CVWO AY2018/19
Mid-Assignment Writeup

Na Nazhou
(A0164627Y)

Use cases

Feature	Description
CREATE	A new task can be created by selecting 'add new task' on the main page and entering relevant information such as title, description, deadline, tag.
READ	The main page displays a list of tasks that have already been created. An existing task can be shown to the user by selecting 'show' on the right of the task.
UPDATE	An existing task can be edited and updated by selecting 'edit' on either the main page or the page showing details of the task.
DELETE	An existing task can be deleted from the to-do list by clicking 'delete' on the right of the task and confirming the deletion.
Tagging	As mentioned in CREATE feature, there is a 'tag' field to be entered when creating a task.
Search by tag	There is a text field on the main page with placeholder 'search by tag..' and all the tasks containing this tag will be displayed after entering a tag and selecting 'search'. I am considering changing the text field to a datalist or collection select so that users do not need to type anything. I am also trying to make the search process a partial re-direct instead of a full re-direct.

Execution plan

Based on my observation and personal experience of other task manager applications, I feel additional features can be introduced to make the website more flexible to users' demands. Except for the required features, I am also attempting to implement the following features.

Feature	Description
Sort	As for now, the to do list is sorted in the order of the time created by default. However, some users may want to sort their tasks according to deadline or tags. I want to put these two options as radio buttons and the users can customize how their tasks are ordered on the main page by selecting the radio button accordingly.
Archive	In the current version, a task is permanently destroyed once the user select 'delete'. I want to add a new attribute 'completed' to the existing task model. Once a task is marked completed, it is not deleted but archived into a another list – a list of completed tasks – so the user can still revisit their completed tasks after marking them completed.
Multiple tags	Currently only one tag is allowed for each task. By creating a separate model for tags and establishing association with the task

	model, a task can be associated with many tags and this increases the flexibility of the tagging system as well.
Login system	The to-do list should be retrieved even if the website is closed and later reopened. To implement this, there are two options. The first one is a user account system. Users need to create their own accounts and sign in in order to fetch their own to-do list. The other choice is through automatic login using cookies. Personally I prefer the latter as it is more convenient (despite less secure).

Furthermore, I would also like to enhance the user interface by using Bootstrap to make the website more aesthetically appealing. If time permits, I would also try out other front-end frameworks.

Currently I am still figuring how to improve the existing features and implement the additional features. Although I took CS1101S this semester, I still found I need to learn more about the application of JavaScript in web programming to incorporate into my to-do list app.

Problems & Challenges

As I am completely new to web programming, I found that I have to start from scratch and I need to have a lot of prior knowledge (like basic understanding of Ruby, HTML, CSS, SQL, RDMS, MVC patterns, HTTP request-response cycle, etc) in order to understand the Ruby on Rails Guide. I started out by following the guide and writing a sample Rails blog app. I was also practicing using Git and pushing repo to Github while I was working on the sample app. Whenever I found any concepts or syntax that I am not familiar with, I went to google relevant online tutorials about them. Through this way, I slowly get to know how everything comes together and how going through a website development cycle is like.

Another challenge is about testing and debugging. As I cannot expect every user using the application the same way as I am, there are likely to be many underlying bugs in the code. I have tried the basic CRUD operations and the search function, they seem working but I am not quite sure they are still going to work when I add more features or I test them with some corner cases. For example, I think I still need to add more validates to the models.

Besides, as I create more models and controllers, the application gets more complex. Things start to get messy as I need to keep track of the associations between different models, update the views and database accordingly. Thankfully, Rails has already simplified the process a lot and provides a systematic way of developing an application. I will try to get used to 'the Rails way' while working on the remaining part of my to-do list app.