

ChatGPT 3.5

JA

```
import React, { useState } from "react"; const MyPantryList = ({ foodList, onDelete }) => {
const [selectedItems, setSelectedItems] = useState([]); const handleCheckboxChange =
(itemId) => { if (selectedItems.includes(itemId)) { setSelectedItems(selectedItems.filter((id)
=> id !== itemId)); } else { setSelectedItems([...selectedItems, itemId]); } }; const
handleDelete = () => { onDelete(selectedItems); setSelectedItems([]); }; const isExpired =
(bestBeforeDate) => { const today = new Date(); const expirationDate = new
Date(bestBeforeDate); return today > expirationDate; }; if (foodList.length === 0) { return
null; // Wenn die Liste leer ist, nichts anzeigen } const columnHeaders = ["ID", "Name",
"Amount", "Unit", "Categorie", "Best Before Date", "Actions"]; return ( <div
className="card-container" style={{ marginBottom: "20px" }}> <table> <thead> <tr>
{columnHeaders.map((header, index) => ( <th key={index}>{header}</th> ))} </tr>
</thead> <tbody> {foodList.map((foodItem) => ( <tr key={foodItem.id}>
<td>{foodItem.id}</td> <td>{foodItem.name}</td> <td>{foodItem.amount}</td>
<td>{foodItem.unit}</td> <td>{foodItem.categorie}</td> <td style={{ color:
isExpired(foodItem.bestBefore) ? 'red' : 'black' }}> {foodItem.bestBefore}
{isExpired(foodItem.bestBefore) && <span> - abgelaufen</span>} </td> <td> <input
type="checkbox" onChange={() => handleCheckboxChange(foodItem.id)}
checked={selectedItems.includes(foodItem.id)} /> </td> </tr> ))} </tbody> </table> <div
className="buttons"> <button class="deleteb" onClick={handleDelete}>Delete
Selected</button> </div> </div> ); }; export default MyPantryList; wie kann man die Liste
nach columnheaders sortieren? Also wenn man auf ein columnheader drückt, soll die ganze
Liste sich anhand der Eigenschaft sortieren.
```

ChatGPT

Um die Liste nach den Spaltenüberschriften zu sortieren, kannst du den `useState`-Hook verwenden, um den aktuellen Sortierstatus zu verfolgen. Hier ist eine mögliche Implementierung:

```
jsx
import React, { useState } from "react";

const MyPantryList = ({ foodList, onDelete }) => {
  const [selectedItems, setSelectedItems] = useState([]);
  const [sortConfig, setSortConfig] = useState({ key: null, direction:
'asc' });

  const handleCheckboxChange = (itemId) => {
    if (selectedItems.includes(itemId)) {
      setSelectedItems(selectedItems.filter((id) => id !== itemId));
    } else {
      setSelectedItems([...selectedItems, itemId]);
    }
  };

  const handleDelete = () => {
    onDelete(selectedItems);
    setSelectedItems([]);
  };

  const isExpired = (bestBeforeDate) => {
    const today = new Date();
    const expirationDate = new Date(bestBeforeDate);
    return today > expirationDate;
  };
};
```

```

const requestSort = (key) => {
  let direction = 'asc';
  if (sortConfig.key === key && sortConfig.direction === 'asc') {
    direction = 'desc';
  }
  setSortConfig({ key, direction });
};

const sortedList = () => {
  let list = [...foodList];
  if (sortConfig.key !== null) {
    list.sort((a, b) => {
      if (a[sortConfig.key] < b[sortConfig.key]) {
        return sortConfig.direction === 'asc' ? -1 : 1;
      }
      if (a[sortConfig.key] > b[sortConfig.key]) {
        return sortConfig.direction === 'asc' ? 1 : -1;
      }
      return 0;
    });
  }
  return list;
};

const columnHeaders = ["id", "name", "amount", "unit", "categorie",
"bestBefore", "actions"];

return (
  <div className="card-container" style={{ marginBottom: "20px"}}>
    <table>
      <thead>
        <tr>
          {columnHeaders.map((header, index) => (
            <th key={index} onClick={() => requestSort(header)}>
              {header}
              {sortConfig.key === header && (
                <span>{sortConfig.direction === 'asc' ? '▲' : '▼'}</span>
              )}
            </th>
          ))}
        </tr>
      </thead>
      <tbody>
        {sortedList().map((foodItem) => (
          <tr key={foodItem.id}>
            {/* ... rest of the code ... */}
          </tr>
        ))}
      </tbody>
    </table>
    <div className="buttons">
      <button className="deleteb" onClick={handleDelete}>Delete
Selected</button>
    </div>
  </div>
);
};

export default MyPantryList;

