

## **Reflection and User Manual for memoit**

Jason Qiu

### **1. Reflection**

This assignment can be described as a lot of firsts for me. Before this project, my experience with web development was limited to HTML, CSS, and simple PHP. It was my first time using React.js and Ruby on Rails with a RESTful API design. Although it was pretty overwhelming at the start to learn everything, this project was a valuable experience for me to formally learn how to build web applications with modern tech stacks. In the end, I am proud of what I have accomplished.

As my first time using a RESTful API design, I like the modularized concept of keeping the client and the server separate. To my interpretation, this is essentially abstraction on a large scale. I find it elegant that the client does not need to know how the server is implemented and only makes calls to retrieve what it needs.

In terms of the front-end, I personally love how memoit looks. Using Bootstrap, I went for a dark and minimalistic aesthetic for the app design. Structure-wise, I followed tutorials on React.js over the winter and applied concepts I'd learned in this project. I was shocked by React's powerful and neat design as I could divide sections into reusable components, instead of lumping everything in one page. When implementing the backend, Rails surprised me on how clean and readable its code can be using the DRY principle. Instead of writing tedious boilerplate code, Rails made it easy for me to implement the backend logic without writing SQL commands to the database.

I also enjoyed creating use cases, user stories, and use case diagrams to organize the functionalities of memoit. When I planned out apps in the past, my ideas usually went all over the place, making it difficult to organize them in a cohesive manner. Writing use cases gave me a systematic way of planning with the user's priorities as the center. This part of the assignment gave me an insight of how startups ideate their initial products.

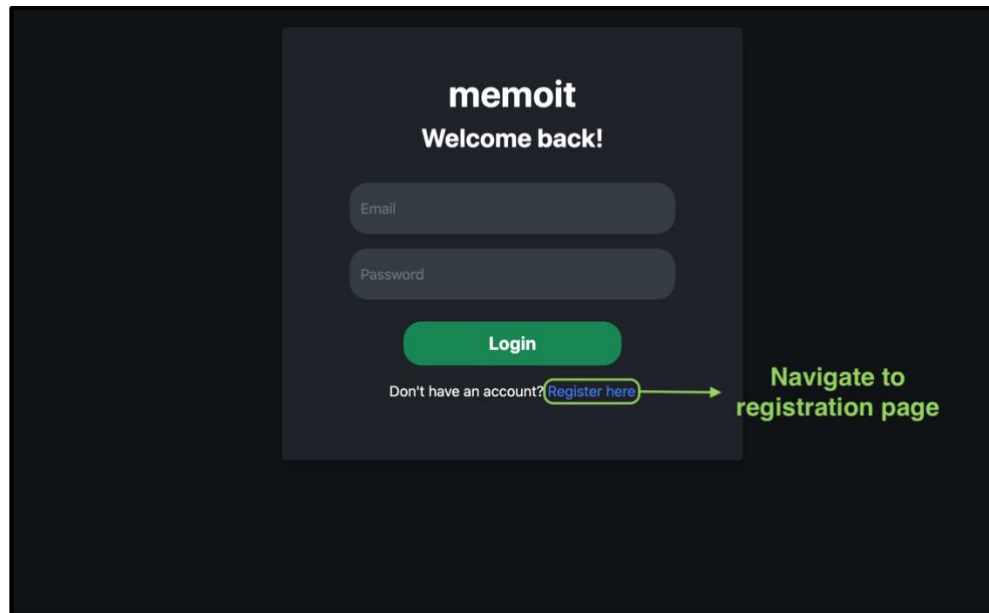
Nonetheless, there are definitely areas of improvement. Firstly, I can improve on writing cleaner React.js code, as it looked very messy at the start when I was writing the code for multiple components. I plan to improve this area by planning more before writing code and not rely on trial-and-error. Secondly, following the tutorial in the CVWO setup Github page, I used a Webpacker to implement a React frontend for Rails. In the future, I hope to try to keep the front-end and back-end components completely separate to modularize the code further.

Looking back, I am genuinely shocked yet proud at how far I had come over these past two months. Before, my knowledge of web development was shallow and

always seemed to be something on-the-fly for me. I did not understand the reasoning, principles, and intricate practices behind the code. Thus, I signed up for this assignment with the primary goal to learn modern web development formally. I am glad to say that I have achieved this goal that I set for myself.

## 2. User Manual

### Login:



**memoit**  
Welcome back!

Email

Password

Login

Don't have an account? [Register here](#)

