FINNEGANS

Componente SearchBar

Sprint deadline	03/11/2023
Tarjeta	SCRUM-34
Responsable	Schwerdt, Oscar Andrés

Tabla 1 Detalle de la tarjeta correspondiente en Jira

1. Objetivos. Contenido de la tarjeta:

- Realizar un componente de búsqueda que pueda ser reutilizable por otros componentes.

2. Dependencias

Fue necesario solamente la implementación del repositorio con un proyecto base. Luego la tarea fue realizada en paralelo con las demás tarjetas, ya que consistió en la adición de un componente aislado, que si bien puede ser utilizado en cualquier lugar, no depende de nada externo a sí mismo.

3. Procedimientos

3.1. Diseño del componente

El componente fue pensado para poder ser lo más aséptico posible, es decir, que no dependa de ningún factor externo a él mismo. Para lograrlo se definieron 3 propiedades que van a ser inyectadas por el componente:

- **rawList:** esta propiedad recibe un array con la lista original de objetos a filtrar. En general esta lista será extraída del estado global de la aplicación.
- **setFilteredList:** en esta propiedad se recibe una referencia a una función del tipo **setState**, que se encargará de enviar al componente padre una lista actualizada de los elementos filtrados por la búsqueda.
- **compareTag:** aquí se recibe un string con el nombre de la propiedad por la cual se quiere filtrar la lista, por ej, si se quiere filtrar una lista de usuarios por su nombre (y como la propiedad correspondiente de ese objeto es username), desde el componente padre se debe enviar “username”.

Con estas propiedades se pudo lograr la autonomía del componente y garantizar la fácil reutilización del mismo.

4. Resultados

La aplicación ahora cuenta con un sistema de búsqueda totalmente reutilizable para su utilización en cualquier componente que lo necesite.

4.1. Especificaciones de uso

Para poder utilizar el SearchBar primero se debe preparar el componente padre de la siguiente manera:

1. Crear un estado local (**useState)** para almacenar la lista filtrada por el componente, cuyo valor inicial corresponderá a la lista original tomada del estado global o de donde fuere.

```
const [filteredList, setFilteredList] = useState(users);
```

Ejemplo tomado del componente UserListComponent.jsx

2. Insertar el componente donde se lo quiera utilizar, mandando las propiedades correspondientes.

```
<SearchBar
  rawList={users}
  setFilteredList={setFilteredList}
  compareTag="username"
/>
```

Ejemplo tomado del componente UserListComponent.jsx

3. Cambiar la lista que va a ser renderizada. En vez de utilizar la lista original se debe utilizar la lista filtrada en su lugar.

Ejemplo tomado del componente UserListComponent.jsx

Con estos pasos se podrá dejar el componente funcionando correctamente.

5. Conclusión

Ahora la aplicación cuenta con la posibilidad de agregar, a cualquier componente que lo necesite, una barra de búsqueda con la capacidad de filtrar una lista en base al input del usuario. Esta característica es muy importante para mejorar la experiencia de usuario, ya que trabajar con listas prolongadas suele ser entorpecedor a la hora de acceder rápidamente a la información.