Type your text

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

IBM

COURSE RECOMMENDATION BOT

Name- Dhruv Batra

Registration No. - 23BCE0074

College,Branch - Vit vellore , Cse Core

Table of Contents

Introduction

With the rapid rise of online learning platforms, users are often overwhelmed by the vast number of available courses. Identifying the most relevant course based on a learner's background, interests, and skill level has become a time-consuming and confusing process.

To solve this problem, this project introduces a **Course Recommendation Bot** powered by **Generative AI technologies**, including **IBM Watson Assistant** and **watsonx.ai**. The bot engages users in a natural conversation, collects basic inputs—such as user type (student, professional), area of interest, and experience level—and recommends suitable courses tailored to their profile.

By applying **prompt engineering** and utilizing **foundation models**, the bot delivers accurate and personalized recommendations. This project showcases a real-world application of generative AI in the education sector, offering a smarter and more efficient alternative to manual course browsing.

Overall, the Course Recommendation Bot simplifies decision-making for learners and demonstrates how AI can enhance digital learning experiences through intelligent automation.

Objective

The primary objective of this project is to build an AI-driven chatbot that provides users with personalized course recommendations based on their background, interests, and skill level.

Specifically, the Course Recommendation Bot aims to:

- **Simplify** the process of identifying relevant online courses or certifications
- **Collect user inputs** such as role (student, professional), interest area, and experience level through conversational interaction
- **Generate personalized suggestions** using prompt-based AI models powered by watsonx.ai
- **Integrate IBM Watson Assistant** with generative AI tools for real-world educational use cases
- **Reduce manual effort** and decision fatigue commonly faced by learners navigating course platforms
- **Demonstrate the effectiveness** of conversational AI in improving user engagement and learning discovery

This solution is scalable and can be integrated into educational institutions, career portals, and e-learning platforms to support personalized upskilling.

Tools and Technologies used:

IBM Watson Assistant

IBM Watsox.ai

IBM Cloud Console , Browser

HTML5 ,CSS3 , JavaScript, GitHub

Methodology/Working:

Introduction

In the age of digital learning, selecting the right course that aligns with an individual's career goals, interests, and skill level can be a challenging task. The **Course Recommendation Bot** was developed to address this need by offering a guided, personalized, and interactive experience to users. Built using **IBM Watson Assistant** and **watsonx.ai**, this intelligent system leverages AI to recommend relevant courses based on user input. Its no-code architecture and logic-based slot handling enable a streamlined development process while maintaining high-quality user experience.

This document provides a comprehensive step-by-step explanation of the bot's workflow, highlighting its components, logic, and data flow.

Step-by-Step Workflow

1. User Interaction Begins

The user journey starts when a visitor interacts with the chatbot interface. As soon as the conversation is initiated, the bot responds with a welcoming message and a brief overview of its capabilities. For example:

"Hi there! I'm your Course Recommendation Assistant. I'll help you find the best courses based on your background and interests."

This initial interaction sets the tone for the rest of the conversation and reassures users that they are engaging with a smart and helpful system.

---

2. User Details are Collected

To offer personalized recommendations, the bot needs specific information about the user. This is done through a sequence of targeted, friendly questions. Using logic-based slot handling, the chatbot ensures that all required inputs are collected before moving forward. The key user inputs include:

- **User Type**:
  The bot asks the user to identify themselves as one of the following:
  - Student
  - Working Professional
  - Job Seeker
- **Interest Area**:
  The user selects their preferred domain or area of interest from options such as:
  - Artificial Intelligence (AI)
  - Web Development
  - Data Science
  - Cybersecurity
  - Cloud Computing

- **Experience Level**:
  The bot then asks about the user's current experience level, allowing selection among:
    - Beginner
    - Intermediate
    - Advanced

The use of slot-based conversation ensures that users do not have to repeat information and the flow remains natural and interactive.

---

3. Prompt Template is Triggered

Once all three inputs are collected, the bot uses a **no-code prompt template** hosted in **watsonx.ai** to process the data. This template is pre-configured to interpret combinations of user responses and generate accurate prompts for course recommendation.

Example prompt used internally:

"Recommend 2–3 online courses for a beginner-level student interested in Data Science."

These templates reduce complexity for developers while ensuring consistency and accuracy in the recommendations generated by the AI model.

---

4. Recommendation Generation

At this stage, the system leverages **watsonx.ai's** powerful large language models (LLMs) to analyze the prompt and generate meaningful course suggestions. The model draws from pre-loaded course datasets or dynamically linked external sources to curate a list of 2–3 courses that fit the user's profile.

