feedback portal

use of SQL

In this Admin Dashboard project, SQL (Structured Query Language) can be integrated to handle all backend data management tasks. Currently, the project is built using HTML, CSS, and JavaScript, which means the interface is static and does not store user inputs. By connecting this dashboard to a server-side backend (using technologies like PHP, Node.js, or Python Flask) and a relational database (like MySQL or PostgreSQL), SQL can be used to store, update, and retrieve data dynamically.

For example, when a user fills out the **Personal Info** form with their name, email, and password, this data can be saved into a database using an INSERT or UPDATE SQL query. Similarly, when a user clicks the **Login** or **Sign Up** buttons in the modal windows, their credentials can be verified against records in the database using SELECT queries. This allows secure and persistent user authentication.

The **Search** functionality can be made dynamic by querying the database for relevant records using SQL LIKE operators, which would allow users to search reports, user names, or other stored data. In the **Settings** section, if a user changes their theme or password, the new preferences can be saved in the database using UPDATE statements.

Moreover, the **Reports** section can generate real-time summaries or analytics by querying and aggregating stored data, such as user activity logs or performance metrics.

To enable this functionality, JavaScript would send the form data to a backend server via API calls (e.g., using fetch or axios). The server processes the request, runs the appropriate SQL commands, and sends the result back to the frontend.