

**Project on Medical Chatbot for Symptom Analysis**

**Name**: Saroj Chaudhuri

**Reg. No.**: 23BCE8908

**College Name**: VIT-Ap University

**Branch**: CSE Core

**Date of Submission** :02.07.2025

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sl No.** | **Topic** | **Page No.** |
| 1. | Introduction | 1 |
| 2. | Objective | 2 |
| 3. | Methodology / Working | 3 |
| 4. | Code Snippets | 6 |
| 5. | Screenshots / Output Results | 11 |
| 6. | Challenges Faced & Solutions | 15 |
| 7. | Conclusion | 16 |

**Introduction**

In today's healthcare systems, patients often struggle to identify which medical specialist to consult based on their symptoms. This results in misdirected appointments, increased waiting times, and poor healthcare experiences.

This project presents a conversational AI chatbot that interacts with users, collects symptoms via natural language, and intelligently recommends the appropriate medical department (e.g., Cardiology, Dermatology, Neurology). It uses a combination of rule-based logic and generative AI for symptom classification.

**Objective**

* Guide users to the correct medical department using natural symptom descriptions.
* Minimize misdirected or incorrect referrals.
* Leverage AI to improve triage accuracy without needing manual intervention.
* Integrate IBM Watson Assistant and watsonx.ai to demonstrate a hybrid conversational + reasoning system.

**Tools & Technologies Used**

|  |  |
| --- | --- |
| Tool/Platform | Purpose |
| IBM Watson Assistant | Handles the chatbot interface and conversation flow |
| IBM watsonx.ai | Performs reasoning using prompt-based LLM inference |
| Prompt Lab | Used to craft and test few-shot prompts for accurate classification |
| Custom Extensions | Connects Watson Assistant with watsonx.ai using API |

**Methodology / Working**

**Method A: Department prediction using watsonx.ai**

**1. Conversation Flow Setup**

* Created an action-based chatbot using **Watson Assistant**.
* The user is asked to describe their symptoms.
* The input is stored in a variable by default.

**2. Prompt Lab for Reasoning**

A few-shot prompt is designed and tested in **Prompt Lab** with examples like:

Symptom: chest pain

Department: Cardiology

Symptom: itchy skin

Department: Dermatology

Symptom: blurry vision and dizziness

Department:

The model responds with the appropriate department name (e.g., “Neurology”).

**3. Integration with watsonx.ai**

* A custom **extension** is created in Watson Assistant using an OpenAPI spec.
* Parameters like model\_id, project\_id, input, and stop sequences (["\n"]) are configured.
* The response is captured via:

$watsonx\_step.body.results[0].generated\_text

**4. Response Generation**

The Assistant replies:

Based on your symptoms, I recommend consulting the {{result}} department.

**Method B: Rule-Based Keyword Matching via Switch Cases**

While the AI-based model in Method A provides dynamic and intelligent classification of symptoms, it is important to have a fast, deterministic, and fallback-friendly alternative. For this purpose, we implement Method B, a rule-based decision system using Switch logic within Watson Assistant.

1. The system checks if the user input (symptom\_input) contains any predefined keywords.
2. These keywords are manually mapped to departments based on medical domain knowledge.
3. Watson Assistant's built-in Switch block is used to match conditions and return the corresponding department.

|  |  |
| --- | --- |
| Keywords in User Input | Department Assigned |
| itching, rash, allergy | Dermatology |
| vomiting, stomach pain | Gastroenterology |
| headache, dizziness | Neurology |
| cough, sore throat | ENT |
| chest pain, breathless | Cardiology |

**When Is It Used?**

* We use Method B first, then fallback to Method A (watsonx.ai) only when no match is found
* Or use Method A by default, and fallback to Method B when API fails or low confidence is detected.

