user hover over the 'Send' button it change color. This effect is done using a pseudo-class selector called ':hover' (lines 27-29 of **Listing 8**).

```
.msger-inputarea {
    display: flex;
    padding: 10px;
    /*border-top: var(--border);*/
    background: #eee;
    width: -webkit-fill-available;
}
.msger-inputarea * {
  padding: 10px;
  border: none;
  border-radius: 3px;
  font-size: 1em;
}
.msger-input {
  flex: 1;
  background: #ddd;
}
.msger-send-btn {
  margin-left: 10px;
  background: rgb(0, 196, 65);
  color: #fff;
  font-weight: bold;
  cursor: pointer;
  transition: background 0.23s;
}
.msger-send-btn:hover {
  background: rgb(0, 180, 50);
}
.scroll{
    background: #000;
    position: absolute;
    height:199px;
    overflow-x: hidden;
    overflow-y: auto;
```

```
}
```

Listing 8: CSS for text input area

JavaScript is used to pass the user entered data from web page and to the back end. The main function is given in **Listing 9**.

```
function getBotResponse() {
          var rawText = $("#textInput").val();
          var userHtml = '<p class="userText"><span>' + rawText + '</span></p>';
          $("#textInput").val("");
          $("#chatbox").append(userHtml);
          document.getElementById('userInput').scrollIntoView({block: 'start', behavior: 'smooth'});
          $.get("/get", { msg: rawText }).done(function(data) {
            var botHtml = '<p class="botText"><span>' + data + '</span></p>';
            $("#chatbox").append(botHtml);
            scrollToBottom();
            document.getElementById('userInput').scrollIntoView({block: 'start', behavior: 'smooth'});
          });
      }
      $("#textInput").keypress(function(e) {
          if(e.which == 13) {
              getBotResponse();
          }
      });
      $("#buttonInput").click(function() {
        getBotResponse();
      })
```

Listing 9: Send data

Also the developer has developed a scrolling function inside chat box where the chat text is visible, so that when ever a new chat text pop up the chat box would automatically scroll to bottom (**Listing 10**). This feature was implemented keeping in mind the user experience. It is this specific feature that allows user not to scroll to the bottom of the chat text each time a new chat text pops up.

The styling of the scroll bar and it's position is done using CSS (lines 31–37 of **Listing 8**).

```
function scrollToBottom() {
          var chatbox = document.getElementById('chatbox');
          chatbox.scrollTop = chatbox.scrollHeight;
      }
```

Listing 10: Scroll to bottom function

Figure 15 shows the chat user interface when a user navigates in to the site for the first time. The chat-bot initiates the conversation by introducing itself to the user and then asks for user consent to get information to book an appointment.

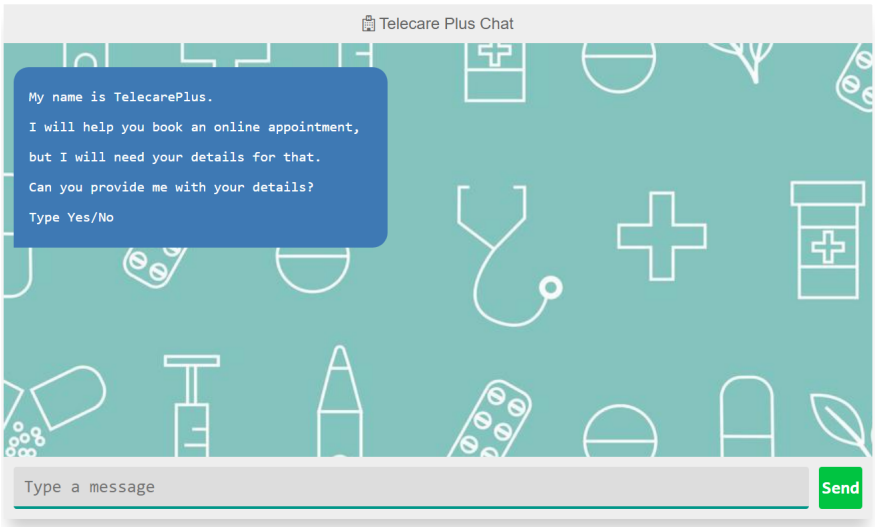

Figure 15: Telecare Plus Chatbot UI

Then the user can type their message in the provided text box and click the 'Send' button to send the reply, see **Figure 16**.

