Interactive Challenge / Mini-Project

Your Turn!

Fetch a list of posts from https://jsonplaceholder.typicode.com/posts using useQuery.

- Add a form to create a new post using useMutation .
- Display loading and error states for both queries and mutations.
- Use DevTools to monitor active queries and mutations.
- Bonus: Implement optimistic updates for the post creation form.

Your Turn!

1. Set Up Your Project

- Create a new React app.
- Install React Query (or SWR) and Axios (optional, you can use fetch).
- Set up the QueryClient and wrap your app with QueryClientProvider .

  2. Fetch and Display Data

- Use the useQuery hook to fetch a list of posts from https://jsonplaceholder.typicode.com/posts .
- Display the posts in a simple list.
- Show a loading state while the data is being fetched.
- Show an error message if the fetch fails.

  3. Add a Form to Create a New Post

- Create a form with fields for title and body .

- Use the useMutation hook to post the new data to https://jsonplaceholder.typicode.com/posts .
- After submitting the form, display a success message and update the list of posts.
4. Implement Optimistic Updates
- When you submit the form, immediately add the new post to the list before the server responds.
- If the server request fails, remove the optimistically added post.
- Show a loading indicator while the mutation is in progress.

5. Invalidate the Cache

- After a successful mutation, invalidate the posts query so the list is refreshed and up-to-date.
- Alternatively, update the cache manually with the response from the server.

6. Bonus: Use DevTools

- Add React Query DevTools to your app to monitor and debug your queries and mutations.

Your Turn!

- Implement a data-fetching hook that uses a type-safe cache key and supports a stale-while-revalidate strategy.
- Use useMemo to cache expensive computations in a component.
- Experiment with React Query or SWR to see how cache keys and revalidation affect UI responsiveness.
- Try invalidang the cache and observe how fresh data is fetched and displayed.