Product Specification Document

Overview

The code implements a conversational web application with persistent chat history. Users can have multiple conversations with different AI models.

Key Components

- **Backend** - Flask server, handles API routes and OpenAI requests
- **Frontend** - HTML, CSS, JS, implements UI and client-side logic
- **Database** - Stores conversation data like message history

Main Files

app.py - Flask server code

- API routes for conversations, chat endpoints
- Makes requests to OpenAI API
- Manages conversation state and history

static/js/main.js - Client-side JS

- Makes API calls to backend to load conversations, submit input
- Displays UI elements like messages, chat box, conversation list
- Handles user interactions like sending messages

templates/chat.html - Conversational web app UI

- Renders chat log, composer, conversation list
- Dynamic elements updated by main.js

Additional Files

- **static/css/styles.css** - CSS styling
- **static/js/prism.js** - Syntax highlighting
- **static/js/marked.min.js** - Markdown rendering

Key Functionality

- **Persisted conversations** - Chat sessions saved in database
- **Real-time chat** - User messages sent to AI, responses in real-time
- **Conversation management** - CRUD operations
- **Multiple AI models** - Switch between different assistants
- **Message history** - Entire chat logs available
- **Interactive UI** - Chat box, messaging, conversational flow

Key Functions

app.py

- `home()` - Renders chat interface
- `chat()` - Handles chat logic and OpenAI requests
- `get_conversations()` - Fetches all conversations
- `get_conversation()` - Fetches single conversation
- `update_conversation_title()` - Updates title of conversation
- `delete_conversation()` - Deletes a conversation

main.js

- `updateConversationList()` - Loads conversation list
- `loadConversation()` - Loads a specific conversation
- `showConversationControls()` - Shows conversation controls
- `modelNameMapping()` - Maps model names to display text
- `sendMessage()` - Sends user message
- `renderOpenAI()` - The OpenAI Playground accent for UI modifications.

Additional

- `generate_summary()` - Generates conversation title