**Explain the need and Benefits of component life cycle**

**Need for Component Lifecycle**

**React components go through different stages:**

Mounting (creation)

Updating (changes in props/state)

Unmounting (removal)

**Understanding the lifecycle allows you to:**

Run specific code at particular times (e.g., fetching data when a component loads).

Optimize performance.

Handle side effects (e.g., timers, subscriptions).

Avoid memory leaks.

**Benefits of Component Lifecycle**

Control - Precise control over what happens at different stages of a component's life.

Performance Optimization - Use lifecycle hooks to avoid unnecessary renders.

Resource Management - Start/stop network requests, event listeners, etc. correctly.

Data Fetching - Fetch data after component mounts (componentDidMount / useEffect).

**Identify various life cycle hook methods**

|  |  |  |
| --- | --- | --- |
| Mounting | * constructor() | * Initialization |

|  |  |  |  |
| --- | --- | --- | --- |
| Mounting | * render() | * Render UI |  |

|  |  |  |
| --- | --- | --- |
| Mounting | * componentDidMount() | * Called after initial render (good for API calls) |

|  |  |  |
| --- | --- | --- |
| Updating | * shouldComponentUpdate() | * Control re-rendering |

|  |  |  |
| --- | --- | --- |
| Updating | * componentDidUpdate() | * Reacts to prop/state changes |

|  |  |  |
| --- | --- | --- |
| Unmounting | * componentWillUnmount() | * Cleanup (like stopping timers, listeners) |

**Sequence of Steps in Rendering a Component**

**In Class Component:**

constructor() – Initialization

getDerivedStateFromProps() (rare)

render() – JSX to DOM

componentDidMount() – After mount

**On update:**

shouldComponentUpdate()

render()

componentDidUpdate()

**On unmount:**

componentWillUnmount()

In Functional Component:

Component function is executed

useEffect(..., []) runs after first render

On updates, useEffect(..., [deps]) runs

On unmount, cleanup from useEffect runs

**App.js**

import React from 'react'

import Posts from './Components/Posts'

function App(){

return(

<div className="App">

<Posts></Posts>

</div>

)

}

export default App;

**Post.js**

import React from 'react'

class Post extends React.Component{

render(){

return(

<div>

<h2>{this.props.title}</h2>

<p>{this.props.body}</p>

</div>

);

}

}

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: []

};

}

loadPosts = () => {

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

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

.then(data => this.setState({posts: data}))

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

};

componentDidMount(){

this.loadPosts();

}

componentDidCatch(error, info){

alert('an error occurred:' + error);

}

render(){

return(

<div>

<h1>Posts</h1>

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

<Post key={post.id} title={post.title} body={post.body} ></Post>

))}

</div>

);

}

}

export default Posts;



