**C-Realization: Project Implementation and Development**

**Tech Stack Overview**

The French Learning Chatbot was developed using the following technologies:

1. **Backend**: Flask (Python), handling server requests and API integration.
2. **Frontend**: HTML, CSS, and JavaScript to create a user-friendly interface.
3. **AI Integration**: OpenAI’s GPT-3.5-turbo model for generating responses.

**Backend Implementation**

The Flask backend is the core of the project, managing user sessions and interacting with the OpenAI API. The key components include:

* **Session Management**: The chatbot uses Flask’s session capabilities to store and maintain conversation history.
* **OpenAI Integration**: Incoming user messages are sent to the GPT model using a structured JSON request, and the generated response is parsed and returned to the frontend.

**Frontend Design**

The user interface consists of a chat window for displaying messages, an input box for typing, and buttons for sending messages and clearing the conversation. The design prioritizes simplicity and usability.

* **JavaScript Logic**: JavaScript is used to send and receive messages via HTTP requests to the Flask server and dynamically update the chat window.

**Integration with OpenAI API**

The OpenAI API is integrated using the openai Python library. Each user message is processed and sent to the GPT model with a defined structure:

*response = openai.ChatCompletion.create(*

*model="gpt-3.5-turbo",*

*messages=session['conversation\_history']*

*)*

The response is then formatted and appended to the session history, allowing the chatbot to maintain context across multiple exchanges.

**Functionality Achieved**

* **Real-Time Interaction**: Users receive immediate responses based on their inputs.
* **Session Management**: The chatbot maintains a conversation history to provide context-aware responses.
* **Clear Chat**: Users can clear their session history and start a new conversation at any time.