VOICE HEALTH CHATBOT

A Project Report

In partial fulfilment of the requirements for the award of the degree BACHELOR OF TECHNOLOGY IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Under the guidance of Mr. DIPON MONDAL

Academy of Skill Development

ASD
Skills For Employment

Submitted by

ANNAPANENI AKSHAYA

INSPIRED BY LIFE
MANIPAL
ACADEMY of HIGHER EDUCATION
(Deemed to be University under Section 3 of the UGC Act, 1956)

MANIPAL INSTITUTE OF TECHNOLOGY, MANIPAL

Certificate from the Mentor

This is to certify that **ANNAPANENI AKSHAYA** has completed the project titled **VOICE HEALTH CHATBOT** under my supervision during the period from **MAY 2024** to **JUNE 2024** which is in partial fulfillment of requirements for the award of the **B. Tech** and submitted to Department **ECE** of **MANIPAL INSTITUTE OF TECHNOLOGY, MANIPAL**

_____

Signature of the Mentor

Date:

Acknowledgment

I take this opportunity to express my deep gratitude and sincerest thanks to my project mentor, **DIPON MONDAL** for giving the most valuable suggestions, helpful guidance, and encouragement in the execution of this project work.

I would like to give a special mention to my colleagues. Finally I am grateful to all the faculty members of the **Academy of Skill Development** for their support.

DECLARATION

I hereby declare that the project work presented in the project proposal entitled “Voice Health Chatbot” partially fulfills the requirements for the award of the Bachelor of Technology in Electronics and Communication Engineering at Academy of Skill Development, Kolkata, West Bengal, is an authentic work carried out under the guidance of Mr. Dipon Mondal. The matter embodied in this project work has not been submitted elsewhere for the award of any degree of our knowledge and belief.

Date:17th June 2024
Name of the Student: Annapaneni Akshaya

CONTENTS

OVERVIEW

Python is a high-level, general-purpose, and prevalent programming language. Python programming language (latest Python 3) is used in web development, Machine Learning applications, and all cutting-edge technology in the Software Industry. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc. Python supports modules and packages, which encourages program modularity and code reuse. The Python Interpreter and the Extensive Standard Library are available in source or binary form without charge for all major platforms and can be freely distributed. Python programming language (The latest version of Python 3) is used in web development, machine learning applications, and all cutting-edge technology in the software industry.

HISTORY OF PYTHON

In the late 1980s, history was about to be written. It was at that time that I started working on Python. Soon after, Guido Van Rossum began its application-based work in December of 1989 at Centrum Wiskunde & Informatica (CWI) in the Netherlands. It was started as a hobby project because he was looking for an exciting project to keep him occupied during Christmas. The programming language in which Python is said to have succeeded is ABC Programming Language, which interfaced with the Amoeba Operating System and had the exception handling feature. He had already helped create ABC earlier in his career and had seen some issues with ABC but liked most of the features. After that, what he did was very clever. He had taken the syntax of ABC and some of its good features. It came with many complaints, too, so he fixed those issues entirely and created a good scripting language that had removed all the flaws.

FEATURES OF PYTHON

1. Easy to Learn and Use:
   - Python's syntax is designed to be intuitive and readable, making it accessible for beginners. This simplicity reduces the learning curve and allows programmers to focus more on problem-solving rather than on the language itself.

2. Interpreted Language:
   - Python is executed line by line by its interpreter at runtime, rather than being compiled into machine code beforehand. This feature allows for easier debugging and more dynamic programming experiences.

3. High-Level Language:
   - Python abstracts many low-level details such as memory management and hardware interactions, providing a more user-friendly programming environment. This abstraction simplifies development and speeds up the coding process.

4. Object-Oriented Programming:
   - Python supports object-oriented programming paradigms, facilitating the creation of reusable and modular code through classes and objects. This approach enhances code organization, readability, and maintainability.

5. Dynamic Typing:
   - Python uses dynamic typing, meaning variables do not need explicit declaration to reserve memory space. This flexibility allows for more rapid development cycles and easier refactoring of code.

6. Extensive Standard Library:
- Python's standard library is comprehensive, offering modules and packages for a wide range of tasks such as file handling, networking, and data manipulation. This feature reduces the need for external dependencies and enhances productivity.

7. Cross-Platform Compatibility:
- Python code can run seamlessly on various platforms including Windows, macOS, and Linux, without requiring modification. This portability makes Python suitable for developing applications that need to operate across different operating systems.

8. Integrated Development Environment (IDE) Support:
- Python is supported by robust integrated development environments (IDEs) such as PyCharm, Visual Studio Code, and Jupyter Notebook. These tools provide features like debugging, syntax highlighting, and code completion, enhancing developer productivity.

