Study-Jar Web application – Portfolio Phase 3

My Idea and Goal

My idea was to build a free to use motivational study tracking web applications where users – basically students, can visualize and track their progress. The goal was to help student build better study habits through visual feedback and by gamifying the study process. Each study session fills a virtual jar with water making progress feel tangible. I also wanted to make the experience more engaging by showing study sessions as gems and offering analytics to reflect on their study patterns be aware of the study time they have put in so far.

I first started by thinking about what possible web applications I could use and build. A study web application is something I knew I and many of my study colleagues would use to stay productive. I then started Design, I wanted to design a user interface that was modern and friendly, so do this is I used Figma, I designed the dashboard, analytics tab and the study jar which would visually represent the amount studied. I used React.js to build the frontend out, tailwind CSS made the styling process easier. For the backend I used Node.js and Express as a Framework for its easy-to-understand features and its ability to scale. When the Frontend looked good it was time to set up the Backend using Node.js. Before I could start programming the back-end logic, I spent time reading documentation, doing Code academies backend career path and watching videos about the latest node.js features, ensuring I fully understood the backend. This research helped me determine the best method for integrating a secure authentication system with minimal complexity. For database Modelling I initially used MySQL (relational database) and later switched to a non-relational database which was MongoDB, I chose it because it excels in scenarios involving large volumes of unstructured data and requires rapid, flexible data retrieval.

From idea to Final Result

I started by sketching a basic concept: an app that makes studying feel rewarding. I then built the interface, implemented user accounts, and added features one by one: time tracking, tag filtering, daily study goal setting, and analytics. Each study session now appears as a gem, and the water level in the jar increases as time accumulates. The analytics graphs give a snapshot of progress over time.

The Final result and Goal Match

The final product is a beautiful fully functional study tracker with all the core features I set out to build. The result matches my original vision for users, which was to track study session and visualize growth and reflect on progress. There were challenges especially on the backend of the web application, the key goals were successfully achieved.

Reflection on Process - What I learned:

I learned how to use version control (Git) and resolve merge conflicts. I also learned how to debug errors in the terminal and fix layout issues with CSS. On the backend side I learned how to integrate charts and track user input, user authentication and build full stack projects in a meaningful way

What I could improve:

I would plan the file structure more carefully from the beginning. I also would add responsive design features earlier in the process to avoid needing to go back and make changes that already worked well. I would do more user testing to gather feedback. I could also add more Jar shapes with different designs to make the user experience different and more engaging

Reflection - Final Thoughts

What worked well

The jar fill-up visual was effective and fun to build but took to much time honestly to get the animation perfect. The chart analytics provided good feedback for users. I successfully managed versions and updates using Git & GitHub.

What didn’t go as planned

I had issues with terminal errors and debugging early on. I struggled with CSS layout bugs especially for different screen devices when making the UI responsive. Time management was tough during testing and polishing phases.

What would I do differently

Start testing earlier in the process. Get feedback from users sooner to improve design decisions. Work on more modular code to make updates and changes easier.