

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi-590018



A MINI PROJECT REPORT

ON

“HEALTHCARE CHATBOT”

*Submitted in partial fulfillment of the requirement for award of degree
of*

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

By

SHIVA PRASAD R

1AK22IS046

DARSHAN GK

1AK22IS008

SANJU A S

1AK22IS044

RAKESH D

1AK22IS036

Under the guidance of

Mr .Madhusudhana V

Assistant. Prof., Dept. of ISE, AIT



AKSHAYA
INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

Akshaya Institute of Technology

Lingapura, Obalapura Post, Koratagere Main Road, Tumakuru, Karnataka 572106

2024-2025



AKSHAYA
INSTITUTE OF TECHNOLOGY

Akshaya Institute of Technology
Lingapura, Obalapura Post, Koratagere Main Road, Tumakuru, Karnataka 572106

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project work entitled “**HEALTHCARE CHATBOT**” is a Bonafide work carried out by **SHIVA PRASAD R [1AK22IS046]**, **DARSHAN GK [1AK22IS008]** , **SANJU A S [1AK22IS044]** , **RAKESH D [1AK22IS036]** in the partial fulfillment of the requirements of V semester of **BACHELOR OF ENGINEERING in INFORMATION SCIENCE AND ENGINEERING** in **VISVESVARAYA TECHNOLOGICAL UNIVERSITY, Belagavi**, during the year **2024-2025**. It is certified that corrections/suggestions recommended for the project have been incorporated in the report.

Signature of Guide
Mr. Madhusudhana V
Asst. Prof. Dept. of ISE,
AIT, Tumakuru

Signature of HOD
Dr. Suhas G K
Head of the Dept. ISE,
AIT, Tumakuru

Name of the Examiners

Signature with date

1.

.....

2.

.....

ACKNOWLEDGEMENT

Firstly, we thank the **Management and Principal of Akshaya Institute of Technology**, Tumakuru for providing us an opportunity to work on this project. It gives us immense pleasure to express our deep sense of gratitude whose words of advice have always been a constant source of inspiration for us.

We would like to express our heartfelt thanks to **Dr. Suhas G K**, Professor and Head of Department of Information Science and Engineering, AIT for his valuable advice and encouragement to us in completing this project work.

We are obliged to **Mr. Madhusudhana V**, Assistant Professor, Dept. of ISE, who rendered valuable assistance as the project coordinator.

We would like to thank our **Parents** and **Friends** for their support, encouragement during the course of our project. Finally, we offer our regards to all the faculty members of ISE department and all those who supported us in any respect during the project.

SHIVA PRASAD [1AK22IS046]

DARSHAN GK [1AK22IS008]

SANJU A S [1AK22IS044]

RAKESH D [1AK22IS036]

ABSTRACT

This project report highlights the development of a **Healthcare Chatbot** designed to assist users with healthcare queries, appointment scheduling, and feedback collection.

The chatbot aims to improve healthcare accessibility, automate appointment processes, and gather valuable feedback from users to enhance services.

Developed using Flask, HTML, CSS, and JavaScript, the system leverages Natural Language Processing (NLP) for accurate query interpretation.

The chatbot is user-friendly, secure, and accessible across devices, reducing patient wait times and improving interaction efficiency.

The addition of feedback functionality ensures continuous improvement of healthcare services.

TABLE OF CONTENTS

ACKNOWLEDGEMENT	III
ABSTRACT	IV
TABLE OF CONTENTS	V
CHAPTER 1: INTRODUCTION.....	6
1.1 Introduction.....	6
1.2 Objectives.....	6
1.3 Importance of Healthcare Applications.....	6
1.4 What is Flask Development.....	6
1.5 Advantages of Flask for Chatbot Development.....	7
1.6 Challenges in Chatbot Development.....	7
CHAPTER 2: LITERATURE SURVEY.....	8
2.1 Existing System.....	8
2.2 Proposed System.....	8
CHAPTER 3: REQUIREMENTS.....	9
3.1 Software Requirements.....	9
3.2 Hardware Requirements.....	9
3.3 Functional Requirements.....	9
3.4 Non-Functional Requirements.....	9
CHAPTER 4: SYSTEM DESIGN.....	10
4.1 System Architecture.....	10
CHAPTER 5: IMPLEMENTATION.....	11
5.1 Project Structure.....	30
5.2 Core Features.....	31
CHAPTER 6: SNAPSHOTS.....	32
CONCLUSION AND FUTURE SCOPE.....	35
REFERENCES.....	36

