innovate achieve lead

BITS Pilani

Artificial Intelligence Assignment

Submitted by: Likhit Gatagat
2022MT13057
Bits, Pilani, WILP
Email: 2022MT13057@wilp.bits-pilani.ac.in

SSZC444 Artificial Intelligence

November 27, 2023

Custom Conversational AI agent (Chatbot) using OpenAI

Installations/Packages needed for this assignment:

- ❑ Node - v20.10.0
- ❑ NPM - 10.2.3
- ❑ Express - 4.18.2 - Framework we'll use to spin up a Node server
- ❑ Airtable - 0.11.6 -
- ❑ OpenAI - 3.3.0 - Node.js library for the OpenAI API

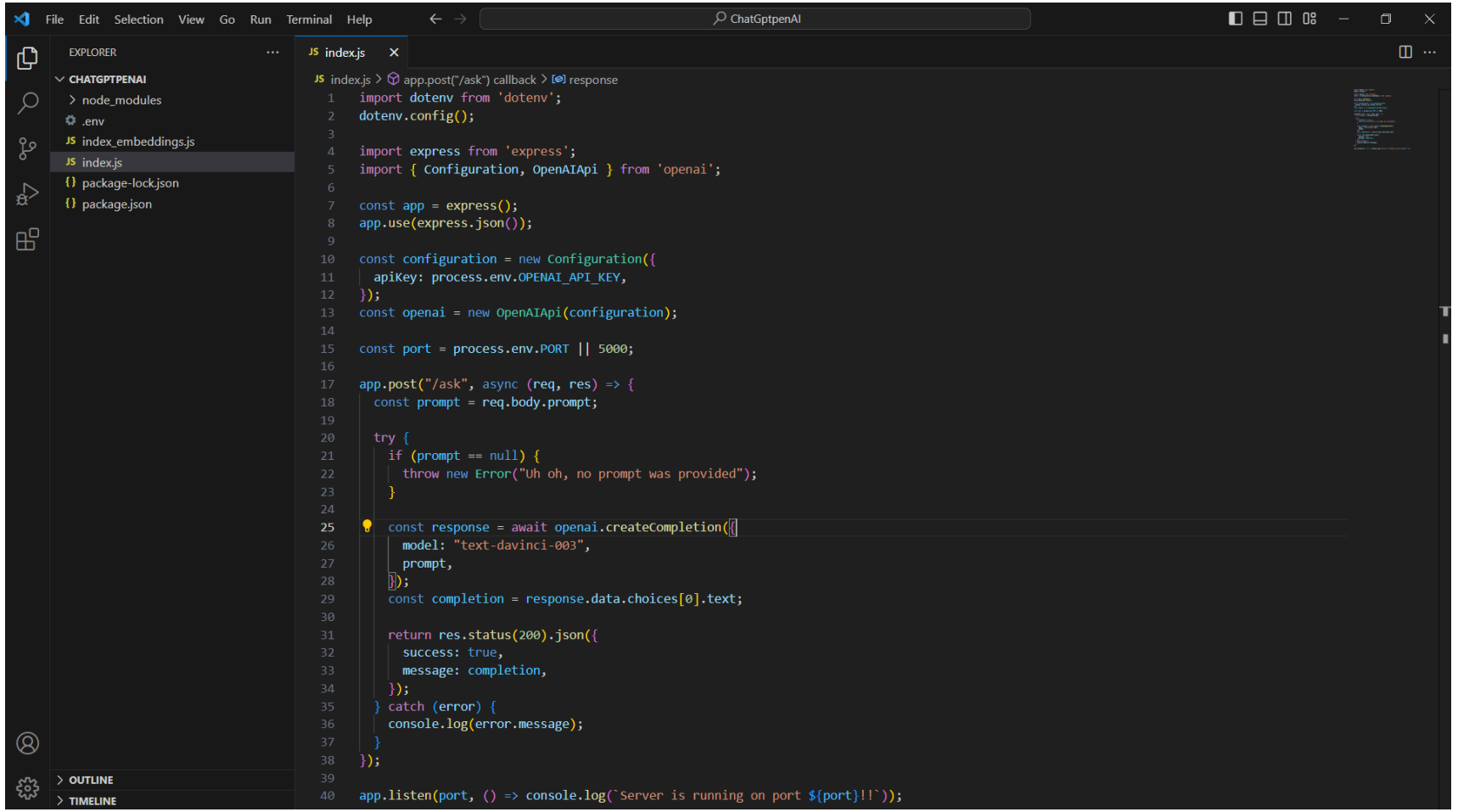
Pre-requisites for the assignment:

- ❑ OpenAI API Key
- ❑ Air Table Base Id
- ❑ Air Table API Key

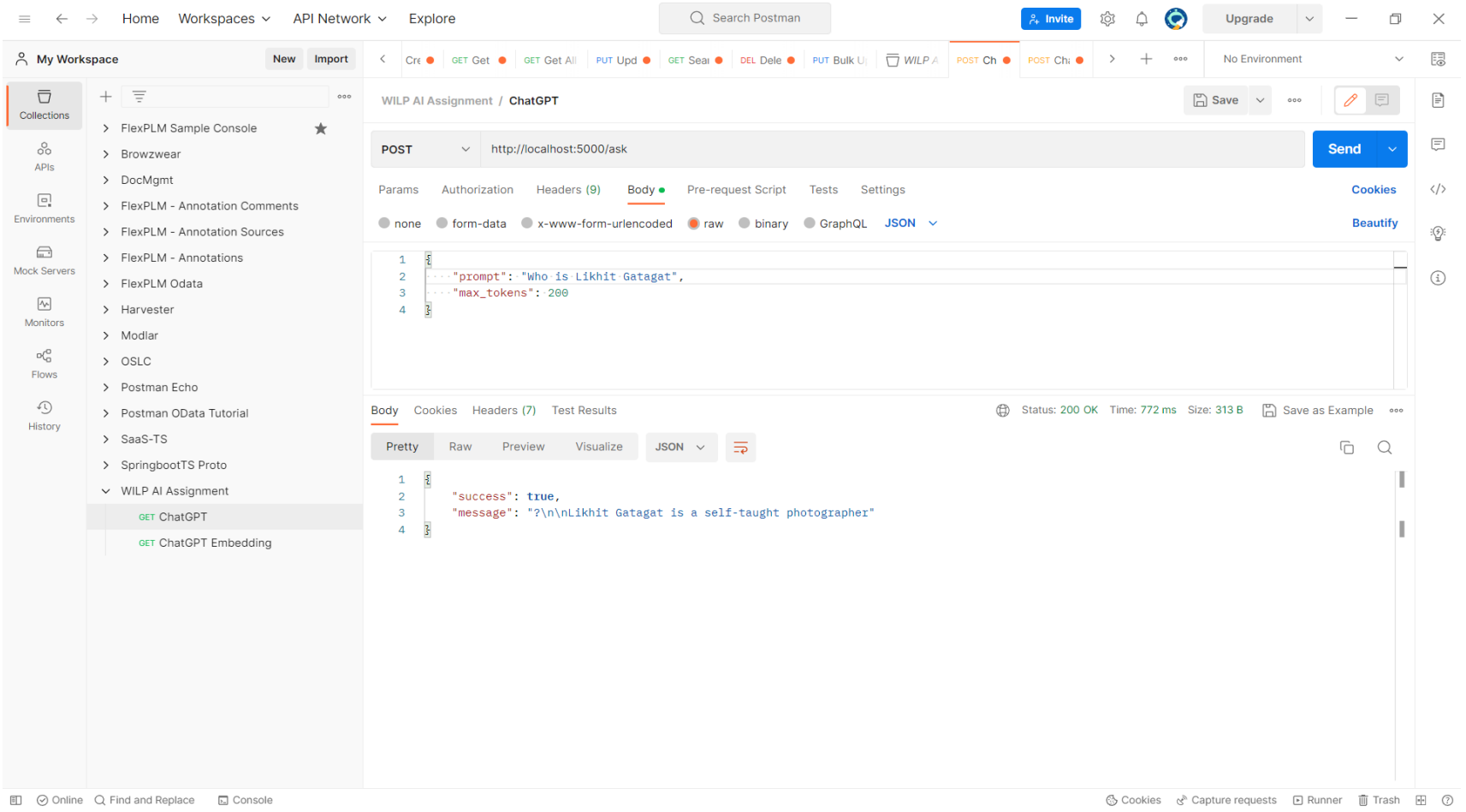
Step 1 - Integrating OpenAI with Node.js/Express

- **Objective:** Communicate with OpenAI API to generate responses.
- **Implementation:** Utilized Node.js and Express to set up an endpoint for sending requests to OpenAI.
- **Key Points:**
  - Backend Setup: Created a Node.js/Express server (index.js) to handle API requests.
  - OpenAI Integration: Configured an endpoint to communicate with OpenAI's API.
  - Random Response Generation: Utilized the OpenAI API to generate random responses based on prompts.
- **Impact:** Facilitated seamless communication between the backend and OpenAI's powerful language processing capabilities.