**Code Snippets**

* 1. **Few-Shot Prompt Format (Passed as input)**

Identify the correct medical department from the following examples:

Symptom: chest pain

Department: Cardiology

Symptom: red itchy skin

Department: Dermatology

Symptom: {{symptom\_input}}

Department:

* 1. **Stop Sequence Parameter**

"parameters.stop\_sequences": ["\n"],

"parameters.include\_stop\_sequence": false

* 1. **Response Mapping in Assistant**

Set variable: result = $watsonx\_step.body.results[0].generated\_text

Assistant says:

"Based on your symptoms, I recommend consulting the {{result}} department."

**4.Sample website**

**Html**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8" />

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

  <title>DocMed • Your Health Partner</title>

  <link rel="stylesheet" href="style.css" />

</head>

<body>

  <header>

    <div class="logo">DocMed</div>

    <nav>

      <ul>

        <li><a href="#about">About Us</a></li>

        <li><a href="#departments">Departments</a></li>

        <li><a href="#membership">Membership</a></li>

        <li><a href="#help">Help</a></li>

        <li><a href="#contact">Contact</a></li>

        <li><button class="signin">Sign In</button></li>

      </ul>

    </nav>

  </header>

  <section class="hero">

    <div class="hero-text">

      <h1>We're determined for your better life.</h1>

      <p>You can get the care you need 24/7 — be it online or in person. You’ll be treated by caring specialist doctors.</p>

      <button class="cta">Make an Appointment</button>

    </div>

    <div class="hero-img">

      <img src="/website/Sources/hospital-1.png" alt="Doctor Team" />

    </div>

  </section>

  <footer>

    <p>© 2025 DocMed. Designed with care </p>

  </footer>

  <script src="script.js"></script>

</body>

</html>

**Css**

body {

  margin: 0;

  font-family: 'Segoe UI', sans-serif;

  background: #eaf6ff;

  color: #333;

}

header {

  display: flex;

  justify-content: space-between;

  padding: 20px 40px;

  background: #007bff;

  color: white;

}

.logo {

  font-size: 24px;

  font-weight: bold;

}

nav ul {

  list-style: none;

  display: flex;

  gap: 20px;

  align-items: center;

  margin: 0;

  padding: 0;

}

nav a {

  color: white;

  text-decoration: none;

  font-weight: 500;

}

.signin {

  background: white;

  color: #007bff;

  padding: 8px 12px;

  border: none;

  border-radius: 4px;

  cursor: pointer;

}

.hero {

  display: flex;

  justify-content: space-between;

  padding: 60px 40px;

  background: #f4f9ff;

}

.hero-text {

  max-width: 600px;

}

.hero-text h1 {

  font-size: 32px;

  margin-bottom: 20px;

}

.hero-text p {

  font-size: 18px;

  margin-bottom: 30px;

}

.cta {

  padding: 12px 24px;

  background: #007bff;

  color: white;

  border: none;

  border-radius: 4px;

  cursor: pointer;

}

.hero-img img {

  width: 400px;

  border-radius: 50%;

  box-shadow: 0 8px 20px rgba(0,0,0,0.1);

}

footer {

  text-align: center;

  padding: 20px;

  background: #dfeffc;

  font-size: 14px;

}

**Javascript**

window.watsonAssistantChatOptions = {

    integrationID: "f010c974-1e1d-4f33-ab2f-f6ba8efbd07e", // The ID of this integration.

    region: "us-south", // The region your integration is hosted in.

    serviceInstanceID: "a85ec0f3-3c56-4345-b0ec-6f114bffd335", // The ID of your service instance.

    onLoad: async (instance) => { await instance.render(); }

};

setTimeout(function(){

    const t=document.createElement('script');

    t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" + (window.watsonAssistantChatOptions.clientVersion || 'latest') + "/WatsonAssistantChatEntry.js";

