

Post.js:

class Post{

    constructor(id,title,body){

        this.id = id;

        this.title = title;

        this.body = body;

    }

}

export default Post;

Posts.js:

import React, { Component } from 'react';

class Posts extends Component {

    constructor(props) {

        super(props);

        // Initialize the state

        this.state = {

            posts: [] // Initialize posts as an empty array

        };

    }

    loadPosts() {

        fetch('https://jsonplaceholder.typicode.com/posts')

            .then(response => {

                if (!response.ok) {

                    throw new Error(`HTTP error! status: ${response.status}`);

                }

                return response.json();

            })

            .then(data => {

                this.setState({ posts: data });

            })

            .catch(error => console.error('Error fetching posts:', error));

    }

    componentDidMount() {

        this.loadPosts();

    }

    render() {

        return (

            <div>

                <h1>Posts</h1>

                <ul>

                    {/\* The condition 'this.state.posts &&' is good practice, but less critical

                        now that posts is initialized to an array \*/}

                    {this.state.posts.map(post => (

                        <li key={post.id}>

                            <h2>{post.title}</h2>

                            <p>{post.body}</p>

                        </li>

                    ))}

                </ul>

            </div>

        );

    }

    // This is a good addition for error handling, keep it!

    componentDidCatch(error, info) {

        console.error("Error in Posts component:", error, info);

    }

}

export default Posts;

App.js:

import logo from './logo.svg';

import './App.css';

import Posts from './Component/Posts';

function App() {

  return (

    <div className="App">

      <Posts/>

    </div>

  );

}

export default App;

Output:



