National University of Singapore
School of Computing
Computing for Voluntary Welfare Organisations (CVWO) AY2022/23
Assignment: Gossip with Go/Rails

Final Writeup

Name : Tan Jing Sheng
Matriculation Number : A0253492B
Date : 25th January 2023

Thoughts

1. Introduction

Undertaking this assignment has been wonderful. Before commencing the project, I believed that it would be smooth sailing and uncomplicated. This belief was quickly demolished when I begin the actual coding of the project. It was my first application and, I quickly realised that I had been hitting roadblock after roadblock while attempting to implement the "harder than I thought" features. Even all the essential features were implemented, there is still much I would like to implement and improve on, in my application.

2. Project Planning

During the planning of my application, I ambitiously drafted out plans and timelines for my project. I wrote out all the features I would like my application to have and drew out wireframes of how my application would look to have a better view of what was ahead. I divided the entire project into different phases to monitor my progress. I created backup plans and ranked the features of my application in order of essentiality. The project planning allowed me to keep track of my progress and ensure that I am hitting the right milestones at the right time and would not exceed any deadlines while implementing as many features as possible. These processes enabled me to push myself to implement more complex features.

3. React, Redux & More | Frontend

This is the first time I used react. I fell in love with the ease of designing react components. The reusability of components and layers of abstraction accelerated the building of the application as I progressed in the application. I was exposed to other toolkits, libraries, and frameworks such as Redux, Redux-Saga, and React-Router-Dom. Each of these was new to me, I had to read through many documentations and watch many tutorial videos to understand what each of them does before beginning to work on the application. I felt a sense of accomplishment for each additional skill I picked up as I added new functionalities to my frontend.

4. Golang, Gorm & Go-chi | Backend

I had a prior experience with MySQL and MySQL workbench. However, I decided to implement my application with PostgreSQL and use the ORM library GORM as I wanted to learn something new. Before I even started coding in Golang and using ORM, I had to also learn how to design my database and normalise it to reduce data redundancy and improve data integrity. I enjoyed using GORM as it simplifies the code making it more readable and easily implemented. When I moved to the web services portion, I was deciding between gin, echo, mux and chi. I chose chi as it was the most lightweight, idiomatic and composable. Setting up the backend of my application was intriguing. It felt very different from implementing a frontend as there was not much to "look" at, apart from the replies I receive through postman during testing.

5. AWS and Heroku | Deploying

Initially, I planned to deploy and host my application entirely on Amazon Web Services (AWS). However, the learning curve for AWS was very steep and time was not on my side. I only managed to host my frontend on AWS amplify, and my database on AWS RDS. My backend was deployed on Heroku. I have also used AWS Route 53 to route traffic to a domain I registered from the Namecheap domain registrar.

6. Coding Structure

I spent a great deal of time thinking what was the optimal code structure that would fit my application. I ensured that my files and folders were organised neatly into folders all with meaningful names. This allowed me to debug issues more quickly and to understand my code on a higher level. I was also learning how to apply the Single Level of Abstraction Principle (SLAP) properly to ensure that logic and data are not mixed when they are not supposed to be mixed.

7. Future Improvements

I would like to implement account-based authentication with JSON Web Tokens (JWT) as it is more secure than my current implementation. Currently, it saves the user-authenticated state using the Web Storage API to persist the redux store upon refresh so that users do not have to log in again on subsequent visits to the forum. Additionally, I would also like to add more functionalities, such as logging in with Gmail and 2FA, customising user profiles and pages, and creating sub-forums for different topics.

