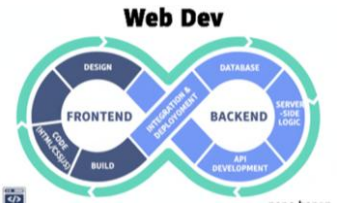
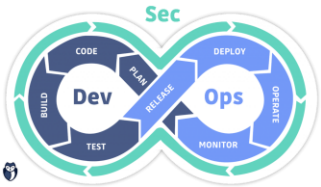




Day 57

"Web Development + Security"

The useRef Hook in React:

What is useRef()?

The useRef() Hook in React is used to access or store a reference to a DOM element or a mutable value that doesn't cause re-renders when updated.

You can think of it like a box that can hold a value — and React will keep that box the same between renders.

Syntax

```
const refName = useRef(initialValue);
```

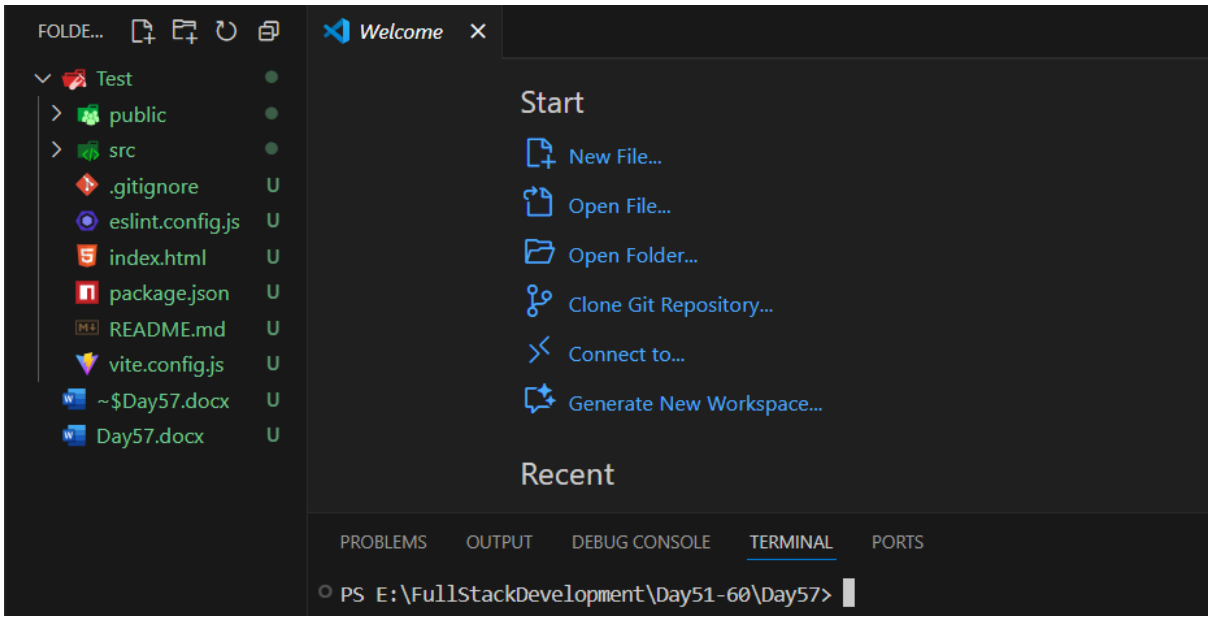
- Returns an object: { current: initialValue }
- The .current property is mutable (you can change it).
- Changing .current does NOT trigger a re-render.

Why Use useRef?

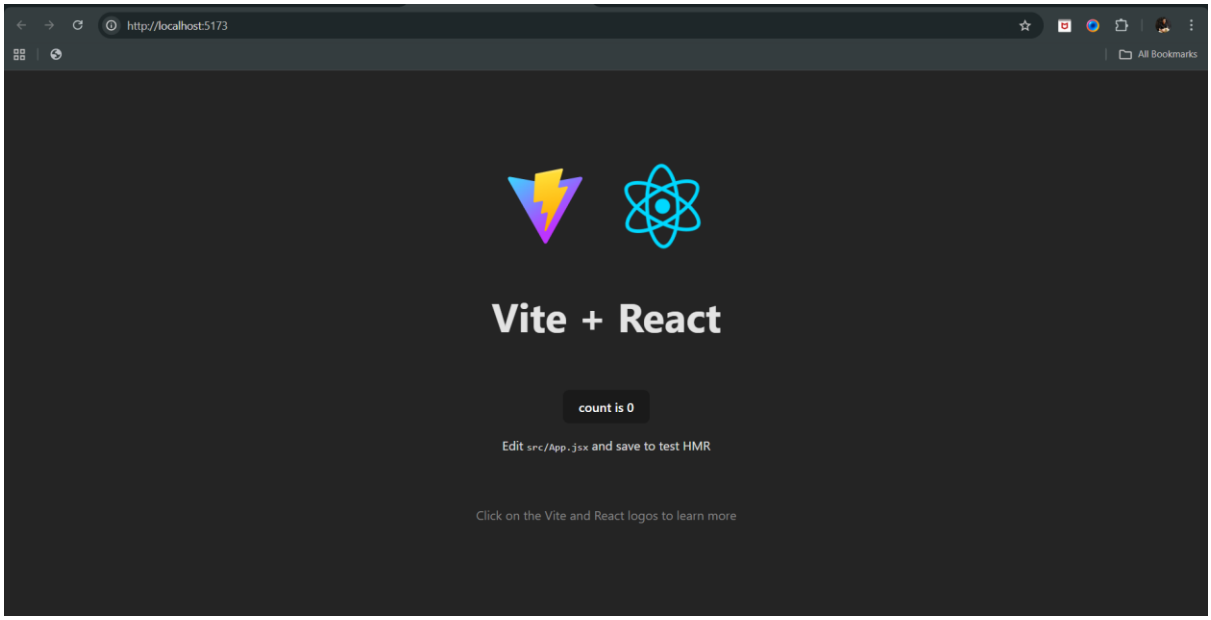
You use useRef when you need to:

1. Access DOM elements directly (e.g., focus an input, scroll a div).
2. Store mutable values that persist between renders (like timers, previous values).
3. Avoid unnecessary re-renders when updating data that isn't needed for the UI.

To understand this, we need to make a folder using the vite, as we did earlier, as shown below:



Then run to see the output we are familiar with:

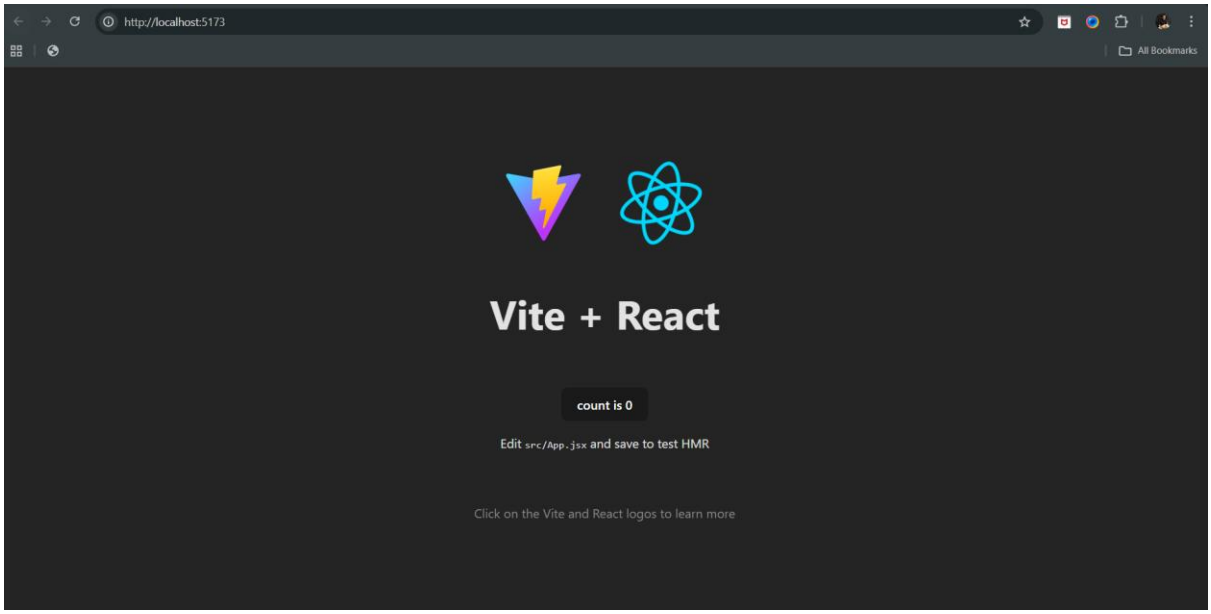


Now, as per the last pdf, we are aware of the rendering, using the useEffect: if removed the [] from the useEffect then for it, the useEffect will work for each time there will be some event in the page.

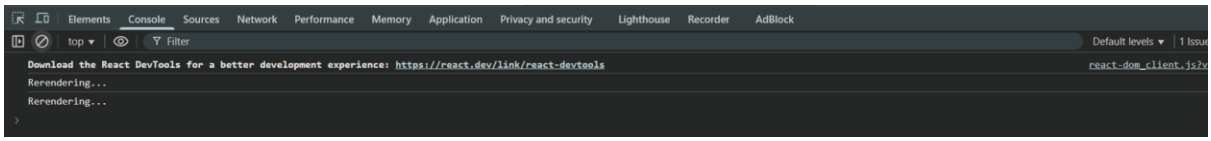
To show this, we will add one useEffect:

App.jsx: added line 9-11

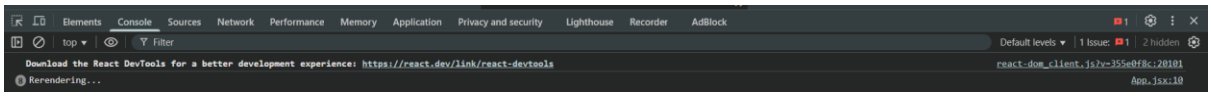
Output:



Console: at the loading of the page

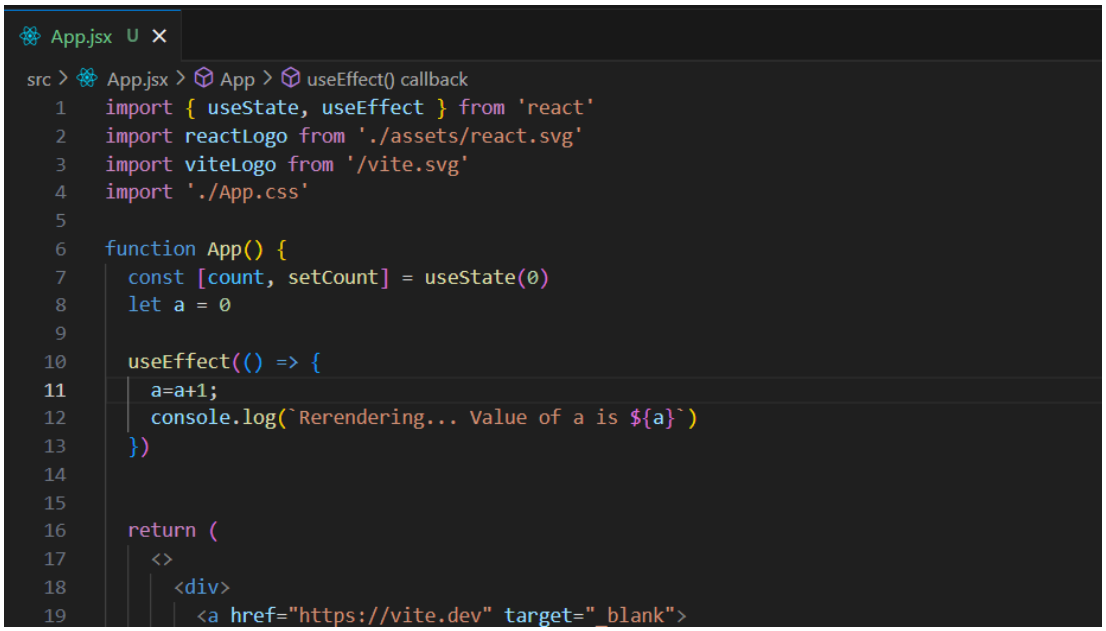


Console: when we click the button for many times. Basically, it occurs because of the useEffect.