9. Scalability:
- Python scales effectively from simple scripts to large-scale applications and enterprise-level systems. Its versatility allows developers to start small and expand functionalities as needed, accommodating diverse project requirements.

10. Strong Community Support:
- Python boasts a vibrant and active community of developers, enthusiasts, and contributors. This community-driven support results in extensive documentation, tutorials, forums, and third-party libraries, fostering collaborative learning and problem-solving.

11. Embeddable:
- Python can be embedded within C/C++ programs through its comprehensive API. This feature enables developers to combine Python's high-level functionalities with the performance benefits of lower-level languages, enhancing application flexibility and performance.

12. Support for Multiple Programming Paradigms:
- In addition to object-oriented programming (OOP), Python supports procedural and functional programming paradigms. This versatility allows developers to choose the best approach for each specific task, improving code structure and maintainability.

13. Portability:
- Python's platform independence and compatibility contribute to its portability. Developers can write code once and deploy it across multiple platforms, reducing development time and ensuring consistent performance across different environments.

14. Readable and Maintainable Code:
- Python emphasizes code readability with its clean and easily understandable syntax. This readability simplifies collaboration among developers, reduces the likelihood of errors, and facilitates code maintenance and updates over time.

15. Extensible:
- Python's extensibility enables integration with other languages and systems. Developers can incorporate modules written in languages like C/C++ to optimize performance-critical sections of code, ensuring efficient execution without sacrificing Python's ease of use.

PYTHON IDENTIFIERS

An identifier is a user-defined name given to a variable, function, class, module, etc. The identifier is a combination of character digits and an underscore. They are case-sensitive, i.e., ‘num’ and ‘Num’ and ‘NUM’ are three different identifiers in Python. It is a good programming practice to give meaningful names to identifiers to make the code understandable.

Rules for Naming Python Identifiers:

- It cannot be a reserved Python keyword.
- It should not contain white space.
- It can be a combination of A-Z, a-z, 0-9, or underscore.
- It should start with an alphabet character or an underscore.
- It should not contain any unique character other than an underscore.

PYTHON KEYWORDS

Python keywords are reserved words that have special meaning in the language. These keywords are used to define the syntax and structure of Python. Here is a list of Python keywords along with a brief description of each:

1. False: Represents the boolean value false.
2. None: Represents the absence of a value or a null value.
3. True: Represents the boolean value true.
4. and: Logical AND operator.
5. as: Used to create an alias when importing a module.
6. assert: Used for debugging purposes, to test if a condition in your code returns True, if not, the program will raise an Assertion Error.
7. async: Used to declare an asynchronous function, which can run independently of the main program flow.
8. await: Used to pause the execution of an async function until the result is ready.
9. break: Terminates the nearest enclosing loop.
10. class: Used to define a class.
11.continue: Skips the rest of the code inside a loop for the current iteration only, and goes back to the top of the loop.
12.def: Used to define a function.
13.del: Deletes objects.
14.elif: Short for else if, used in conditional statements.
15.else: Used in conditional statements, to include a block of code to be executed if the condition is false.
16.except: Used with exceptions, what to do when an exception occurs.
17.finally: Used with exceptions, a block of code that will be executed no matter if there is an exception or not.
18.for: Used to create a for loop.
19.from: Used to import specific parts of a module.

20. global: Declares a variable global.
21. if: Used to make a conditional statement.
22. import: Used to import a module.
23. in: Used to check if a value is present in a list, tuple, etc., or used in for loops.
24.is: Tests for object identity.
25.lambda: Used to create an anonymous function.
26.nonlocal: Declares a variable as non-local.
27.not: Logical NOT operator.
28.or: Logical OR operator.
29.pass: A null statement, a placeholder in loops, functions, classes, or conditionals.
30.raise: Used to raise an exception.
31.return: Exits a function and returns a value.
32.try: Specifies exception handlers.
33.while: Creates a while loop.
34.with: Used to simplify exception handling.
35. yield: Ends a function and returns a generator.

INDENTATION

Python indentation tells a Python interpreter that the statements belong to a particular code block. A block is a combination of all these statements. Block can be regarded as the grouping of statements for a specific purpose. Most programming languages like C, C++, and Java use braces {} to define a code block. Python uses indentation to highlight the blocks of code. Whitespace is used for indentation in Python. All statements with the same distance to the right belong to the same block of code. If a block has to be more deeply nested, it is simply indented further to the right. You can understand it better by looking at the following lines of code.

