Major Project: Search And Chat

Post Functionality

Post UI:

Firstly we created a post UI and added a text area and add-post button. Then we created hooks for adding posts. With this, we completed the post UI part.

Post Functionality:

Here in this session, we created event handlers on buttons and their corresponding functions. Then fetched the API call for adding posts and fetched that in the browser. We also added toast notifications.

Then we fixed the issue due to which we were unable to see the newly added posts by creating providers and context API. (similar to auth provider) and making our post global. We also added loader functionality when a post will be newly added.

Comment Functionality:

For comment functionality, When a comment is added, an API is called. If the call is successful the state of posts is updated and the comment gets concatenated in the list of comments.

Like Functionality:

For like functionality, we created a toggle like function on the like button which toggles the button with a boolean variable which accepts two parameters itemId and itemType. This functionality was added to both the post and comment of the user. So, In itemId we can specify whether it is a comment or a post. But due to the limitation of the api, we cannot update the like count but it works the same way with any other api.

Search Functionality:

For implementing Search, We first created its UI and then we created a hook for search query. Then we used useEffect hook for fetching the data when user enters the query in the search box. If the response is successful then we will display the results. If search query's length is greater than 2 only then we will display the search results.

Some Important Concepts

We don't really need to perform API requests and DOM updates on each letter typed. We need to wait until the user either stop typing or types for a considerable amount of time. We need to wait until the user scrolls for an amount of time or stops scrolling. This will ensure better performance of our web app. So, For this some techniques are used which are mentioned below:

- **Debouncing** enforces that a function won't be called again until a certain amount of time has passed without it being called.
- **Throttling** enforces a maximum number of times a function can be called over time.

Summarising It

Let's summarise what we have learnt in this module

- Learned about how the flow goes for post UI.
- Learned about Context API and various functionalities.
- Learned about Throttling and Debouncing.