Lappeenrannan teknillinen yliopisto
School of Business and Management

Sofware Development Skills

Antti Keronen, 001855050

LEARNING DIARY, FULL-STACK MODULE

Declaration of Ai use:

Chatgpt was used to give an idea of a mern project. It gave me the idea of budgeting tool.

Chatgpt was used to solve coding errors, to show what is wrong with the code.

LEARNING DIARY

20.1.2026

I started the course. I had some time to scroll through the moodle page and realized that I started this course a bit late in my studies. I have Advanced Web Applications course going on at the same time so let's see if I can learn something that the AdWeb course doesn't have. At least I get to strengthen my skills.

21.1.2026

I decided early on that I would not simply copy the demo project but instead build my own MERN stack application while still following the course requirements. From the Node.js introduction video, I reinforced my understanding of core Node.js concepts like modules, file system handling, and building servers without Express. Although most of the content was familiar, the section on encryption was new to me and helped me better understand how security functionality sits in backend development.

While setting up MongoDB Atlas, I learned how cloud-based databases differ from local setups. This gave me a more realistic understanding of how databases are typically used in production environments. The Express.js crash course mainly worked as a review. It strengthened my understanding of middleware, routing, authentication, and error handling. The React crash course overlapped with previous courses, but it still reinforced best practices related to components, hooks, routing, and state handling.

22.1.2026

After completing the videos, I started building my own project, a monthly budgeting tool. The main idea of the application is to track expenses per month and compare them against a defined budget.

I began by setting up MongoDB Atlas and then built the backend using Node.js and Express. Unlike the demo project, my application required three data models, which are users, budgets, and expenses. Creating and modifying these models helped me learn how to adapt an existing project structure to fit new requirements. Authentication is implemented using JWT and bcrypt. One of the more challenging backend tasks was handling monthly queries for expenses. Working with dates in MongoDB was not straightforward, and I had to research additional resources to implement the logic correctly. This taught me the importance of understanding how dates work in databases.

On to the forntend

I created the frontend using React with Vite. My goal was to connect the user interface to the backend and build a clear, usable structure for the application. I implemented routing using React Router and created the main pages register, login, and dashboard. While building these. I used Axios to communicate with the backend and implemented authentication handling using React Context. I encountered several issues where authentication state did not update correctly after login or logout. Through debugging, I realized that centralizing authentication logic inside a context the application much easier to maintain.

I continued developing the frontend by creating reusable components such as budget cards, expense forms, expense lists, month selectors, summary cards, and protected routes. I learned how important consistent error handling is, especially when dealing with authentication errors.

I encountered multiple issues related to API requests failing due to missing headers or incorrect environment variables. In the final stage, I focused on styling the application and improving usability. I'd say the css was the funniest part of the task.