**Server Documentation**

**1. Introduction**

**Purpose**

The server acts as the backend for the Chrome extension chatbot, facilitating communication between the extension frontend and external APIs. It provides Q&A functionality, generates suggested questions based on webpage content, and ensures context-aware responses.

**Technologies Used**

* **Python**: Programming language used for server development.
* **Flask**: Microframework for handling web requests and APIs.
* **Flask-CORS**: Extension for handling Cross-Origin Resource Sharing (CORS) in Flask applications.
* **langchain**: Library for natural language processing tasks such as text splitting and embeddings.
* **ObjectBox**: Vector storage for managing document embeddings.
* **Ollama**: Language model used for generating responses and suggestions.

**2. Installation**

**Prerequisites**

* Python 3.7+
* pip (Python package installer)

**Setup Instructions**

1. **Clone the Repository:**

git clone https://github.com/Joel-Mathew-777/chat-bot.git

cd chat-bot/backend

1. **Install Dependencies:**

pip install -r requirements.txt

**3. Usage**

**Starting the Server**

To start the server locally:

python server.py

The server will run on http://localhost:5000 by default.

**4. API Documentation**

**API Endpoints**

**/ask Endpoint**

The /ask endpoint handles the chat functionality (Q&A) with the user.

* **Method:** POST
* **Request Body:**

{ "question": "your question" }

**question**: A string containing the user's question related to the current webpage content.

* **Response:**

{ "response": "AI's answer" }

**response**: A string containing the AI's answer to the user's question.

**/getUrl Endpoint**

The /getUrl endpoint checks the current URL where the context is generated for the model.

* **Method:** POST
* **Request Body:**

{ "url": "webpage url" }

**url**: A string representing the webpage URL to be checked.

* **Response:**

{ "match": "true/false" }

**match**: A boolean value (true or false) indicating whether the provided URL matches the current context URL.

**/start-chat Endpoint**

The /start-chat endpoint initializes the chat and generates suggested questions based on the provided webpage URL.

* **Method:** POST
* **Request Body:**

{ "url": "webpage url" }

**url**: A string representing the webpage URL for which the chat is initialized and questions are generated.

* **Response:**

{

"message": "Conversational chain created",

"chunk\_count": number,

"suggested\_questions": ["Question 1", "Question 2", "Question 3"]

}

**message**: A string confirming the creation of the conversational chain.

**chunk\_count**: An integer indicating the number of text chunks generated from the webpage content.

**suggested\_questions**: A list of strings representing suggested questions generated based on the webpage content.

**5. Development**

**Code Structure**

The server application is structured as follows:

* server.py: Main application file containing Flask routes and logic.
* requirements.txt: List of dependencies.

**Key Components**

* **Flask App**: Initializes Flask application and defines API routes.
* **Langchain Integration**: Handles natural language processing tasks.
* **ObjectBox Database**: Stores and retrieves document embeddings.
* **Ollama Language Model**: Used for generating responses and suggestions.

**Development Workflow**

* Follow coding standards and best practices.
* Use version control (e.g., Git) for managing codebase changes.
* Conduct code reviews and testing to ensure functionality and performance.

**Using the chrome extension**

**Installing the extension**

**Setup Instructions**

**Clone the Repository:**

git clone https://github.com/Joel-Mathew-777/chat-bot.git

cd chat-bot/Chat Bot Application

**Add the extension in chrome:**

Open **chrome extension manager**:

Option 1: type chrome://extensions in the url bar and press enter.

Option 2: click on the tree dots in the top right of the browser, then click "More tools" then click "Extensions".

**Activate developer mode**

Turn on the switch on the top right of the page that says "Developer mode";

**load unpacked extension**

Click on the button on the top left of the page that says "Load unpacked".

Browse to where the repository is being cloned and select the Chat Bot Application folder.

**Accessing the Extension**

1. **Open the Extension:**
   * At the top right of your Chrome browser window, click on the Extensions icon (puzzle piece).
   * Look for **ChatBot Extension** and click on it to open the popup window.
2. **Start Chat:**
   * In the extension popup window, click on the button labeled **Start chat**.
3. **Learning the Webpage:**
   * The extension will redirect to a new page displaying "Learning the webpage".
   * Wait for the extension to process the webpage content.
4. **Chat Window with Suggested Questions:**
   * Once processing is complete, the extension will display a chat window with suggested questions based on the webpage content.
5. **Chatting with the Bot:**
   * You can now interact with the chatbot:
     + Type your own question into the chat window and press Enter to receive an answer.
     + Alternatively, click on one of the suggested questions displayed to get an immediate response.