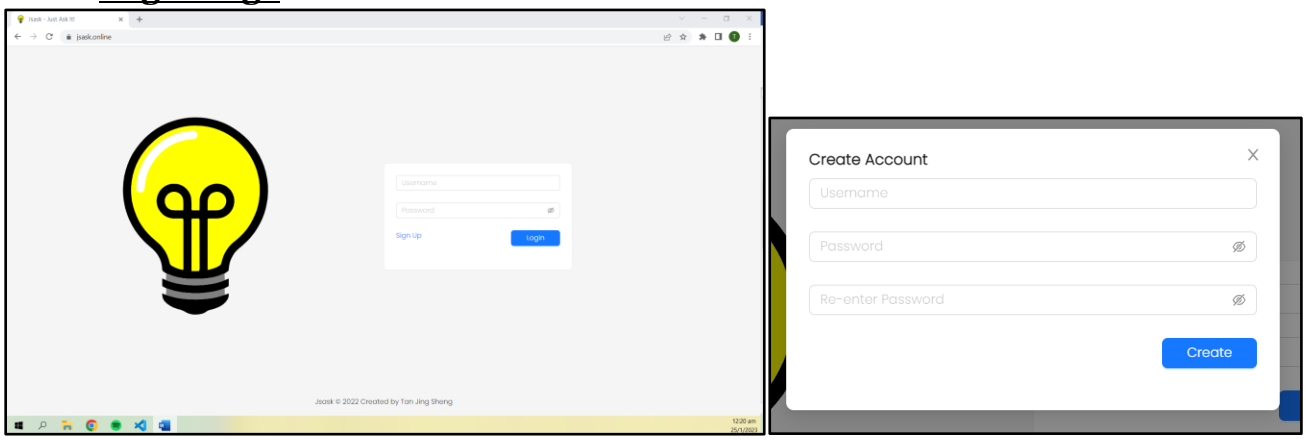
8. Conclusion

I have learnt immensely doing this assignment. Developing and deploying an application of this scale was a new and enriching experience. If you were to tell the "me" from eight weeks ago that I could accomplish what I did for this assignment, the "me" would have laughed. If given a chance, I would definitely do an assignment like this again. Many thanks to the CVWO team that put together this assignment.

User Manual

The Jsask Forum can be accessed at "https://www.jsask.online/".

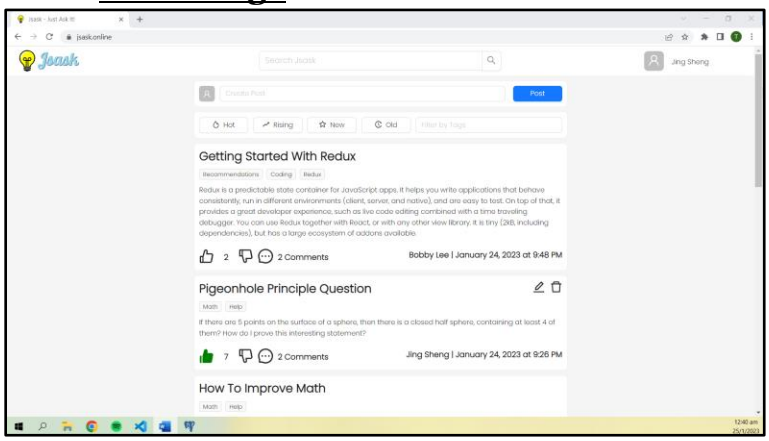
1. Login Page

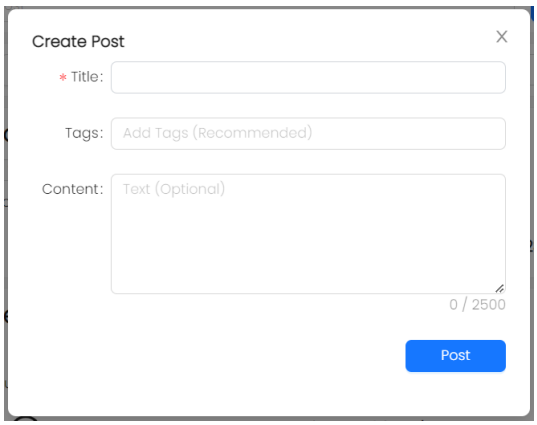


Upon visiting the URL, the user will be directed to this landing page (Refer to image, top left). Users may log in with their credentials or create an account by clicking "Sign Up". Clicking "Sign Up" will render a modal with a create user form (Refer to image top right).