Step 1 – Code Snippet



Step 1 – OpenAI Integration In Action



Step 2 – Front-end Setup with Vite + React

- **Objective:** Develop a front-end interface to interact with OpenAI for AI-driven responses.

- **Technologies Used:**
  - Front-end Framework: Vite + React
  - Other Technologies: HTML, CSS

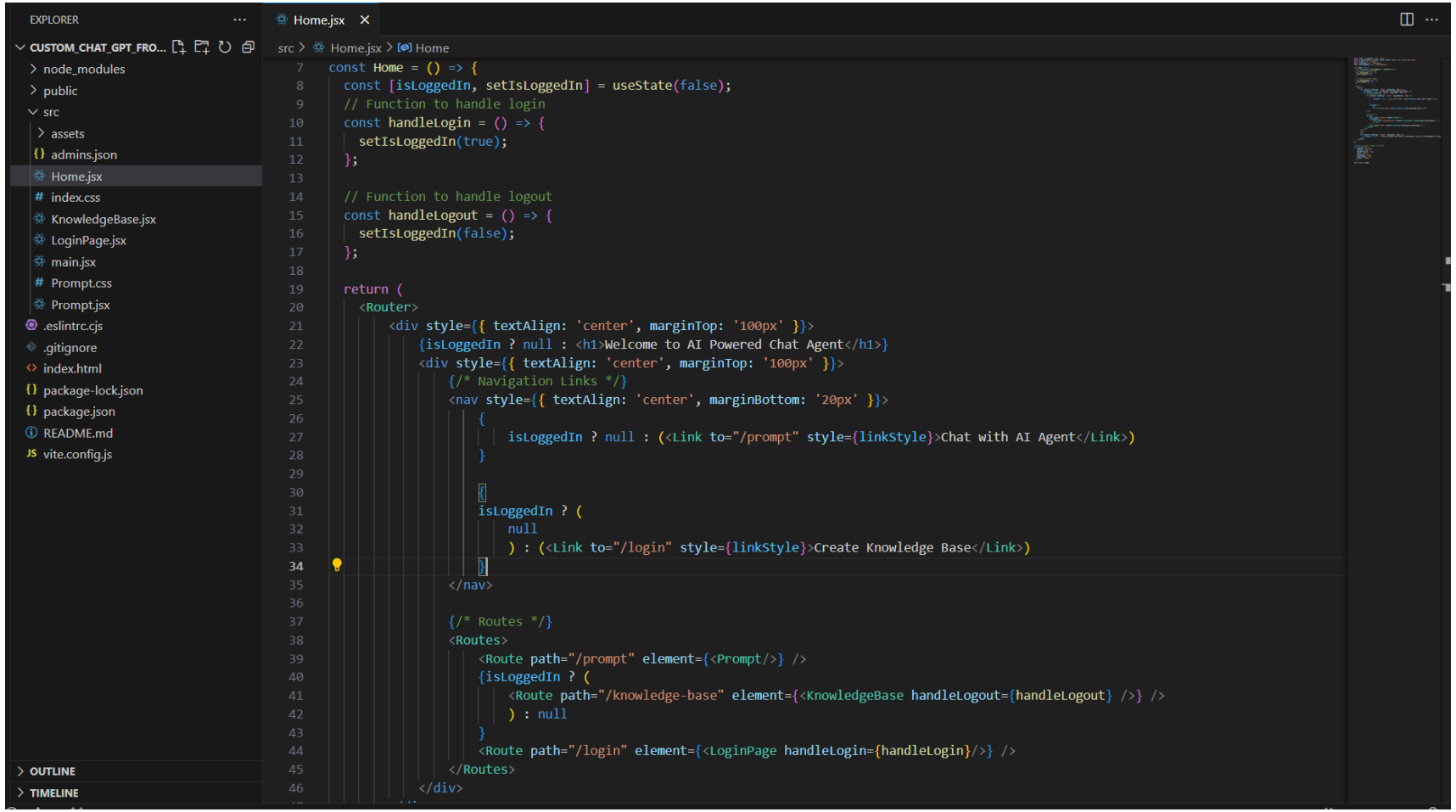
- **Components and Navigation Logic:**
  - Home Page: Serves as the launch point of the application.
  - Navigation Links: "Chat with AI Agent" and "Create Knowledge Base" are two main links.