Each course recommendation typically includes:

- **Course Name**
- **Provider or Platform** (e.g., Coursera, edX, Udemy)
- **Estimated Duration**

Example Output:

1. **"Introduction to Data Science"** – Coursera – 4 weeks
2. **"Data Science Crash Course"** – Udemy – 6 hours
3. **"Beginner's Guide to Data Analytics"** – edX – 5 weeks

These recommendations are relevant, short-listed, and actionable, making it easier for the user to make a decision.

---

5. Response Delivery

After generating the recommendations, the bot formats the information neatly and sends it back to the user through the chat interface. This response is user-friendly and often includes a follow-up question to extend the conversation:

“Here are some course suggestions for you! Would you like more options or want to explore another interest area?”

This keeps the user engaged and allows them to navigate back or repeat the process with different criteria.

---

6. Loop or Exit

Finally, the bot offers the user the choice to:

- **Get More Recommendations** (using same or new inputs)
- **End the Conversation**

If the user opts to continue, the bot either loops through the input collection again or adjusts parameters like the interest area or experience level to generate fresh results. If the user chooses to exit, the bot ends the session politely:

“Glad I could help! Wishing you success in your learning journey. ”

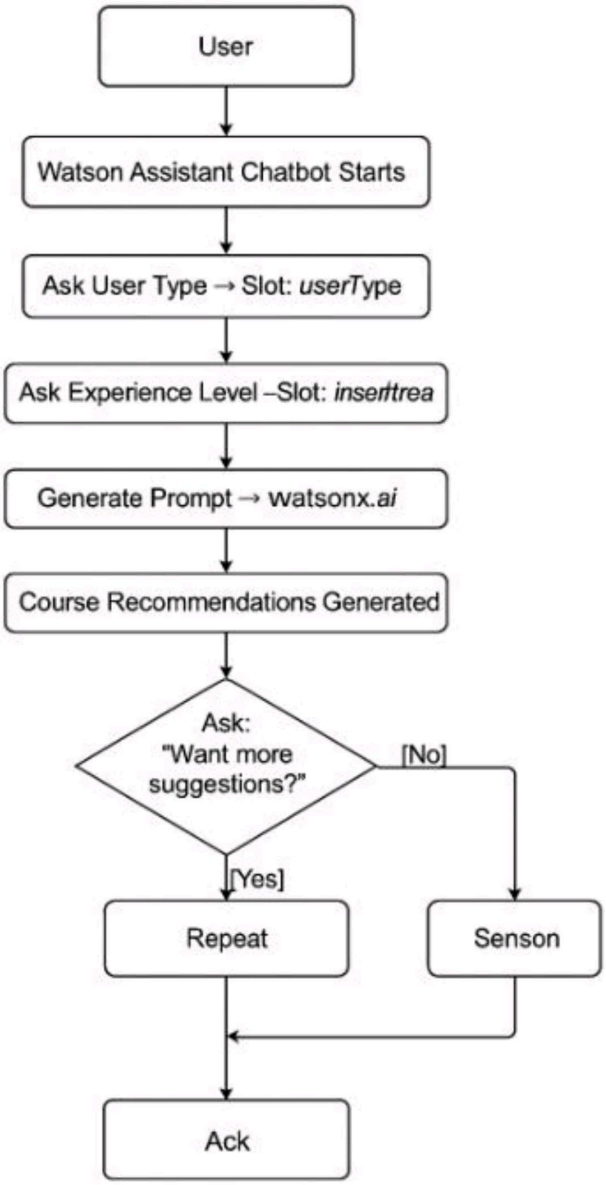
This looping logic allows the user full control of the interaction without needing to refresh or restart the chat.

Conclusion

The **Course Recommendation Bot** provides a smart, responsive, and personalized solution for users seeking online courses. With the integration of **IBM Watson Assistant** for dialogue management and **watsonx.ai** for prompt-based AI generation, the bot ensures a seamless and interactive experience. The combination of user-centric design, logical slot handling, and advanced AI models makes it an efficient tool for students, professionals, and job seekers alike.

By following this clear workflow, the bot not only enhances user engagement but also simplifies the course discovery process in an increasingly complex digital education environment.

Model Flowchart



Codes and Snippets:

Prompt Template for watsonx.ai

Code:

User Input:

"I'm a student interested in Web Development at a beginner level."

Prompt:

"Based on this user input, recommend 2 beginner-level Web Development courses. Mention the course name, platform, and duration."

