

Example 1 : With normal Class component without hook:

import React, { Component } from 'react'

 class ClassComponentWithOneExample extends Component {

    constructor(props) {

      super(props)

      this.state = {

        count:0

      }

    }

    componentDidMount(){

        document.title=`Clicked ${this.state.count} times`

    }

    componentDidUpdate(prevProps,prevState){

        document.title=`Clicked ${this.state.count} times`

    }

  render() {

    return (

      <div>

        <button onClick={()=> this.setState({count: this.state.count+1})}>

            Click {this.state.count} times

        </button>

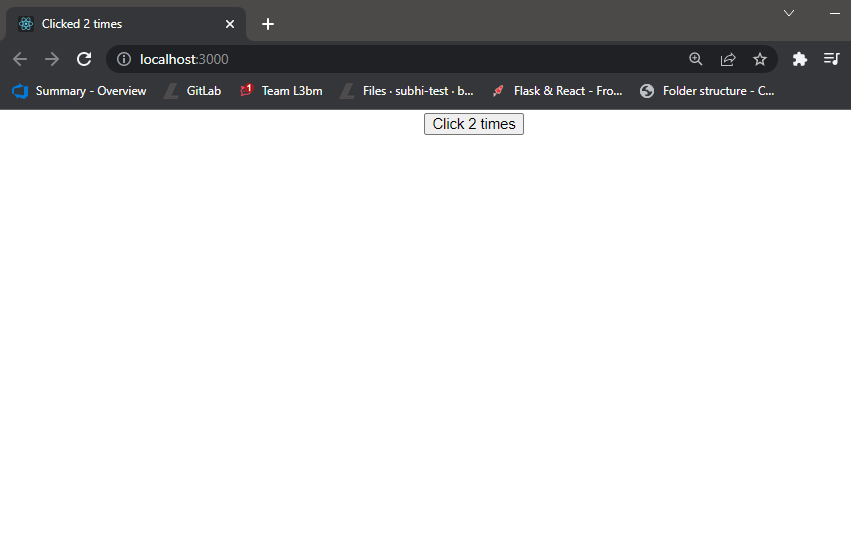
      </div>

    )

  }

}

export default ClassComponentWithOneExample



Example 2 : Using Hook Achieve Class Component Features

import React,{useState,useEffect} from 'react'

function UseEffectAfterRender() {

    const [count,setCount]= useState(0)

    useEffect(()=>{

        document.title=`Clicked ${count} times`

    })

  return (

    <div>

      <button onClick={()=> setCount(count+1)}>

        Click {count} times

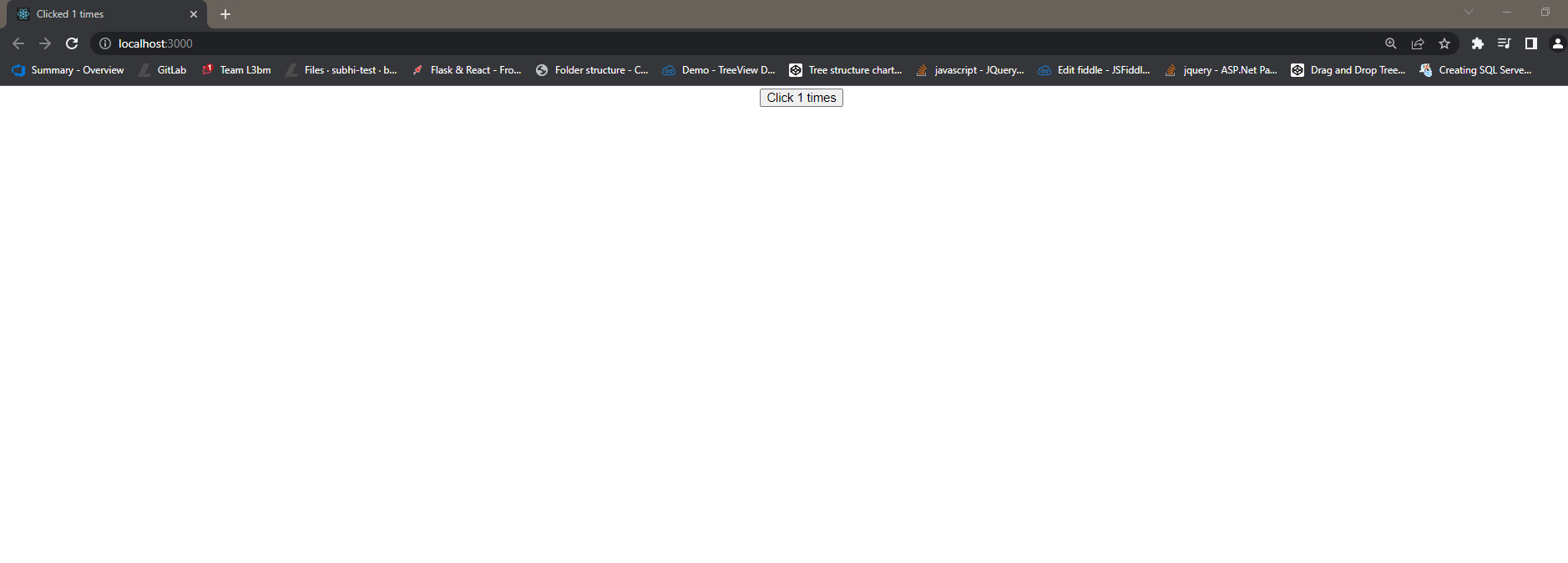
      </button>

    </div>

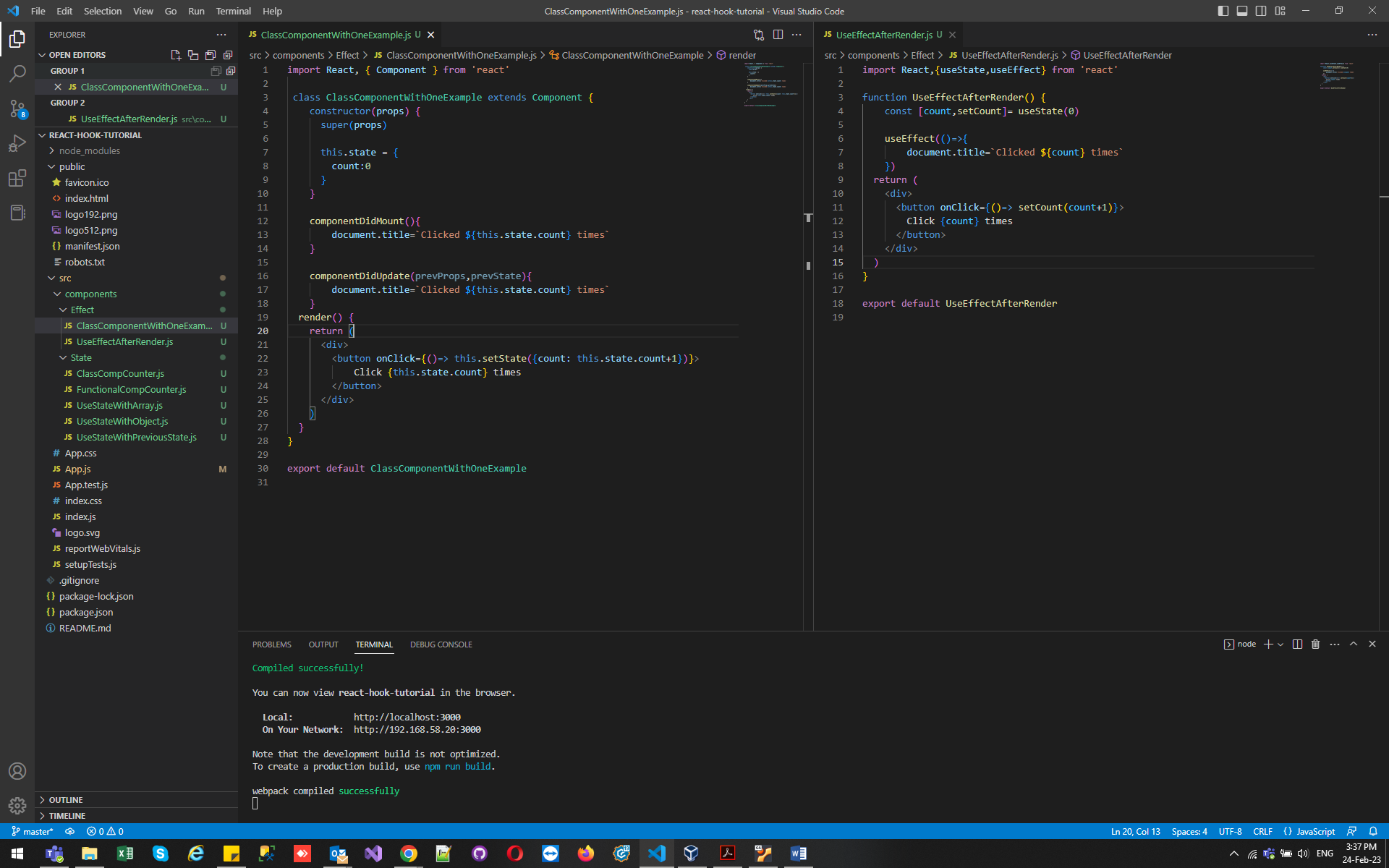
  )

