Chatbot Web Project Documentation

1. Project Requirements

Software & Accounts:

- Node.js: JavaScript runtime environment required to run backend code and development tools on your computer.

- npm: Node.js package manager (comes with Node.js) used to install project dependencies.

- Vercel account: Required for deploying your project to the Vercel platform and for using the Vercel CLI for local development.

- OpenAI API key: Needed to access OpenAI's language models for generating chatbot responses.

Project Files:

- index.html — The frontend (chat UI)

- api/chat.js — The backend (Vercel serverless function for chat)

- package.json — Project dependencies

- .env — (You need to create this) Contains your OpenAI API key:

OPENAI\_API\_KEY=your\_openai\_api\_key\_here

------------------------------------------------------------

2. How to Set Up and Run Locally

Step-by-Step Setup:

1. Clone or download the project to your computer.

2. Install dependencies:

npm install

3. Install Vercel CLI globally (if not already):

npm install -g vercel

4. Add your OpenAI API key:

- Create a file named .env in your project root.

- Add this line (replace with your actual key):

OPENAI\_API\_KEY=sk-xxxxxxxxxxxxxxxxxxxx

5. Run the project locally:

vercel dev

- The first time, you’ll log in and set up the project.

- After that, just run vercel dev to start the local server.

6. Open your browser to http://localhost:3000 to use the chatbot.

------------------------------------------------------------

3. How the Web Chatbot Works

Frontend (index.html):

- Displays a chat interface (messages, input box, send button).

- When you type a message and hit send:

1. The message is shown in the chat window.

2. The message (and a session ID) is sent to the backend at /api/chat using fetch.

3. When a response is received from the backend, it’s displayed as a bot message.

Backend (api/chat.js):

- Receives POST requests at /api/chat with { message, sessionId }.

- Stores conversation history in memory (per session).

- Sends the conversation to OpenAI’s API (using your API key).

- Gets the AI’s response and sends it back to the frontend.

Session Handling:

- Each user gets a unique session ID (stored in their browser’s localStorage).

- This keeps conversation context for each user.

------------------------------------------------------------

4. How to Deploy to Vercel

1. Push your project to GitHub (or another git provider).

2. Connect your repo to Vercel (via the Vercel dashboard).

3. Set your OpenAI API key in Vercel’s environment variables.

4. Deploy!

- Vercel will serve your static files and run your /api/chat.js as a serverless function.

-----------------------------------------------------------

5. Summary Diagram (Text Version)

User types message and clicks Send ->

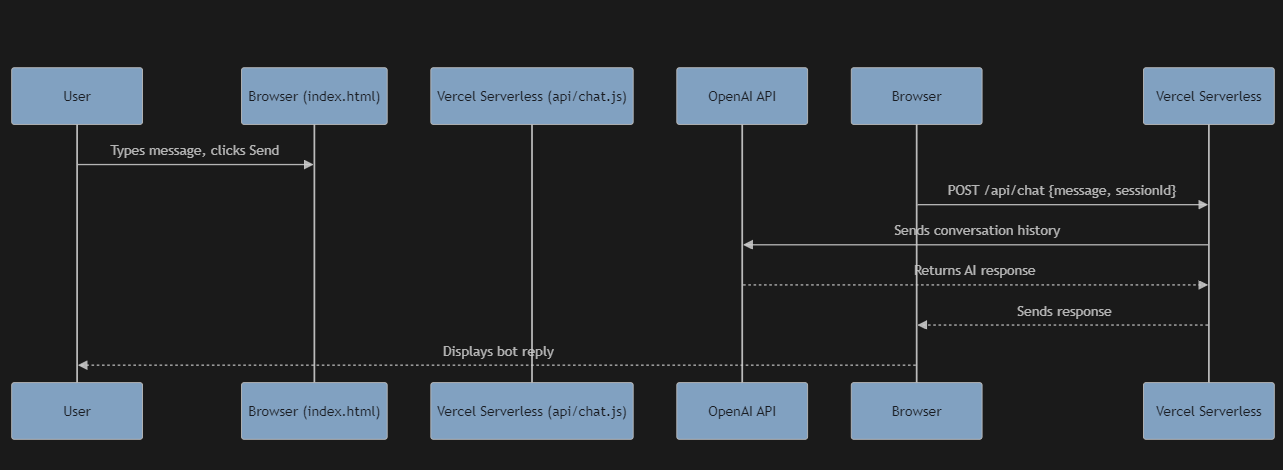
Browser (index.html) sends POST /api/chat {message, sessionId} ->

Vercel Serverless (api/chat.js) sends conversation to OpenAI API ->

OpenAI API returns AI response ->

Vercel Serverless sends response to Browser ->

Browser displays bot reply to User



------------------------------------------------------------

6. Key Points

- All you need to run locally is Node.js, npm, Vercel CLI, and your OpenAI API key.

- The frontend and backend are integrated: the frontend sends messages to the backend, which talks to OpenAI and returns responses.

- Vercel makes it easy to deploy and run both static frontend and serverless backend together.