**Lesson Summary – Error Boundaries**

Errors can occur in applications that developers cannot prevent or handle directly. These errors are often used to convey information about something that went wrong from one part of the application to another. For example, if an HTTP request fails due to a server being offline, the application will generate an error. In such cases, errors can be used as a means of transporting information rather than being considered negative.

While regular JavaScript provides the "try-catch" syntax to handle errors, this cannot be directly used in JSX code within React components. Instead, React provides a solution called error boundaries. An error boundary is a class-based component that implements the "componentDidCatch" lifecycle method. This method is triggered whenever a child component throws or generates an error.

To create an error boundary, you need to:

* Create a new component as a class-based component with the desired name (e.g., "ErrorBoundary").
* Add the "componentDidCatch" lifecycle method to the component.
* Wrap the components that need to be protected by the error boundary in the component using the props.children property.
* Handle the error by accessing the error object passed as a parameter to the "componentDidCatch" method.
* Optionally, manage the state of the error boundary component and perform any additional actions, such as logging the error or sending it to a server.

In development mode, the error boundary will display an error message provided by the React development server. However, in production mode, these error messages are not shown to the users. By implementing error boundaries, you can ensure that your entire application does not crash due to errors and handle them in a more controlled and graceful manner.