GABRIELE ROMANATO

Come creare un filtro di ricerca per i dati in React

React è una delle librerie JavaScript più popolari per la creazione di interfacce utente dinamiche e reattive. Quando si lavora con grandi set di dati, può essere estremamente utile implementare un filtro di ricerca per consentire agli utenti di trovare rapidamente le informazioni di loro interesse. In questo articolo, esploreremo come creare un filtro di ricerca per i dati in una applicazione React.

Creare il componente dei dati

Per scopi di esempio, creeremo un array di oggetti JSON che rappresenteranno i dati da filtrare. Questi dati possono essere sostituiti con un'API esterna o un database, a seconda delle esigenze del tuo progetto.

```
// src/data.js

const data = [
  { id: 1, name: "Prodotto A", category: "Elettronica" },
  { id: 2, name: "Prodotto B", category: "Abbigliamento" },
  { id: 3, name: "Prodotto C", category: "Elettronica" },
  { id: 4, name: "Prodotto D", category: "Casa" },
  // Aggiungi più dati qui
];

export default data;
```

Creare il componente di filtro

Creeremo un componente React chiamato `Filter` che consentirà agli utenti di inserire una query di ricerca e visualizzerà solo i dati che corrispondono alla query. Ecco come puoi farlo:

```
// src/Filter.js

import React, { useState } from "react";

const Filter = ({ data, onDataFiltered }) => {
  const [query, setQuery] = useState("");

  const handleInputChange = (e) => {
    const newQuery = e.target.value;
    setQuery(newQuery);

    const filteredData = data.filter((item) =>
      item.name.toLowerCase().includes(newQuery.toLowerCase())
    );

    onDataFiltered(filteredData);
  };

  return (
    <div>
```

```
      <input
        type="text"
        placeholder="Cerca..."
        value={query}
        onChange={handleInputChange}
      />
    </div>
  );
};

export default Filter;
```

Nel componente `Filter`, utilizziamo lo stato locale per gestire la query di ricerca. Ogni volta che l'utente modifica l'input di ricerca, filtriamo i dati in base alla query e utilizziamo la funzione `onDataFiltered` per inviare i dati filtrati al componente principale.

Creare il componente principale

Ora creeremo il componente principale in cui verranno visualizzati i dati filtrati. Il componente principale importerà sia i dati che il componente di filtro e passerà i dati filtrati come prop al componente di visualizzazione.