If the user does not allow to share the details then the chat-bot will terminate the conversation, as shown in **Figure 17**.

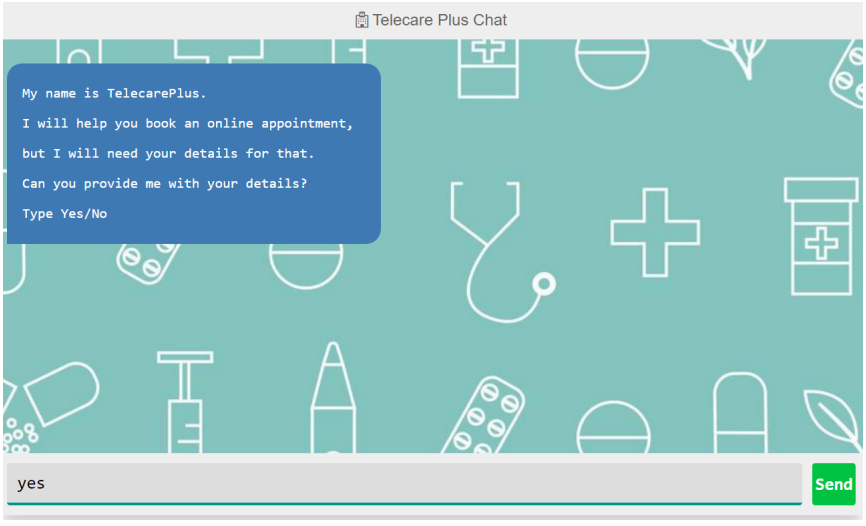

Figure 16: User input

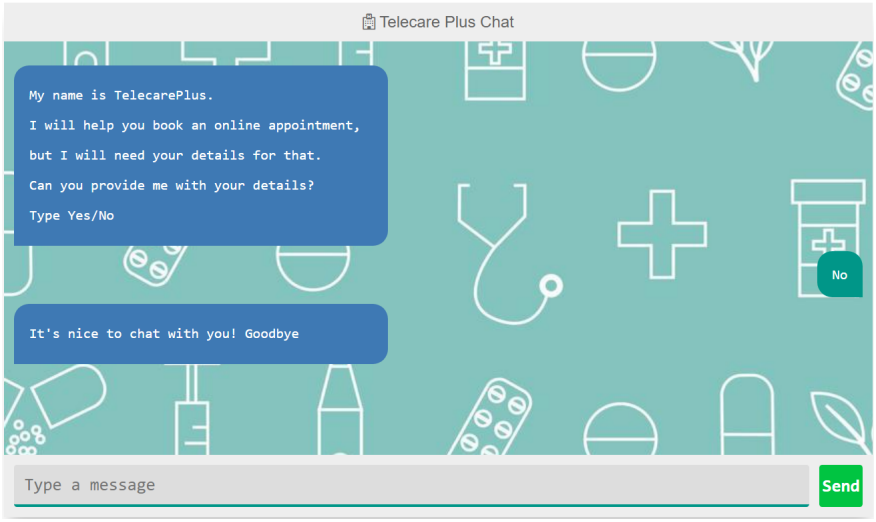

Figure 17: Chat terminate

If only the user allows to share his/her detail then the bot will ask for user name. When the user input the name chat-bot is able to store it in a session and output it within the next chat text, as seen in **Figure 18**.

