# Classroom Activity: Demonstrate the use of useFetch to perform the same function multiple times.

# Custom Hook: useFetch

Suppose we need to use data fetching in multiple components. We can create a custom hook called useFetch that provides the same data fetching functionality across those components.

### Code :

|  |
| --- |
| import { useState, useEffect } from 'react';  const useFetch = (url) => {  const [data, setData] = useState(null);  const [loading, setLoading] = useState(true);  const [error, setError] = useState(null);   useEffect(() => {  const fetchData = async () => {  try {  setLoading(true);  const response = await fetch(url);  if (!response.ok) {  throw new Error('Network response was not ok');  }  const result = await response.json();  setData(result);  } catch (error) {  setError(error);  } finally {  setLoading(false);  }  };   fetchData();  }, [url]);   return { data, loading, error }; };  export default useFetch; |

### Component A:

|  |
| --- |
| DataFetchingComponentA.js:  import React from 'react'; import useFetch from './useFetch';  const DataFetchingComponentA = () => {  const { data, loading, error } = useFetch('https://jsonplaceholder.typicode.com/todos/1');   if (loading) return <div>Loading...</div>;  if (error) return <div>Error: {error.message}</div>;   return (  <div>  <h1>Fetched Data A</h1>  {data && <p>{data.title}</p>}  </div>  ); };  export default DataFetchingComponentA; |

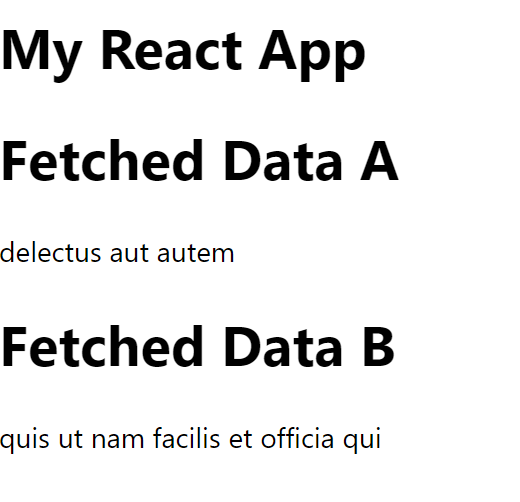
### Component B:

|  |
| --- |
| DataFetchingComponentB.js:  import React from 'react'; import useFetch from './useFetch';  const DataFetchingComponentB = () => {  const { data, loading, error } = useFetch('https://jsonplaceholder.typicode.com/todos/2');   if (loading) return <div>Loading...</div>;  if (error) return <div>Error: {error.message}</div>;   return (  <div>  <h1>Fetched Data B</h1>  {data && <p>{data.title}</p>}  </div>  ); };  export default DataFetchingComponentB; |

### App.js:

|  |
| --- |
| import React from 'react'; import DataFetchingComponentA from './DataFetchingComponentA'; import DataFetchingComponentB from './DataFetchingComponentB';  function App() {  return (  <div className="App">  <header className="App-header">  <h1>My React App</h1>  <DataFetchingComponentA />  <DataFetchingComponentB />  </header>  </div>  ); }  export default App; |

**Output :**



**Execution Steps:**

1. **Create a new React project:**
   * If you don't have a React project, create one using the command: npx create-react-app my-app.
2. **Add the custom hook:**
   * Inside the src folder of your React project, create a file named useFetch.js and add the custom hook code provided.
3. **Add the components:**
   * Create files named DataFetchingComponentA.js and DataFetchingComponentB.js in the src folder, and add the component codes provided.
4. **Update App.js:**
   * Replace the contents of App.js with the code provided to import and use DataFetchingComponentA and DataFetchingComponentB.
5. **Save the files:**
   * Ensure that useFetch.js, DataFetchingComponentA.js, DataFetchingComponentB.js, and App.js are saved after making the changes.
6. **Run the React app:**
   * Open your terminal.
   * Navigate to the project directory and run the command: npm start.
7. **View the output:**
   * Open your browser and go to http://localhost:3000.
   * You should see the fetched data from both endpoints displayed on the screen.