Computing for Voluntary Welfare Organisations, AY2021/22
Final Submission

Name:	Tiang Hui Zheng
Matric No.:	A0234713L

GitHub: Main	GitHub: Frontend	GitHub: Backend	Link to working App

Introduction

For the past two months, I have been working on my task manager web application, and it has been an interesting and eventful experience for me. In this report, I would like to share the final state of my app and my reflections.

Execution plan status

Do by	Features	Importance	Difficulty to implement	Status
4 Jan	Use Case 0 – User accounts	★★★★★	★★★★★	✔
5 Jan	Use Case 1 – (Marking of tasks)	★★★★☆	★☆☆☆☆	✔
9 Jan	Use Case 3 – Search, sort and filter	★★★★☆	★★★☆☆	✔
12 Jan	Vim-based keyboard shortcuts	★★☆☆☆	★★☆☆☆	✔
12 Jan	Toggling of dark mode	★★★★☆	★☆☆☆☆	✔
20 Jan	[In consideration] New tab browser extension	★★☆☆☆	★★★☆☆	✘
20 Jan	Short in-app tutorial	★★★★☆	★★☆☆☆	✘
22 Jan	Compatibility and responsiveness tests	★★★★★	★★★★☆	✔

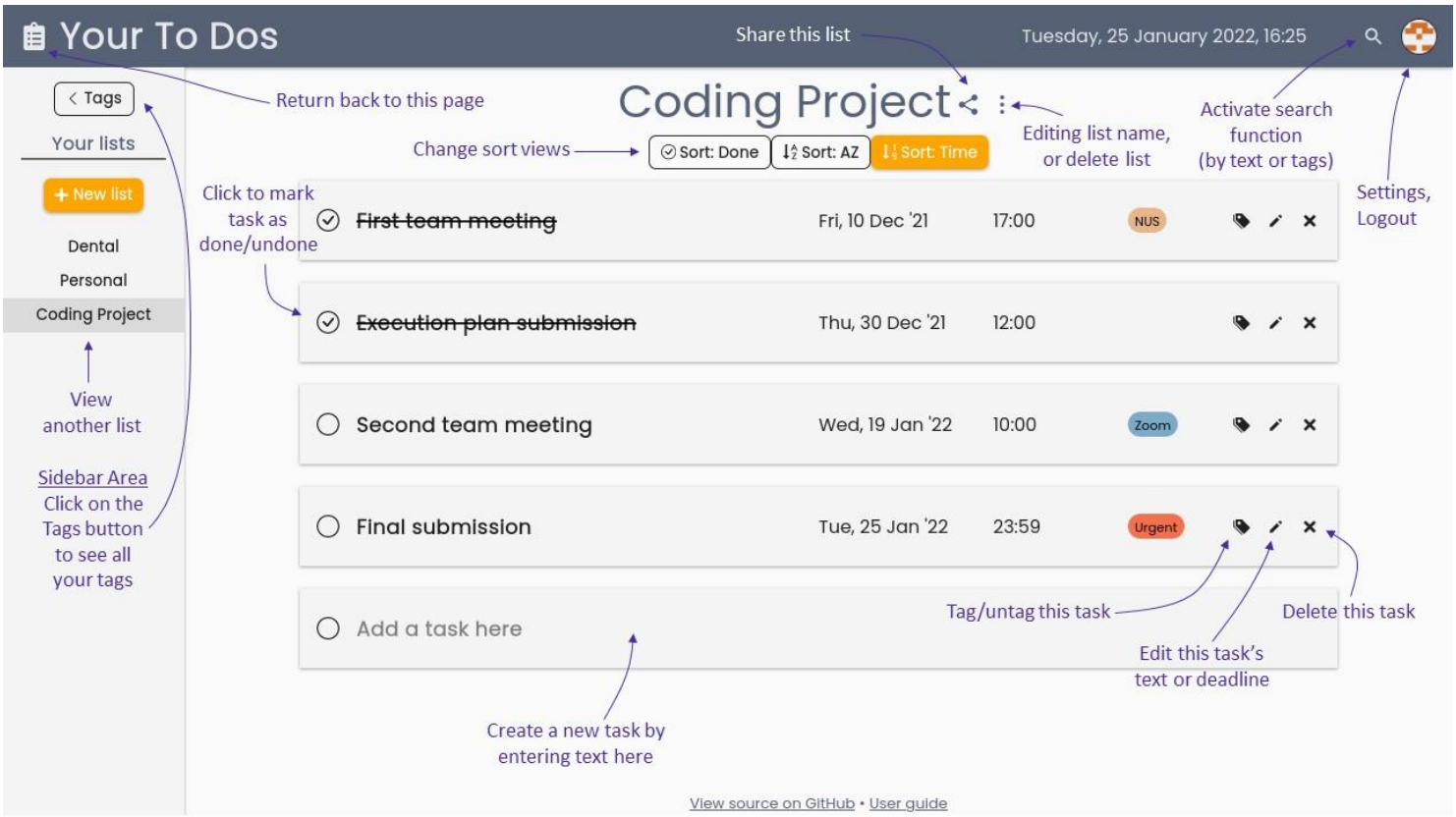
I was unable to complete two of the features I had set out to do in my mid-assignment report. This is mainly because while working on the app, I thought of some other features and my priorities shifted to them instead. For some of these features, implementing them to the existing codebase required me to re-think and refactor how my code was structured.