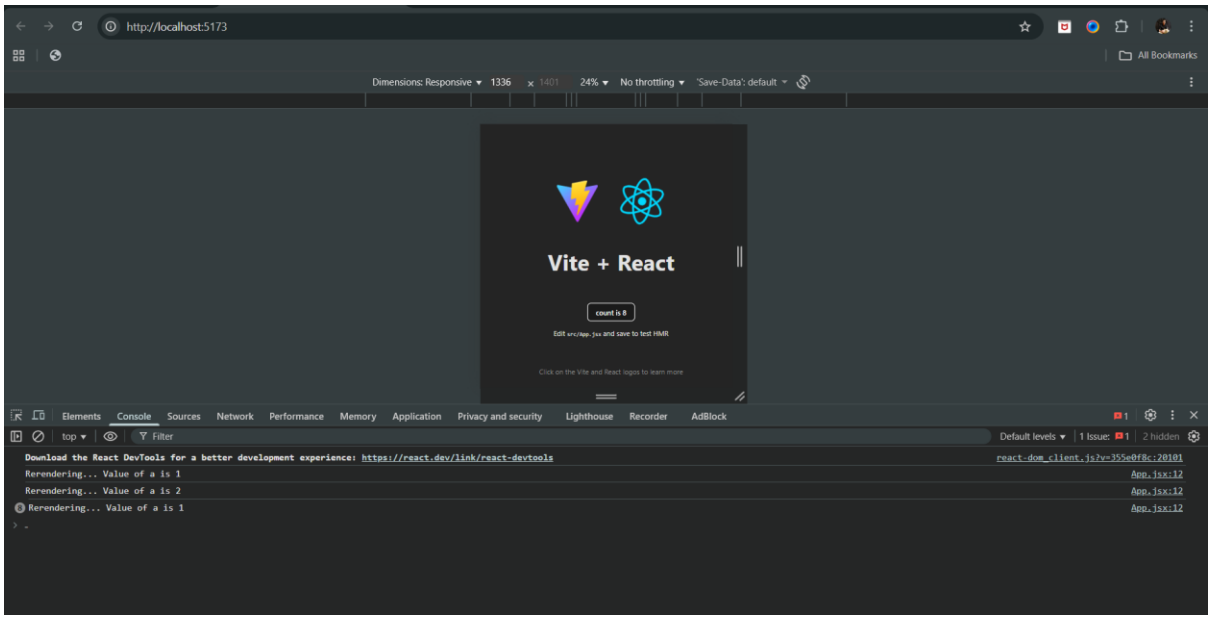
Now, suppose we don't want that re rendering to happen if the state changes, then for this we can define a variable, and assign it a value, which will change as per the logic given by you.

For this let's update the App.jsx: snippet



Output: we expect that whenever we will press the button, it will update the value of the a, but in reality, it doesn't do so. Instead it re renders the content again and again, and hence value of a = 0 is considered as a = 0 for every rendering.

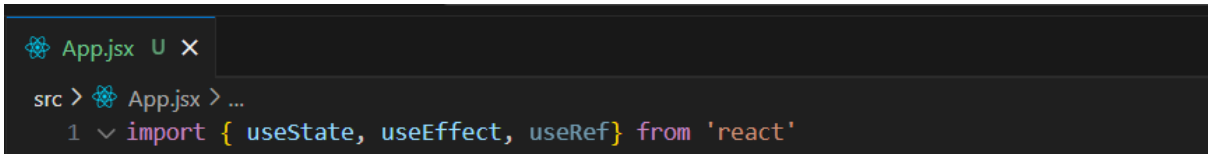
Output: console



Now, how to get out of this issue? We will use useRef.

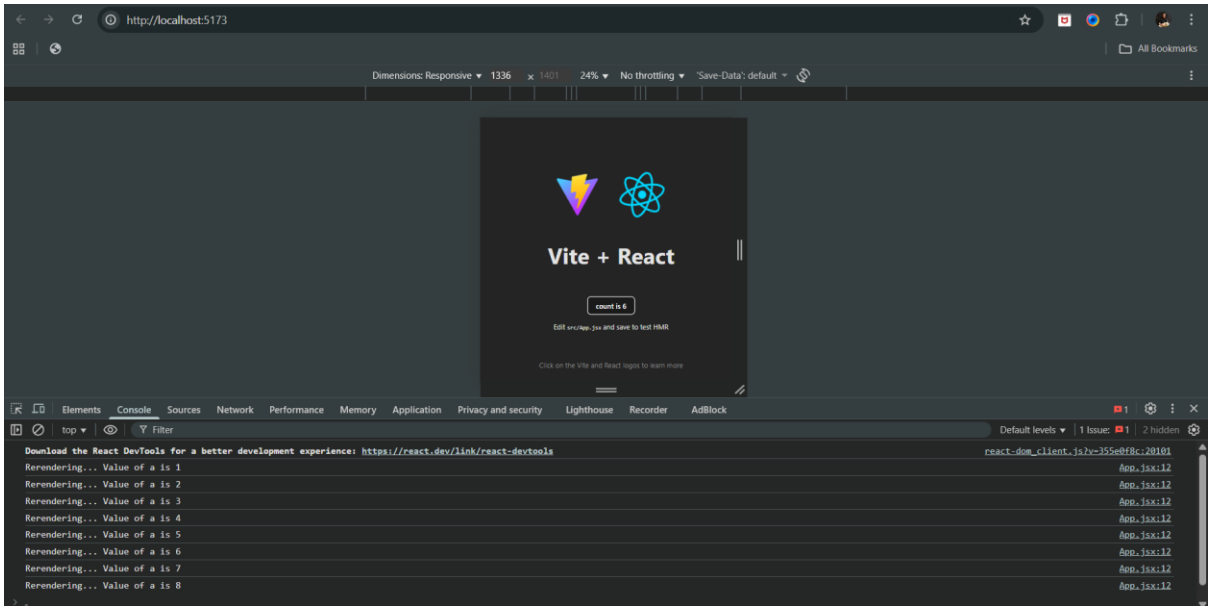
How to include it in the code?

