System Documentation: Chat 8 UI

Application Name: Chat 8 (Version 7.x series)
Project Lead: Mike Holland
System Architect: Gemini Pro

1. Overall Purpose

The Chat 8 UI is a comprehensive, single-page web application (SPA) designed to serve as the primary user interface and control center for a sophisticated, multi-workflow RAG (Retrieval-Augmented Generation) memory system. Its core purpose is to provide a seamless and intuitive user experience for managing complex AI projects, enabling users to conduct conversations, manage long-term memory, and switch between distinct project contexts, all while orchestrating communication with a suite of backend n8n workflows.

2. Core Functionality & User Scenarios

The UI is built around two primary user scenarios, which are supported by a robust set of features:

- **Scenario 1: Live Conversation & Memory Commitment**
    - **Create Simple Project:** Users can initiate a new, empty project with a unique name. The UI triggers a backend workflow to create the necessary database records and returns the session IDs.
    - **Live Chat:** Users can engage in a standard, turn-by-turn conversation with the AI.
    - **Commit to Memory:** At any point, the user can click the "Commit to Memory" button. This triggers a backend workflow that processes the recent conversation, distills key facts, and saves them to the project's permanent, long-term memory store (rag_store).
- **Scenario 2: Bootstrap Memory from External Source**
    - **Create Project from URL:** Users can create a new project by providing a URL to an external JSON file containing a pre-existing conversation or knowledge base.
    - **Backend Indexing:** The UI triggers a dedicated RAG Indexing Workflow that fetches, chunks, embeds, and saves the entire content of the file to the new project's long-term memory.

- **Immediate Retrieval:** Upon completion, the UI loads the new project, allowing the user to immediately begin asking questions and retrieving information from the freshly indexed knowledge.

3. Key Architectural Features

a. State Management

The UI's most critical architectural feature is its client-side state management system.

- **Global State Variables:** A set of live JavaScript variables (currentProjectId, currentChatId, currentRagId) holds the active session's identifiers.
- **Session Persistence:** The application state is persisted across page reloads using the browser's sessionStorage. A robust saveState() and loadState() utility system, complete with error handling for data corruption, ensures a stable user experience.
- **State Synchronization:** All major user actions (creating a new project, loading an existing one) trigger functions that explicitly update both the live global variables and the persisted sessionStorage, ensuring data consistency.

b. Workflow Orchestration

The UI acts as the central orchestrator, making POST requests to a suite of distinct n8n webhook URLs. It is responsible for:

- Constructing and sending the correct JSON payloads for each specific action, adhering to a strict **Master Data Contract**.
- Handling the JSON responses from the workflows, whether it's a simple AI reply, a list of projects, or a set of new session IDs.
- Displaying status messages and handling errors gracefully.

c. User Interface Components

- **Project Launcher:** The initial screen that allows users to choose between starting a new project or continuing an existing one.
- **Project Creation Forms:** Separate UI sections for creating a simple project or creating one from a URL.
- **Project Selection Dropdown:** A dynamically populated <select> menu that lists all existing projects, allowing users to resume previous sessions.

- **Main Chat Container:** The core conversational interface, including the message display area, text input, and action buttons ("Send," "Commit to Memory," etc.).
- **Configuration & Management:** A set of buttons ("New Session," "Clear Chat," "Dashboard," "Config") provides access to higher-level management functions.