CVWO Mid-Assignment Submission

Name: Dorcas Tabitha Tan *Matriculation number: A0190356A*

This write-up contains two sections: first, the basic use cases and possible extensions; second, my plan for prioritizing and implementing these use cases.

I recently learnt the basics of React (before the assignment was released), so my focus for this project is to understand Rails/relational databases and to learn to integrate React and Rails using JSON API, as suggested in the React on Rails installation guide. I also intend to learn and incorporate TypeScript before the final submission.

---

1. Use Cases

The to-do list application will support basic CRUD (add/edit/delete) functionality as well as tagging and searching of to-dos. A possible additional feature is the sorting of to-dos by their title or tag.

The following table provides details of each use case:

<table>
<tr>
<td>UC01 – Adding a to-do

MSS:
1. User adds a new to-do with the following information: title, body (i.e. details), and tag. The body and tag are optional.
2. System shows the list of to-dos.
Use case ends.

Extensions:
1a. Required information (i.e. title) is not provided.
1a1. System shows an error message.
Use case resumes from step 1.</td>
<td>UC02 – Deleting a to-do

MSS:
1. User indicates a to-do to be deleted.
2. System shows that the to-do is no longer present.
Use case ends.

Extensions:
1a. The selected to-do does not exist (e.g. was deleted using another window).
1a1. System shows an error message and the list of to-dos.
Use case ends.</td>
</tr>
<tr>
<td>UC03 – Editing a to-do

MSS:
1. User selects a to-do to edit.
2. System displays the to-do details.</td>
<td>UC04 – Tag a to-do

MSS:
1. User selects a to-do to tag.</td>
</tr>
</table>

3. User enters the edited to-do. 4. System shows the updated (edited) to-do. Use case ends. Extensions: 1a. The selected to-do does not exist. 1a1. System shows an error message and the list of to-dos. Use case ends.	2. System displays the to-do's tag, if any. 3. User enters a tag name. 4. System shows the updated (tagged) to-do. Use case ends. Extensions: 1a. The selected to-do does not exist. 1a1. System shows an error message and the list of to-dos. Use case ends.
UC05 – Untag a to-do MSS: 1. User selects a to-do to untag. 2. System shows the updated (untagged) to-do. Use case ends. Extensions: 1a. The selected to-do does not exist. 1a1. System shows an error message and the list of to-dos. Use case ends.	**UC06 – Search for a to-do** MSS: 1. User specifies a search query (e.g. a tag name). 2. System shows all to-dos that match the search query. Use case ends. Extensions: 2a. No to-dos match the search query. 2a1. System displays a "*No to-dos were found*" message. Use case ends.
UC07 – Sort to-dos MSS: 1. User specifies a sort critera (e.g. descending ID, ascending title). 2. System displays the sorted list of to-dos. Use case ends.	

---

2. Execution Plan

Database Schema

The database is structured as a single table. Column names and column types are indicated below, with the primary key being **ID**. Each *To-Do* has an (automatically created) ID, title, body, and tag. The body and tag are optional.

To Dos
ID: `INT` Title: `VARCHAR(255)` Body: `TEXT` Tag: `VARCHAR(255)`

Timeline

I started on this assignment on 24 December 2019. As such, I had 1 week to prepare the mid-assignment submission (30 Dec), and 4.5 weeks for the final submission (25 Jan '20).

This is a summary of what I intend to complete in each week:

Week	Dates	Tasks
1	24–30 Dec	Write out use cases and execution plan. Implement some CRUD operations (*UC01 add*, *UC02 delete*).
2	31 Dec–6 Jan	Implement remaining CRUD operations (*UC03 edit*). Implement tagging (*UC04 tag*, *UC05 untag*).
3	7–13 Jan	Implement searching (*UC06 search*). Design UI. Ensure cross-browser compatibility.
4	14–20 Jan	Host on Heroku. Add TypeScript to the project. *Implement sorting (*UC07 sort*).
4.5	21–24 Jan	Tidy up UI and documentation.

**To be done if time allows.*

Current progress

I am on track with the above timeline. So far, the biggest challenges have been to ensure that various packages and extensions work together properly and to figure out how to integrate React and Rails. Searching for online resources has not been easy; since some tutorials use different tools/packages, I have needed to develop an understanding of how front-end and back-end software communicate in order to figure out which tools are equivalent.

Moving forward, I anticipate spending more time to learn how Rails works. I might also need to relook the database schema. I look forward to gaining a better mental model of how the different web development stack layers work together.

My daily progress and learning points are documented in `/docs/Journal.md`.