

Debounced Search API React

search input with debounce lodash react ts call api example

To implement a search input with a debounce in a React TypeScript application, you can use `lodash.debounce` to delay the API call until the user has stopped typing for a specified duration. Here's a practical example:

Step 1: Install lodash

First, install lodash if you haven't already:

```
bash

npm install lodash
```

Step 2: Set up the Component

Here's a sample component using `lodash.debounce`:

```
typescript

import React, { useState, useCallback } from 'react';
import debounce from 'lodash.debounce';
import axios from 'axios';

const SearchComponent: React.FC = () => {
  const [query, setQuery] = useState('');
  const [results, setResults] = useState([]);

  // API call with a debounced function
  const fetchResults = async (searchQuery: string) => {
    try {
      const response = await axios.get(`/api/search`, {
        params: { q: searchQuery },
      });
      setResults(response.data);
    } catch (error) {
      console.error('Error fetching data:', error);
    }
  };

  // Debounced function to avoid API calls on every keystroke
```

```

const debouncedFetchResults = useCallback(
  debounce((searchQuery: string) => fetchResults(searchQuery), 500),
  []
);

// Handler for the input change
const handleChange = (e: React.ChangeEvent<HTMLInputElement>) => {
  const { value } = e.target;
  setQuery(value);
  debouncedFetchResults(value); // Call debounced function
};

return (
  <div>
    <input
      type="text"
      value={query}
      onChange={handleChange}
      placeholder="Search..."
    />
    <ul>
      {results.map((result, index) => (
        <li key={index}>{result}</li>
      ))}
    </ul>
  </div>
);
};

export default SearchComponent;

```

Explanation:

1. State Management:

- `query` stores the current search input.
- `results` stores the fetched search results.

2. API Call:

- `fetchResults` makes an API call to fetch data based on the search query.

3. Debounce with `useCallback`:

- `debouncedFetchResults` is a debounced version of `fetchResults`, only firing when the user stops typing for 500 milliseconds.

4. Input Handling:

- `handleChange` updates the `query` and triggers the debounced fetch function with the latest input value.

Notes:

- Adjust the debounce time (`500 ms`) based on the desired delay.
- This example assumes your API endpoint is `/api/search`, so update it as needed.