Expected Response Format:

1. Course Name – Platform

2. Course Name – Platform

Explanation:

This prompt is used in **IBM watsonx.ai** to interact with a foundation model like **FLAN-T5** or **Mistral**. The AI model uses this structured input to return accurate course recommendations based on user preferences. It's a classic use of **prompt engineering** in GenAI.

2. Slot-Based Action in Watson Assistant

Code:

```
{
  "name": "AskInterest",
  "type": "slot",
  "variable": "interestArea",
  "question": {
    "text": "What subject are you interested in? (e.g., AI, Web Development, Cybersecurity)"
  },
  "next_step": {
    "behavior": "jump_to",
    "dialog_node": "AskLevel"
  }
}
```

Explanation:

This code defines a **slot-based action** in Watson Assistant to capture the user's area of interest. Once the user replies (e.g., "Data Science"), the value is stored in a variable (interestArea) and used later to generate a prompt or response. It's part of the chatbot's logical flow.

4. Webpage Integration Script
   Code:

```
<script>
  window.watsonAssistantChatOptions = {
    integrationID: "fb18aedf-766e-4bb4-866c-703e5d74e049", // The ID of this integration.
    region: "us-south", // The region your integration is hosted in.
    serviceInstanceID: "069e48cc-1702-471d-8e5c-15097daba302", // 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);
  });
</script>
```

   Explanation:
   This is the **embed script** used to integrate the Watson Assistant chatbot into a webpage. After setting up the integration in IBM Cloud, you copy your **Integration ID**, **region**, and **Service Instance ID** into this script and include it in your index.html. When the page loads, the chatbot appears in the bottom corner.

5 Webpage Html+Css code:

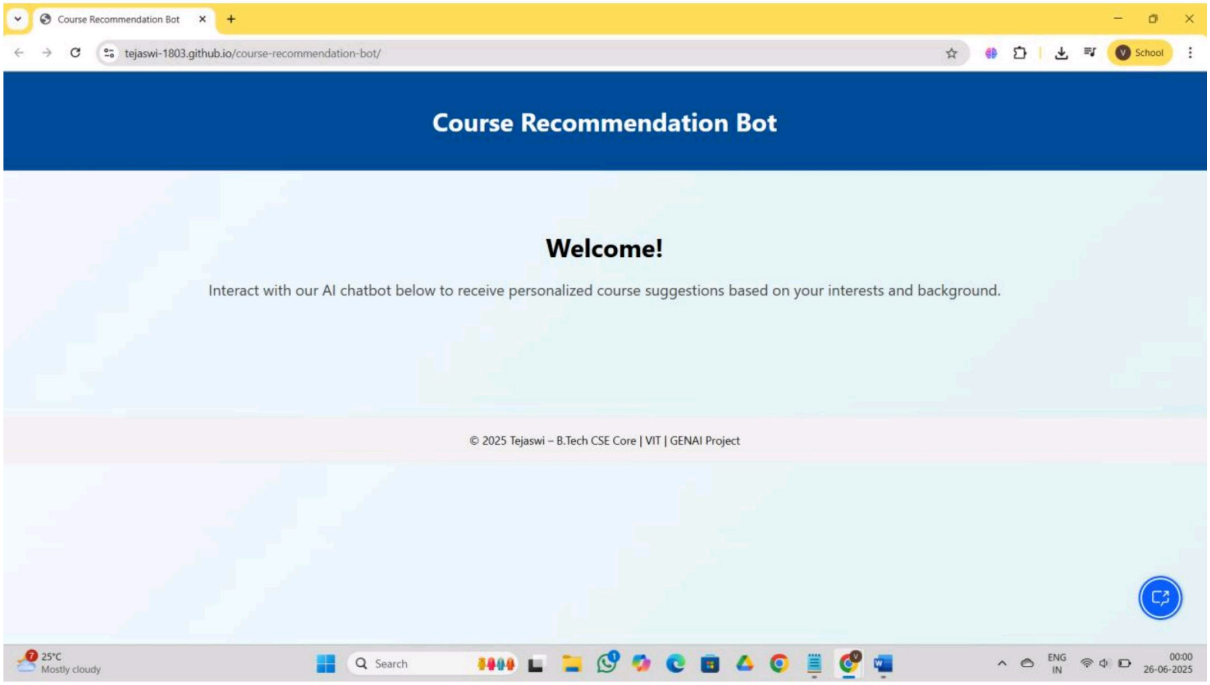
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Course Recommendation Bot</title>
    <style>
        body {
            font-family: 'Segoe UI', sans-serif;
            margin: 0;
            background: linear-gradient(135deg, #f1f9ff, #e0f2f1);
        }
        header {
            background-color: #004d99;
           color: white;
            padding: 20px;
```

