System Documentation: Setup Workflow - Project Launcher

Workflow Name: Setup Workflow: Project Launcher for Documentation
Source File: Setup_Workflow__Project_Launcher_for_Documentation.json
Project Lead: Mike Holland
System Architect: Gemini Pro

1. Overall Goal

This workflow serves as the primary "receptionist" and setup engine for the Chat 8 UI. It is a dual-function workflow that automates both the creation of new, simple projects and the retrieval of the complete list of existing projects. Its core purpose is to manage the project_contexts table, ensuring each project has a unique set of identifiers before handing control back to the UI.

2. Key Components & Architecture

- **Trigger:** A unique n8n **Webhook** (...a61a290c...) listens for POST requests from the Chat 8 UI's initial launcher screen.
- **Database:** It interacts exclusively with the project_contexts table in the Postgres database to create new project records and retrieve the full project list.
- **Logic:** The workflow uses an If node to act as a switch, directing the incoming request down one of two distinct paths based on an action property in the request body.

3. Step-by-Step Data Flow

The workflow begins with a single entry point and immediately branches based on the user's intent.

1. **Webhook3:** The workflow is triggered by a POST request from the UI. The payload contains a JSON body with an action key, which will be either create_project or list_projects.

Branch A: "List Existing Projects"

This is the True path of the If1 node.

1. **Get Project List:** A Postgres node executes a SELECT query to retrieve the project_name, chat_session_id, rag_session_id, and xata_id for all records in the project_contexts table.

2. **If1:** The workflow logic for this branch is slightly counter-intuitive. The If node checks if the result of the Get Project List query has more than 0 items. If True, it proceeds down this "List Projects" path.
3. **Code3:** A Code node takes the list of projects from the database and transforms it to match the data contract expected by the UI, ensuring all necessary keys (project_name, session_id, project_id, rag_session_id) are present.
4. **Respond to Webhook4:** The final Respond to Webhook node sends the formatted list of projects back to the UI, which then uses it to populate the project selection dropdown.

Branch B: "Create New Project"

This is the False path of the If1 node.

1. **Code2:** A Code node receives the projectName from the webhook. It generates a new, unique chat_session_id and rag_session_id based on the project name and the current timestamp.
2. **Postgres1:** This Postgres node executes an INSERT query, using the data from the Code2 node to create a new record in the project_contexts table. It uses RETURNING xata_id to get the permanent project ID back from the database.
3. **Respond to Webhook5:** The final Respond to Webhook node sends a success response back to the UI, providing it with the newly created chat_session_id and rag_session_id so it can initialize the new chat session.