New features	Description
CRUD lists	A group containing related tasks
Sharing of lists	Users can get a uniquely generated URL to share with others to collaborate on the same list
Guest mode	Use the app without creating an account, relying on browser storage to save tasks instead
Offline mode	App functions if user's computer disconnects from the internet. The app will resync changes once back online.

From the original execution plan, the two features which I chose to "sacrifice" were ones I judged to be less important. The new browser extension feature can be emulated by setting the browser to open previous tabs on start-up. The user will then be able to continue using the app, even through computer restarts and internet disconnects. For the short in-app tutorial, I have placed a link to my repository's wiki page, where the user guide of the app is placed.

Mini user-guide

A link to the full user guide on my repo's wiki: Link



Creating your first task

1. Initially, the app looks blank because you have no lists yet. Create one by clicking on the orange "New list" button in the sidebar.
2. Fill up the title of the list and press OK.
3. A yet-to-be created task appears. Create your first task by clicking anywhere on it.
4. Enter the text of the task, and then the date and time of the deadline.
5. When done, click anywhere outside the task. The task will be automatically saved.

Creating your first tag

1. Click on the orange "< Tags" button in the sidebar to toggle to the tags drawer.
2. Click on the "New Tag" button and fill up name of the tag.
3. If you don't like the random color of the tag, you can click on the color wheel icon to select another color. Alternatively, specify the exact hex code of the color of your preference.

Tagging

1. Click the tag icon of the task you would like to tag.
2. A menu appears showing all your available tags. You can search through the available tags using the input box. Click on your desired tag.
3. The tag will appear on your task. Repeat until desired.

Some available shortcuts

Key sequence	**Description**
j/k	Select the below/above task
i	Edit selected* task
dd	Delete selected task
Enter/Space	Mark selected task as done or undone
l	Switch to next list
h	Switch to previous list
x	Toggle dark mode
/	Jump to searchbar

Reflection

Looking back, I should have had a better plan of what my app's use cases were going to be like. The sharing of tasks feature was one that I already had in mind from the beginning, and naturally, that meant I needed some sort of way to group tasks. However, I only realised that in the middle of developing the app. As a result, I had to make a major redesign of the app in both the frontend and backend, which could have been avoided if I planned better from the start.

From what I see, the way I managed state in my app is quite out of the norm. It is rather opposing to the current pattern of React, which has changed from being class-based to one that uses functional paradigm in the recent years. In my app, I am using a large context object acting as a combined prop, which holds helper functions and values required by multiple components. Sure this works, but I have come to realise it is less scalable and things can quickly become unwieldy. This is probably why Redux is commonly used to manage state for React.

In the past few weeks, I have truly learnt a lot about web development. I've experienced modern technologies that are commonly used in today's world, and learnt about the more nuanced guidelines of UI design. I've expanded my knowledge of JavaScript to the more advanced syntax that were beyond the scope of what I had learnt in my introductory programming course, CS1101S.

On the less technical side of things, I would like to say I am grateful for this assignment. When I started on this project nearly two months ago, I definitely did not expect myself to learn so much solely off creating one web application. I have been through many frustrating moments and countless hours combing through Stack Overflow and MDN Web Docs. But at the end of the day, I am glad to have made this decision to attempt the assignment, and I am definitely proud of what I have created.