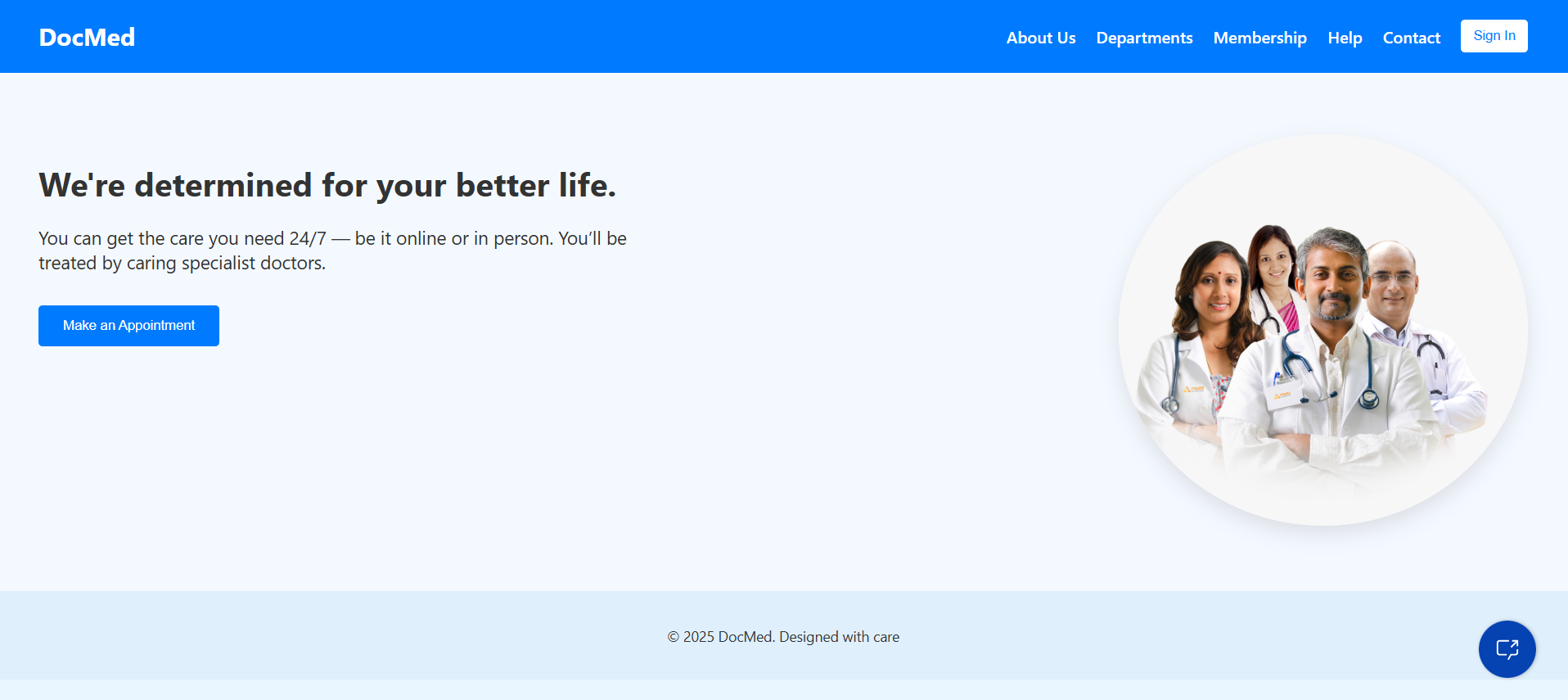
    document.head.appendChild(t);

});

