Plataforma de Usuarios con Preferencias

Objetivo: Crear una pequeña aplicación que maneje el estado global de usuarios con Redux, la autenticación simulada con useContext, rutas protegidas con react-router-dom, persistencia de preferencias del usuario en localStorage, y uso de useState y useMemo para optimizar el renderizado de una lista.

Requerimientos:

Rutas (React Router):

1. /login: Página de inicio de sesión.
2. /dashboard: Página principal que muestra una lista de usuarios.
3. La ruta /dashboard es una ruta protegida: solo accesible si el usuario está autenticado.

Autenticación (useContext):

1. Crear un contexto AuthContext que contenga información sobre si el usuario está autenticado.
2. Un componente AuthProvider que use useState para mantener el estado de autenticación (isAuthenticated: boolean) y un método login() que cambie este estado a true.
3. En la página de login, al enviar un formulario falso, se llamará a login() del contexto.

Estado Global (Redux):

1. Un store de Redux que contenga una lista de usuarios en su estado: users: [{id: 1, name: 'Alice'}, {id:2, name:'Bob'}, ...].
2. Un reducer para manejar acciones que modifiquen o filtren a los usuarios.
3. Una acción LOAD_USERS que cargue usuarios predefinidos al iniciar la app.
4. Un componente Dashboard que use useSelector para obtener la lista de usuarios del store.

LocalStorage:

1. Al iniciar la aplicación, leer desde localStorage si hay alguna preferencia del usuario, por ejemplo, el número de usuarios a mostrar por página (itemsPerPage) o si prefiere mostrar en modo "dark".
2. Estas preferencias se guardan y leen en localStorage para que persistir entre recargas.
3. Un pequeño panel en el Dashboard para cambiar la preferencia itemsPerPage (ej: 5, 10, 20) y guardarla en localStorage.

useMemo:

En el Dashboard, al renderizar la lista de usuarios, usar useMemo para calcular la lista paginada, de forma que si itemsPerPage o la página actual cambian, se recalcula la lista, pero no en otros casos.

De esta manera se evita recalcular filtrados o paginaciones costosas si otros estados no relacionados cambian.

Flujo sugerido:

1. El usuario entra a /login.
2. Al hacer clic en un botón "Iniciar Sesión", se setea isAuthenticated a true en el contexto y se navega a /dashboard.
3. En /dashboard, se obtiene la lista de usuarios del store de Redux.
4. Se lee itemsPerPage desde localStorage. Si no existe, se usa un valor por defecto (ej: 10).
5. Un menú en el Dashboard permite cambiar itemsPerPage. Al cambiarlo, se actualiza el estado local, se guarda en localStorage y se recalcula la lista mostrada.
6. useMemo se utiliza para memorizar la lista de usuarios paginada según itemsPerPage y la página actual.
7. Si el usuario no está autenticado y trata de ir a /dashboard, se redirige de vuelta a /login.

Retos opcionales:

- Añadir un buscador para filtrar usuarios por nombre y usar useMemo para no recalcular el filtrado a menos que cambie el término de búsqueda.
- Añadir un botón de "Cerrar sesión" que cambie isAuthenticated a false y redirija a /login.