- **Functionality:**
  - Chat with AI Agent: Opens a prompt box for user queries, and fetches responses from OpenAI on enter.
  - Create Knowledge Base: To be discussed in the context of Embeddings.

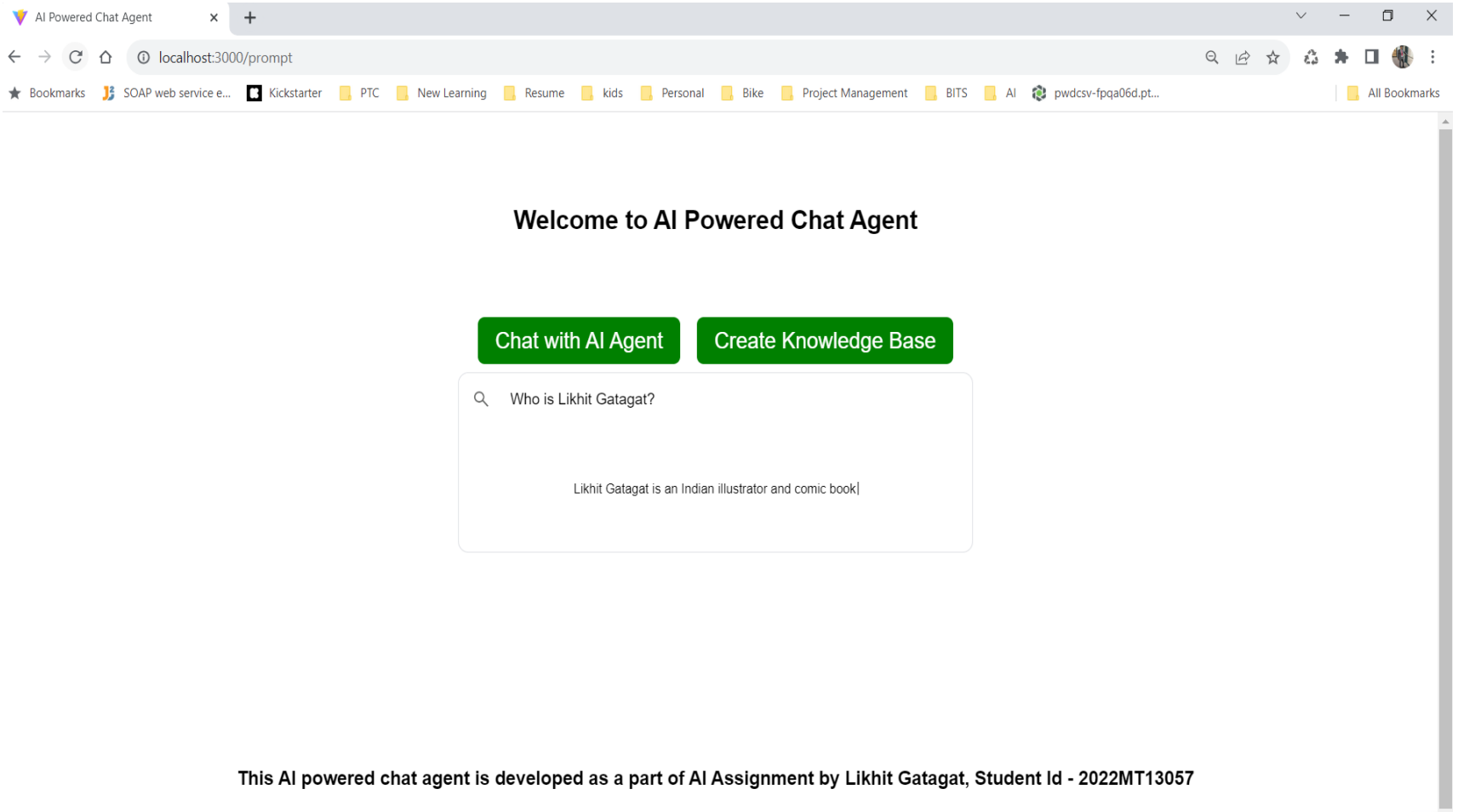
- **Implementation:**
  - Vite + React: Utilized Vite as a build tool and React for UI components.
  - HomePage: Manages navigation and links to different functionalities.
  - Prompt: Handles the interaction with OpenAI for user queries.

- **Impact:** Created an intuitive front-end allowing users to engage with OpenAI for query-based responses.

Step 2 – Code Structure Overview



Step 2 – UI Screens



Integrated AI Agent with PDF Upload and Context Setting

- **Objective:** Integration of PDF upload, context setting, and AI agent for enriched chatbot responses.