Example of Python Indentation:

- Statement (line 1), if condition (line 2), and statement (last line) belong to the same block, which means that after statement 1, if condition will be executed. If the condition becomes false, Python will jump to the last statement for execution.
- The nested if-else belongs to block 2, which means that if nested if becomes False, then Python will execute the statements inside the else condition.
- Statements inside nested if-else belong to block 3, and only one statement will be executed depending on the if-else condition.

PYTHON DATA TYPES

Python Data types are the classification or categorization of data items. It represents the value of what operations can be performed on a particular data. Since everything is an object in Python programming, Python data types are classes, and variables are instances (objects) of these classes.

The following are the standard or built-in data types in Python:

- Numeric
- Sequence Type
- Boolean
- Set
- Dictionary
- Binary Types(memory view, byte array, bytes)

DATA TYPE CONVERSION

Data type conversion is the process of converting data from one type to another. This can be necessary when different data types need to be combined or when data needs to be processed in a specific way. In Python, data type conversion can be categorized into two types: implicit type conversion and explicit type conversion.

- Implicit Type Conversion

Implicit type conversion, also known as automatic type conversion, is performed by Python automatically. This type of conversion occurs when Python converts one data type to another without any user involvement. It usually happens when performing operations between different data types, where Python converts smaller data types to larger data types to prevent data loss. For example, when adding an integer and a float, Python will automatically convert the integer to a float before performing the addition.

- Explicit Type Conversion

Explicit type conversion, also known as type casting, requires the programmer to manually convert one data type to another. This is done using predefined functions that convert the data type explicitly. The most used functions for explicit type conversion in Python include:

- int (): Converts a value to an integer type.
- float (): Converts a value to a float type.
- str (): Converts a value to a string type.
- list (): Converts a value to a list type.
- tuple (): Converts a value to a tuple type.
- set (): Converts a value to a set type.
- dict (): Converts a value to a dictionary type (when applicable).

Explicit type conversion is necessary when specific operations require data to be in a particular format, such as when concatenating strings with non-string values or when performing mathematical operations on string representations of numbers.

PYTHON FUNCTION

Functions in Python are reusable blocks of code that perform a specific task. They help in breaking down complex problems into smaller, manageable pieces, promoting code reuse and modularity. Here are the key theoretical aspects of Python functions:

Function Definition
- A function is defined using the `def` keyword, followed by the function name and parentheses `()`.
- Inside the parentheses, parameters can be defined (optional).
- The function body contains the code to be executed and is indented.

Function Parameters and Arguments
- Parameters: Variables listed inside the parentheses in the function definition.
- Arguments: Values passed to the function when it is called.
- Parameters can have default values, making them optional when calling the function.

Function Call
- A function is called by using its name followed by parentheses.
- If the function requires parameters, arguments are passed inside the parentheses.

Return Statement
- The `return` statement is used to exit a function and return a value to the caller.
- If no return statement is present, the function returns `None` by default.

Types of Functions
- Built-in Functions: Pre-defined functions in Python (e.g., `print ()`, `Len()`, `range()`).
- User-defined Functions: Functions created by the user to perform specific tasks.

Scope and Lifetime of Variables
- Variables defined inside a function have local scope, meaning they are accessible only within that function.
- The lifetime of these variables is limited to the function execution.

Recursive Functions
- A function can call itself, which is known as recursion.
- Recursive functions can be used to solve problems that can be broken down into smaller, repetitive tasks.

Lambda Functions
- Lambda functions, also known as anonymous functions, are defined using the `lambda` keyword.
- They can have any number of parameters but only one expression.
- Used for creating small, one-time, and inline functions.

Benefits of Using Functions
1.Modularity: Functions allow you to break down a complex problem into smaller, manageable parts.
2.Reusability: Functions can be reused across different parts of a program or in different programs.
3.Maintainability: Functions make it easier to manage and update code.
4.Readability: Well-defined functions with meaningful names enhance the readability of the code.

PYTHON MODULES

A module allows us to organize our Python code logically. Grouping related code into a module makes the code easier to understand and use. A module is a Python object with arbitrarily named attributes you can bind and reference. The Python code for a module named ‘name’ normally resides in a file named ‘aname.py.’ Here's an example of a simple module, support.py def printfunc(par): print("Hello: ", par) The Import Statement You can use any Python source file as a module by executing an import statement in another Python source file. The import has the following syntax – import module1[, module2[,… module n].

GRAPHICAL USER INTERFACE

The standard and valuable component of electronics such as computers, tablets, and smartphones is a visual interface called a Graphical User Interface (GUI). The graphical user interface (GUI) shows valuable items that indicate actions the user can perform and transmit information as per requirement. When interacting with the objects, the user can modify or simplify their size, color, and visibility. As the text command-line interfaces in the system were complex and challenging to master, GUIs were developed. It is currently the accepted practice in software applications or system programming for required user-centered design.

