CSE134B HW4 Summary

Features Implemented

Authentication

We implemented a login system using Firebase and the login system allows users to create an account through email or by logging into their Google account.

Create

Users can create a favorite meal by clicking on the ‘Find Meals’ option on the menu bar to navigate to the search page. Once on the search page, the user can input a search query such as ‘Salad’, click the search button, and see the results. Next to each result is a ‘Add to Favorites’ button that will add the meal to the user’s favorites list which is stored in the database.

Read

Users can read their stored favorite meals from the homepage. If the user has added two favorite meals, the user will see those two added favorite meals on the home page. The user will also be able to see the information that is stored for each favorite meal.

Update

The user can update the ranking of each meal on the homepage by clicking on the button ‘Rank Up Meal’. This will increase the selected favorite meal’s ranking and move it up in the list of favorite meals in the database.

Delete

The user can also delete a favorite meal on the homepage by clicking on the button ‘Remove’. This will delete the favorite meal for that user from the database.

Implementation Process and Challenges

For the login system, we took full advantage of Firebase built in authentication system. We could simply call Firebase defined functions accessed through ‘Firebase.auth()’ to log the user in and out. Firebase even provided functions to login a user through Google. A challenge we faced was getting the current user’s information. Get the current user information was essential for the CRUD actions since we needed to read and write data based on who is currently logged in. The problem was that sometimes the current user data would be null despite there being a user logged in. What we found out was that the Firebase auth object could be in an intermediate state when we use it in one of our functions. The solution was to use an observer provided by Firebase which would be able to successfully get the current user’s data after it was initialized.

To implement the CRUD actions and display each user's information on the homepage, we used a combination of JavaScript, Vue, and Firebase. To search for meals, we used a jQuery post to an API we found online that provided millions of meal information/recipes. Once we received the data, we created a Vue instance that would render the data we received from the API to the DOM. We also used the Vue instance for handling events like when the user clicks 'Add to Favorites' button. Inside the event handler methods, we used Firebase to complete the CRUD actions based on what the user clicked on. The main challenge when implementing CRUD actions was learning how to use all the new technologies. The basics of Vue was simple to pick up, but a lot of experimentation and documentation reading was involved in order to successfully have our app do what we wanted it to do. We also never had experience getting data from online API's so it also took quite a long time before we figured out how to get from retrieving the data from the online API to rendering the data to the DOM. Firebase was the easiest aspect to learn and the read/write operations were simple to use. The structure of the database was familiar to us as well since it was basically JSON.

Optimization

We barely finished the core functionality of our app after much experimentation with Vue and Firebase frameworks and we realized that there is much to be optimized. For example, before I found out about Vue event handlers, our other JavaScript functions didn't have access to the data being rendered on the page. Our first solution was to add hidden inputs everywhere. After learning about Vue event handlers, it was much easier to handle onclick events and access the rendered data. Another aspect we can still optimize is how we read and write data. Currently, our app is reading all of the user's data and then setting all of the user's data for the CRUD actions. Right now the amount of data for each user is tiny, but once the user starts storing lots of information, it will probably slow down the CRUD actions. We need to learn better how to use Firebase references and other database functions other than just set. Another aspect of our app we can optimize is the JavaScript files that we include. We can improve performance by storing Firebase JavaScript files in our app instead of using CDN links and by choosing which specific Firebase JavaScript files we need.