- **Process Overview:**
  1. **Customizing Chatbot:**
     - ❑ Used PDF content to enhance the chatbot's knowledge base.
     - ❑ Utilized OpenAI's text embeddings for measuring relatedness between text strings.

  2. **Embedding Model API Usage:**
     - ❑ Generated embeddings for PDFs using OpenAI's embedding model API.
     - ❑ Stored embeddings in AirTable Base for domain-specific responses.

  3. **Cosine Similarity for Relevance:**
     - ❑ Computed cosine similarity between prompt embeddings and stored database embeddings.
     - ❑ Identified the most similar text using the highest cosine score.

  4. **Super Prompt Construction:**
     - ❑ Constructed a "super" prompt by embedding the most relevant text into a query.
     - ❑ Sent the refined prompt to OpenAI for a more contextually relevant answer.

Continued…

- **Features Integrated:**
  1. **PDF Upload Functionality:**
     - ❑ Users can browse and upload multiple PDFs within the application.
     - ❑ Uploaded PDFs are utilized to enrich the chatbot's knowledge base.
  2. **Embedding Model API Usage:**
     - ❑ Users have the ability to set the context for the conversation.
     - ❑ Contextual information from uploaded PDFs guides the AI agent's responses.
  3. **AI Agent Integration:**
     - ❑ The AI agent utilizes the supplied PDFs as a knowledge base for answering queries.
     - ❑ Responses are generated based on the collective information from uploaded PDFs.
  4. **GUI Description:**
     - ❑ **User Interface:** A user-friendly interface allowing seamless PDF uploads and context setting.
     - ❑ **Interaction Flow:** Users navigate, upload PDFs, set context, and engage with the AI agent for responses.