COMPONENTS OF GUI:

● Pointers: The pointer appears on the user's screen as a marking symbol. The pointer moves on to choose instructions and objects as per requirement.

● Icons: Icons allude to tiny visual representations of windows, documents, actions, and other things on the display screen to simplify. A pointer and pointing device can be used by the user to carry out the initial tasks for the overall processes.

● Pointing tool: At the initial stages, the pointing tool enables the user to select and move the required pointer items on the screen, including a trackball or mouse. It is the most beneficial tool in GUI.

● Desktop: The desktop is the screen contained within the icons and user beneficial.

INTRODUCTION TO TKINTER

Python offers multiple options for developing GUI (Graphical User Interface). Tkinter is the most commonly used method of all the GUI methods. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python Tkinter is the fastest and easiest way to create GUI applications. Creating a GUI using Tkinter is an easy task.

To create a Tkinter Python app, you follow these basic steps:

● Import the tkinter module: This is done just like importing any other module in Python. Note that in Python 2. x, the module is named 'Tkinter,' while in Python 3. x, it is named 'tkinter'.

● Create the main window (container): The main window serves as the container for all the GUI elements you'll add later.

● Add widgets to the main window: You can add any number of widgets, like buttons, labels, entry fields, etc., to the main window to design the interface as desired.

● Apply event triggers to the widgets: You can attach them to the widgets to define how they respond to user interactions.

TKINTER WIDGETS

A Tkinter user interface is made up of individual widgets. Each widget is represented as a Python object, instantiated from classes; there are various pre-defined widgets available in Tkinter:

● Label: Used to contain text or images.

● Buttons: Similar to a Label but provides additional functionality for mouse hovers, presses, and releases, as well as keyboard activity/events.

● Canvas: Provides the ability to draw shapes (lines, ovals, polygons, rectangles); can contain images or bitmaps.

● Frame: Pure container for other widgets.

● Entry: Single-line text field with which to collect keyboard input. and many more. We have used many such widgets in our project, including those mentioned above.

WIDGETS USED IN THE PROJECT

1. tk.Tk(): Creates the main application window.

2. tk.Text: Multi-line text area widget for displaying and editing text.

3. tk.Scrollbar: Widget that allows scrolling through another widget's content.

4. tk.Entry: Single-line text entry widget for user input.

5. tk.Button: Button widget that triggers an action when clicked.

