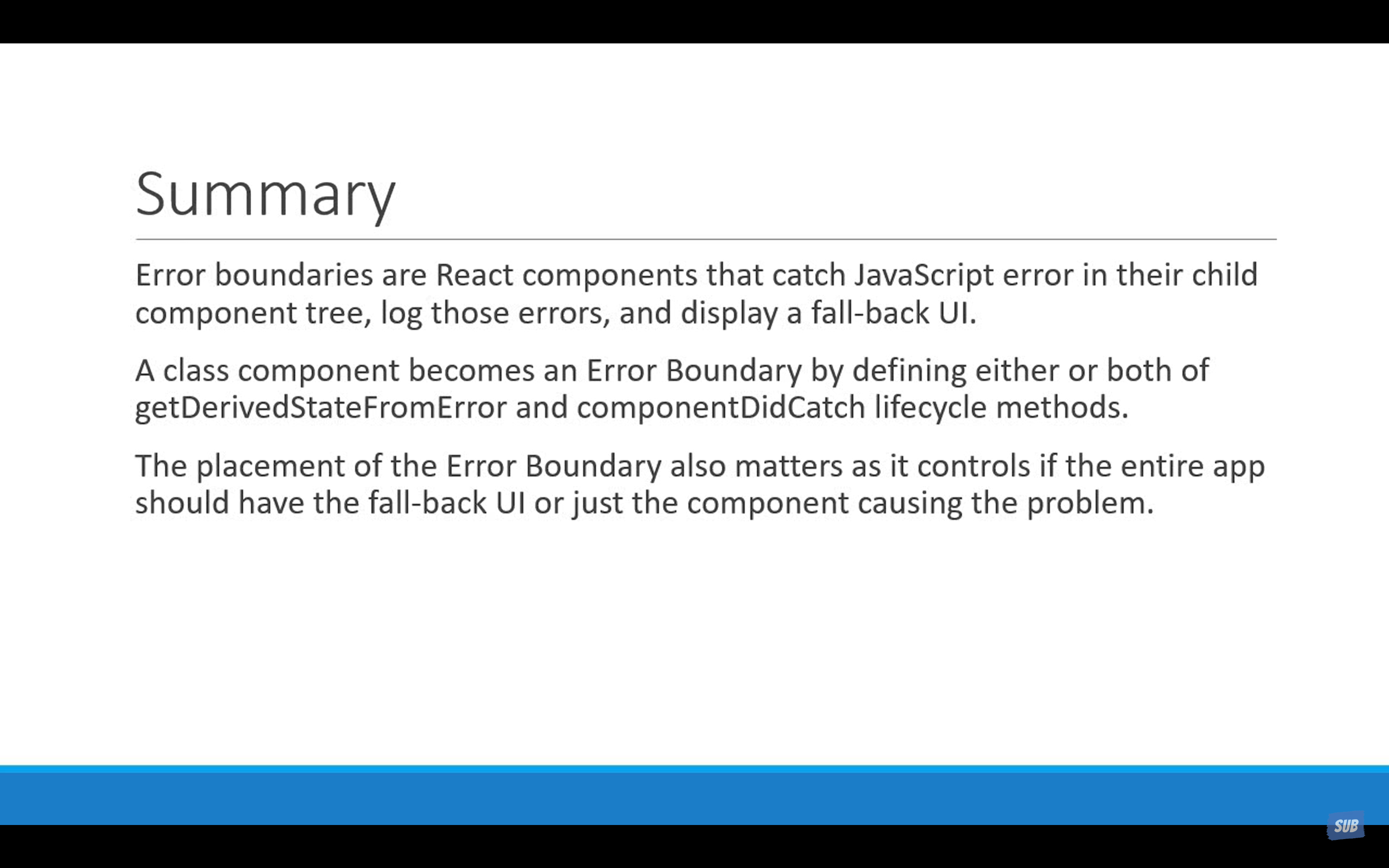
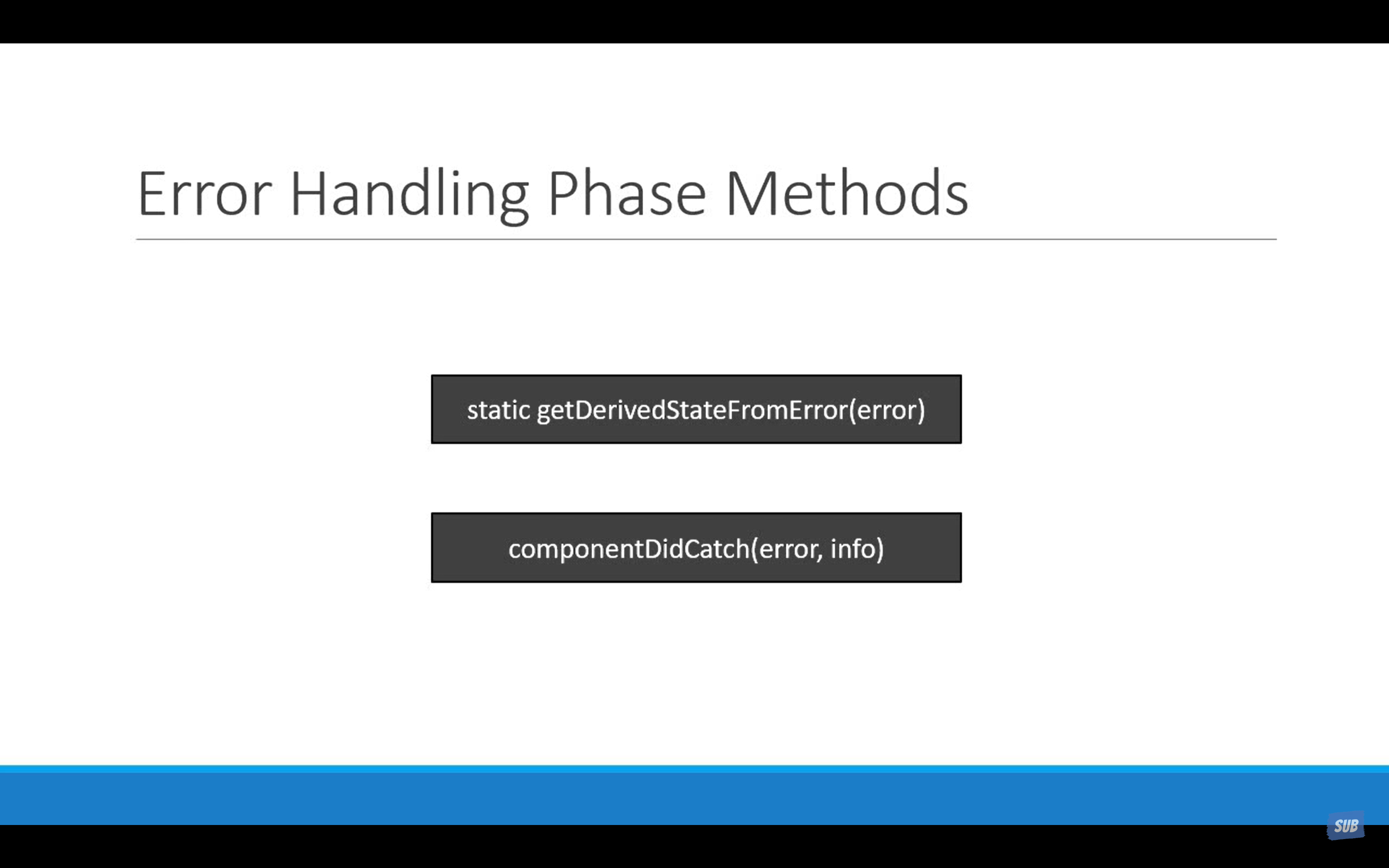
<https://builtin.com/software-engineering-perspectives/react-error-boundary>