}

export default UseEffectAfterRender



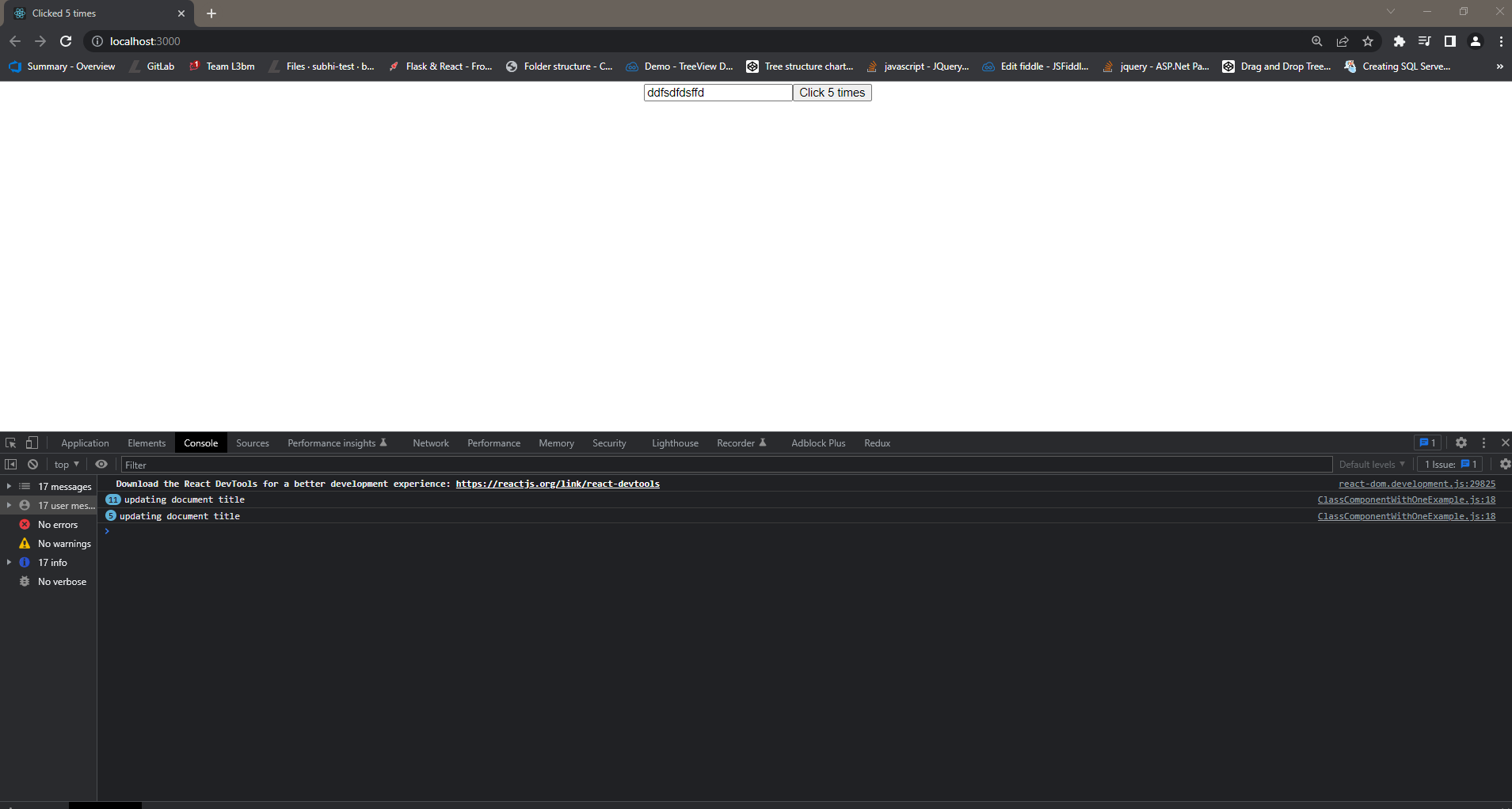
Comparison Hook and Normal Class component Code :