CHAPTER 1:

INTRODUCTION

1.1 Introduction:

Healthcare systems often face challenges such as delayed appointment processes, limited access to healthcare providers, and lack of personalized interaction.

The healthcare chatbot addresses these issues by providing real-time assistance to users for healthcare-related queries, scheduling appointments, and collecting user feedback to improve services.

1.2 Objectives:

- Provide an interactive chatbot for healthcare services.
- Automate appointment scheduling with token generation.
- Collect user feedback to improve service quality.
- Ensure user-friendly navigation and 24/7 availability.
- Minimize patient wait times and streamline healthcare communication.

1.3 Importance of Healthcare Applications

The increasing demand for digital healthcare solutions has made chatbots indispensable in providing instant support, accessibility, and automation.

1.4 What is Flask Development

Flask is a lightweight Python web framework ideal for developing scalable and flexible applications, especially chatbots.

1.5 Advantages of Flask for Chatbot Development

- Simple and easy to implement.
- Integrates seamlessly with APIs (e.g., Twilio, Dialogflow).
- Lightweight for small to medium-scale applications.
- Facilitates modular code organization.

1.6 Challenges in Chatbot Development

- Handling complex and ambiguous user queries.
- Ensuring high-level data security and privacy.
- Providing real-time responses with minimal latency.
- Scaling the system to handle increased user traffic.

CHAPTER 2:

LITERATURE SURVEY

2.1 Existing System

- Manual appointment scheduling systems are time-consuming and prone to human error.
- Most existing healthcare applications lack interactive query resolution and instant feedback collection.

2.2 Proposed System

- An interactive chatbot that provides instant responses to healthcare queries.
- Automates appointment scheduling with token generation for patients.
- Includes a feedback system to enhance healthcare services.
- Features a secure, responsive, and user-friendly design.

CHAPTER 3:

REQUIREMENTS

3.1 Software Requirements

- **Programming Language:** Python
- **Framework:** Flask
- **Database:** SQLite or MySQL
- **Frontend:** HTML, CSS, JavaScript

3.2 Hardware Requirements

- **Processor:** Intel i5 or higher
- **RAM:** 8 GB or more
- **Storage:** Minimum 20 GB free space

3.3 Functional Requirements

- Real-time handling of user queries through a chat interface.
- Token generation for appointment scheduling.
- Feedback collection and storage for analysis.
- Secure storage and retrieval of user and appointment data.