CODES USED IN THE PROJECT

```python
import tkinter as tk
import webbrowser
import random
from tkinter import Canvas
import speech_recognition as sr

# Dictionary containing predefined responses for different user inputs
responses = {
    "hi": ["Hello!", "Hi there!", "Hey!"],
    "how are you": ["I'm doing well, thank you!", "Great, thanks for asking!"],
    "bye": ["Goodbye!", "See you later!", "Bye!"],
    "symptom checker": "Please input your symptoms separated by commas.",
    "book appointment": "Click here to book an appointment!",
    "disease information": "Please enter the disease name.",

}
# Database of symptoms, diseases, and medication information (for demonstration)

symptom_db = {
    "fever": {
        "description": "An elevated body temperature.",
        "related_diseases": [
            "common cold",
            "influenza",
            "COVID-19",
            "typhoid fever",
            "malaria",
        ],
        "medication": "Take rest and drink plenty of fluids.",
    },
    "cough": {
        "description": "Expelling air from the lungs with a sudden sharp sound.",
        "related_diseases": [
            "common cold",
            "influenza",
            "pneumonia",
            "bronchitis",
            "asthma",
            "COPD",
        ],
        "medication": "Use cough syrup and warm liquids.",
    },
    "headache": {
        "description": "A continuous pain in the head.",
        "related_diseases": [
            "migraine",
            "tension headache",
            "sinusitis",
            "cluster headache",
        ],
        "medication": "Take pain relievers and get enough sleep.",
    },
    "fatigue": {
        "description": "Feeling of extreme tiredness or lack of energy.",
        "related_diseases": [
            "anemia",
            "chronic fatigue syndrome",
            "fibromyalgia",
            "depression",
        ],
        "medication": "Rest, balanced diet, regular exercise.",
    },
    "nausea": {
        "description": "Feeling of discomfort in the stomach with an inclination to vomit.",
        "related_diseases": [
```

```
            "food poisoning",
            "viral gastroenteritis",
            "migraine",
            "pregnancy",
        ],
        "medication": "Stay hydrated, eat bland foods, medications if severe.",
    },
    "shortness of breath": {
        "description": "Difficulty in breathing; feeling of suffocation or tightness in the chest.",
        "related_diseases": [
            "asthma",
            "pneumonia",
            "COPD",
            "anxiety disorders",
            "heart failure",
        ],
        "medication": "Bronchodilators, inhalers, oxygen therapy as per cause.",
    },
    "abdominal pain": {
        "description": "Pain or discomfort felt in the area between the chest and pelvis.",
        "related_diseases": [
            "appendicitis",
            "gastroenteritis",
            "ulcers",
            "gallstones",
            "kidney stones",
        ],
        "medication": "Pain relievers, antibiotics if bacterial infection, surgery in severe cases.",
    },
    "joint pain": {
        "description": "Discomfort, aches, or soreness in any part of the body where two or more bones meet.",
        "related_diseases": ["arthritis", "fibromyalgia", "gout", "Lyme disease"],
        "medication": "Pain relievers, anti-inflammatory drugs, physical therapy.",
    },
    "dizziness": {
        "description": "A sensation of lightheadedness, unsteadiness, or faintness.",
        "related_diseases": ["vertigo", "dehydration", "anemia", "inner ear problems"],
        "medication": "Rest, hydration, medications as per the cause.",
    },
    "rash": {
        "description": "Change in the skin's appearance, often characterized by redness, itching, or irritation.",
        "related_diseases": [
            "allergies",
            "eczema",
            "psoriasis",
            "measles",
            "chickenpox",
        ],
        "medication": "Topical creams, antihistamines, steroids as per the cause.",
    },
    "sore throat": {
        "description": "Pain, scratchiness, or irritation of the throat often worsened by swallowing.",
        "related_diseases": ["common cold", "flu", "strep throat", "mononucleosis"],
        "medication": "Rest, hydration, lozenges, pain relievers.",
    },
    "vomiting": {
        "description": "Forcefully expelling the stomach's contents through the mouth.",
        "related_diseases": [
            "gastroenteritis",
            "food poisoning",
            "pregnancy",
            "viral infections",
        ],
        "medication": "Stay hydrated, eat bland foods, medications if severe.",
    },
    "diarrhea": {
```

```
        "description": "Frequent and watery bowel movements.",
        "related_diseases": [
            "food poisoning",
            "viral gastroenteritis",
            "bacterial infections",
            "IBS",
        ],
        "medication": "Stay hydrated, eat bland foods, medications if severe.",
    },
    "chest pain": {
        "description": "Pain or discomfort felt anywhere along the front of the body between the neck and upper abdomen.",
        "related_diseases": [
            "heart attack",
            "angina",
            "pneumonia",
            "GERD",
            "panic attack",
        ],
        "medication": "Depends on the cause; immediate medical attention for suspected heart issues.",
    },
    "back pain": {
        "description": "Pain felt in the back, which may originate from muscles, nerves, bones, or other structures in the spine.",
        "related_diseases": [
            "muscle strain",
            "herniated disc",
            "spinal stenosis",
            "kidney stones",
        ],
        "medication": "Rest, pain relievers, hot or cold therapy, physical therapy.",
    },
    "sweating": {
        "description": "Increased perspiration often due to heat, exercise, or emotional stress.",
        "related_diseases": [
            "hyperhidrosis",
            "menopause",
            "anxiety disorders",
            "infections",
        ],
        "medication": "Depends on the underlying cause; may involve antiperspirants or medical treatment.",
    },
    "swelling": {
        "description": "Enlargement or puffiness in a body part due to fluid retention or inflammation.",
        "related_diseases": ["injuries", "edema", "infections", "allergic reactions"],
        "medication": "Depends on the cause; may involve rest, elevation, compression, or medications.",
    },
    "muscle weakness": {
        "description": "Reduced strength or inability to exert force with muscles.",
        "related_diseases": [
            "muscle diseases",
            "neuromuscular disorders",
            "thyroid disorders",
        ],
        "medication": "Physical therapy, medication as per underlying condition.",
    },
    "vision problems": {
        "description": "Difficulties with sight, including blurriness, double vision, or vision loss.",
        "related_diseases": ["myopia", "cataracts", "glaucoma", "macular degeneration"],
        "medication": "Corrective lenses, surgery, medication as per eye condition.",
    },
    "irregular heartbeat": {
        "description": "Abnormal heart rhythm, palpitations, or sensations of skipped or extra heartbeats.",
        "related_diseases": [
            "arrhythmia",
            "heart disease",
            "anxiety",
            "thyroid disorders",
```

```
        ],
        "medication": "Depends on the cause; may involve medications or procedures.",
    },
    "swollen glands": {
        "description": "Enlargement or tenderness in the lymph nodes, often in the neck, armpits, or groin.",
        "related_diseases": ["infections", "cancers", "immune system disorders"],
        "medication": "Treat the underlying cause; may involve antibiotics or other medications.",
    },

}


# Expanded disease database
disease_db = {
    "common cold": {
        "description": "A viral infection affecting the nose and throat.",
        "medication": "Rest, hydration, over-the-counter cold medications.",
    },
    "influenza": {
        "description": "A highly contagious viral infection affecting the respiratory system.",
        "medication": "Rest, hydration, antiviral medications.",
    },
    "COVID-19": {
        "description": "A contagious viral infection caused by SARS-CoV-2.",
        "medication": "Isolation, medical care, vaccination.",
    },
    "typhoid fever": {
        "description": "A bacterial infection causing high fever, diarrhea, and abdominal pain.",
        "medication": "Antibiotics, hydration, rest.",
    },
    "malaria": {
        "description": "A mosquito-borne infectious disease causing fever, chills, and flu-like symptoms.",
        "medication": "Antimalarial drugs, prevention of mosquito bites.",
    },
    "pneumonia": {
        "description": "Infection that inflames air sacs in one or both lungs, which may fill with fluid.",
        "medication": "Antibiotics, oxygen therapy, rest.",
    },
    "bronchitis": {
        "description": "Inflammation of the lining of the bronchial tubes.",
        "medication": "Rest, hydration, cough medicine.",
    },
    "asthma": {
        "description": "A condition in which a person airways become inflamed, narrow, and swell, and produce extramucus.",
        "medication": "Inhalers, corticosteroids, long-term control medications.",
    },
    "COPD": {
        "description": "A chronic inflammatory lung disease that causes obstructed airflow.",
        "medication": "Bronchodilators, corticosteroids, oxygen therapy.",
    },
    "migraine": {
        "description": "A type of headache characterized by severe throbbing pain, often accompanied by nausea and sensitivity to light and
        "medication": "Pain relievers, rest in a quiet, dark room.",
    },
    "tension headache": {
        "description": "A common type of headache often related to stress or muscle tension.",
        "medication": "Pain relievers, relaxation techniques.",
    },
    "sinusitis": {
        "description": "Inflammation or swelling of the tissue lining the sinuses.",
        "medication": "Nasal decongestants, antibiotics (if bacterial), saline nasal spray.",
    },
    "cluster headache": {
        "description": "Severe headaches that occur in clusters, often on one side of the head.",
        "medication": "Oxygen therapy, sumatriptan injections, preventive medications.",
    },
```

```
    "anemia": {
        "description": "A condition in which there is a deficiency of red cells or of hemoglobin in the blood.",
        "medication": "Iron supplements, vitamin supplements, blood transfusions.",
    },
    "chronic fatigue syndrome": {
        "description": "A disorder characterized by extreme fatigue that does not improve with rest.",
        "medication": "Symptom-based treatment, lifestyle changes, therapy.",
    },
    "fibromyalgia": {
        "description": "A disorder characterized by widespread musculoskeletal pain, fatigue, and tenderness in localized areas.",
        "medication": "Pain relievers, antidepressants, physical therapy.",
    },
    "food poisoning": {
        "description": "Illness caused by consuming contaminated food or drink.",
        "medication": "Hydration, rest, anti-nausea medication if needed.",
    },
    "viral gastroenteritis": {
        "description": "Inflammation of the stomach and intestines caused by a virus.",
        "medication": "Hydration, rest, anti-diarrheal medication.",
    },
    "pregnancy": {
        "description": "State of carrying a developing embryo or fetus within the female body.",
        "medication": "Prenatal vitamins, regular check-ups.",
    },

}
# Function to open the appointment booking website
def open_appointment_website():
    # Replace this URL with the actual URL of the appointment booking website
    booking_url = "https://docpulse.com/products/online-doctor-appointment-app/"
    webbrowser.open_new(booking_url)
# Function to simulate symptom checking
def send_text_message(user_message):
def send_text_message(user_message):
    chat_log.config(state=tk.NORMAL)
    chat_log.insert(tk.END, "You: " + user_message + "\n\n")

    if user_message == "exit":
        root.destroy()
    elif user_message in responses:
        chat_log.insert(
            tk.END, "ChatBot: " + random.choice(responses[user_message]) + "\n\n"
        )
    elif user_message.startswith("disease information"):
        disease_name = user_message.replace("disease information ", "")
        if disease_name in disease_db:
            response = f"Details for {disease_name}:\n"
            response += f"Description: {disease_db[disease_name]['description']}\n"
            response += f"Medication: {disease_db[disease_name]['medication']}\n\n"
        else:
            response = f"No information available for {disease_name}\n\n"
        chat_log.insert(tk.END, "ChatBot: " + response)
    elif "symptom checker" in user_message:
        chat_log.insert(tk.END, "ChatBot: " + responses["symptom checker"] + "\n\n")
        user_symptoms = user_message.replace("symptom checker ", "")
        symptoms_list = [symptom.strip() for symptom in user_symptoms.split(",")]
        response = check_symptoms(symptoms_list)
        chat_log.insert(tk.END, "ChatBot: " + response + "\n\n")
    elif user_message == "book appointment":
        chat_log.insert(tk.END, "ChatBot: " + responses[user_message] + "\n\n")
        open_appointment_website()
    else:
        chat_log.insert(
            tk.END,
            "ChatBot: I'm sorry, I didn't understand that. Please try again.\n\n",
```

