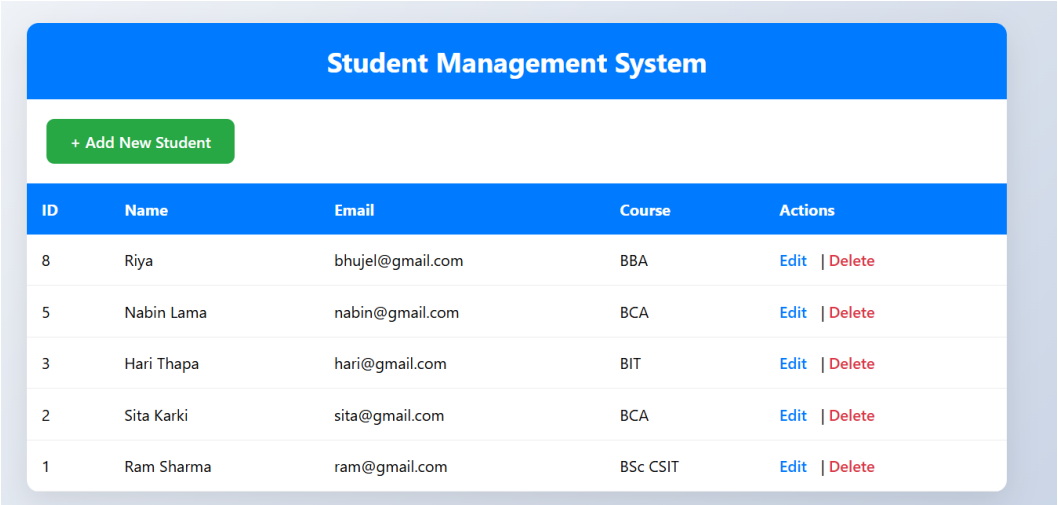
Student Management System

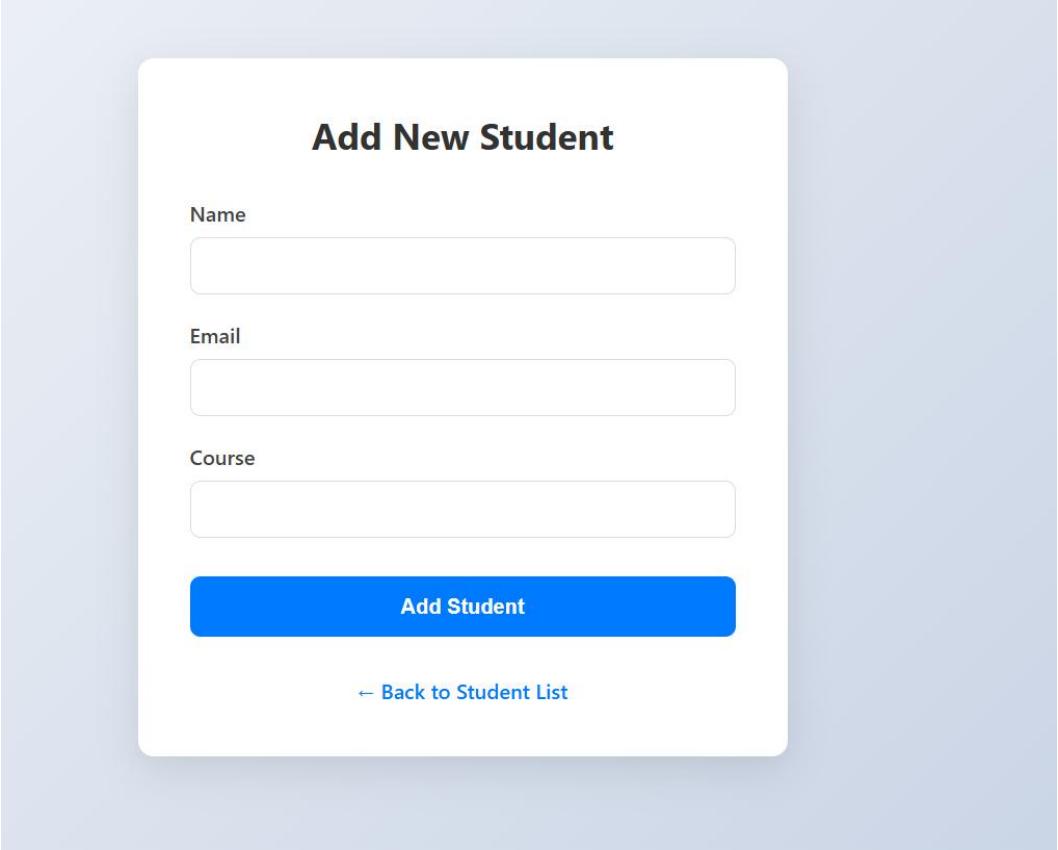
This document explains the Student Management System logically from a system perspective. Each screenshot is followed by a short, clean explanation focusing on how the system works, how data flows, and how different components interact. The explanations are not code-heavy, but small syntax references are used where necessary for clarity.