3.4 Non-Functional Requirements

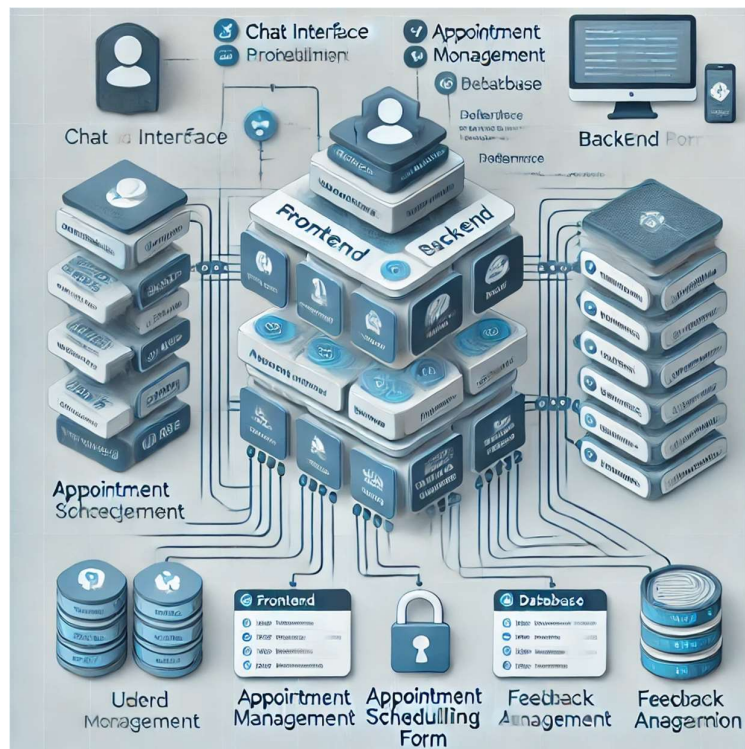
- Scalability to handle increased traffic.
- A responsive interface across devices.
- Data security with encryption for sensitive information.

CHAPTER 4:

SYSTEM DESIGN

4.1 System Architecture

- **Frontend:** Displays chat interface, appointment booking, and feedback form.
- **Backend:** Processes user queries, manages appointments, and stores feedback.
- **Database:** Stores user details, appointments, and feedback data.



CHAPTER 5:

IMPLEMENTATION

5.1 Project Structure

Project/

|

|— app.py

|— templates/

| |— index.html # Chat interface

| |— appointment.html # Appointment booking form

| |— feedback.html # Feedback form

|

|— static/

| |— css/

| | |— style.css

| |— images/

| |— background.jpeg

|

|— db/

| |— database.py # Database connection logic

App.py

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

import random

app = Flask(__name__)

# Sample data for doctors and health tips

doctors_info = {

    "cardiology": {"name": "Dr. Smith", "experience": "10 years", "availability": "Mon, Wed, Fri"},

    "neurology": {"name": "Dr. Doe", "experience": "8 years", "availability": "Tue, Thu"},

    "pediatrics": {"name": "Dr. Clark", "experience": "5 years", "availability": "Mon-Fri"},

}

health_tips = [

    "Drink plenty of water to stay hydrated.",

    "Exercise regularly for at least 30 minutes a day.",

    "Get regular check-ups with your healthcare provider.",

]

thank_you = [

    "Welcome, I will always assist you and Thank You"

]

hi = [

    "Hello, I am Health care chatBot,How can I assist you today?"]

# Function to generate an appointment token

def generate_token():

    return f'APT-{random.randint(1, 999)}'
```

```

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/book-appointment', methods=['POST'])
def book_appointment():
    doctor = request.form.get("doctor")
    date = request.form.get("date")
    time = request.form.get("time")

    # Save the data or process it

    return jsonify({"message": f'Appointment successfully booked with {doctor.capitalize()} on {date} at {time}.'})

# Route for handling user queries
@app.route('/query', methods=['POST'])
def query():
    user_input = request.form.get("user_input", "").lower().strip()

    # Response logic

    if "appointment" in user_input:
        token = generate_token()

        response = f'Your appointment has been scheduled. Your token is {token}. Please mention it when you arrive.'

    elif "doctor" in user_input or "specialty" in user_input:

        # Extract specialty from user input

        for specialty in doctors_info:
            if specialty in user_input:
                doctor = doctors_info[specialty]

                response = f'Our {specialty.capitalize()} specialist is {doctor["name"]} with {doctor["experience"]} experience. Available on {doctor["availability"]}.'

```

```
        break

    else:

        response = "We have specialists in doctor cardiology, doctor neurology, and doctor
pediatrics. Please specify one of these."

elif "health tip" in user_input:

    response = random.choice(health_tips)