```

In diesem Beispiel wird die Funktion `requestSort` aufgerufen, wenn auf eine Spaltenüberschrift geklickt wird. Diese Funktion aktualisiert den Status `sortConfig` und löst eine Neusortierung der Liste aus. Der Pfeil (▲ oder ▼) neben der Spaltenüberschrift zeigt die Sortierrichtung an.

JA

kannst du den Code komplett azeige, also auch den wo rest of the code steht?

ChatGPT

Sicher, hier ist der vollständige Code mit dem fehlenden Abschnitt:

```
jsx
import React, { useState } from "react";

const MyPantryList = ({ foodList, onDelete }) => {
  const [selectedItems, setSelectedItems] = useState([]);
  const [sortConfig, setSortConfig] = useState({ key: null, direction: 'asc' });

  const handleCheckboxChange = (itemId) => {
    if (selectedItems.includes(itemId)) {
      setSelectedItems(selectedItems.filter((id) => id !== itemId));
    } else {
      setSelectedItems([...selectedItems, itemId]);
    }
  };

  const handleDelete = () => {
    onDelete(selectedItems);
    setSelectedItems([]);
  };

  const isExpired = (bestBeforeDate) => {
    const today = new Date();
    const expirationDate = new Date(bestBeforeDate);
    return today > expirationDate;
  };

  const requestSort = (key) => {
    let direction = 'asc';
    if (sortConfig.key === key && sortConfig.direction === 'asc') {
      direction = 'desc';
    }
    setSortConfig({ key, direction });
  };

  const sortedList = () => {
    let list = [...foodList];
    if (sortConfig.key !== null) {
      list.sort((a, b) => {
        if (a[sortConfig.key] < b[sortConfig.key]) {
          return sortConfig.direction === 'asc' ? -1 : 1;
        }
        if (a[sortConfig.key] > b[sortConfig.key]) {
          return sortConfig.direction === 'asc' ? 1 : -1;
        }
        return 0;
      });
    }
    return list;
  };
}
```

```

    };

    const columnHeaders = ["id", "name", "amount", "unit", "categorie",
    "bestBefore", "actions"];

    return (
      <div className="card-container" style={{ marginBottom: "20px"}}>
        <table>
          <thead>
            <tr>
              {columnHeaders.map((header, index) => (
                <th key={index} onClick={() => requestSort(header)}>
                  {header}
                  {sortConfig.key === header && (
                    <span>{sortConfig.direction === 'asc' ? '▲' : '
▼'}</span>
                    )}
                </th>
              ))}
            </tr>
          </thead>
          <tbody>
            {sortedList().map((foodItem) => (
              <tr key={foodItem.id}>
                <td>{foodItem.id}</td>
                <td>{foodItem.name}</td>
                <td>{foodItem.amount}</td>
                <td>{foodItem.unit}</td>
                <td>{foodItem.categorie}</td>
                <td style={{ color: isExpired(foodItem.bestBefore) ? 'red' :
'black' }}>
                  {foodItem.bestBefore}
                  {isExpired(foodItem.bestBefore) && <span> -
abgelaufen</span>}
                </td>
                <td>
                  <input
                    type="checkbox"
                    onChange={() => handleCheckboxChange(foodItem.id)}
                    checked={selectedItems.includes(foodItem.id)}
                  />
                </td>
              </tr>
            ))}
          </tbody>
        </table>
        <div className="buttons">
          <button className="deleteb" onClick={handleDelete}>Delete
Selected</button>
        </div>
      </div>
    );
  };

export default MyPantryList;

```

Hier habe ich den Teil ergänzt, der die Daten in den Tabellenzeilen rendert. Dieser Abschnitt enthält die Zeilen für jede Spalte des Lebensmittelartikels.