CVWO Assignment Submission

Name: Daniel Lim Wei En
Matriculation Number: A0233730N

1. What I feel about my accomplishments in this assignment.

I came from a background with only very limited knowledge about HTML, CSS and JavaScript. Therefore, when this assignment was first given out, much of the concepts were entirely new to me. From writing a backend using the Ruby on Rails framework, to having to write the frontend using React and using a RESTful architecture to link the frontend and backend. Many of these concepts were foreign to me and therefore there was a steep learning curve before I could even begin attempting the assignment.

To that end, in order to bridge the gap, I took the advice from the assignment and planned out how I was going to tackle learning these new concepts. I began with attempting to learn about version control software and specifically Git using Youtube tutorials and by practicing with small trivial examples. Next, I thought to learn about JavaScript and React. Having come from CS1101S, I had knowledge about a subset of JavaScript called Source. Thus, the first thing I did was attempt to learn the full JavaScript language so that I could make use of React without worrying about syntax. I looked up guides and made use of freeCodeCamp's JavaScript course to practice and to learn. Next up was React. I turned to the documentation and visited their website and attempted their tutorial to learn by doing and to learn from the ground up. After that, was using freeCodeCamp's React section to put the new knowledge to practice. Finally, I moved on to HTML and CSS. I followed the guide on HTMLDog, practiced and read up on what they were. So that covered basic knowledge of the frontend.

Next up was the backend. Ruby and Ruby on Rails were next. I made use of online recourse, guides, videos and the like to learn about these technologies and put the knowledge to practice with simple example programs. I also followed along Youtube guides to use Rails to make simple web applications in order to learn by doing and to put whatever I have learnt about to practice. At the same time, I learnt about relational databases, what SQL was and how to write SQL queries to query databases for information. This allowed me to link the knowledge about SQL to how Rails handles managing databases. It also allowed me to begin thinking about the schema for my application.

I could make simple applications using Rails and using React alone. But I was still missing the thing that linked them together. The final step was for me to learn about RESTful architecture and how I could get the frontend to interact with the backend. I read up on HTTP messages and JSON format for passing data as well as on APIs. I then put the skills to practice. I created two separate applications, a Rails API only application for rendering JSON data to the frontend, and a React application for making requests to the backend for data to be displayed and attempted to get them to interact by running them on different ports.

Throughout the learning process, it was tough because of the sheer volume of new information coming in at once. But I am glad that I pressed on. I tackled the new topics systematically and I tried my best to put new knowledge to practice so that I could have the knowledge stick. I am glad that I went through the process of scouring the internet

for resources to learn new technologies as it has taught me to be more resourceful and to be more independent in learning new things instead of waiting to be taught. It has also taught me how to learn new technologies more effectively and that is through practicing. With small examples and toy programs first, then following along guides to practice more complex concepts a few times and then finally practicing writing small projects independently. Overall, I am glad that I attempted this project as it has really taught me a lot about web development and given me valuable experience learning and trying new things by myself.

2. Short User Manual.

Creating Tasks:

1. On first entering the application, users will be able to view the main to-do list page.
2. The top of the page contains a form that allows users to add new tasks to the to-do list.
3. The form contains 3 fields, a task description, further details about the task and an urgency selector to categorize tasks into urgent or not urgent.
4. Once these fields are filled, users will be able to add tasks and the updated tasks will be shown on the to-do list page as individual task cards.

Updating Tasks:

1. Each task card contains an edit button, that upon clicking by the user, will toggle the edit form to show up on the task card.
2. Users can then edit the task by filling in the new task description, the task details and the urgency of task and then submitting the form.
3. Upon the next refresh, the users' previous task will be updated to have the new information.
4. The tasks are read from the database and then displayed on the webpage.

Deleting Tasks:

1. Each task card also contains a delete button that upon clicking, removes the task card from the database and removes the task card.
2. The updated tasks are then shown on the webpage.

Categorizing Tasks:

1. In the form to enter new tasks and when updating tasks, users can select a level of urgency for their tasks whether urgent or not urgent and they are able to update the level of urgency for their tasks as required.
2. This is recorded in the database and displayed to the user in a small note in the task cards. With red representing urgent tasks and blue representing not urgent tasks.
3. Moreover, users can filter the tasks that they would like to see only by using the filter buttons at the top. They can view only not urgent tasks, only urgent tasks and are able to redisplay all tasks as desired.
4. And upon refreshing, all tasks will be displayed once again.