# elif "token" in user_input:

#     response = random.choice(token)

elif "hi" in user_input:

    response = random.choice(hi)

elif "thank you" in user_input:

    response = random.choice(thank_you)

else:

    response = "Welcome to Official Health Hospital! \t \"Ask about appointments, doctors,
or health tips...\""

    return jsonify({"response": response})

if __name__ == '__main__':

    app.run(debug=True)
```

index.html

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Official Health CareChatbot</title>

  <link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">

</head>

<body>

  <div class="background">


    <header>

      <h1>Welcome to Official Health CareChatbot</h1>

      <p>Your health, our priority. How can we help you today?</p>

    </header>


    <div class="appointment-container">

      <h2>Book Your Appointment</h2>

      <form id="appointment-form">

        <label for="doctor">Select Doctor:</label>

        <select id="doctor" name="doctor" required>

          <option value="" disabled selected>Select a specialty</option>

          <option value="cardiology">Cardiology</option>

          <option value="neurology">Neurology</option>

          <option value="pediatrics">Pediatrics</option> </select>

      </form>

    </div>

  </div>

</body>

</html>
```

<label for="date">Choose Date:</label>

<input type="date" id="date" name="date" required>

<label for="time">Choose Time:</label>

<input type="time" id="time" name="time" required>

<button type="submit">Book Appointment</button>

</form>

<div id="appointment-message"></div>

</div>

<div class="chat-container">

<div id="chat-box">

<div id="chat-output"></div>

<form id="chat-form" method="post" action="/query">

<input type="text" id="user-input" name="user_input" placeholder="Ask about appointments, doctors, or health tips..." autocomplete="off" required>

<button type="submit">Send</button>

</form>

</div>

</div>


```
</div>

<div class="feedback-container">

  <h3>Feedback</h3>

  <form id="feedback-form">

    <textarea id="feedback-input" placeholder="Leave your feedback..."
required></textarea>

    <button type="submit">Submit Feedback</button>

  </form>

  <div id="feedback-message"></div>

</div>

<script>

  document.getElementById("chat-form").onsubmit = async (e) => {

    e.preventDefault();

    const userInput = document.getElementById("user-input").value;

    // Display the user's input

    const chatOutput = document.getElementById("chat-output");

    chatOutput.innerHTML += `<p><b>User:</b> ${userInput}</p>`;

    // Send the request to the server

    const response = await fetch("/query", {

      method: "POST",

      headers: { "Content-Type": "application/x-www-form-urlencoded" },

      body: new URLSearchParams({ user_input: userInput })

    });

  });
```

```
const data = await response.json();

// Display the bot's response
chatOutput.innerHTML += `<p><b>Bot:</b> ${data.response}</p>`;
document.getElementById("user-input").value = "";

// Scroll to the bottom of the chat
chatOutput.scrollTop = chatOutput.scrollHeight;
};

document.getElementById("feedback-form").onsubmit = async (e) => {
e.preventDefault();

const feedbackInput = document.getElementById("feedback-input").value;

// Send feedback to the server
const response = await fetch("/feedback", {
  method: "POST",
  headers: { "Content-Type": "application/x-www-form-urlencoded" },
  body: new URLSearchParams({ feedback: feedbackInput })
});

const data = await response.json();

// Show confirmation message
const feedbackMessage = document.getElementById("feedback-message");
feedbackMessage.textContent = data.message;
```

```
document.getElementById("feedback-input").value = "";

};

document.getElementById("appointment-form").onsubmit = async (e) => {

    e.preventDefault();

    const doctor = document.getElementById("doctor").value;

    const date = document.getElementById("date").value;

    const time = document.getElementById("time").value;

    const response = await fetch("/book-appointment", {

        method: "POST",

        headers: { "Content-Type": "application/x-www-form-urlencoded" },

        body: new URLSearchParams({ doctor, date, time })

    });

    const data = await response.json();

    const appointmentMessage = document.getElementById("appointment-message");

    appointmentMessage.textContent = data.message;

    document.getElementById("appointment-form").reset();

};
```

```
// Show the popup when the page loads

window.onload = function() {

    const popupHeader = document.getElementById("popup-header");

    popupHeader.style.display = "block"; // Display the popup


    // Close the popup when the close button is clicked

    const closeButton = document.getElementById("close-popup");

    closeButton.onclick = function() {

        popupHeader.style.display = "none"; // Hide the popup
    };


    </script>

    <footer>

        <p>&copy; HealthCare ChatBot 2024</p>

    </footer>


    </body>

    </html>
```

