

1. The component lifecycle in React refers to the series of phases a component goes through from creation to removal. It is important because it allows developers to execute code at specific points during a component's existence. The benefits include the ability to initialize values before the component appears, perform side-effects such as data fetching after the component has rendered, and clean up resources before the component is removed from the page. Understanding the component lifecycle improves code organization and allows for better control over component behavior.
2. React provides several lifecycle hook methods, particularly in class components. These include:

* constructor(): Used to initialize state and bind methods.
* render(): Returns the JSX to be displayed on the screen.
* componentDidMount(): Executes after the component is inserted into the DOM, commonly used for API calls.
* componentDidUpdate(): Called immediately after updating occurs, useful for reacting to state or prop changes.
* componentWillUnmount(): Invoked just before the component is removed from the DOM, commonly used for cleanup like removing event listeners.
* componentDidCatch(): Handles errors that occur during rendering or in lifecycle methods of child components.

1. The typical sequence of lifecycle methods during the mounting phase is:

* constructor() – Initializes the component.
* render() – Describes what should be rendered.
* componentDidMount() – Executes after the component has been rendered into the DOM.

During the updating phase, the sequence is:

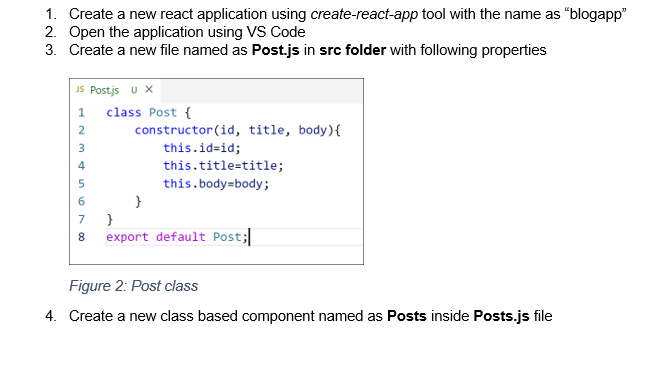
* render() – Called when props or state changes.
* componentDidUpdate() – Runs after the component updates in the DOM.

During the unmounting phase:

* componentWillUnmount() – Called just before the component is removed from the DOM.

In the case of runtime errors:

* componentDidCatch() – Called when a child component throws an error.



**Post.js**

class Post {

constructor(id, title, body) {

this.id = id;

this.title = title;

this.body = body;

}

}

export default Post;

**Posts.js**

import React from 'react';

import Post from './Post';

class Posts extends React.Component {

constructor(props) {

super(props);

this.state = {

posts: [

new Post(1, 'First Post', 'This is the first post.'),

new Post(2, 'Second Post', 'This is the second post.'),

new Post(3, 'Third Post', 'This is the third post.')

]

}; }

loadPosts() {

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

.then(response => response.json())

.then(data => {

const posts = data.map(post => new Post(post.id, post.title, post.body));

this.setState({ posts });

})

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

}

componentDidMount() {

this.loadPosts();

}

componentDidCatch(error, info) {

console.error('Error caught in Posts component:', error);

console.error('Error info:', info);}

render() {

return (

<div>

<h2>Posts</h2>

<ul>

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

<li key={post.id}>

<h3>{post.title}</h3>

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

</li>

))}

</ul>

</div>

);

}

}export default Posts;

**App.js**

// import logo from './logo.svg';

import './App.css';

import Posts from './Posts';

function App() {

return (

<div className="App">

<h2>Student Score Calculator</h2>

<Posts />

</div>

);

}

export default App;

**Output**

