# ErrorBoundary UI States

CodeScribe AI - Production-Ready Error Handling Component

`ErrorBoundary.jsx` • 48 Tests Passing • October 14, 2025

## Overview & Key Metrics

Catches JavaScript errors anywhere in the component tree, prevents app crashes, and displays user-friendly fallback UI. Features intelligent error handling with development and production modes.

| 48 | 3 | 2 | 100% |
|---|---|---|---|
| Tests Passing | Recovery Options | Display Modes | Coverage |

## Development Mode

Shows detailed technical info when `import.meta.env.DEV === true`

### Something went wrong

Unexpected error occurred

We're sorry for the inconvenience. Here's what you can try:

- Try again - temporary issues resolve themselves
- Reload the page to start fresh
- Check your internet connection

Try Again | Reload Page | Go Home

▶ **Technical Details**

docs | report issue

**Includes:**
- Full error messages
- Complete stack traces
- Component stack display
- Expandable technical details

## Production Mode

Clean UI with error ID when `import.meta.env.DEV === false`

### Something went wrong

Unexpected error occurred

We're sorry for the inconvenience. Here's what you can try:

- Try again - temporary issues resolve themselves
- Reload the page to start fresh
- Check your internet connection

Try Again | Reload Page | Go Home

**Error ID:** 1729025847-k3j9x2m1q

Contact support with error ID if problem continues.

docs | report issue

**Features:**
- Clean, professional UI
- Unique error IDs for support
- No sensitive info exposed
- User-friendly guidance

## Development vs Production Comparison

| Feature | Development | Production |
|---|---|---|
| Error Message | ✓ Full details | ✓ User-friendly |
| Stack Trace | ✓ Visible | ✗ Hidden |
| Component Stack | ✓ Full tree | ✗ Hidden |
| Error ID | ✗ Not shown | ✓ Unique ID |
| Recovery Actions | ✓ All 3 options | ✓ All 3 options |
| Console Logging | ✓ Detailed | ✓ Basic |

## Recovery Actions

### 🔄 Try Again

- Resets error state
- Re-renders components
- Preserves app state
- Best for temporary issues

**Use:** Network glitches, race conditions

### 🔃 Reload Page

- Full page refresh
- Clears all state
- Reloads resources
- Guaranteed fresh start

**Use:** Persistent errors, corrupted state

### 🏠 Go Home

- Navigates to root
- Clears route state
- Returns to safe state
- Gentle recovery

**Use:** Route-specific errors

## Implementation & Integration

**Component Structure:**

- **Class:** React.Component for error lifecycle
- **State:** hasError, error, errorInfo, errorCount
- **Methods:** getDerivedStateFromError(), componentDidCatch()
- **Recovery:** handleReset(), handleReload(), handleGoHome()
- **Environment:** Detects via import.meta.env.DEV

**Integration (main.jsx):**

```
import ErrorBoundary from './ErrorBoundary'

createRoot(document.getElementById('root'))
  .render(
    <StrictMode>
      <ErrorBoundary>
        <App />
      </ErrorBoundary>
    </StrictMode>
  )
```

💡 **Best Practice:** Place ErrorBoundary at root level to catch all errors.

**Test Coverage:** 48 tests covering normal rendering, error catching, recovery actions, dev/prod modes, accessibility, edge cases, and responsive design

CodeScribe AI • ErrorBoundary Component • client/src/components/ErrorBoundary.jsx • October 14, 2025