**Error Boundary:**

* Error Boundaries are **React components** that catch **JavaScript errors** in their child component tree during **rendering, lifecycle methods(componentDidMount, componentDidUpdate, etc.), and constructors** of child components.
* Instead of breaking the whole React app, they let you show a **fallback UI**.

They are implemented using either:

* **class components** with componentDidCatch and static getDerivedStateFromError
* or in newer React, helper libraries like react-error-boundary provide hooks for functional components.( npm install react-error-boundary, import { ErrorBoundary } from "react-error-boundary",..)

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer error

AI-generated content may be incorrect.

\*\*TO HANDLE ASYNC ERRORS WITH EB:You must **store the error in state**, then let the component throw during render:storing in error in state(MAY BE IN CATCH BLOCK) makes error boundary catch error, with try catch alone EB will not catch error

**Difference between try/catch and Error Boundaries?**

* **try/catch** → only works inside the same function scope, including event handlers and async code.
* **Error Boundaries** → work across the **React component tree** but only catch errors during **render/lifecycle/constructor**.
* A screenshot of a computer

  AI-generated content may be incorrect.

These operators(OC AND NC) are **proactive safety nets** — they don’t “catch” runtime errors like try/catch or Error Boundaries, but they **prevent the error from happening in the first place**.

* TRY/CATCH CANNOT CATCH ERRORS OCCURING DURING REACT RENDERING PROCESS (AS THIS PROCESS IS DECLARATIVE), try catch only handles imperative code like event handlers,async api calls etc.so,EBs are used

**Custom hooks:**

**A screenshot of a chat

AI-generated content may be incorrect.**

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

\*\*even without naming custom hook starting with ‘use’ , react custom hooks still works expected but its not correct convention,will face difficulty in debugging ,linting ttc so, best practice to name custom hook starting with ‘use\*’.

**Redux:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A white background with black text

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer code

AI-generated content may be incorrect.**

**A computer code on a white background

AI-generated content may be incorrect.**

**A screenshot of a computer code

AI-generated content may be incorrect.**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer program

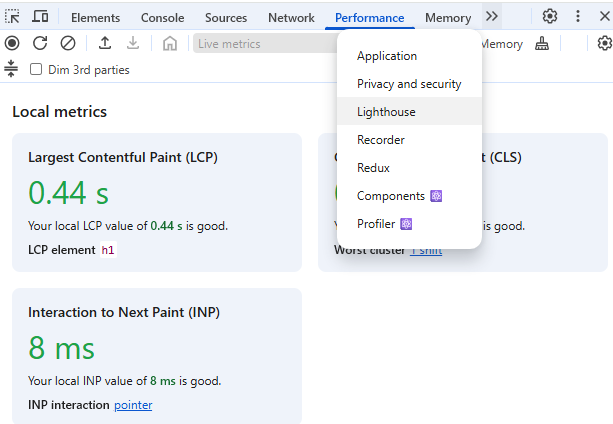
AI-generated content may be incorrect.**

Refer ‘redux-multipleSlices’ project to know code abt lazyloading,suspense,redux,reduxasyncthunketc;

**Light House:**

c;

found in inspect:

****

LIGHTHOUSE SCORE Sample:

A screenshot of a computer

AI-generated content may be incorrect.

A white text on a black background

AI-generated content may be incorrect.

A screenshot of a web page

AI-generated content may be incorrect.

A screenshot of a web page

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a web site

AI-generated content may be incorrect.

A screenshot of a web page

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Test in **production build** (npm run build && npm run preview)