Style.css

```
body, html {  
    margin: 0;  
    padding: 0;  
    height: 100%;  
    font-family: Arial, sans-serif;  
    background-image: url("../css/images/background.jpeg");  
    background-size: cover;  
    background-position: center;  
    background-repeat: no-repeat;  
    /* background-attachment: fixed; */  
    image-rendering: -webkit-optimize-contrast; /* Chrome */  
    image-rendering: crisp-edges;           /* Firefox */  
    image-rendering: pixelated;  
    background-color: rgb(17, 17, 17);  
}
```

```
.background {  
    position: relative;  
    display: flex;  
    justify-content: center;  
    align-items: center;  
    height: 100%;  
    color: #ffffff;  
}
```

```
header {  
    position: absolute; /* Ensures the header stays at the top */  
    top: 0; /* Aligns the header to the top of the page */  
    left: 0; /* Ensures it starts from the left edge */  
    width: 100%; /* Makes the header span the full width of the page */  
    text-align: center; /* Centers the text within the header */  
    padding: 20px;  
    background-color: rgba(200, 200, 200, 0.20); /* Blue background */  
    color: white; /* White text color */  
    border-radius: 0 0 8px 8px; /* Rounded corners only at the bottom */  
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2); /* Add shadow for depth */  
    z-index: 1000; /* Ensures the header stays above other elements */  
}  
  
h1 {  
  
    background-color: #007bff;  
    border-radius: 0px;  
    /* background-size: text; */  
    margin: auto;  
    box-sizing: border-box;  
  
}
```

```
/* align-items:top; */
```

```
.chat-container {  
    background-color: rgba(255, 255, 255, 0.4);  
    border-radius: 8px;  
    width: 400px;  
    padding: 20px;  
    color: #333;  
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);  
    position: relative;  
    margin: 20px auto; /* Center horizontally */  
    top: 100px; /* Adjust spacing from the header */  
    text-align: center;  
    transform: translate(-50%, -50%);  
    margin-top: 200px;  
    position: relative;  
}
```

```
#chat-box {  
    padding: 10px;  
    border-radius: 8px;  
    background-color: #f1f1f1;  
    max-height: 300px;  
    overflow-y: auto;
```

```
    font-size: 16px;
}
```

```
#chat-output p {
    margin: 5px 0;
}
```

```
#chat-form {
    display: flex;
    margin-top: 10px;
}
```

```
#user-input {
    flex: 1;
    padding: 10px;
    font-size: 16px;
    border-radius: 4px;
    border: 1px solid #ccc;
}
```

```
button {
    padding: 10px 20px;
    font-size: 16px;
    background-color: #007bff;
    color: white;
    border: none;
    border-radius: 4px;
```



```
        cursor: pointer;
    }

    button:hover {
        background-color: #0056b3;
    }

    .feedback-container {
        margin-top: 20px;
        background-color: rgba(34, 35, 45, 0.592);
        padding: 15px;
        border-radius: 0px;
        color: #fff;
        margin-top: 60px;
    }

    .feedback-container textarea {
        width: 100%;
        height: 100px;
        padding: 10px;
        border: 1px solid #cccccc74;
        border-radius: 8px;
        font-size: 16px;
        color: rgb(255, 255, 255);
        background-color: rgba(33, 37, 100, 0.059);
    }

    .feedback-container button {
        margin-top: 10px;
```

```
background-color: #28a745;

color: white;

padding: 10px;

border: none;