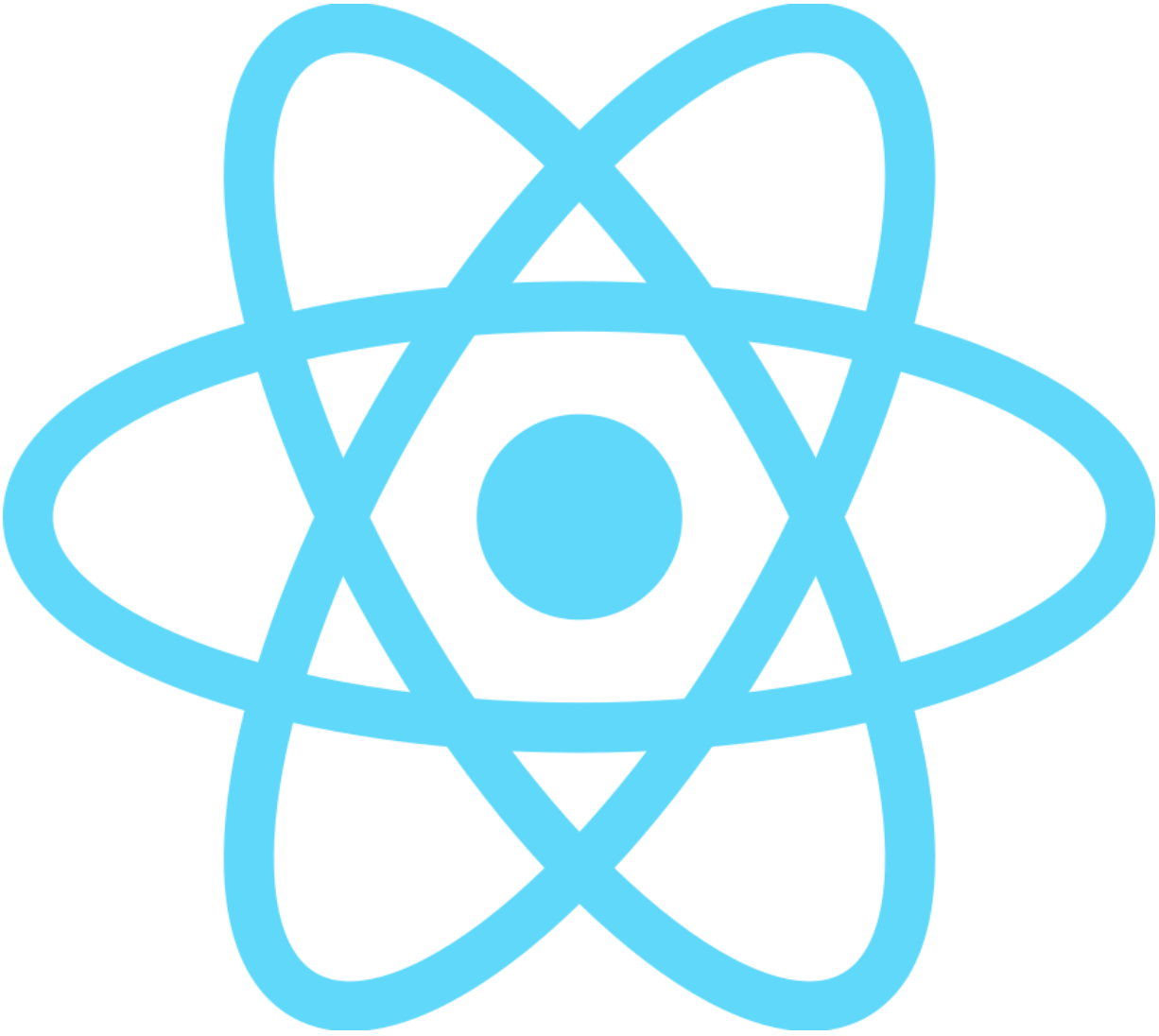
```
// src/App.js

import React, { useState } from "react";
import data from "./data";
import Filter from "./Filter";

const App = () => {
  const [filteredData, setFilteredData] = useState(data);

  const handleDataFiltered = (filteredData) => {
    setFilteredData(filteredData);
  };

  return (
    <div>
      <h1>Filtro di Ricerca</h1>
      <Filter data={data} onDataFiltered={handleDataFiltered} />
      <ul>
        {filteredData.map((item) => (
          <li key={item.id}>
            {item.name} - Categoria: {item.category}
          </li>
        ))}
      </ul>
    </div>
  );
};

export default App;
```

In questo componente principale, importiamo sia i dati che il componente `Filter`. Utilizziamo lo stato per gestire i dati filtrati e passiamo la funzione `handleDataFiltered` al componente `Filter` per ricevere i dati filtrati.

Conclusioni

La creazione di un filtro di ricerca per i dati in React è un modo efficace per consentire agli utenti di trovare rapidamente le informazioni di loro interesse. Con la giusta implementazione, è possibile migliorare l'usabilità delle tue applicazioni e fornire un'esperienza utente più positiva. Speriamo che questo articolo ti abbia aiutato a comprendere come realizzare un filtro di ricerca in una applicazione React.

Applicazioni Correlate



- **Book Store**

  Un'applicazione in React con Node.js e Fastify come backend.
  DockerDocker ComposeNode.jsJavaScriptFastifyReact