**Screenshots / Output Results**

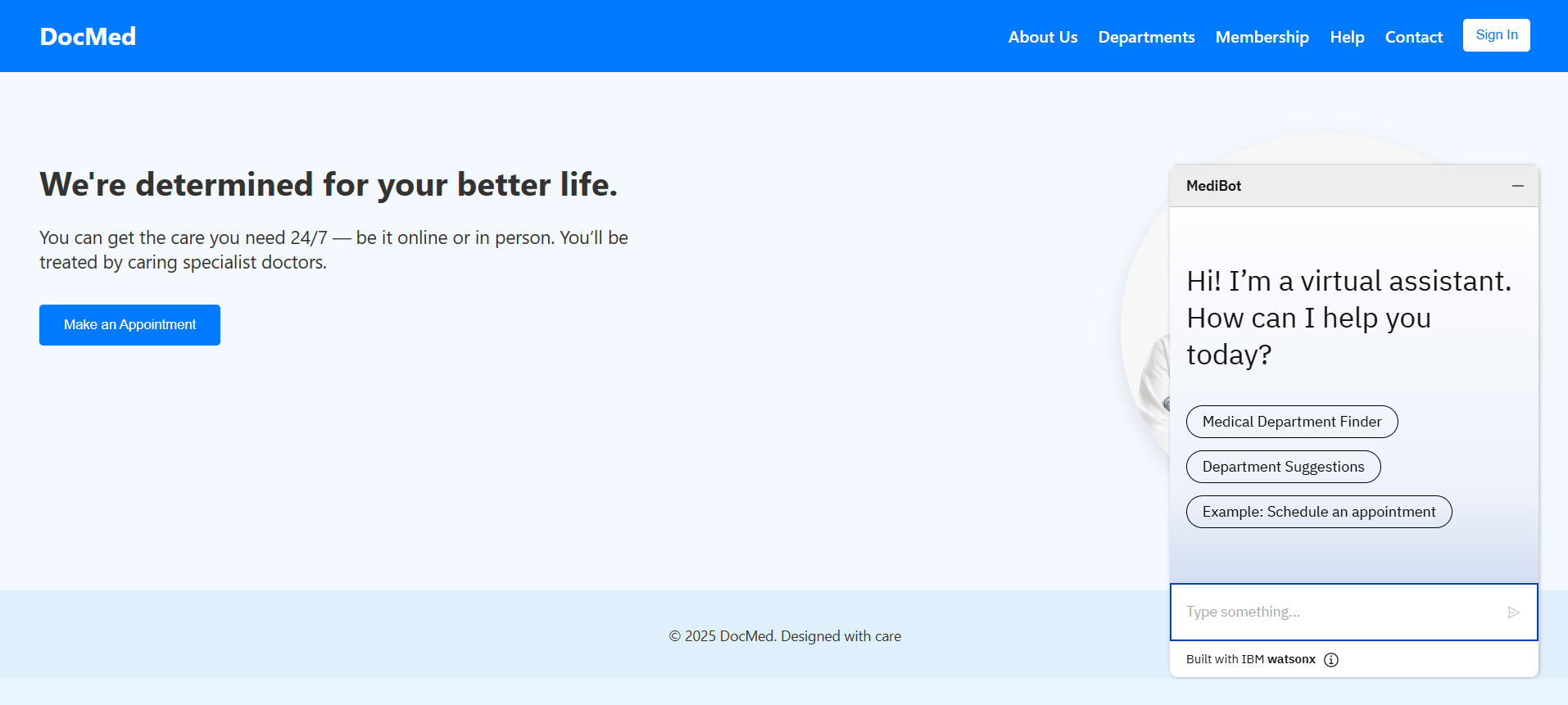
1. **Chatbot Embedded on Website**

The chatbot is successfully integrated into the DocMed website as a floating widget, ready to assist users with medical queries.

****

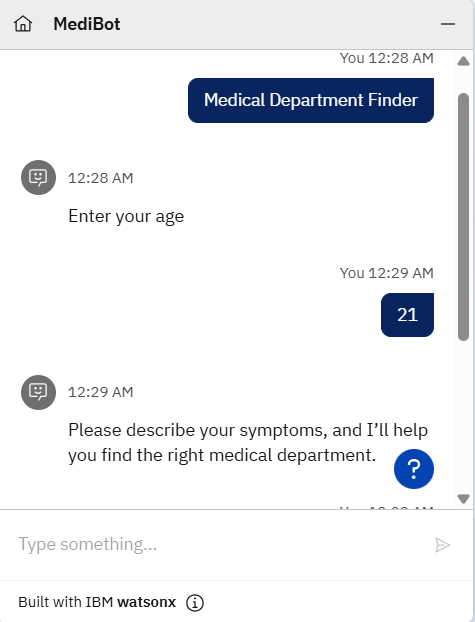
**2.The Chatbot is initiated**

By clicking the chatbot icon the chatbot pops up and greets the user.