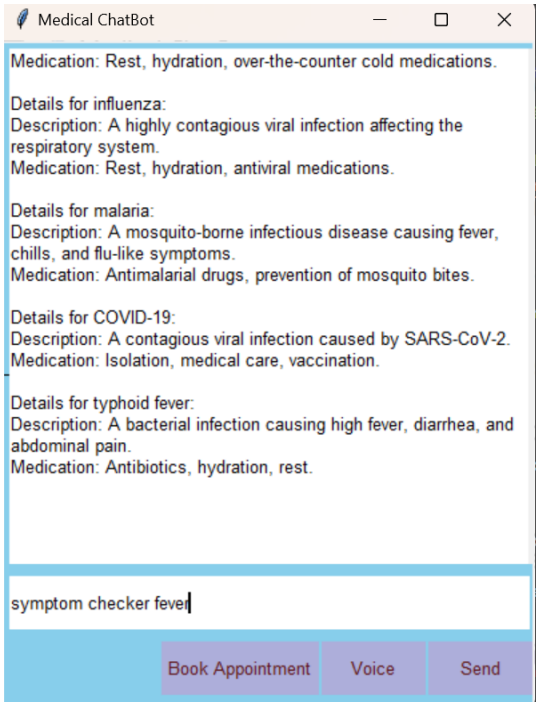
```
        )

    chat_log.config(state=tk.DISABLED)
    chat_log.yview(tk.END)

# Function to simulate symptom checking
def check_symptoms(symptoms):
    matched_diseases = []
    medication_suggestions = []

    for symptom in symptoms:
        if symptom in symptom_db:
            matched_diseases.extend(symptom_db[symptom]["related_diseases"])
            medication_suggestions.append(symptom_db[symptom]["medication"])

    if matched_diseases:
        all_diseases = list(set(matched_diseases))
        response = ""

        for disease in all_diseases:
            if disease in disease_db:
                response += f"Details for {disease}:\n"
                response += f"Description: {disease_db[disease]['description']}\n"
                response += f"Medication: {disease_db[disease]['medication']}\n\n"
            else:
                response += f"No information available for {disease}\n\n"

        return response
    else:
        return "Your symptoms do not match any specific condition. Please consult a healthcare professional."