import React from 'react'

export const App5 = () => {

  return (

    <div>

        <h1> Error Handling </h1>

        <Test message="hi"/>

    </div>

  )

}

export const Test = (props) => {

  return (

    <div>

    <h2> Test {props.message} </h2>

    </div>

  )

}

import React from 'react'

export const App5 = () => {

  return (

    <div>

        <h1> Error Handling </h1>

        <Test message={100}/>

    </div>

  )

}

export const Test = (props) => {

    if(typeof(props.message)!=="string")

        throw new Error("not in string")

  return (

    <div>

    <h2> Test {props.message} </h2>

    </div>

  )

}

import React from 'react'

export const App5 = () => {

  return (

    <div>

        <h1> Error Handling </h1>

        <Test message=""/>

    </div>

  )

}

export const Test = (props) => {

    if(typeof(props.message)!=="string" || !props.message)

        throw new Error("not in string")

  return (

    <div>

    <h2> Test {props.message} </h2>

    </div>

  )

}

import React, { Component } from "react";

// import { HashRouter } from "react-router-dom";

// class ErrorBoundary extends Component

// {

//     /\*\*

//      \*

//      \*/

//     constructor(props) {

//         super(props);

//      this.state={hasError:false}

//     }

//     componentDidCatch()

//     {

//         this.setState({hasError:true})

//     }

//     render()

//     {

//         if(this.state.hasError)

//         return <p>Something went wrong. Please try again later.</p>;

//         return this.props.children;

//     }

// }

// export default  ErrorBoundary;

class ErrorBoundary extends React.Component {

    constructor(props) {

      super(props);

      this.state = { hasError: false };

    }

    static getDerivedStateFromError(error)

    {

        return {

        hasError:true

        }

    }

    // componentDidCatch(error, info) {

    //     console.log("eb")

    //   this.setState({ hasError: true });

    //   // Log the error to a centralized service

    // //   logErrorToService(error, info);

    // }

    render() {

        // {this.state.hasError}

      if (this.state.hasError) {

        // Render fallback UI

        return <p>Something went wrong. Please try again later.</p>;

      }

      return this.props.children;

    }

  }

  export default ErrorBoundary;

import React from 'react'

import ErrorBoundary from './ErrorBounday'

export const App5 = () => {

  return (

    <div>

        <h1> Error Handling </h1>

        <ErrorBoundary>

        <Test message=""/>

        </ErrorBoundary>

    </div>

  )

}

export const Test = (props) => {

    if(typeof(props.message)!=="string" || !props.message)

        throw new Error("not in string")

  return (

    <div>

    <h2> Test {props.message} </h2>

    </div>

  )

}

import React, { Component } from "react";

// import { HashRouter } from "react-router-dom";

// class ErrorBoundary extends Component

// {

//     /\*\*

//      \*

//      \*/

//     constructor(props) {

//         super(props);

//      this.state={hasError:false}

//     }

//     componentDidCatch()

//     {

//         this.setState({hasError:true})

//     }

//     render()

//     {

//         if(this.state.hasError)

//         return <p>Something went wrong. Please try again later.</p>;

//         return this.props.children;

//     }

// }

// export default  ErrorBoundary;

class ErrorBoundary extends React.Component {

    constructor(props) {

      super(props);

      this.state = { hasError: false };

    }

    static getDerivedStateFromError(error)

    {

        return {

        hasError:true

        }

    }

    componentDidCatch(error, info) {

        console.log("error" + error    )

    console.log("info"  + info)

    //   this.setState({ hasError: true });

      // Log the error to a centralized service

    //   logErrorToService(error, info);

    }

    render() {

        // {this.state.hasError}

      if (this.state.hasError) {

        // Render fallback UI

        return <p>Something went wrong. Please try again later.</p>;

      }

      return this.props.children;

    }

  }

  export default ErrorBoundary;

Npm run build

Cd build

http-server

Next Demo

import React, { useState, useEffect } from 'react';

const MyComponent = () => {

  const [data, setData] = useState(null);

  const [error, setError] = useState(null);

  useEffect(() => {

    const fetchData = async () => {

      try {

        const response = await fetch('https://jsonplaceholder.typicode.com/users');

        const result = await response.json();

        setData(result);

      } catch (err) {

        setError(error);

      }

    };

    fetchData();

  }, []);

  return (

    <div>

      {data?<h1> Data </h1> : error}

    </div>

  )

}

export default MyComponent