```
            text-align: center;
        }
        .container {
            text-align: center;
            padding: 50px;
        }
        .container h2 {
            font-size: 2em;
            margin-bottom: 20px;
        }
        .container p {
            font-size: 1.2em;
            color: #444;
        }
        footer {
            text-align: center;
            font-size: 0.9em;
            padding: 20px;
            background-color: #f2f2f2;
            margin-top: 80px;
        }
    </style>
</head>
<body>
    <header>
        <h1>Course Recommendation Bot</h1>
    </header>

    <div class="container">
        <h2>Welcome!</h2>
        <p>Interact with our AI chatbot below to receive personalized course suggestions based
on your interests and background.</p>
    </div>

    <footer>
        &copy; 2025 Tejaswi – B.Tech CSE Core | VIT | GENAI Project
    </footer>

    <!-- 🧠 IBM Watson Assistant Integration -->

<script>
  window.watsonAssistantChatOptions = {
    integrationID: "fb18aedf-766e-4bb4-866c-703e5d74e049", // The ID of this integration.
    region: "us-south", // The region your integration is hosted in.
    serviceInstanceID: "069e48cc-1702-471d-8e5c-15097daba302", // The ID of your service
instance.
    onLoad: async (instance) => { await instance.render(); }
  };
  setTimeout(function(){
    const t=document.createElement('script');
```

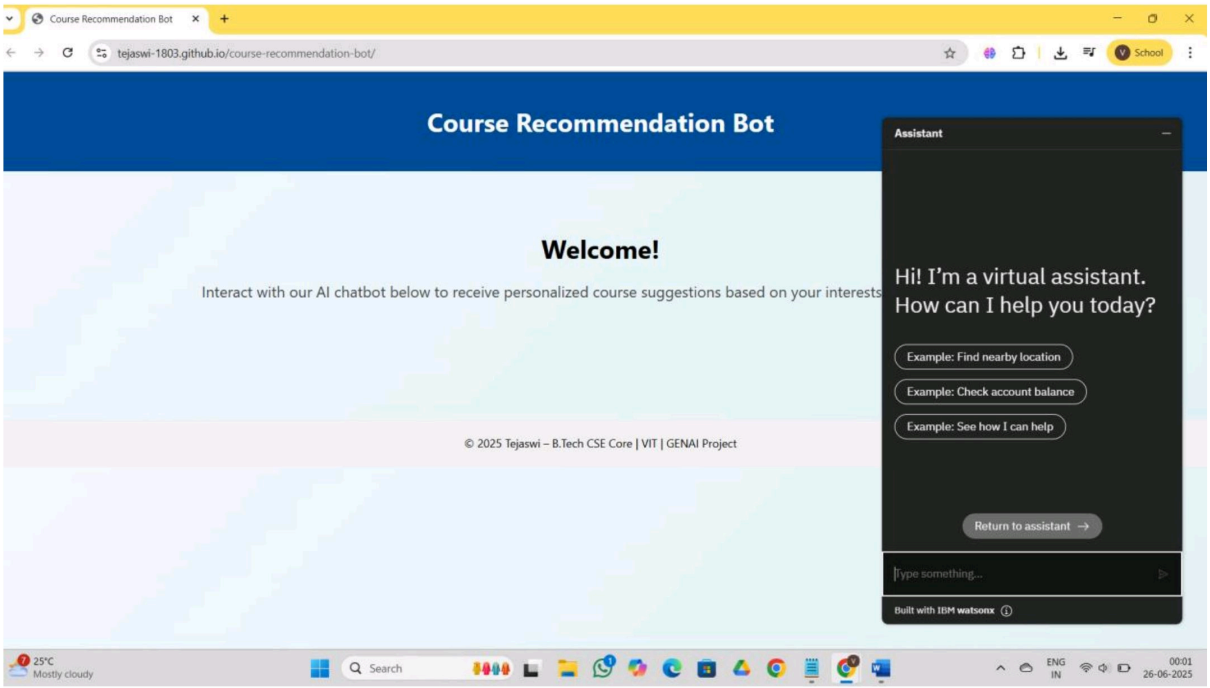
```
    t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') +
"/WatsonAssistantChatEntry.js";
    document.head.appendChild(t);
  });
</script>
</body>
</html>
```

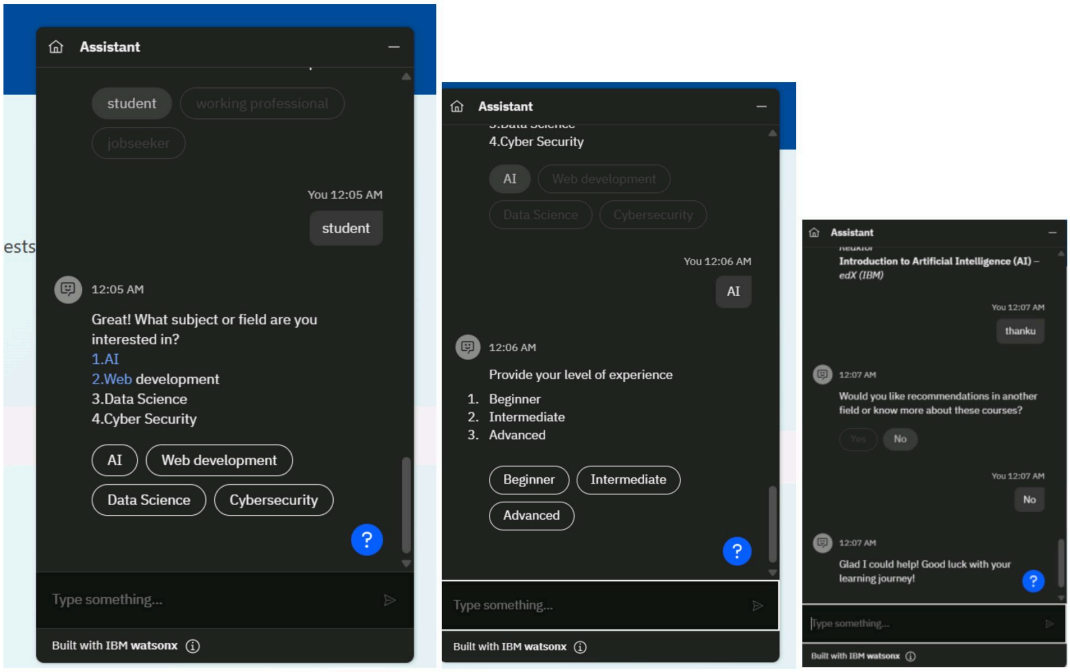
ScreenShots or Output Results

Webpage Screenshot:

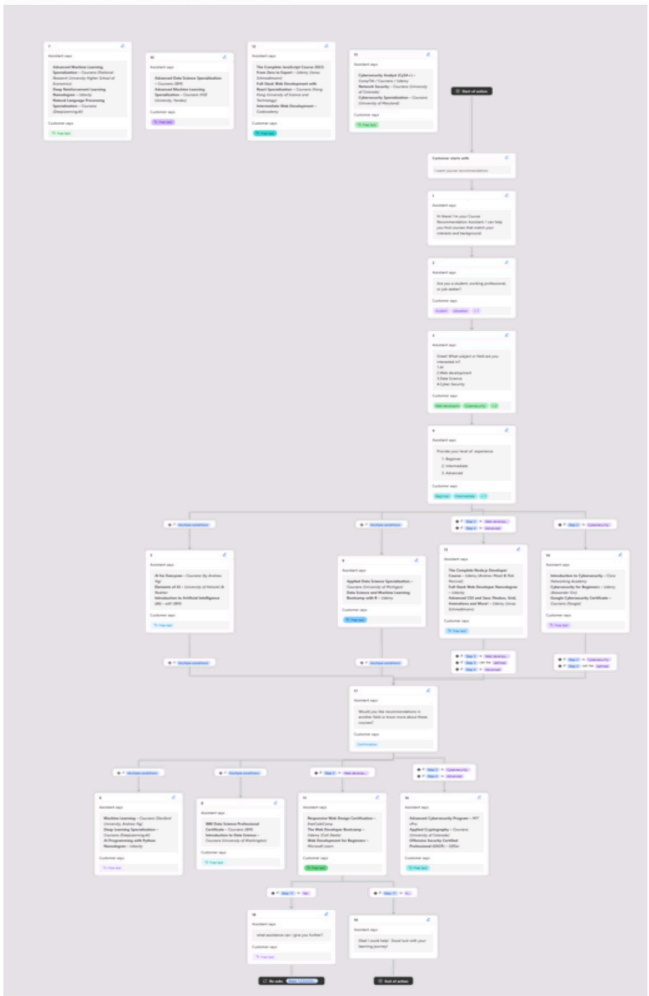


ChatBot ScreenShots:





ChatBot Visualization:



GitHub Link:

https://github.com/dbatra20/genaiibm