Expense Tracker

This project is a simple, single-page web application built to help an individual user track their personal financial transactions, classifying them as either income or expense. The interface has a clean, neon-themed design defined in style.css.

1. Core Assumptions

The application is built on the assumption that a dedicated backend is required, which is provided by Supabase. This means all user accounts and transaction records are stored securely in the cloud. Critically, the system is designed to use the user's ID (user_id) to ensure that users can only see their own data in the database (relying on Supabase's Row Level Security). A fixed list of transaction categories (like "Food," "Salary," "Bills") is predefined in the dashboard's HTML.

2. Design and Flow

The application follows a two-part flow:

Authentication Flow: The user starts at index.html. The auth.js script handles the sign-in logic, using the Supabase Auth API to verify credentials. Upon successful login, the user is immediately redirected to the main dashboard.html. If the user's session expires later, the dashboard.js script forces them back to the login page.

Dashboard Flow: The dashboard.html page is the main hub. The dashboard.js script handles all activity here:

- **Fetching Data**: Immediately after login, the script calls the fetchRecords() function to pull all the user's historical transactions from Supabase.
- **Summary**: The updateSummary() function runs to calculate the Total Income, Total Spending, and Net Balance using all the fetched data, displaying the result prominently.
- **Display and Filtering**: The renderTable() function builds the main transaction list. It is dynamic, meaning it can filter the displayed records based on a selected Category or a Date Range (From/To) chosen by the user in the top bar.
- **CRUD**: The script supports full data management:
    - New transactions are saved using the addRecord function.
    - Existing records can be modified using the startEdit and saveEdit functions, which trigger a modal for data entry.
    - Transactions can be permanently removed using deleteRecord.

3. Sample Interactions

When a user adds a new record, such as a $5000 Income classified as "Salary," the system inserts the data into the database and immediately updates the main table and the financial summaries. Similarly, if the user applies a filter for "Food" expenses in March, the renderTable function will recalculate and show only those specific items, providing a focused view of spending.