Final Project Presentation

By: Cody Crosby

CST-391

Project Overview

- This project’s purpose was to create a full-stack gym equipment management application that supports all CRUD operations. The backend was built using Node.js, Express, and MySQL, while the front end was developed in both Angular and React.
- Both front ends use the same REST APIs to perform adding, editing, viewing, and deleting operations.

Backend Development

- The backend implements a REST API using Node.js and Express to handle all CRUD operations. The APIs interact with a MySQL database for persistence. Middleware was used for logging and error handling.
- This section of the project taught valuable lessons in handling async calls, validating user input, and the importance of detailed logging.

Angular Front-End

- The Angular front end uses modular components for listing, editing, creating, and viewing equipment. It integrates with the REST APIs for interacting with the data.
- A major design goal for Angular version was to keep the interface clean and intuitive while focusing on component-based design.

React Front-End

- The React front end using functional components and hooks. It mirrored the same CRUD functionality as Angular but emphasis on state management and reusable components.
- One goal for the React font end was to implement functionality that was missed on the Angular version, as well as maintain consistent styling between the two front ends.

Challenges and Lessons Learned

- The first challenge came from handling API connections between the front ends and the backend server. Small mismatches in routes or request formatting could break the link and force me to trace the server responses to find the source of the issue.
- Timing issues with async calls were another challenge, especially deleting equipment. UI components were refreshing before the database confirmed the change, causing the view to not be properly updated.

Christian Worldview

- Accessibility and ensuring equal access for everyone closely parallels the Christian Worldview belief of serving others. For developers, this means designing applications that all users can use, regardless of disabilities that may alter how they interact with it. Best practices like clear navigation and semantic HTML for screen readers help to fulfill the developers responsibility according the Christian Worldview.

Conclusion

- Creating two separate front ends showed how different frameworks handle concepts like routing and component structure while still following the same core coding principles.
- This project greatly improved my understanding of how front-end frameworks connect to backend systems and gave me the confidence to build and troubleshoot full-stack applications.