Final Writeup

Reflection

Ruby on Rails:

As the first stage of the assignment, the backend component was not the most difficult to learn but it did take a while to fully understand the code and structure, and more importantly, type out my own controller code without referring to the templates. The most challenging part in this stage for me was to imagine how the code interact with the invisible database in the background, like how data is returned from the backend (in an array of data objects) and how create/post requests send the new data to the backend. And after a lot of practice, I no longer need to refer to the ruby guide frequently to perform tasks like creating tables in the database. Practice makes perfect!

React:

Developing the frontend using react is the most fun part of the assignment. I spent most of my time exploring different kinds of react components, testing them on my webpage, and changing different parameters to see how they perform. I also tried to use more trivial icon buttons instead of plain text buttons to make the application more fun (or just not lame) to use. The one thing I am not satisfied about my frontend is that I did not implement it systematically, like using classes to categories different components. As such, there are a lot of repeated code for components like text fields and their styling. Moreover, I hard coded some of the URL for the routing, which cause some problems later in the project. Apart from these imperfections, I still learnt a lot about react functionalities, like using react.state to keep track of different variables. I also applied some basic concepts of OOP in my application, like encapsulation.

Axios api:

This stage was the most painful of the all. I tried many apis, like fastjson api, and their online tutorials. I wasted a lot of time skipping from one api to another:

every time I faced some obstacles, I tried to move to another api thinking that one may be easier to learn. Then I realized I can never make progress by evading problems. So, I tried my best to preserve through the painful learning curve. Apart from all that, I had to modified most of my backend and frontend codes to accommodate the axios api, like the backend controllers need to render via json. So comprehensive planning in the beginning of the project is imperative to minimize repeating and waste of effort in the future.

Heroku:

It was the last stage of the assignment. Learning how to upload the project files onto GitHub using Git wasn't as troublesome as the previous tasks. But I still came across many problems like failing to git push. Implementing Heroku took a lot of time as Heroku does not accept sqlite3 (the database I used in my initial app) many problems surfaced during the installation of pg, like the wrong directory and version. Another major problem (unable to access the backend controllers on the Heroku server) is caused by the hard coding of URLs in my react application.

Throughout the learning journey, Google has been my best friend. Every time I got stuck at some error messages or mysteries bugs, I can almost always find some posts or forums discussing a similar problem. Although they may not provide a direct solution to my problem, they provide valuable insights for me to start the debugging. Nevertheless, it is extremely satisfying to solve the bug after hours of searching and trying, and witness how my website grows in functionalities.

User manual

The application consists of two main segments: student and subject records. For each segment, the user can perform CRUD operations on the respective data. For each student, the user can also add and delete exam results in the view student tab. Only subjects exist in the database can be added. For more detailed information about the structure of the database please refer to the readme file in Github.