- **All users are to be authenticated before viewing or interacting with the posts.**
- **All users' usernames are unique**
- **All passwords are salted and hashed with SHA512 before being dispatched.**

2. Home Page

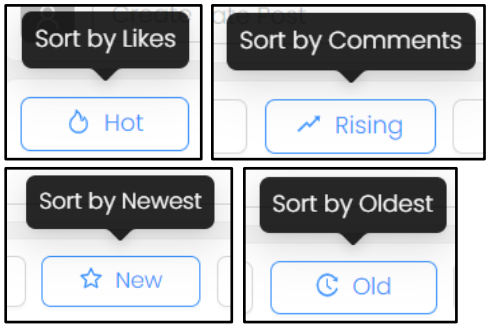


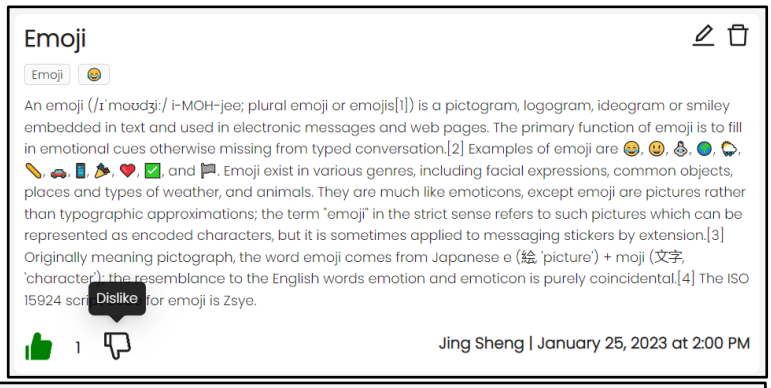


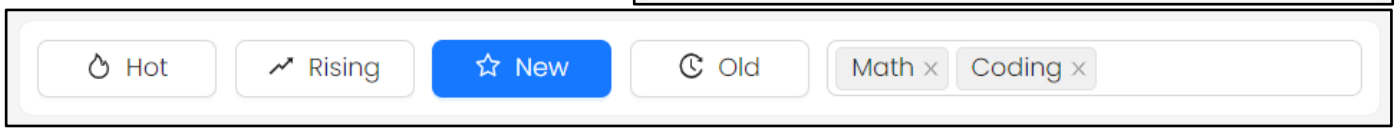
Upon Authentication, users will be directed to the home page (Refer to image, top left). Users may log out by clicking the logout button or view their profile by clicking the profile button in the menu.

On this page, users can:

- **Create posts**
- **Edit/Delete posts***
- **Like/Unlike/Dislike/Undislike posts** | Tooltips visible upon hover (Refer to image, below)
- **Sort posts by a variety of options.** | Tooltips visible upon hover, (Refer to image, below)
- **Filter posts by tags** | Type out the tag, Select from a dropdown or Click any of the tags in a post
- **Click into a post** | User will be directed to the Post Page

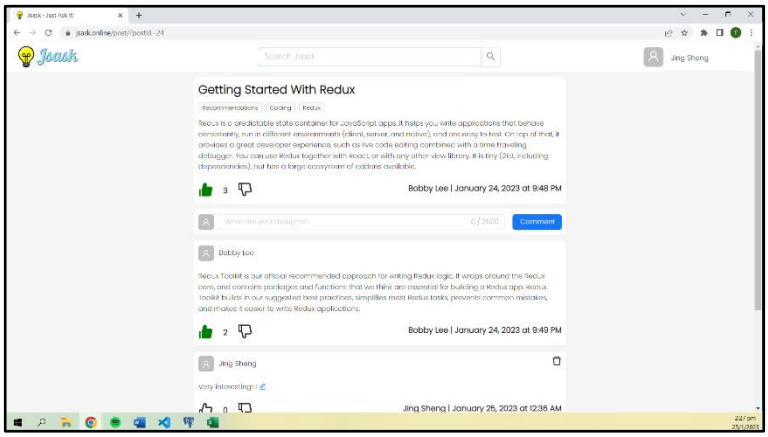






Post Page

Upon clicking on a post or the comment icon on the Home Page, users will be directed to the Post Page (Refer to image, right). The user will see the post they clicked, followed by the comments made on the particular post.



On this page users can:

- **Edit/Delete the post***
- **Comment on the post**
- **Like/Unlike/Dislike/Undislike the post or its comments.**

*Users can only edit/delete posts they created, editable/deletable posts will have an “Edit” and “Delete” Icon to edit/delete the post