useUserData

Overview

The `useUserData` hook provides a simple interface for accessing and updating a user object within your component. It wraps `useUpdateUser` to offer a higher-level API for updating user details, while also managing local state through a provided `setUser` function. It is useful in components that already have user data and want to support real-time updates (e.g., updating a name, avatar, or bio).

Usage Example

```
import { useUserData } from "@replyke/react-js";

function ProfileSettings({
  user,
  setUser,
}: {
  user: UserLean | null;
  setUser: (newUser: UserLean) => void;
}) {
  const { updateUser } = useUserData({ user, setUser });

  const handleChangeName = async () => {
    try {
      await updateUser({ name: "New Name" });
    } catch (err) {
      console.error("Failed to update user:", err);
    }
  };

  return (
    <div>
      <button onClick={handleChangeName}>Change Name</button>
    </div>
  );
}
```

Parameters & Returns

Parameters

The hook accepts an object with the following fields:

Parameter	Type	Required	Description
`user`	`UserLean | null`	Yes	The user object to be managed and updated.
`setUser`	`(newUser: UserLean) => void`	Yes	Function to update the local user state after a successful mutation.

Returns

Return Value	Type	Description
`user`	`UserLean | null | undefined`	The current user object passed to the hook.
`updateUser`	`(update: UpdateUserParams) => Promise<void>`	Function to update the user by passing a partial update object. Automatically applies the changes to local state via `setUser`.

Last updated on May 8, 2025