Example 3 : Conditionally run effects:

1. Without hook with class component :
2. import React, { Component } from 'react'
3. class ClassComponentWithOneExample extends Component {
4. constructor(props) {
5. super(props)
7. this.state = {
8. count:0,
9. name:""
10. }
11. }
12. componentDidMount(){
13. document.title=`Clicked ${this.state.count} times`
14. }
16. componentDidUpdate(prevProps,prevState){
17. console.log("updating document title")
18. document.title=`Clicked ${this.state.count} times`
19. }
20. render() {
21. return (
22. <div>
23. <input type="text" value={this.state.name} onChange={
24. e=>{
25. this.setState({name:e.target.value})
26. }
27. }/>
28. <button onClick={()=> this.setState({count: this.state.count+1})}>
29. Click {this.state.count} times
30. </button>
31. </div>
32. )
33. }
34. }
35. export default ClassComponentWithOneExample

If u see output its every time updating when u click button or input:



So for stop we can also add condition if any count change then update then update

import React, { Component } from 'react'

 class ClassComponentWithOneExample extends Component {

    constructor(props) {

      super(props)

      this.state = {

        count:0,

        name:""

      }

    }

    componentDidMount(){

        document.title=`Clicked ${this.state.count} times`

    }

    componentDidUpdate(prevProps,prevState){

        if(prevState.count !== this.state.count){

        console.log("updating document title")

        document.title=`Clicked ${this.state.count} times`

    }

    }

  render() {

    return (

      <div>

        <input type="text" value={this.state.name} onChange={

            e=>{

                this.setState({name:e.target.value})