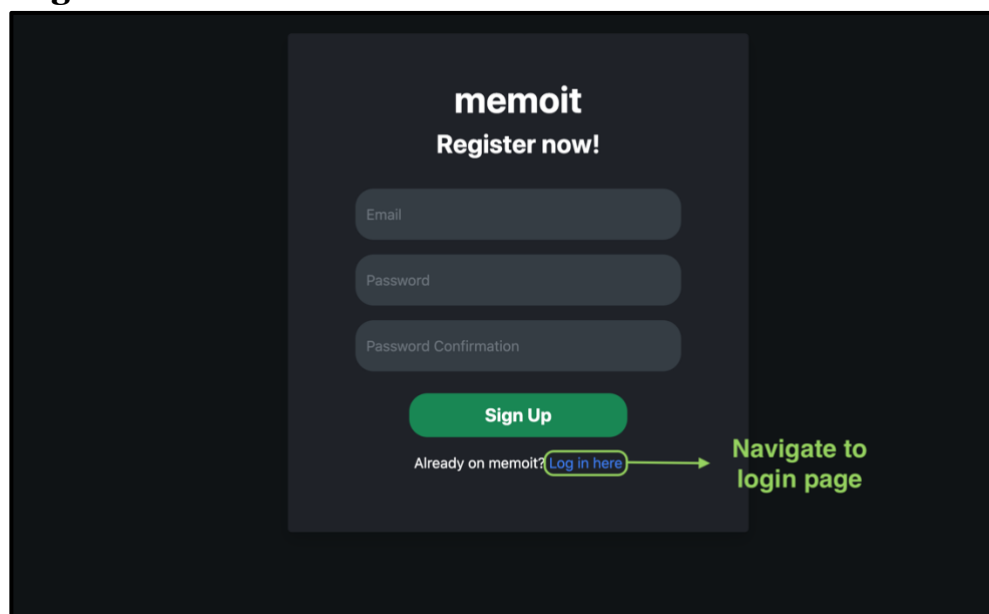
Navigate to registration page

For testing purposes, users can use the following login details:

Email: test@test.com

Password: 12345678

### Registration:



**memoit**  
Register now!

Email

Password

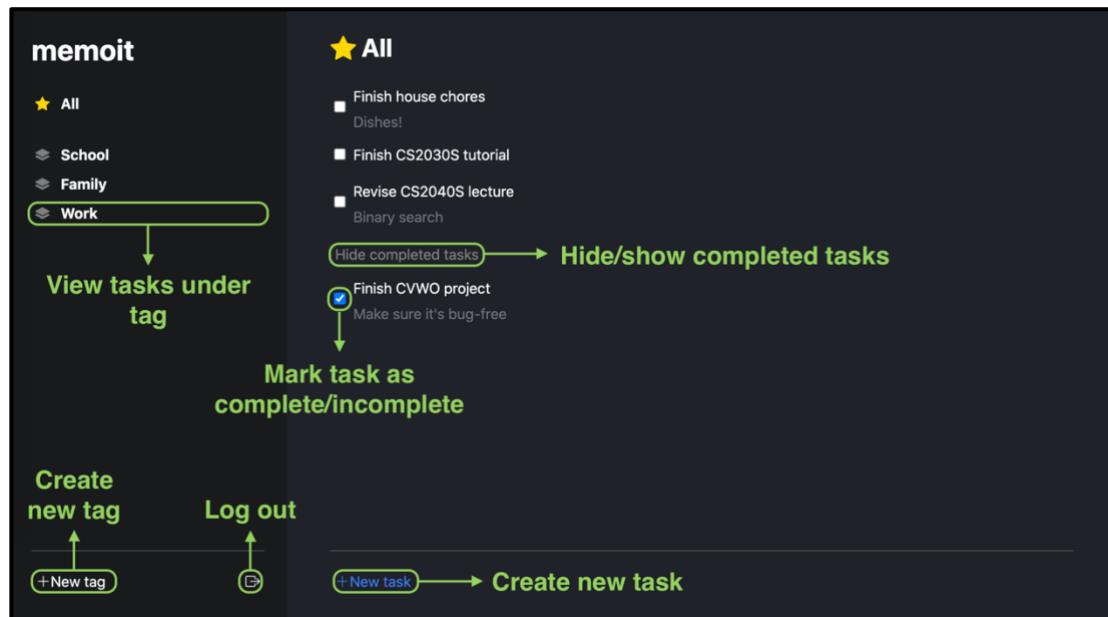
Password Confirmation

Sign Up

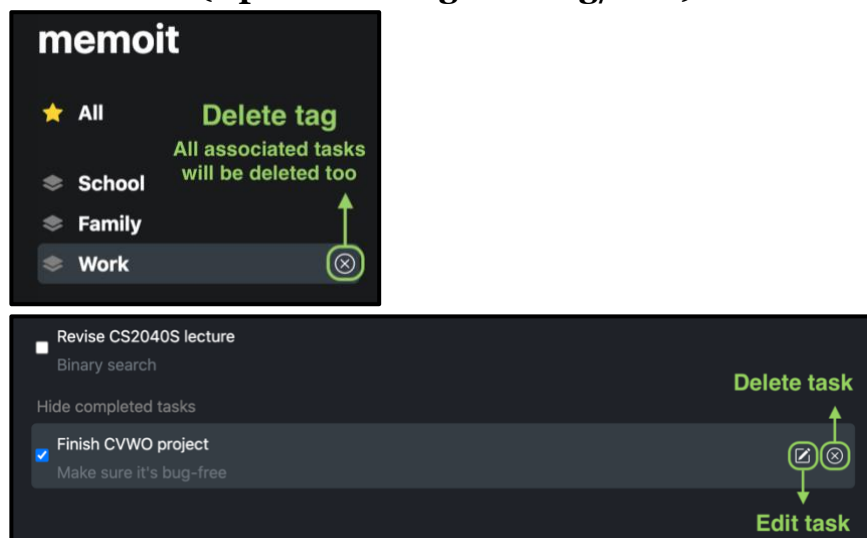
Already on memoit? [Log in here](#)

Navigate to login page

## Dashboard:



## Dashboard (Upon hovering over tag/task):



## New Task:

