

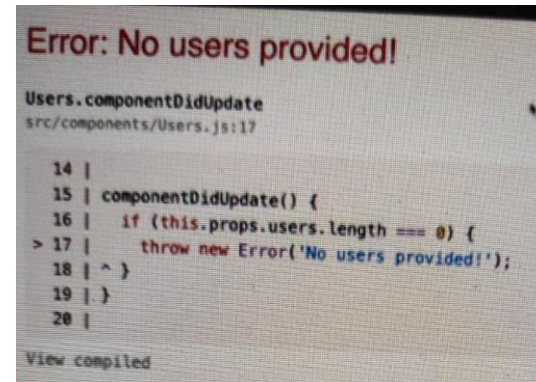
# ERROR BOUNDARIES

---

# Using Error Boundaries

```
const person = (props) => {  
  const rnd = Math.random();  
  if(rnd > 0.7){  
    throw new Error("Something went wrong");  
  }  
  return (  
    ...  
  )  
};  
export default person;
```

```
class Users extends Component{  
  
  componentDidMount(){  
    if(this.props.users.length == 0)  
      throw new Error("No users in list!")  
  }  
  ...  
}
```



- A JavaScript error in a part of the UI shouldn't break the whole app.
- To solve this problem for React users, React 16 introduces a new concept of an "error boundary".

# Using Error Boundaries

```
import React, {Component} from 'react';
class ErrorBoundary extends Component{
  state = {
    hasError:false,
    errorMessage:''
  }
  componentDidCatch = (error, info) => {
    this.setState({hasError:true, errorMessage:error});
  }
  render(){
    if(this.state.hasError)
      return <h1>{this.state.errorMessage}</h1>
    else
      return this.props.children
  }
}
export default ErrorBoundary;
```

```
import React from 'react'
function Artists({artistName}) {
  if (artistName === 'peruzzi') {
    throw new Error ('not performing tonight!')
  }
  return (
    <div>
      {artistName}
    </div>
  )
}
export default Artists
```

**Use it like this:**  
<ErrorBoundary>  
 <Artists />  
</ErrorBoundary>

- Error boundaries are React components that **catch JavaScript errors anywhere in their child component tree, log those errors, and display a fallback UI** instead of the component tree that crashed. Error boundaries catch errors during rendering
- <https://reactjs.org/docs/error-boundaries.html>
- [React.Component – React \(reactjs.org\)](#)

# Using Error Boundaries

```
function App() {  
  return (  
    <div className="App">  
      <h1> Welcome to React</h1>  
      <ErrorBoundary >  
        <Person />  
      </ErrorBoundary>  
    </div>  
  );  
}
```

Welcome to React

Error: Something went wrong

Welcome to React

From Error throwing Person component - all is well!



localhost:3000

```
> 45 | result = __webpack_require__(__webpack_require__.s = deferredModule[0]);  
46 | ^  
47 | }  
48 |
```

View compiled

Array.webpackJsonpCallback [as push]  
E:/FreelanceTrg/ReactJS/Demo-Nov2021/second-app/webpack/bootstrap:32

```
29 | deferredModules.push.apply(deferredModules, executeModules || []);  
30 |  
31 | // run deferred modules when all chunks ready  
> 32 | return checkDeferredModules();  
33 | };  
34 | function checkDeferredModules() {  
35 | var result;
```

View compiled

(anonymous function)  
http://localhost:3000/static/js/main.chunk.js:1:73

This screen is visible only in development. It will not appear if the app crashes in production.  
Open your browser's developer console to further inspect this error. Click the 'X' or hit ESC to dismiss this message.