💥 CSS Can Block Routing & Clicks? Yes! Let’s Understand How 👇

📌 Why It Happens?

Sometimes, due to certain **CSS styles**, your app **UI looks fine** but:

- ❌ `onClick` events **don’t work**
- ❌ Buttons are **unclickable**
- ❌ Routes don’t change (especially using `react-router`)
- ❌ Input fields are **not focusable**
- ❌ Page seems **frozen**

Let’s dive into causes with 🔍 examples and solutions ✅

🔒 1. **Overlay Elements Blocking Clicks**

If you use **absolute/fixed positioned elements** (like modals, banners, shimmer loaders), they might **cover the entire screen**, preventing interactions underneath.

❌ Problem Code:

```css
.loader {
  position: fixed;
  top: 0;
  left: 0;
  height: 100vh;
  width: 100vw;
  background-color: rgba(255, 255, 255, 0.7);
  z-index: 9999;
}
```

```jsx
{isLoading && <div className="loader">Loading...</div>}
```

🧨 Problem:

- This `div` **covers the entire screen** even when it’s **transparent**.
- You **can’t click** on anything under it.

✅ Solution:

➡️ Hide loader after loading ➡️ Add a proper condition ➡️ Or use `pointer-events`

```css
.loader {
  pointer-events: none; /* 👈 allows clicks to pass through */
}
```

🧼 2. Using `pointer-events: none / auto` Smartly

🔥 Fixing Overlays, Loaders, and Decorations

✅ Example 1: Allow clicks to pass through

```css
.overlay {
  pointer-events: none;
}
```

❌ Mistake: Block all interactions

```css
.blocker {
  pointer-events: auto;
  z-index: 999;
}
```

> This will **block any click** unless you specifically allow elements inside to be interactable.

🧱 3. Z-Index & Layering Chaos

🤯 You see the button, but can't click it?

It's probably behind an **invisible or higher-z-index element**

❌ Problem Example:

```css
.header {
  position: fixed;
  top: 0;
  z-index: 1000;
}
.nav-wrapper {
  z-index: 9999;
}
```

```
<button onClick={() => navigate("/login")}>Login</button>
```

> Even though button is clickable, it's **under the** `nav-wrapper`

✅ Fix

- Ensure the clickable component is at the **topmost z-index**
- Or move blocking elements behind

🧩 4. **CSS That Makes Elements Unusable**

⚠️ CSS Properties That Disable Events:

🔧 **Property**	💧 **Effect**
`pointer-events: none`	Clicks won't work
`display: none`	Element is gone
`visibility: hidden`	Visible but not clickable
`opacity: 0`	Invisible & not clickable
`z-index`	Layered below something else
`overflow: hidden`	Content gets cut off
`height: 0`, `width: 0`	Not visible/clickable

🚀 Routing Fails? Check These!

❌ Broken Navigation Example (React):

```
<Link to="/login">
  <button className="nav-btn">Login</button>
</Link>
```

But the `button` doesn’t navigate?

🔍 Check:

1. Any **overlay** element covering the `button`
2. Any **CSS** making the `Link` or `button unclickable`
3. **JS errors** in the console that prevent re-render

💡 Practical Example

🔧 Example: Shimmer Loader Blocks Routing

```
{loading && <div className="shimmer"></div>}

<Link to="/about">
  <button className="route-btn">About</button>
</Link>
```

```
.shimmer {
  position: fixed;
  top: 0;
  left: 0;
  height: 100vh;
  width: 100vw;
  background: rgba(255, 255, 255, 0.8);
  z-index: 10000;
}
```

❌ Issue:

`button` won't work — shimmer is **blocking everything**.

✅ Fix:

```
.shimmer {
  pointer-events: none;
}
```

🔁 Common Fix Checklist ✅

✅ **Check**	🔍 **Fix**
☐ Invisible overlays	`pointer-events: none`
☐ High z-index	Lower it or bring interactive items on top
☐ Wrong layout flow	Use `flex`, `grid` properly
☐ Event bubbling issues	Use `e.stopPropagation()` cautiously
☐ Transparent blockers	Set `opacity: 0.5` with `pointer-events: none`

📦 Bonus: Use DevTools to Debug 🛠️

👉 Right-click → **Inspect Element**

- Use **"Elements"** tab to see if your button is **covered**
- Hover around the button area and observe overlays
- Try toggling off CSS (`display`, `position`, `z-index`) live!

🌈 Summary in Emojis

- 👀 **Looks fine** ≠ works fine!
- 🧱 CSS can create invisible blocks
- 🔍 Debug using browser tools
- 🛡️ Use `pointer-events`, `z-index` wisely
- 🧹 Always clean up overlays/loaders

✅ Best Practice Suggestion

```
{loading && (
  <div className="overlay">
    <div className="loader">Loading...</div>
  </div>
)}
```

```
.overlay {
  position: fixed;
  inset: 0;
  background: rgba(255, 255, 255, 0.5);
  display: flex;
  justify-content: center;
  align-items: center;
  z-index: 10000;
  pointer-events: none; /* 👈 This ensures routing and clicks work */
}
```

🧠 Final Tips:

1. ✔️ Always **inspect what is on top** of your clickable areas
2. 🧪 Use `pointer-events` for **temporary UI** elements
3. 🛠️ Use `z-index` only when needed — don't go wild!
4. 📏 Make sure `height`, `width` aren't set to 0