border-radius: 4px;

cursor: pointer;

align-content: center;

}

.feedback-container button:hover {

background-color: #218838;

align-content: center;

}

.appointment-container {

background-color: rgba(255, 255, 255, 0.134);

padding: 15px; /* Reduced padding */

border-radius: 8px; /* Slightly smaller corner radius */

width: 320px; /* Reduced width */

margin: 15px auto; /* Center horizontally with less margin */

text-align: center;

box-shadow: 0 6px 12px rgba(0, 0, 0, 0.448); /* Slightly smaller shadow */

position: relative;

top: 100px; /* Adjust spacing from the chat box */

margin-top: 50px;

}
```

```
.appointment-container h2 {  
font-size: 20px; /* Adjust font size */  
color: #fff; /* Text color for contrast */  
margin-bottom: 15px;  
font-weight: bold;  
background-color: #0000009b; /* Background color for the heading */  
padding: 10px; /* Add padding around the text */  
border-radius: 6px; /* Optional: round the corners */  
text-align: center;  
}
```

```
.appointment-container select,  
.appointment-container input {  
width: 90%; /* Keep full width for readability */  
padding: 8px; /* Reduced padding */  
margin-bottom: 10px; /* Reduced spacing between inputs */  
border: 1px solid #151515f7;  
border-radius: 4px; /* Slightly smaller border radius */  
font-size: 14px; /* Reduced font size */  
}
```

```
.appointment-container button {  
padding: 10px 15px; /* Reduced button size */  
font-size: 14px; /* Reduced font size */  
background-color: #007bff;  
color: white;
```

```
border: none;

border-radius: 4px; /* Slightly smaller corner radius */

cursor: pointer;

transition: background-color 0.3s ease;
}


.appointment-container button:hover {

    background-color: #0056b3;
}


footer {

    text-align: center;

    padding: 20px;

    background-color: rgba(0, 0, 0, 0.23);

    color: #ddd;

    border-radius: 0;

    /* align-items: top; */

}

/* Start off-screen to the left */

/* Smooth transition */


/* When the cursor hovers over the form container, slide it in */
.appointment-container:hover {

    transform: translateX(0); /* Slide in to its original position */
}
```

```
}
```

```
.appointment-container {
```

```
    position: relative;
```

```
    transform: translateX(-100%); /* Start off-screen to the left */
```

```
    transition: transform 0.5s ease-in-out; /* Smooth transition */
```

```
}
```

```
/* Popup Header Styles */
```

```
.popup-header {
```

```
    position: fixed;
```

```
    top: 0;
```

```
    left: 0;
```

```
    right: 0;
```

```
    background-color: rgb(0, 4, 255); /* Dark background */
```

```
    color: #fff;
```

```
    padding: 20px;
```

```
    text-align: center;
```

```
    display: none; /* Hidden by default */
```

```
    z-index: 1000; /* Ensure it stays on top */
```

```
}
```

```
.popup-content {
```

```
    max-width: 600px;
```

```
    margin: 0 auto;
```

```
}
```

```
.popup-content h2 {
```

```
    font-size: 24px;
    margin-bottom: 10px;
}
```

```
.popup-content ul {
    list-style: none;
    padding: 0;
}
```

```
.popup-content li {
    margin: 5px 0;
    font-size: 18px;
}
```

```
.close-popup {
    margin-top: 15px;
    padding: 10px 20px;
    background-color: #28a745;
    color: white;
    border: none;
    border-radius: 4px;
    cursor: pointer;
}
```

```
.close-popup:hover {
    background-color: #218838;}
}
```

