TECHTONIC Evaluation Report

Problem: Design and implement a web-based Student Management System

Date: 2025-05-07 08:43

Total Score: 66.0 / 100

Problem Statement

Design and implement a web-based Student Management System that allows users to perform basic CRUD (Create, Read, Update, Delete) operations on student records. The frontend should be developed using React and styled with HTML, CSS, and Bootstrap to ensure a clean and responsive user interface. The backend should be built using Express.js, exposing RESTful API endpoints to handle requests and interact with a MySQL database for persistent storage of student information such as name, email, and course. The system should include input validation, modular code organization, and follow best practices in API design and component-based architecture.

Student Solution

For the application, I created a form and a list using React. I used `useState` for storing form inputs and student data. I implemented API calls using the Fetch API to interact with the Express backend, which is connected to a MySQL database. I used SQL queries to insert and retrieve student records. The interface uses Bootstrap for layout and styling. I focused on creating the add and delete functionalities first and structured the code to be extendable. Input fields include basic validation. The project demonstrates use of React concepts and REST principles.

Feedback Details

Category	Feedback	Score
React UI Components	The student demonstrated use of a form and list components in React, showing a basic but effective use of UI components.	8
RESTful API Design	The solution uses Fetch API and an Express backend, although only a subset of the full CRUD operations is fully detailed.	7
MySQL CRUD Operations	The implementation shows SQL queries for inserting and retrieving records, but it lacks clear details on update and delete operations at the database level.	7
Form Validation	Basic validation is mentioned, which meets minimal requirements but could be more comprehensive.	7
State Management	Use of React's useState hook for managing form inputs and student data is appropriately demonstrated.	8
Code Readability	The explanation suggests an organized structure aimed at extensibility, though more explicit details or examples would strengthen this area.	7
Responsiveness (CSS)	Relying on Bootstrap provides a responsive and clean layout, adequately addressing the styling requirement.	8

Category	Feedback	Score
Modular Structure	The code is indicated to be extendable, but details on how modules and components are separated are sparse.	6
Error Handling (API)	There is little evidence of robust error handling in API calls, reducing the overall effectiveness of this feature.	5
Deployment Readiness	There is no clear indication of deployment configuration or preparation steps, implying the project may not be fully production-ready.	3