CVWO Mid-Assignment Report

Name: RUPPA NAGARAJAN SIVAYOGA SUBRAMANIAN
Matriculation Number: A0217379U
Assignment: Riding on Rails

Before Starting this assignment:

When I first read the assignment requirements, I was initially overwhelmed by the sheer number of things that I had to learn before starting. I had no prior experience in web development, and therefore, everything was new to me. After the assignment was released, I spend close to 2 weeks reading about react and rails. I also watched various YouTube videos about react and rails before starting on this assignment which gave me a rough understanding about how everything works. After I started the assignment, I found myself watching more tutorials and reading more online on how to go about implementing what I want to implement. This is that part that proved to be the most useful as I learned a lot when I was faced with a problem (no matter how small) and had to solve it myself by searching it up online.

About Just To-do it:

Just To-do it is a simple To-do manger that is simple to use and has features that some To-do mangers such as Google's Tasks app lack. Firstly, Just To-do it supports subtodo creation which means that a user can add as many subtasks as desired to the main task which itself has CRUD features. The user can also add a description, a deadline and most importantly tags for the main To-do.

Adding tags is the most important aspect of Just To-do it as it helps the user better organize their To-dos efficiently. Just To-do it sorts the To-dos automatically which is reflected in the navigation drawer of the website. The navigation drawer is always present in all views of the website so that the user can jump around quickly to what they want to do. The navigation drawer has 2 sections. The main section will have tabs that display All To-dos, To-dos that are due on the current day, To-dos that are due on that week and finally completed To-dos. The second section will have tabs that display all tags that the user has created and clicking on one will bring up all To-dos that have that tag.

Just To-do it is also fully mobile, and tablet optimized given that Material UI is used as the UI framework of choice which means that the user can use the website no matter what device that they are using.

Use Cases:

When using Just To-do it, the user should be able to create an account or login with an existing account. The authentication will be done using JSON Web Tokens. Once logged in, the user can proceed to add To-dos.

The website will support CRUD features of these To-dos as well as the CRUD of the Subtodo of a To-do.

One of the most important aspect of Just To-do it is that adding To-dos is only supported in the Home page of the website so that the user can quickly create a To-do on the fly. The other views only support RUD features to ensure that Just To-do it is as simple as it can be.

Main Views of Just To-do it:

Home/All View [Completed]

This view is that view that has all To-dos that the user has in a list view. At the bottom right corner of the page, there is a floating action button that the user can click to add a new To-do.

Each To-do has a checkbox to mark it as not completed/completed, a text field that contains the title for the To-do and 3 buttons (Save, Edit, Delete). Save and Delete buttons support UD features of the main To-dos. The edit button will bring up the Edit View when the user clicks on it.

Edit View [Completed]

The Edit view is the main view that supports advanced U features for the main To-do. Here the user can add a description, a deadline, and tags for the main To-do. This view also supports CRUD of subtodos.

The subtodos have a checkbox to mark them as completed/not completed. Each subtodo can also be deleted using the delete button from this view.

Navigation Tab Views (All [Completed], Due Today [Completed], Due This Week [Completed], By tags [In-progress])

The main section where the user can click the respective buttons to sort their To-do list by some attribute or by their tags. Sorting by tags will allow the user to view all To-dos with a specific tags and update or delete them quickly.

The view will be like the home view but will only have the sorted To-dos that meets the criteria of the different tabs. All views will support the edit and delete features.

Calendar View [In-Progress]

The calendar view shows all To-dos organized neatly into a calendar format so that the user can have a good overview of the tasks they need to complete and completed tasks.

My Account Page [In-Progress]

This view will allow users to update their password if they choose to do so.

What I hope to achieve by the final assignment deadline:

- ~ Complete the remaining views that are in-progress.
- ~ Improve the design of the website. Most importantly add loading spinners to signal to the user that the frontend is waiting for a response from the API.
- ~ Comment and make code cleaner.
- ~ Host both the frontend and API on netlify.
- ~ Add email verification when a new user creates an account.
- ~ Add third party authentication (Sign in with Google/Facebook)
- ~ Notification Emails on To-dos approaching deadline.
- ~ Learn and explore how to integrate Cron Jobs.
- ~ Learn and explore how to integrate Docker.

Difficulties faced during this assignment:

- Ruby on Rails:

  Learning ruby was relatively straightforward given that it was made to be simple and easy to write. However, it took be a lot of time in the beginning to understand Rails. As I was learning Rails, I started understanding the power of rails because a lot of things that wanted to achieve was created using the terminal commands that rails provides. Even though I have made a working API, I find myself still weak with backend because Rails abstracts a lot of things which is good to get the application up and running but does not help with understanding. I still have a lot more to learn and will continue to do so.

- Typescript

  I initially started my frontend project with Typescript. However, it was initially quite difficult for me personally because I do not have any experience in statically typed languages. Furthermore, given that I was fresh out of CS1101S, I was more comfortable using JavaScript instead. Even though I could have spent my time learning typescript and then implementing it, I decided to dedicate my time and effort into learning React, Redux and Material UI which I thought would be better given the short timeframe given for this project.

- React and Redux

  React and Redux proved to be the most difficult aspect of this project. I watched a 10-hour YouTube react course to gain a better understanding about the different aspects of React like hooks, react router and many others. The most difficult part was executing. While doing so, I got to learn much more about react such as creating custom hooks and using redux as a global state management tool so that I do not have to drill my props through components to have something simple done.

  Redux and Redux Thunk also made it easier to handle API calls more efficiently with the use of Axios.

  Even though initially I struggled a lot just to start the project, I find myself more comfortable with React the most which I am proud of as I did not have any frontend experience prior to this assignment.

- Styling and Design

  I had no experience in UI/UX design which made designing this app very challenging.

  Fortunately, Material UI provides an easy and elegant way to override the existing styling using the makeStyles hook generator which I took full advantage of.

Conclusion:

Even though everything is still new to me, I feel that I am more confident in using Rails and React. There was a lot of reading done and videos watched to get where I am today. I still know that I have just scratched the surface and have a lot more to learn. I am confident that I will continue to learn and implement the remaining features into Just To-do it. I am also very excited to host my very first website and have real people try it out!