5.2 Core Features

1. Chat Interface:

- Enables interaction between the user and the bot.
- Processes healthcare queries using NLP techniques.

2. Appointment Scheduling:

- Allows users to select a doctor, date, and time.
- Automates token generation and notifies the user.

3. Feedback Collection:

- Displays a simple form for users to submit feedback.
- Stores feedback in the database for future analysis.

4. Background Design:

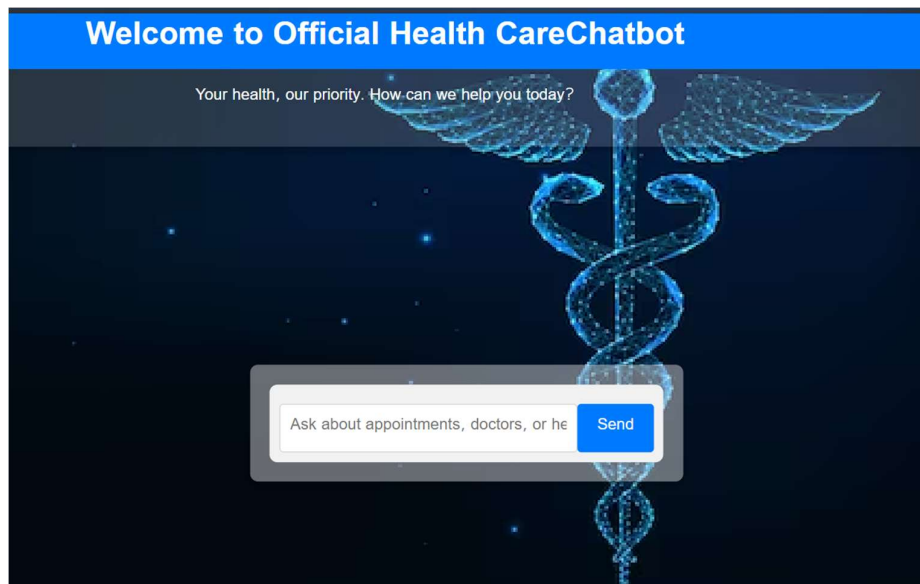
- A visually appealing interface with responsive and interactive design.

CHAPTER 6:

SNAPSHOTS

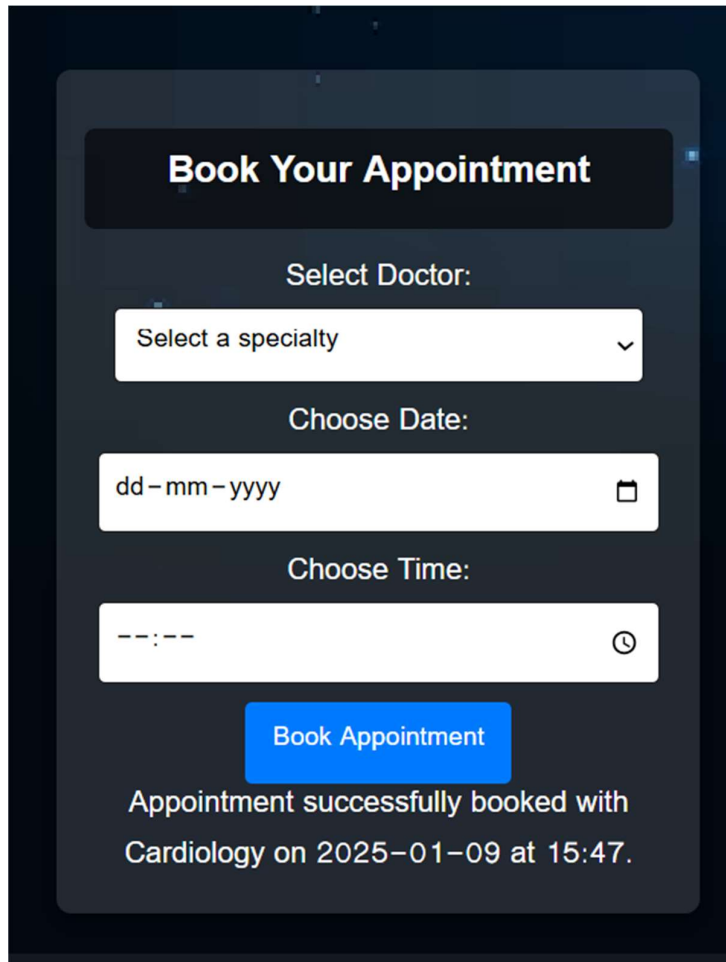
1. Homepage with Chat Interface:

Displays a chat box where users can interact with the chatbot.