```

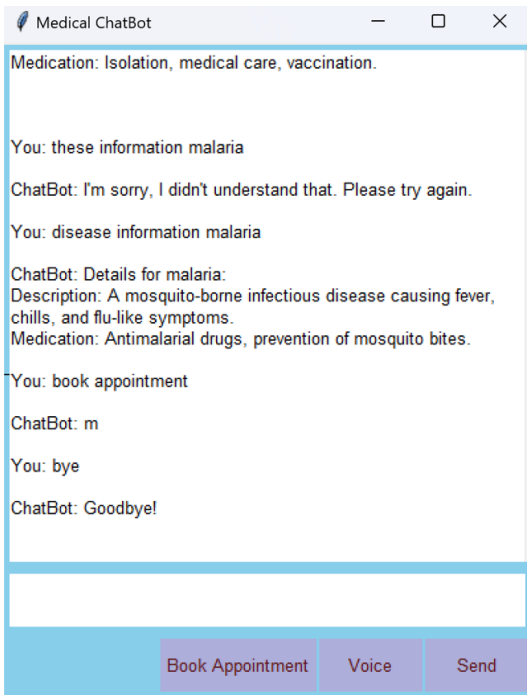
```
# Function to handle voice inputs
def send_voice_message():
    recognizer = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening... Speak something.")
        audio = recognizer.listen(source)

    try:
        user_message_voice = recognizer.recognize_google(audio).lower()
        print(f"You said: {user_message_voice}")
        send_text_message(user_message_voice)  # Pass the recognized voice input to the text message handler
    except sr.UnknownValueError:
        print("Sorry, I could not understand the audio.")
    except sr.RequestError:
        print("Could not request results. Check your internet connection.")

def activate_voice_input():
    send_voice_message()

def send_message(event=None):
    user_message = user_input.get().lower()

    if user_message == 'exit':
        root.destroy()
    else:
        send_text_message(user_message)

# GUI setup
root = tk.Tk()
root.title("Medical ChatBot")
root.geometry("400x500")
root.configure(bg="#e0e0e0")