**3. Department Finder Flow Initiated**

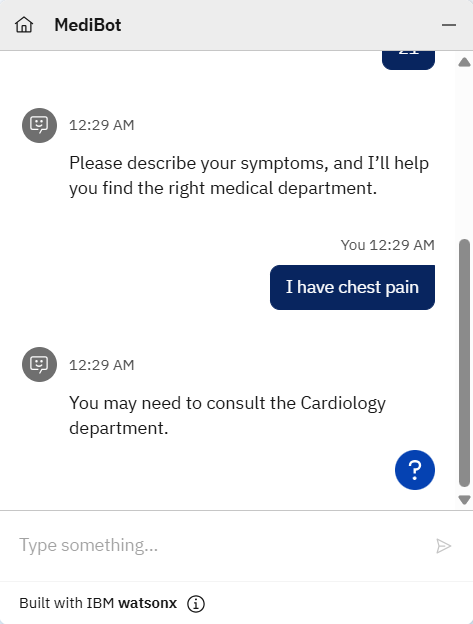
After the user clicks **Medical Department Finder**, the chatbot initiates the flow by asking for the user's **age**, followed by a prompt to describe their symptoms. This is the input stage for natural language symptom collection, which will be analyzed in the next step.



**4. Symptom Analyzed – Output Displayed**

Here, the user enters the symptom:

"I have chest pain"  
The assistant forwards this input to **watsonx.ai** via a custom extension with a prompt template, and receives the response:  
"Cardiology"  
The assistant then constructs a natural reply:  
“You may need to consult the **Cardiology** department.”  
This confirms that the prompt logic and LLM integration is functioning correctly and interpreting symptoms accurately.



Links for your project (Github if Possible)

**Challenges Faced & Solutions**

**1. Integration Between Watson Assistant and watsonx.ai**

One of the major challenges was integrating Watson Assistant with watsonx.ai using custom extensions. The configuration required specific parameters like project\_id, model\_id, and formatting the input prompt precisely. Initially, this led to errors such as “project not found” or 404 responses.

**Solution:**  
The issue was resolved by carefully checking and setting all required parameters in the extension setup, including using valid project\_id from watsonx and the correct LLM model (flan-t5-xxl). Also, configuring stop sequences (["\n"]) ensured the model output was clean and didn’t overgenerate text.

**2. Model Overgeneration and Inconsistent Output**

When using generative models for department classification, the model sometimes produced verbose or inaccurate results, like "Cardiology.com – Heart disease specialist" instead of just "Cardiology".

**Solution:**  
To fix this, decoding settings were fine-tuned — greedy decoding method, low temperature, and adding a newline \n as a stop sequence limited the model to return only the department name. The prompt was also restructured to be more controlled and example-driven (few-shot format).

**3.Deploying on a Website**

Deploying the assistant on a custom HTML website and ensuring it rendered properly in all screen sizes required extra attention, especially with chatbot positioning and styling.

**Solution:**  
The IBM Watson Assistant Web Chat integration script was added to the site with responsive layout design, ensuring the bot loaded smoothly and remained accessible across desktop and mobile.

**Conclusion**

The Medical Chatbot for Symptom Analysis demonstrates how conversational AI, powered by IBM Watson Assistant and watsonx.ai, can be effectively used to guide patients toward the right medical departments based on their symptoms. By combining few-shot learning through Prompt Lab with real-time user interactions, the chatbot provides a practical solution to one of the most common challenges in healthcare: misdirected appointments.

The project showcases a hybrid approach—leveraging both AI reasoning for complex inputs and rule-based logic for common symptoms—to ensure speed, accuracy, and reliability. It also emphasizes user-centric design by enabling smooth web integration, natural language understanding, and optional feedback mechanisms.

This solution not only improves the triage experience for patients but also has the potential to reduce administrative workload and enhance the efficiency of healthcare systems. With further enhancements such as symptom history tracking, multi-language support, and integration with hospital databases, this chatbot can evolve into a powerful front-line digital health assistant.