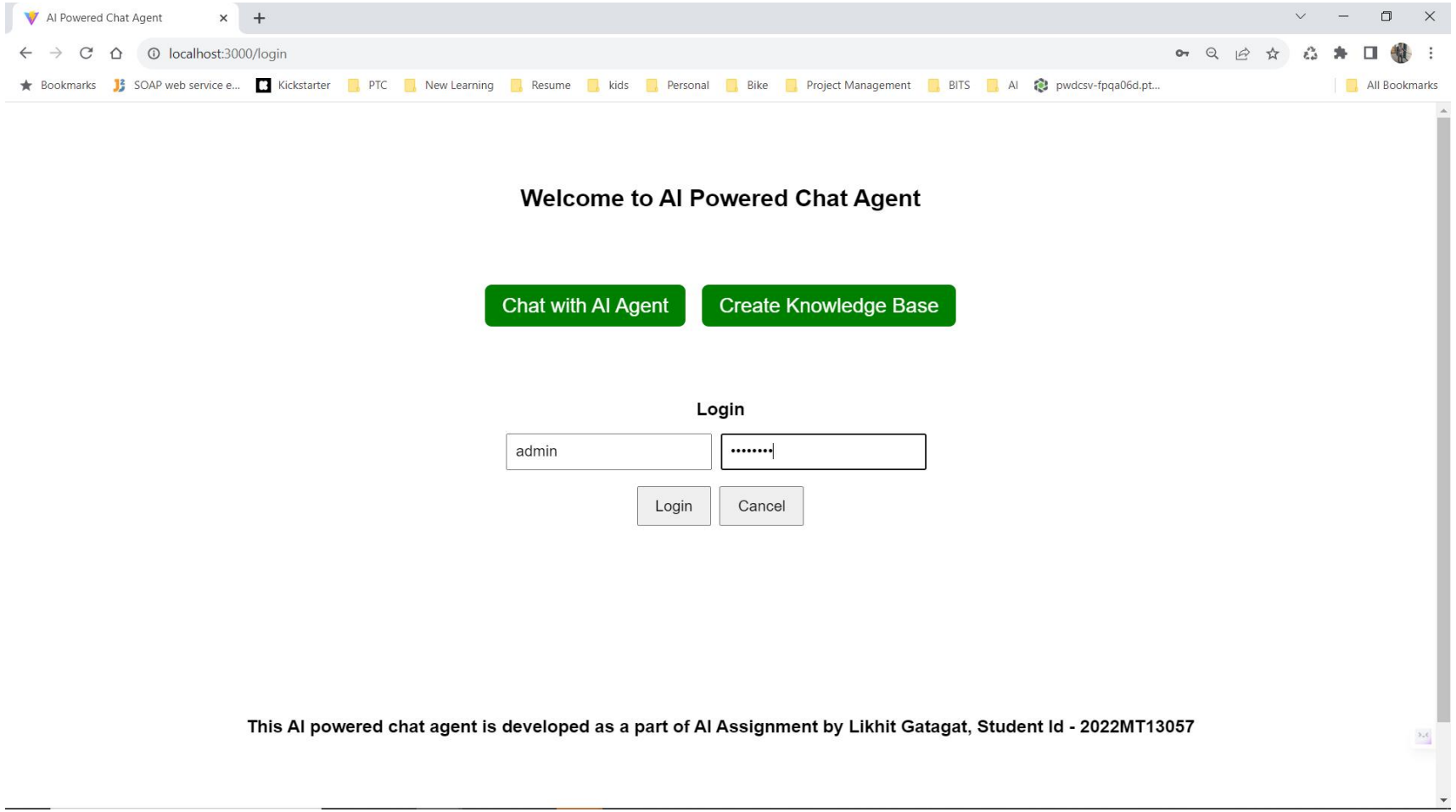
Example Scenario

- **User Query:** "Who is Likhit Gatagat"

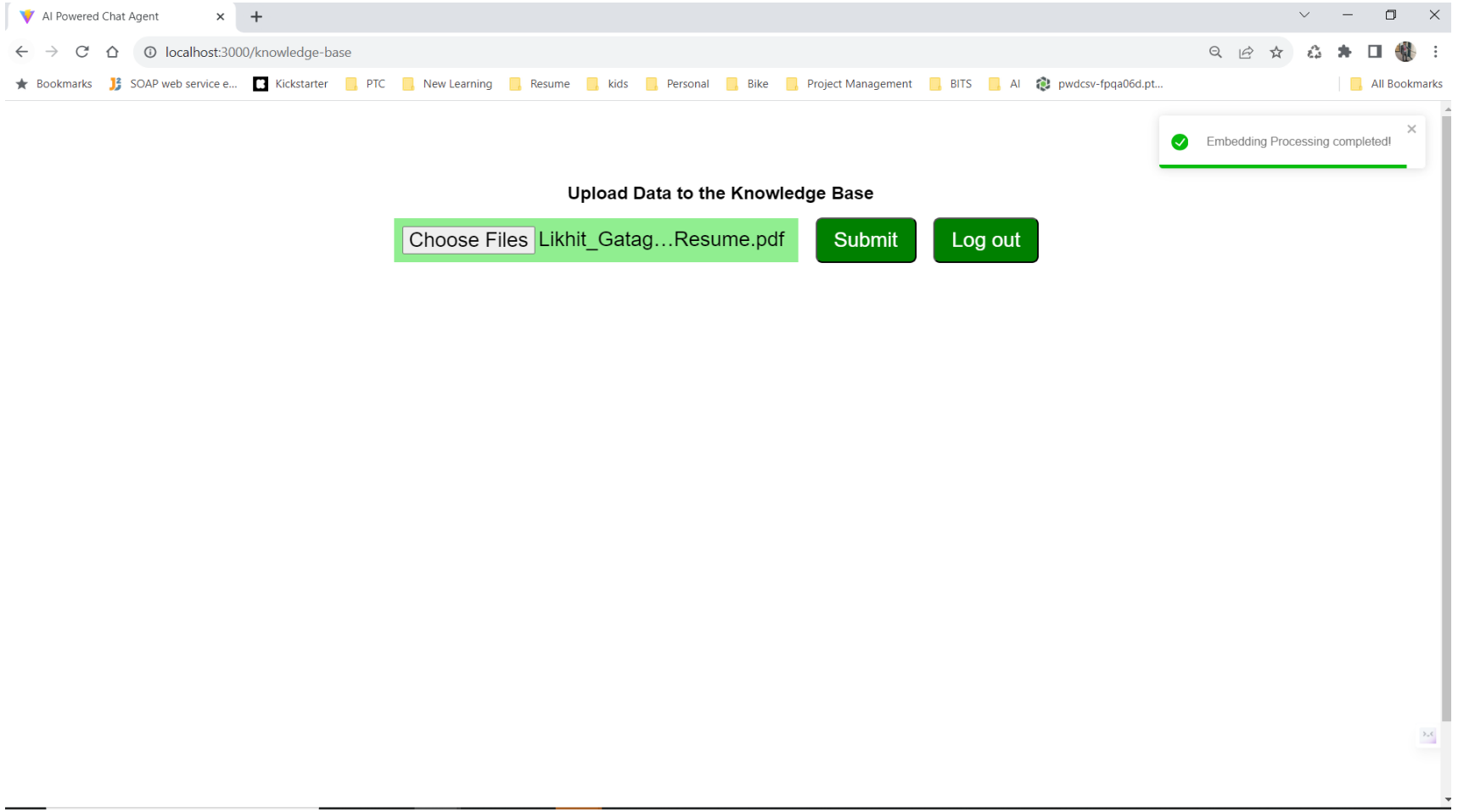
- **Process Flow:**
  - Received embeddings data for the specific prompt.
  - Computed cosine similarity with database embeddings.
  - Constructed a refined prompt for OpenAI based on the most relevant text.
  - Sent the refined prompt and obtained a more contextually relevant response.