App.jsx:



App.jsx: notice the line 8, 10 to 13.

Output: console. Yup, it worked as we expected.



Now, how to work with the [] involved in the useEffect?

For that we will refer the content we want to change:

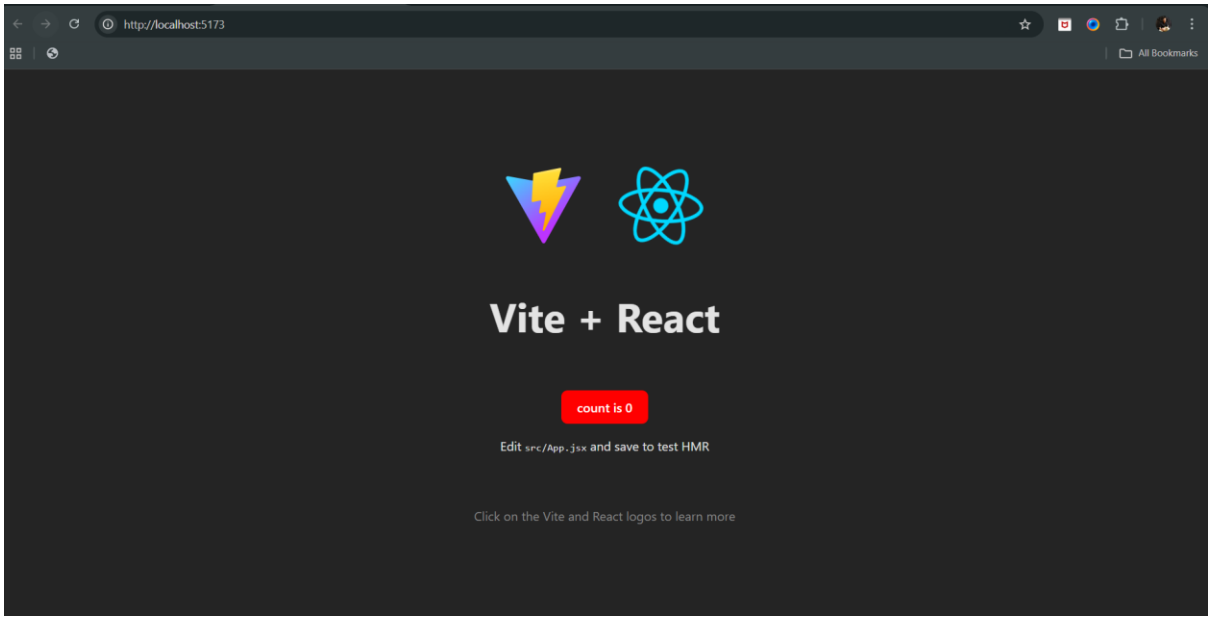
App.jsx: snippet. Notice line 8,11,12,28.

App.jsx U

src > App.jsx > App

```
1  import { useState, useEffect, useRef} from 'react'
2  import reactLogo from './assets/react.svg'
3  import viteLogo from '/vite.svg'
4  import './App.css'
5
6  function App() {
7    const [count, setCount] = useState(0)
8    const ref  = useRef(0)
9
10   useEffect(() => {
11     console.log(`First Rendering...`)
12     ref.current.style.backgroundColor = "red"
13   },[])
14
15
16   return (
17     <>
18       <div>
19         <a href="https://vite.dev" target="_blank">
20           <img src={viteLogo} className="logo" alt="Vite logo" />
21         </a>
22         <a href="https://react.dev" target="_blank">
23           <img src={reactLogo} className="logo react" alt="React logo" />
24         </a>
25       </div>
26       <h1>Vite + React</h1>
27       <div className="card">
28         <button ref={ref} onClick={() => setCount((count) => count + 1)}>
29           count is {count}
30         </button>
```

Output:



--The End--