**React HOL-4**

**Explain the Need and Benefits of components of life cycle**

* Helps manage what happens when a component **starts**, **updates**, or **ends**.
* Useful for tasks like **fetching data**, **starting timers**, or **cleaning up**.

**Identify various life cycle hook methods**

**Mounting (when component loads):**

* constructor()
* render()
* componentDidMount()

**Updating (when props/state change):**

* render()
* componentDidUpdate()

**Unmounting (when component is removed):**

* componentWillUnmount()

**List the sequence of steps in rendering a component**

 constructor() – Setup

 render() – Draw UI

 componentDidMount() – Run after UI is shown

**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';

import Post from './Post';

class Posts extends Component {

constructor(props) {

super(props);

this.state = {

posts: [],

error: null

};

}

loadPosts = () => {

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

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

.then(data => {

const postObjects = data.map(

post => new Post(post.userId, post.id, post.title, post.body)

);

this.setState({ posts: postObjects });

})

.catch(err => this.setState({ error: err }));

};

componentDidMount() {

this.loadPosts();

}

componentDidCatch(error, info) {

alert("An error occurred: " + error);

}

render() {

return (

<div>

<h1>Blog Posts</h1>

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

<div key={post.id}>

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

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

<hr />

</div>

))}

</div>

);

}

}

export default Posts;

**App.js**

import React from 'react';

import Posts from './Posts';

function App() {

return (

<div className="App">

<Posts />

</div>

);

}

export default App;

**Output**

**A screenshot of a computer

AI-generated content may be incorrect.**