Lappeenrannan teknillinen yliopisto

School of Business and Management

Software Development Skills

Natunen Aleksi, 001153516

LEARNING DIARY, FULL STACK MODULE

LEARNING DIARY

Date: 03.09.2025

Activity: Initial setup

Learning Outcome:

I checked the general information and understood the main focus of the course. I chose my main code editor to be VS Code as I have used it for many other courses and mainly for all of my coding. From this I looked at the needed addons and such for this course, and noticed that I already have all that are needed/recommended as I have completed other courses where such are recommended.

From here on I started to set up my project on VS Code, by creating some of the files/file directories and setting up React on the frontend/client side. From there on I loaded react on to my frontend and started to imagine the wireframe of my project in my mind. My project is supposed to be a rating platform for LUT-University courses (only Technology related courses). I chose this as there are data handling and functions I am interested in practising.

Date: 20.10

Activity: Nodejs

Learning Outcome:

Here I reinforced my knowledge of Node.js by creating a simple Node.js webpage without the help of Express. I learned that it is possible to make a fully functional and simple web server with only Node.js, which helped me better understand what Express actually abstracts away and why frameworks are so useful.

This step also refreshed my understanding of how HTTP requests and responses work at a lower level, including setting headers, managing routes manually, and writing responses directly to the client. It gave me a better appreciation of what Express does behind the scenes.

Date: 20.10

Activity: MongoDB

Learning Outcome:

Here I reinforced my knowledge of MongoDB. I already knew how to make a database, but I learned how to import data directly into it. As I had scraped the courses from the Moodle webpage into a JSON file, I imported all of the data straight to MongoDB.

I created the Mongoose controller in the app.ts file and defined models for the web application to structure and rewrite the database. Additionally, I learned how to update the whole database efficiently, and in the process, I added two new fields: likes and Dislikes.

Additionally I created the models, users, and courses in the back-end directory so the application could use them.

In addition to this, I created the whole app.ts file during this stage. The app.ts works as the main controller for the server. In it, I initialized an Express.js server with port number 3001, connected it to MongoDB.

Date: 20.10

Activity: Express.js

Learning Outcome:

Here I reinforced my knowledge on Express.js. In addition to the previously created app.ts I created some simple routes in a separate routes file to improve code readability and modularity. I reinforced my knowledge on how to use express.Router() to define routes and export them for use in the main server application file. These routes included routes like getCourses, likeCourse, dislikeCourse, login, and register. The routes use GET and POST HTTP methods. These I tested with a simple html “front-end” that I created simply to test my backend logic.

I also implemented a custom middleware function called validateToken to handle token validation for user registration and login logic. The middleware and backend utilize JWT (JSON Web Tokens), JwtPayload, hashing, and bcrypt to securely identify and authenticate users. Through this process, I gained a better understanding of how to

integrate authentication and authorization mechanisms in an Express.js application, as well as how to protect routes and manage user sessions securely.

Date: 20.10

Activity: React

Learning Outcome:

Here I reinforced my knowledge on React. I developed Login, Register, and Home components for the web application. Within these components, I used various useState hooks to manage and update state information dynamically.

I also structured the components in a way that keeps the code organized and readable. I didn't implement any additional hooks beyond useState, as they were not required for the current functionality, but this stage deepened my understanding of state-driven UI updates and reactivity in React.

Date: 21.10

Activity: MERN-Stack

Learning Outcome:

Here I integrated the whole MERN stack. I connected the React frontend to an Express + Node backend and a MongoDB database (via Mongoose). I learned how to increment fields in MongoDB ($inc), and I practiced hashing passwords with bcrypt. The front-end and back-end I connected via cors. I gabe the back-end port 3001, and front-end port 3002. I told the back-end that it can take calls from port 3002 only, and the front-end that it must make the calls to 3001.

On the frontend I implemented fetch calls to the back-end that include the Authorization: Bearer <token> header, parsed responses and handled errors (redirecting to /login if the token is missing/invalid). I practised optimistic UI updates by immediately updating local React state for likes/dislikes and then syncing with the backend. With the fetch calls to the back-end I was able to get the courses from the mongoDB, login, and register.

In the front-end I queried the courses in a list to the homepage. The courses I of course queried from the mongoDB via the back-end API. The courses are queried as courseItem, which are REACT components. The courseItem components are built as

MUI card materials. This I chose for the reason that it makes the UI look better, and to make my life easier, as the cards are fairly easy to make and develop.

Additionally I added MUI React UI tools to make the web application look more modern, and to ease my development process. I used MUI materials like: TextField, Box, Button, Select, MenuItem, FormControl, InputLabel, Typography. Additionally I used iconButtons to make the like, and dislike buttons look more appealing to the user.

Additionally to the front-end I added some filters for the amount of courses to show at once, and a search query to search for courses. The courses can be searched by name, teacher, or subject.

Overall this course solidified my understanding of the full request flow (React → Express → MongoDB → Express → React), authentication/authorization with JWTs, and practical concerns like error handling, and basic API.

AI statement

AI was used to help write this learning diary. The AI tool used was ChatGPT.