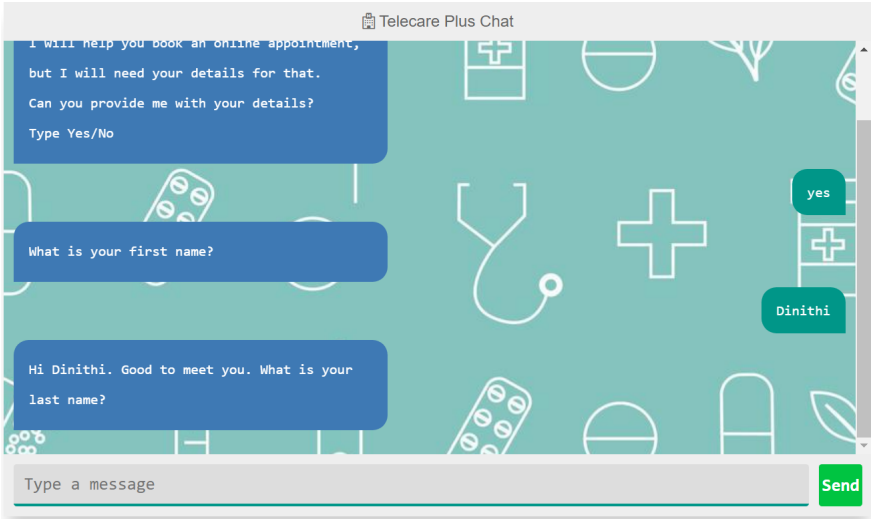

Figure 18: Chat-bot storing user input

After retrieving necessary information the chat-bot will then ask the user to upload any medical documents. Here the user have to type the word 'upload' if she or he has any medical documents as shown in **Figure 19**.

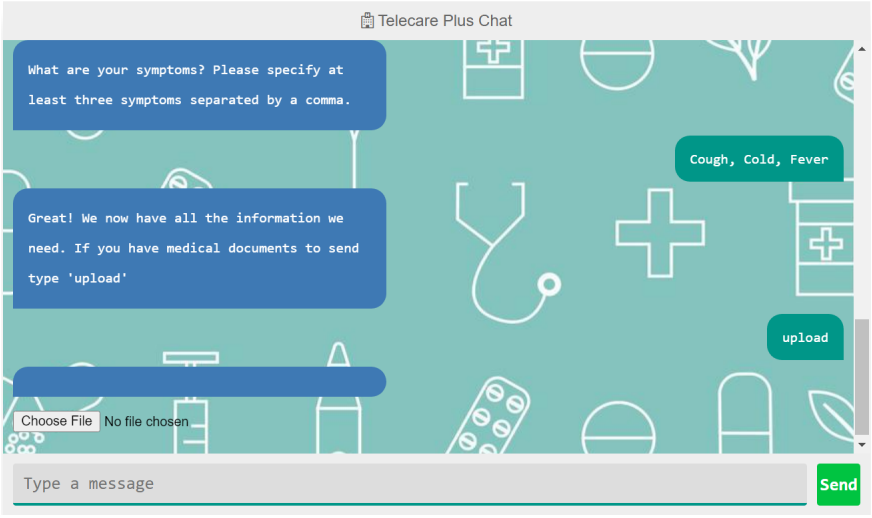

Figure 19: Requesting to upload medical documents

When the user click on 'Choose File' button, it is navigating to a file picker where user can select the documents. **Figure 20** shows this.

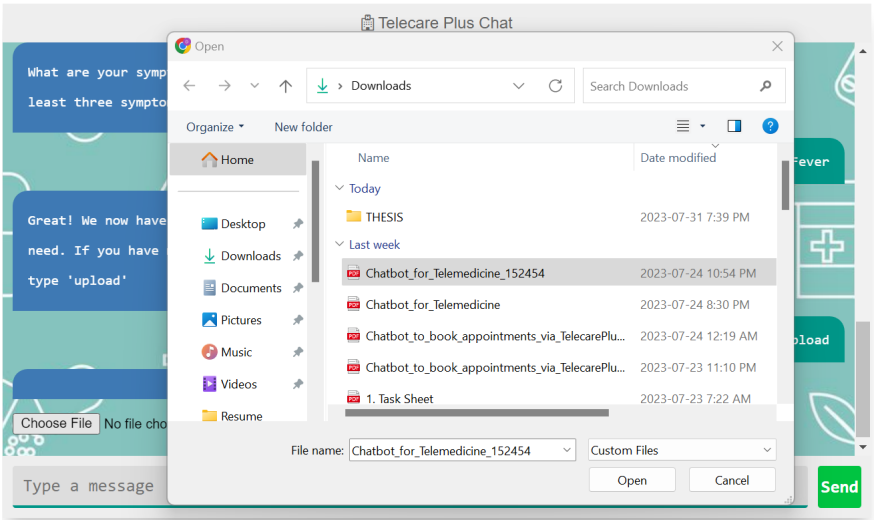

Figure 20: File picker to upload medical documents

Finally the developer has also considered the responsiveness of the web user interface among different devices. **Figure 21** shows an example of a responsive screen of the user interface.