Student List Dashboard



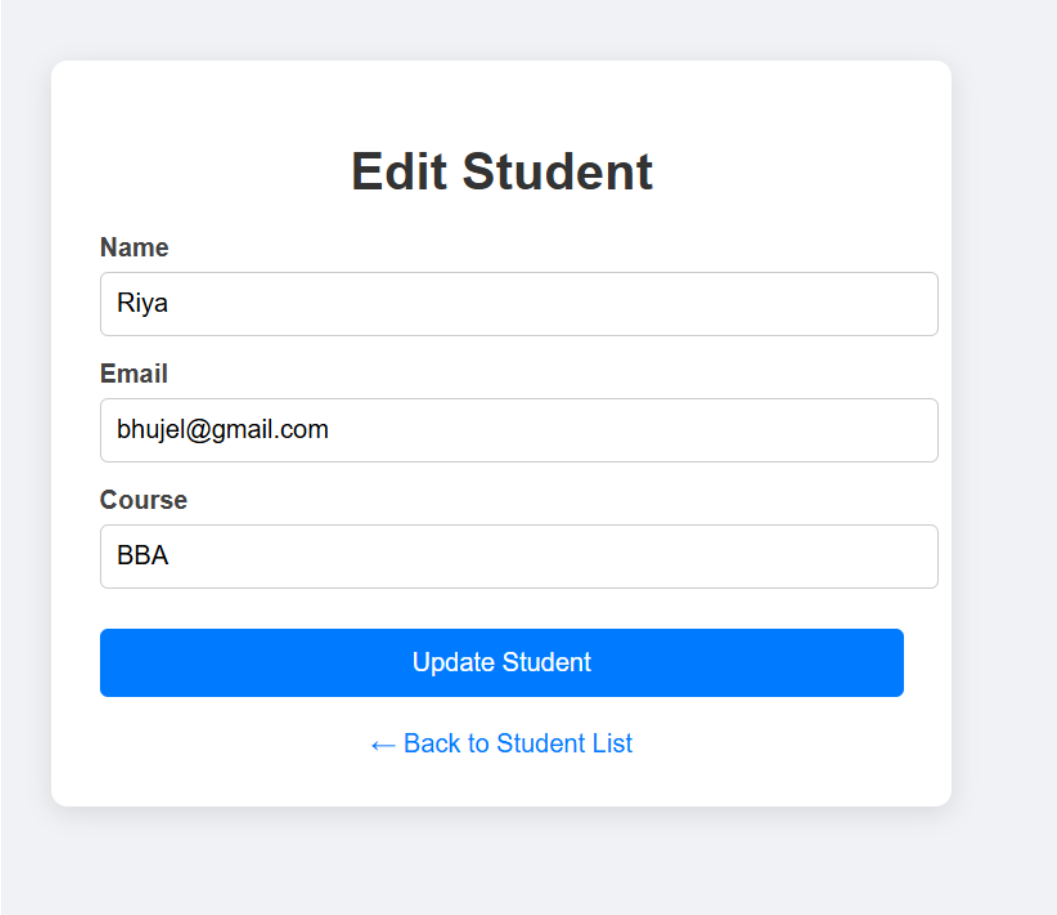
This is the main dashboard of the system. It displays all student records retrieved from the database. The system performs a READ operation by fetching student data using a database connection. Each row represents one student entity, showing structured data like name, email, and course. The Actions column allows navigation to update or delete operations, making this page the central control point.

Add New Student Form



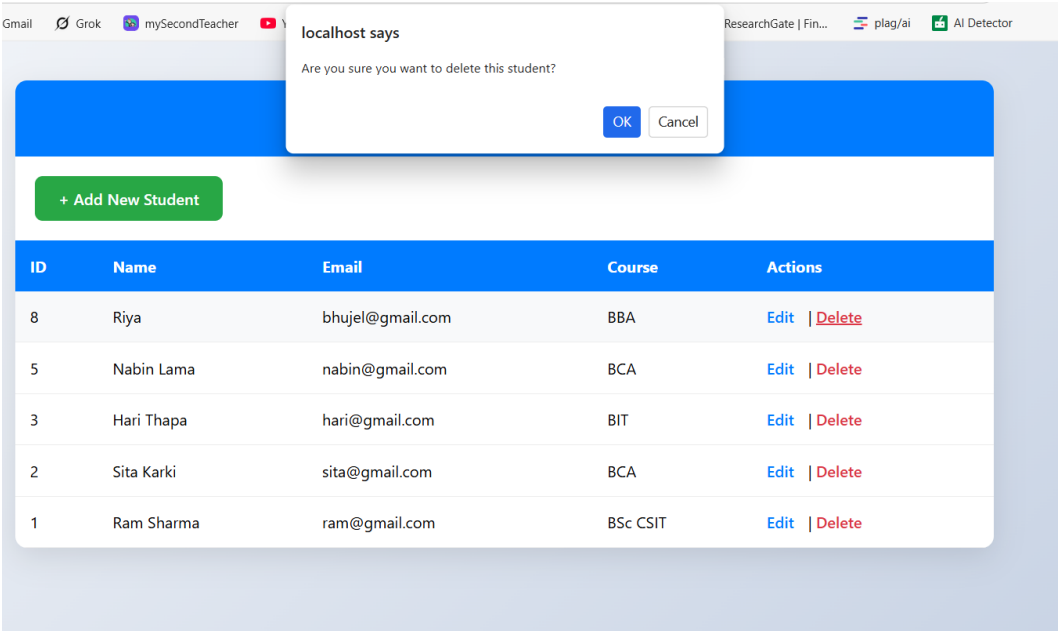
This screen handles the CREATE operation. The system collects user input through a form interface. Once submitted, the data is validated and sent to the backend for insertion into the database. This separation between form display and data processing improves system clarity and maintainability.

Edit Student Form



This page supports the UPDATE operation. Existing student data is first retrieved using a unique identifier (ID). The form is pre-filled to avoid data loss and improve user experience. After modification, the system updates only the selected record, ensuring controlled data consistency.

Delete Confirmation Dialog



This confirmation dialog acts as a safety mechanism before performing the DELETE operation. The system uses a confirmation check to prevent accidental data loss. Only after user approval does the system execute the delete request and refresh the student list.

System Flow Summary

The system follows a clear CRUD-based architecture. User actions from the interface trigger backend logic, which interacts with the database and returns updated results to the interface. This logical separation between presentation, processing, and data storage makes the system efficient, easy to understand, and suitable for beginner-level backend development projects.