1. **ReactJS-HOL**

**Question:** Explain the need and Benefits of component life cycle Identify various life cycle hook methods. List the sequence of steps in rendering a component. Implement componentDidMount() hook Implementing componentDidCatch() life cycle hook.

**Code:**

**App.js**

import logo from './logo.svg';

import './App.css';

import Posts from './Posts';

function App() {

  return (

    <div className='App'>

      <Posts />

    </div>

  );

}

export default App;

**index.js**

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

  <React.StrictMode>

    <App />

  </React.StrictMode>

);

reportWebVitals();

**Post.js**

class Post{

    constructor(id, title, body){

        this.id = id;

        this.title = title;

        this.body = body;

    }

}

export default Post;

**Posts.js**

import { Component } from "react";

import Post from "./Post";

class Posts extends Component{

    constructor(props){

        super(props);

        this.state = {

            posts: [],

            hasError: false

        }

    }

    loadPosts(){

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

            .then(res => res.json())

            .then(data => {

                const postsArray = data.map(item => new Post(item.id, item.title, item.body));

                this.setState({posts: postsArray})

            })

            .catch(err => {

                console.error("Error fetching posts", err);

                this.setState({hasError: true});

            })

    }

    componentDidMount(){

        this.loadPosts();

    }

    componentDidCatch(error, info){

        alert("Something went Wrong!");

        console.error("Error Caught:", error, info);

    }

    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;

**OUTPUT:**

