

# React Recipe Website

## User

Ich will eine React-Website erstellen, die Rezeptvorschläge ausgibt (angebunden über eine API) auf Grundlage von eingaben in einem Textfeld durch den Benutzer. Wie wollte ich die App aufbauen und welche Komponenten sind relevant?

## ChatGPT

Um eine React-Website zu erstellen, die Rezeptvorschläge auf Grundlage von Benutzereingaben aus einem Textfeld über eine API abrufen, benötigst du mehrere Komponenten. Hier ist eine grundlegende Struktur, die du als Ausgangspunkt verwenden kannst:

### 1. **\*\*App-Komponente (App.js):\*\***

- Hier wird die gesamte Anwendung gerendert.
- Hält den Zustand für die Benutzereingaben und die Rezeptvorschläge.

```
```jsx
```

```
import React, { useState } from 'react';
```

```
import SearchBar from './SearchBar';
```

```
import RecipeList from './RecipeList';
```

```
function App() {
```

```
  const [userInput, setUserInput] = useState('');
```

```
  const [recipes, setRecipes] = useState([]);
```

```
  const handleSearch = async () => {
```

```
    // Hier wird die API-Aufruf-Logik implementiert
```

```
    // Verwende userInput, um die API-Anfrage zu erstellen
```

```
    // Setze dann die erhaltenen Rezepte mit setRecipes
```

```
  };
```

```
  return (
```

```
    <div>
```

```

    <h1>Rezeptvorschläge</h1>

    <SearchBar onSearch={handleSearch} onChange={ (value) => setUserInput(value) } />

    <RecipeList recipes={recipes} />

  </div>

);
}

export default App;
...

```

## 2. \*\*SearchBar-Komponente (SearchBar.js):\*\*

- Enthält ein Textfeld und einen Button zum Auslösen der Suche.

```

```jsx

import React from 'react';

function SearchBar({ onSearch, onChange }) {

  return (

    <div>

      <input type="text" placeholder="Geben Sie Zutaten ein..." onChange={(e) =>
onChange(e.target.value)} />

      <button onClick={onSearch}>Suchen</button>

    </div>

  );
}

export default SearchBar;
...

```

### 3. **\*\*RecipeList-Komponente (RecipeList.js):\*\***

- Zeigt die Liste der gefundenen Rezepte an.

```
```jsx
```

```
import React from 'react';
```

```
function RecipeList({ recipes }) {
```

```
  return (
```

```
    <ul>
```

```
      {recipes.map((recipe, index) => (
```

```
        <li key={index}>{recipe.title}</li>
```

```
        // Du kannst hier weitere Informationen aus dem Rezeptobjekt anzeigen
```

```
      )}}
```

```
    </ul>
```

```
  );
```

```
}
```

```
export default RecipeList;
```

```
```
```

Diese Struktur ist ein einfacher Ausgangspunkt. Du könntest sie erweitern, indem du zusätzliche Funktionen und Styling hinzufügst. Beachte auch, dass du die API-Aufrufe möglicherweise asynchron gestalten musst (z. B. mit `fetch` oder `axios`). Die genaue Implementierung hängt von der API ab, die du verwendest.

Denk daran, dass du React installieren musst, wenn du es noch nicht getan hast. Du kannst dies mit dem Befehl `npm install react` tun. Wenn du es mit `create-react-app` erstellst, wird React automatisch installiert.

#### **User**

Ich würde gerne das Eingaben im Textfeld sofort umgesetzt werden ohne erst auf einen Submit-Button zu drücken

#### **ChatGPT**

Um die Eingaben im Textfeld sofort umzusetzen, ohne auf einen Submit-Button zu drücken, kannst du die Suchfunktion bei jeder Änderung im Textfeld aufrufen. Hier ist eine modifizierte Version der `SearchBar`-Komponente und der `App`-Komponente, um dies zu erreichen:

```
```jsx
```

```
// SearchBar.js
```

```
import React from 'react';
```

```
function SearchBar({ onSearch, onChange }) {
```

```
  return (
```

```
    <div>
```

```
      <input type="text" placeholder="Geben Sie Zutaten ein..." onChange={(e) =>
onChange(e.target.value)} />
```

```
      {/* onSearch wird bei jeder Änderung im Textfeld aufgerufen */}
```

```
    </div>
```

```
  );
```

```
}
```

```
export default SearchBar;
```

```
```
```

```
```jsx
```

```
// App.js
```

```
import React, { useState, useEffect } from 'react';
```

```
import SearchBar from './SearchBar';
```

```
import RecipeList from './RecipeList';
```

```
function App() {
```

```

const [userInput, setUserInput] = useState('');

const [recipes, setRecipes] = useState([]);

// useEffect wird bei jeder Änderung in userInput aufgerufen
useEffect(() => {

  const handleSearch = async () => {

    // Hier wird die API-Aufruf-Logik implementiert

    // Verwende userInput, um die API-Anfrage zu erstellen

    // Setze dann die erhaltenen Rezepte mit setRecipes

  };

  // Führe die Suche aus, wenn sich userInput ändert

  handleSearch();

}, [userInput]);

return (

  <div>

    <h1>Rezeptvorschläge</h1>

    <SearchBar onChange={ (value) => setUserInput(value)} />

    <RecipeList recipes={recipes} />

  </div>

);

}

export default App;
...

```

In diesem Beispiel wird `useEffect` verwendet, um die `handleSearch`-Funktion bei jeder Änderung in `userInput` aufzurufen. Dadurch wird die Suche automatisch ausgelöst, wenn der Benutzer etwas in das Textfeld eingibt. Beachte, dass dies zu vielen API-Aufrufen führen kann, wenn der Benutzer schnell tippt. Du könntest eine Verzögerung (Debounce) hinzufügen, um dies zu steuern, aber das ist optional und hängt von deinen Anforderungen ab.