Figure 21: Responsive UI screen

6 Future Work and Recommendation

As this artifact described in this work has proven that it has a high potential to be useful for the healthcare sector. So there are many improvements that could be done to the application to make it better. Some of the ideas are stated below derived out of user's evaluations and the others are personal opinions of the researcher.

User profiles could be created for users who enable storing details. So that if a user already has an account in the website then that detail could be used by the bot to output to the user. And if the user does not need to add more detail then the bot can use the previous account details to book an appointment easily. This functionality should be linked with users phone number or the email address. So the conversational flow should be changed in a way that the user could enter the email address and the phone number at first rather that waiting till the end of the conversation.

When entering the gender a function could be implemented to check the age is below 18 years. If the patient is a child under 18 years then the chat-bot could navigate to ask specific set of questions about the guardian. As an example see **Listing 11**.

```
1
2           if user_age <= 18:
3               guardian_fname = "Please provide the first name of
    guardian."
4               print("ROBO: {}".format(guardian_fname))
5               guardian_fname = input()
6
7               guardian_lname = "Please provide the last name of
    guardian."
8               print("ROBO: {}".format(guardian_lname))
9               guardian_lname = input()
10
11              guardian_full_name = "{} {}".format(guardian_fname,
    guardian_lname)
```

```
                result_dict.update({"name_guardian":
    guardian_full_name})

                # validating guardian pn number
                pattern = r'^\d{10}$'
                while True:
                    print("ROBO: Please provide 10 digit phone
    number of your guardian")
                    guardian_number = input()

                    if re.match(pattern, guardian_number):
                        print("ROBO: Phone number is valid.")
                        result_dict.update({"number_guardian":
    guardian_number})
                        break
                    else:
                        print("ROBO: Invalid phone number.")
```

Listing 11: Ask guardian information if the pation is a child

Also, the verification functionality of the mobile number entered by the user is another important functionality, which the researcher will be developing in the future. The current function checks whether the user input is a number but it does not check the number of digits it contains. Developer hopes to add a validation method similar to the **Listing 12**

```

pattern = r'^\d{10}$'
                while True:
                    print("ROBO: Please provide a 10 digit phone
    number")
                    user_pn = input()

                    if re.match(pattern, user_pn):
                        print("ROBO: Phone number is valid.")
                        result_dict.update({"number_user": user_pn})
                        break
                    else:
                        print("ROBO: Invalid phone number.")
```

Listing 12: Validate phone number

Not only that but also email validation is also by default considered to be implemented by the developer after this. In order to improve the security of the application these two validation methods are essential. See **Listing 13** for a sample email validation used in a Python chat-bot.

The advantage of using both of these validation functions are, it allows the user to insert input until they insert the correct input, without terminating that chat.

```
pattern_mail = r'^[\w\.-]+@[\w\.-]+\.\w+$'

            while True:
```

```
                print("ROBO: Please provide email address:")
                user_email = input("You: ")

                if re.match(pattern_mail, user_email):
                    result_dict.update({"email_user": user_email})
                    break
                else:
                    print("ROBO: Invalid email address.")
```

Listing 13: Validate email address

It is also taken into account to send appointment details to user for their given mobile number as a text message or email the appointment details to the given email address. Where this could be used as a confirmation of the booking.

Enabling the voice input/output function will be considered to be added in the future, so that it expands the ways users can input there information and allows more user friendly control over the application.

7 Conclusions

This report provides a clear idea about the chat-bot application, how it is implemented and a base level insight of the application. In conclusion, chat-bots have demonstrated significant promise for healthcare sector to people of various skill levels. Chat-bots can deliver a customised and dynamic experience thanks to developments based on natural language processing, machine learning, gamification, interaction with IDEs, and voice assistants. Chat-bots can make online appointment booking more approachable by allowing patients to enter required detail in a convenient way, which is similar to texting. The use of chat-bots in the healthcare sector has come a long way already, but there is still potential for growth. As a result, we may anticipate further advancements in the future [13].

References

[1] Bala, Kumkum & Kumar, Mukesh & Hulawale, Sayali & Pandita, Sahil. (2018). Chat-Bot For College Management System Using A.I. 2395-0056.