```

```
393   # Create a gradient background
394   canvas = Canvas(root, width=400, height=500)
395   canvas.pack()
396
397   canvas.create_rectangle(0, 0, 400, 500, fill="#87CEEB")  # Pik
398   canvas.create_rectangle(0, 0, 400, 250, fill="#87CEEB")  # Pink
399
400   # Create GUI components
401   chat_log = tk.Text(
402       root,
403       bd=0,
404       bg="#ffffff",
405       height="10",
406       width=300,
407       font=("Arial", 10),
408       wrap="word",
409       fg="#000000",
410   )
411   chat_log.config(state=tk.DISABLED)
412   scrollbar = tk.Scrollbar(root, command=chat_log.yview, cursor="arrow")
413   chat_log["yscrollcommand"] = scrollbar.set
414
415   user_input = tk.Entry(root, bd=0, bg="#ffffff", font=("Arial", 10), fg="#000000")
416   user_input.bind("<Return>", send_message)
417
418   # Updated button backgrounds to use #808000 color
419   send_button = tk.Button(
420       root,
421       text="Send",
422       width="9",
423       height="2",
424       bd=0,
425       bg="#adaeda",  # Updated button color
```

```
    activebackground="#adaeda",  # Updated active background color
    fg="#5d1717",
    font=("Arial", 10),
    command=send_message,
)

voice_button = tk.Button(
    root,
    text="Voice",
    width="9",
    height="2",
    bd=0,
    bg="#adaeda",  # Updated button color
    activebackground="#adaeda",  # Updated active background color
    fg="#5d1717",
    font=("Arial", 10),
    command=activate_voice_input,
)

appointment_button = tk.Button(
    root,
    text="Book Appointment",
    width="14",
    height="2",
    bd=0,
    bg="#adaeda",  # Updated button color
    activebackground="#adaeda",  # Updated active background color
    fg="#5d1717",
    font=("Arial", 10),
    command=open_appointment_website,
)

# Place components on the window
chat_log.place(x=6, y=6, height=386, width=388)
scrollbar.place(x=394, y=6, height=386)
user_input.place(x=6, y=401, height=40, width=390
)
send_button.place(x=320, y=450)  # Placed under the text input bar
voice_button.place(x=240, y=450)  # Placed under the text input bar
appointment_button.place(x=120, y=450)  # Placed under the text input bar

root.mainloop()

```

OUTPUT


Medical ChatBot
Medication: Rest, hydration, over-the-counter cold medications.
Details for influenza:
Description: A highly contagious viral infection affecting the respiratory system.
Medication: Rest, hydration, antiviral medications.
Details for malaria:
Description: A mosquito-borne infectious disease causing fever, chills, and flu-like symptoms.
Medication: Antimalarial drugs, prevention of mosquito bites.
Details for COVID-19:
Description: A contagious viral infection caused by SARS-CoV-2.
Medication: Isolation, medical care, vaccination.
Details for typhoid fever:
Description: A bacterial infection causing high fever, diarrhea, and abdominal pain.
Medication: Antibiotics, hydration, rest.
symptom checker fever
Book Appointment
Voice
Send


Medical ChatBot
Medication: Isolation, medical care, vaccination.
You: these information malaria
ChatBot: I'm sorry, I didn't understand that. Please try again.
You: disease information malaria
ChatBot: Details for malaria:
Description: A mosquito-borne infectious disease causing fever, chills, and flu-like symptoms.
Medication: Antimalarial drugs, prevention of mosquito bites.
You: book appointment
ChatBot: m
You: bye
ChatBot: Goodbye!
Book Appointment
Voice
Send

FUTURE SCOPE OF THE PROJECT AND CONCLUSION

Using a medical chatbot using Tkinter for a graphical user interface and functionality and communication platform for users seeking basic medical information using Python libraries for voice recognition, default answers and medical information database Provides chatbot for users can ask about symptoms, diseases, book policies, and chat using both text and voice input.

While the current implementation provides a solid foundation for medical chatbots, there are many avenues for future growth and development: expanding medical databases and using machine learning to generate matching signals.

BIBLIOGRAPHY

- https://pythongeeks.org
- https://www.programiz.com

THANK YOU