- **Result:**
  - **Enhanced Relevance:** Obtained more relevant responses by incorporating contextually relevant prompts.
  - **Expanded Knowledge Base:** Enabled users to contribute by uploading PDFs, enriching the chatbot's knowledge.

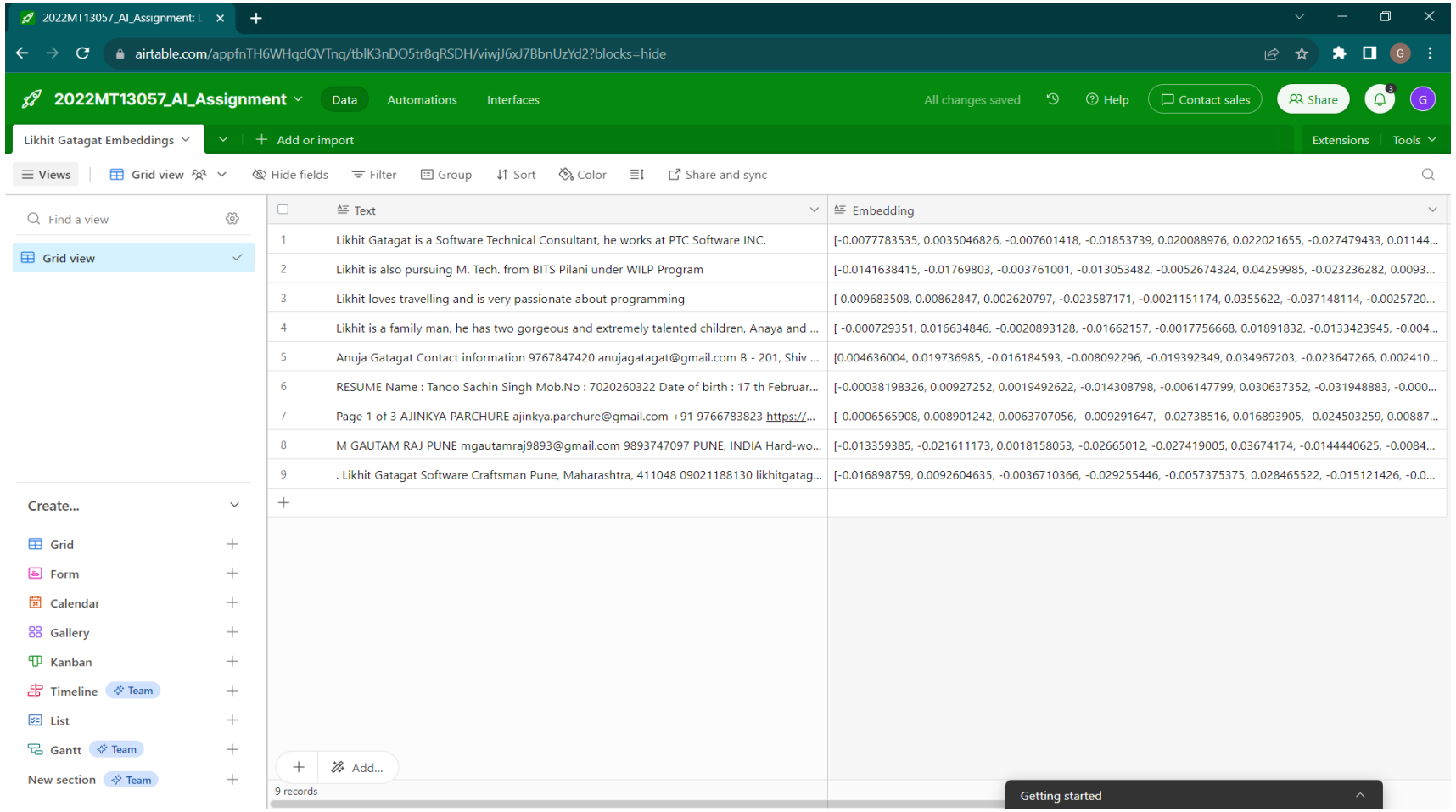
Step 3 – UI Screens – User Login