[2] Chao, C., Huang, Y., & Chen, L. (2018). Integrating a Chatbot with a Programming Learning Platform: A Design-based Research Framework. Educational Technology & Society, 21(3), 32-44, https://doi.org/10.1016/j.caeai.2021.100033

[3] Pal, A., & Ghosh, S. (2019). Chatbots in Education: A Review of Current Research. International Journal of Emerging Technologies in Learning, 14(6), 163-178, https://educationaltechnologyjournal.springeropen.com/articles/10.1186/s41239-021-00302-w

[4] Guo L, Wang D, Gu F, Li Y, Wang Y, Zhou R. Evolution and trends in intelligent tutoring systems research: a multidisciplinary and scientometric view Asia Pacific Education Review. 2021 Jan;22(3) 441-461. PMCID: PMC8095475.

[5] Dimitriadou E, Lanitis A. A critical evaluation, challenges, and future perspectives of using artificial intelligence and emerging technologies in smart classrooms. Smart Learn. Environ. 2023;10(1):12. doi: 10.1186/s40561-023-00231-3. Epub 2023 Feb 6. PMCID: PMC9900563.

[6] Health care financing review - Effects and effectiveness of telemedicine, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4193577/, note = Accessed: 2023-05-22

[7] Approach to Reviewing Telehealth, https://www.medicaid.gov/medicaid/benefits/telehealth/index.html, note = Accessed: 2023-06-01

[8] ChatGPT - How has the popularity of telemedicine increased over the years?, https://chat.openai.com/chat, note = Accessed: 2023-05-28

[9] C. Wibhowo and R. Sanjaya, "E-Learning Design for Psychologists to Implement Chatbots for Clients with Borderline Personality Disorder," 2022 12th International Conference on Information Technology in Medicine and Education (ITME), Xiamen, China, 2022, pp. 189-192, doi: 10.1109/ITME56794.2022.00049.

[10] A. M. Rahman, A. A. Mamun and A. Islam, "Programming challenges of chatbot: Current and future prospective," 2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC), Dhaka, Bangladesh, 2017, pp. 1-4, doi: 10.1109/R10-HTC.2017.8288910.

[11] ARTIFICIAL INTELLIGENCE MARKUP LANGUAGE: A BRIEF TUTORIAL, https://arxiv.org/ftp/arxiv/papers/1307/1307.3091.pdf, (Accessed: 2023-05-24)

[12] B. R. Ranoliya, N. Raghuwanshi and S. Singh, "Chatbot for university related FAQs," 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, India, 2017, pp. 1525-1530, doi: 10.1109/ICACCI.2017.8126057.

[13] N. P. K S, S. S, T. T N, Y. Yuvraaj and V. D A, "Conversational Chatbot Builder – Smarter Virtual Assistance with Domain Specific AI," 2023 4th International Conference for Emerging Technology (INCET), Belgaum, India, 2023, pp. 1-4, doi: 10.1109/INCET57972.2023.10170114.

[14] What is Flask Python, https://pythonbasics.org/what-is-flask-python, (Accessed: 2023-06-23)

[15] M. Salehi and I. N. Kmalabadi, "Attribute-based recommender system for learning resource by learner preference tree," 2012 2nd International eConference on Computer and Knowledge Engineering (ICCKE), Mashhad, Iran, 2012, pp. 133-138, doi: 10.1109/ICCKE.2012.6395366.

[16] About ChatterBot, https://chatterbot.readthedocs.io/en/stable, (Accessed: 2023-05-05)

[17] The Systems Development Life Cycle Essay, https://ivypanda.com/essays/the-systems-development-life-cycle/, (Accessed: 2023-07-05)

[18] Esko Kilpi, Neural networks as the architecture of human work, from Michael Nielsen's book on neural nets, https://medium.com/@EskoKilpi/neural-networks-as-the-architecture-of-human-work-3f9d20f019a3, (Accessed: 2023-06-04)

[19] Sumit Saha, A Comprehensive Guide to Convolutional Neural Networks - the ELI5 way, Published in Towards Data Science (https://towardsdatascience.com/), 2018. (Accessed: 2023-06-10)

[20] Examples: Mixture Modeling with Cross-Sectional Data, Chapter 7, https://www.statmodel.com/HTML_UG/chapter7V8.htm, (Accessed: 2023-06-30)

[21] Chat GUI background image, Medical outline seamless pattern with pill syringe thermometer and stethoscope vector illustration on green background, from https://www.vecteezy.com/vector-art/, (Accessed: 2023-06-20)