Major Project: User Profile

Settings Functionality

Settings UI :

In this session, we created UI for the settings page through which users can update their details like a name. Then we defined the route for the settings page in app.js and added a link for the settings page in the navbar.

Settings Functionality :

Here in this session, we created event handlers on buttons and their corresponding functions. Then applied a condition over them that if settings are in edit form then show form else show details of the user.

When we updated the profile the Navbar reflects the old name after refreshing because user info is extracted from the Authentication Token present in local storage which contains the old user info and we were not updating the token.

Route Issues

The two issues that we were facing here were:

1. If the user is logged in, the user should not see the login and register page.
2. If the user is logged out, then the user should not see the settings profile page.

Fixing the first issue:

In this session, we found out that if we try to access the settings which is a private route, the application takes us to the login page. But when the authenticated user logs in it take to the home page. So we solved this issue by adding a dynamic object in the Redirect component which ensures that if a user logs in he goes to the page for which he requested before logging in.

Fixing the second issue:

If the user is logged in then he/she should not be able to access login and signup pages but he/she should be able to access the settings which is a private route. For solving this issue we created a Private Route Component with this logic: if a user is logged in then we will use the render prop for the children and return the requested route else we will redirect the user to the login/signup page.

Adding User Profile

UI of the User Profile:

In this session, we created the UI for the user profile page which is similar to the settings page. But we only need to show the name, email, and a button for adding or removing friends depending upon whether the user is already a friend or not. And user profile page is also a private route. The details that are going to be displayed are extracted from the user id present in the path. We used <Link> component here.

Improving User Profile Logic:

In this session, we added the loader functionality and we used the useParams hook for getting the user id from the URL. After grabbing the User Id, an API call is made for fetching the user profile. We also used react-toast-notifications module here for getting notifications when an event is handled. We used useHistory hook here which will redirect you to the Home Page if the user is not logged in.

Friendships

Check whether a user is a friend or not:

For this, we simply created a function in which we fetched user_ids of friends and checked for the user_id of a particular user in that array. If the user is a friend then we replace the Add Friend button with the Remove Friend button for that particular user.

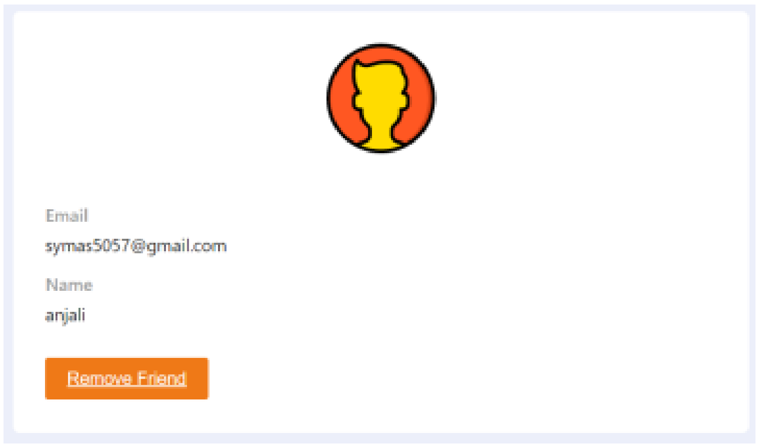
Strategy for Adding and Removing Friends:

Firstly, an API call will be made which will return a friendship Object as a response, and then we will add that response to user.friends. And then we will update the state accordingly.

For removing a friend we will search that user id in the user.friends object and when it is found we will remove that user id from the array. Then the state gets updated.

For adding a friend we do not need to search. We will add that user id at the end of the array.

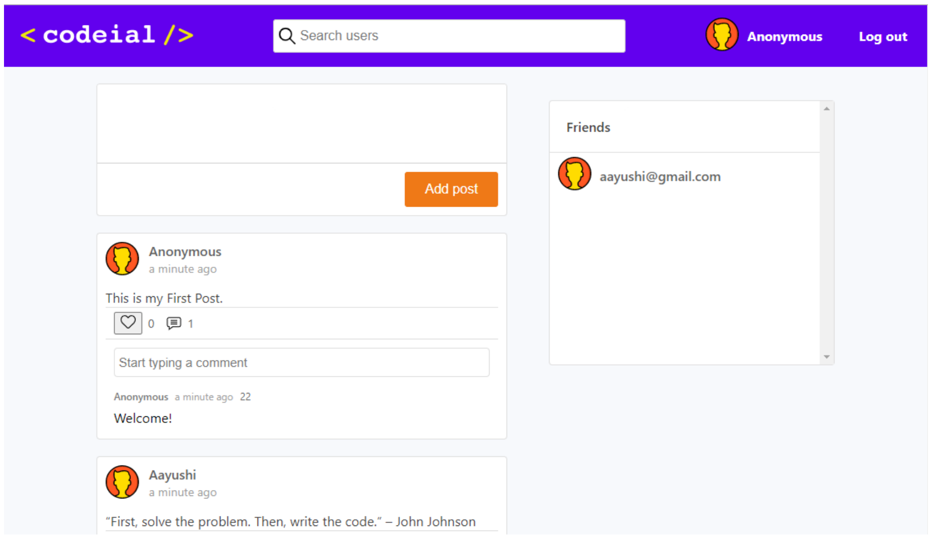
But there is an issue here, when we will refresh the page we will not to able to see the updated data because the token only has user information. So, what we will do here is we will fetch user friends by making another API call. And we will store the friends in the user object and then the token gets decoded and after that state will be updated.



Email
symas5057@gmail.com
Name
anjali

Creating Friend List:

We created a new component for friendlist and then called an API request which will fetch data.data.friends where the initial state is in an empty array and it returns the list of friends. After this, we created two components friendList.js and friendListItem.js. In friendList we have the condition if friendslist. length is zero then “No friends found!” else we mapped over the friend’s list. In friendListItem.js we simply had a user profile picture and email id.



Anonymous
a minute ago
This is my First Post.
0 1
Start typing a comment
Anonymous a minute ago 22
Welcome!

Aayushi
a minute ago
“First, solve the problem. Then, write the code.” – John Johnson

Friends
aayushi@gmail.com

Summarising It

Let’s summarise what we have learned in this module:

- Learned about how the flow goes between things to create settings functionality.
- Learned about adding user-profiles and adding friendships in the application.
- Learned about useEffect and other miscellaneous hooks.