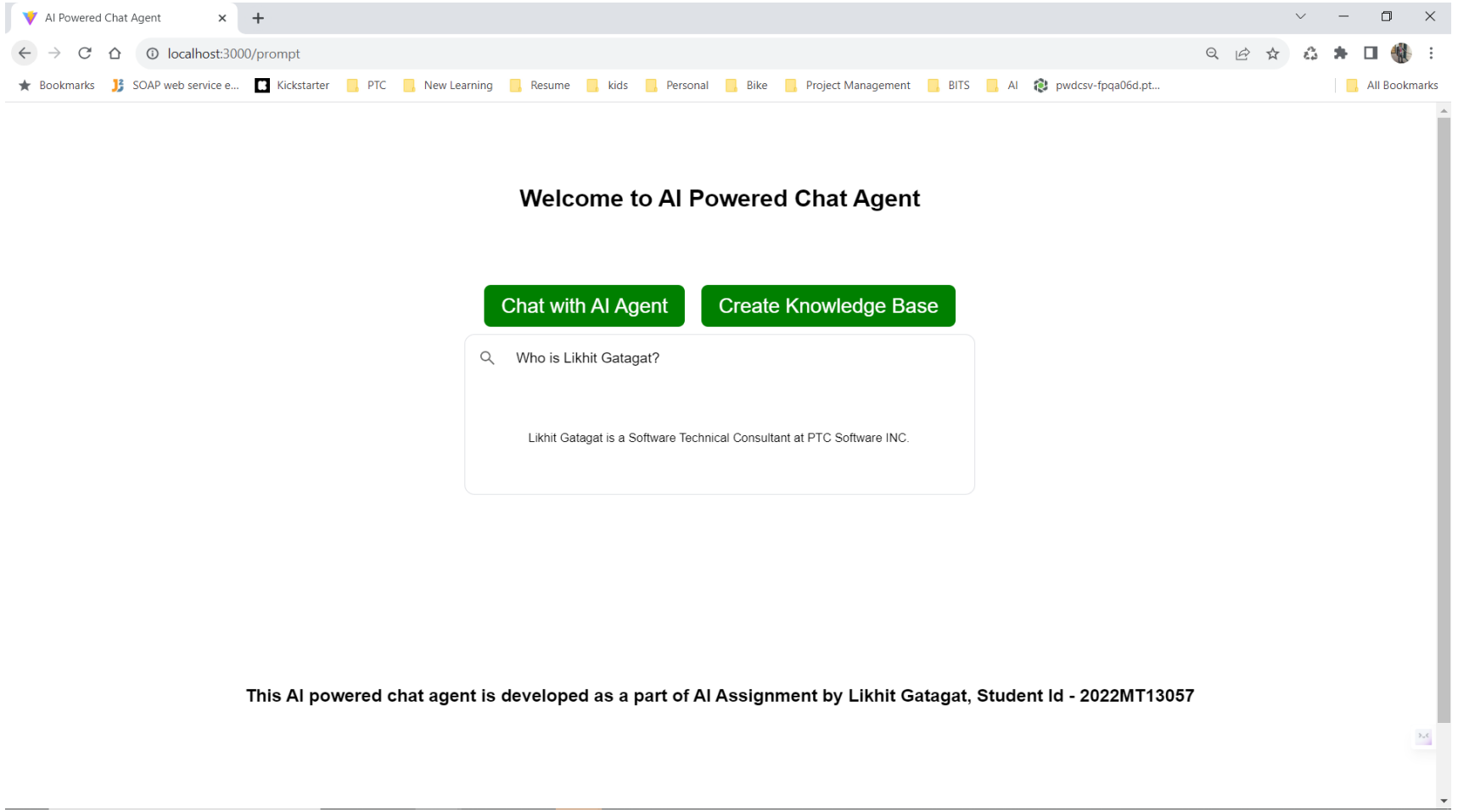
UI Screens – PDF Upload, Update Knowledge Base



UI Screens – Embeddings Database (Excel)



UI Screens – Relevant response as per knowledge base



Demo

[Link to the Demo]

Source Code

[Link to the Source Code]


THANK YOU FOR
YOUR ATTENTION