**who is MosaicBot?**

MosaicBot is the AI chatbot that you're creating! It's designed to answer questions specifically about **you**, Raiyan Yasawy. The idea is that MosaicBot will provide information related to your professional background, skills, education, experience, goals, and other personal details that you've inputted into the system.

Here’s how MosaicBot works:

* **Purpose**: MosaicBot is a chatbot that answers questions based on the information you’ve provided (like your name, skills, education, experience, contact info, etc.).
* **Limitations**: It will only respond with answers based on the data you've given it. For example, if someone asks about your programming skills or previous job roles, it will provide relevant details about that. Casual questions like "Hi" or "How are you?" will be politely redirected back to talking about your professional background.

In short, MosaicBot is a personalized assistant created to answer questions about your professional self, designed to help recruiters, employers, or anyone interested in learning more about you.

**what did we do in this project and what are the techs that we used**

In this project, we created a personalized chatbot called **MosaicBot**, designed to answer questions specifically about you (Raiyan Yasawy). The bot responds with professional details such as your skills, education, experience, goals, and contact information. It uses both predefined data you provided and integration with OpenAI's GPT model to answer dynamic queries while staying focused on professional topics.

**Key Steps in the Project:**

1. **Data Structure**:
   * We started by defining a userData object that holds your personal and professional information (skills, education, experience, goals, etc.). This is the core data that MosaicBot uses to generate responses.
2. **Chatbot Interaction**:
   * We created a chat interface with HTML, CSS, and JavaScript.
   * The user inputs a question in a text box, and the bot responds with a message from predefined data.
3. **Greeting & Casual Conversation Handling**:
   * We handled basic greetings ("Hi", "How are you?", etc.) by providing polite responses but guiding the user back to professional questions.
4. **API Requests**:
   * We configured API calls to OpenAI for professional responses, making the bot more flexible in answering a range of questions, all while staying within the scope of your personal and professional details. // cancelled.
5. **Styling**:
   * We used basic **CSS** to style the chatbot and make it visually appealing and user-friendly.
6. **Server Setup**:
   * The project includes a basic **server.js** file to handle API requests, especially if we plan to deploy the chatbot on a server and enable backend communication with OpenAI.

**Technologies Used:**

* **HTML**: For the structure of the chatbot, including the input field and message display area.
* **CSS**: For styling the chatbot interface and making it visually appealing.
* **JavaScript**: For handling the logic of user input, generating bot responses, and integrating with the OpenAI API.
* **Node.js (Server.js)**: Node.js handles server-side operations, like making API calls. It can also be used for routing and other backend functionality (optional for deployment).
* **Fetch API**: To send requests to the API and receive responses asynchronously.