2. Appointment Scheduling Page:

- Features a form to select a doctor, date, and time.
- Includes confirmation of token generation.



The image shows a dark-themed mobile application interface for booking an appointment. At the top, a dark blue header contains the text "Book Your Appointment" in white. Below this, the form is organized into three sections: "Select Doctor:", "Choose Date:", and "Choose Time:". Each section has a corresponding input field. The "Select Doctor:" field is a dropdown menu with the placeholder text "Select a specialty" and a downward arrow. The "Choose Date:" field is a date picker with the placeholder text "dd-mm-yyyy" and a calendar icon. The "Choose Time:" field is a time picker with the placeholder text "--:--" and a clock icon. Below these fields is a blue button labeled "Book Appointment". At the bottom of the form, a confirmation message reads: "Appointment successfully booked with Cardiology on 2025-01-09 at 15:47."

Book Your Appointment

Select Doctor:

Select a specialty

Choose Date:

dd-mm-yyyy

Choose Time:

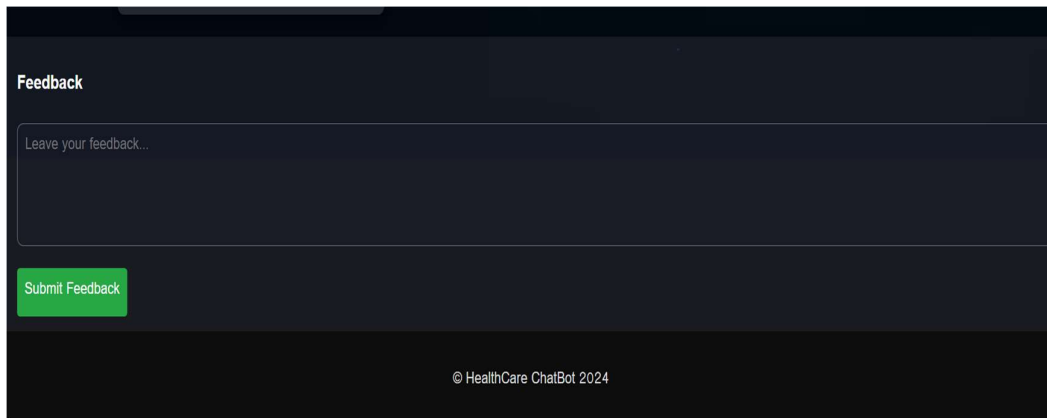
--:--

Book Appointment

Appointment successfully booked with
Cardiology on 2025-01-09 at 15:47.

3. Feedback Form:

- Allows users to submit feedback about their experience.
- Stores feedback in the database for further analysis.



Feedback

Leave your feedback...

Submit Feedback

© HealthCare ChatBot 2024

CONCLUSION AND FUTURE SCOPE

Conclusion

The healthcare chatbot bridges the gap between patients and healthcare providers by offering instant support, automated appointment scheduling, and feedback collection.

It is a secure, efficient, and scalable solution to modern healthcare needs.

Future Scope

1. Integration with voice assistants for voice-based commands.
2. Use of AI/ML for advanced query handling and predictive analytics.
3. Multi-language support for global reach and inclusivity.
4. Advanced feedback analysis using sentiment analysis.
5. Real-time notifications for reminders and health updates.

REFERENCES

1. Flask Official Documentation: <https://flask.palletsprojects.com>
2. Bootstrap for Front-End Design: <https://getbootstrap.com>
3. MySQL Official Documentation: <https://dev.mysql.com/doc/>
4. Twilio API Documentation: <https://www.twilio.com/docs/>