            }

        }/>

        <button onClick={()=> this.setState({count: this.state.count+1})}>

            Click {this.state.count} times

        </button>

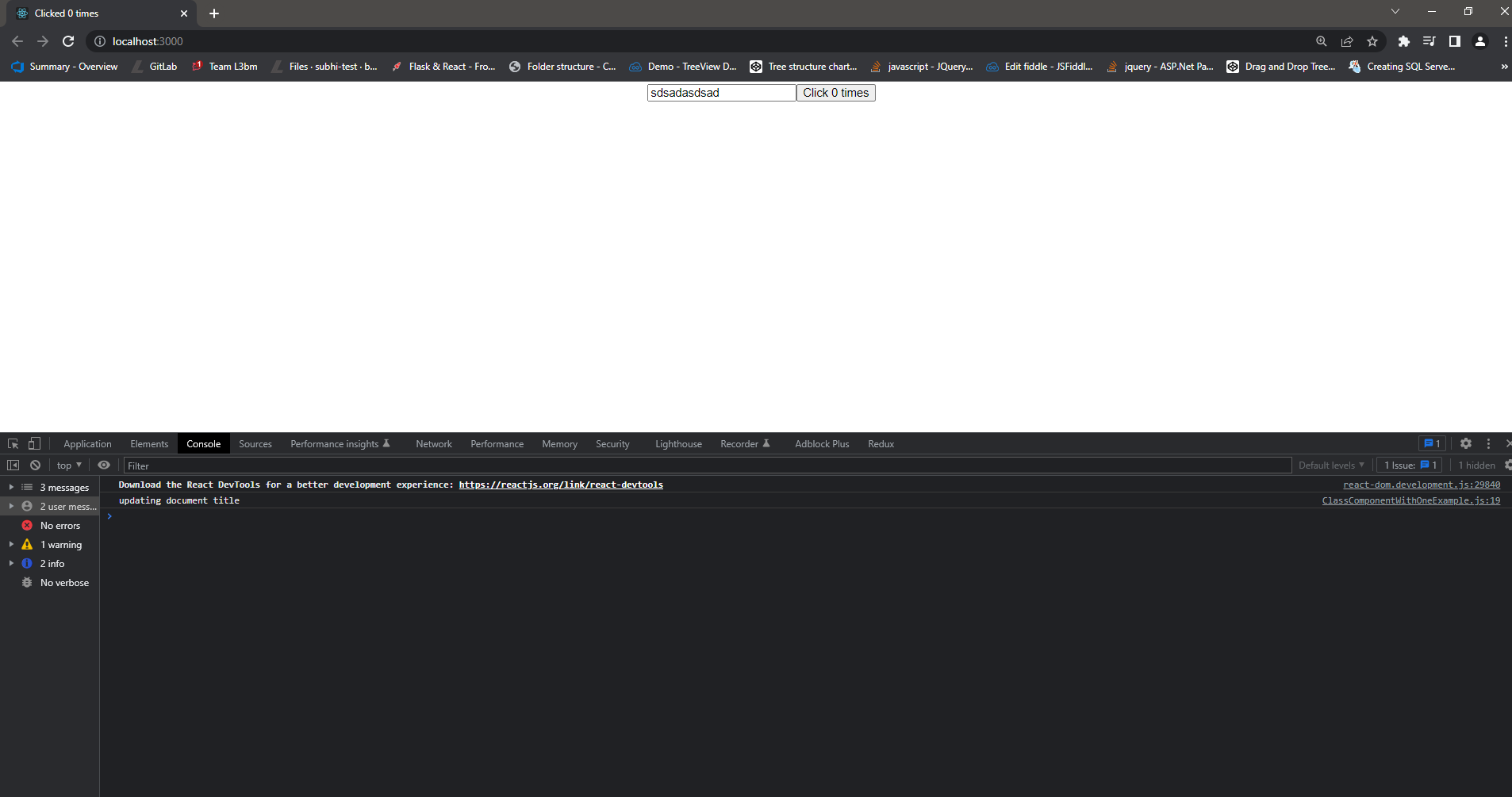
      </div>

    )

  }

}

export default ClassComponentWithOneExample



2. hook useEffect With Condition based (achieve same output likw we done with class component upper)

import React,{useState,useEffect} from 'react'

function UseEffectRunBasedOnCondition() {

    const [count,setCount]= useState(0)

    const [name,setName]= useState('')

    useEffect(()=>{

        console.log('useEffect - Updating document title')

        document.title=`Clicked ${count} times`

    },[count]) //we have added condition here

  return (

    <div>

        <input type='text' value={name} onChange={e=>setName(e.target.value)}/>

    <button onClick={()=> setCount(count+1)}>

      Click {count} times

    </button>

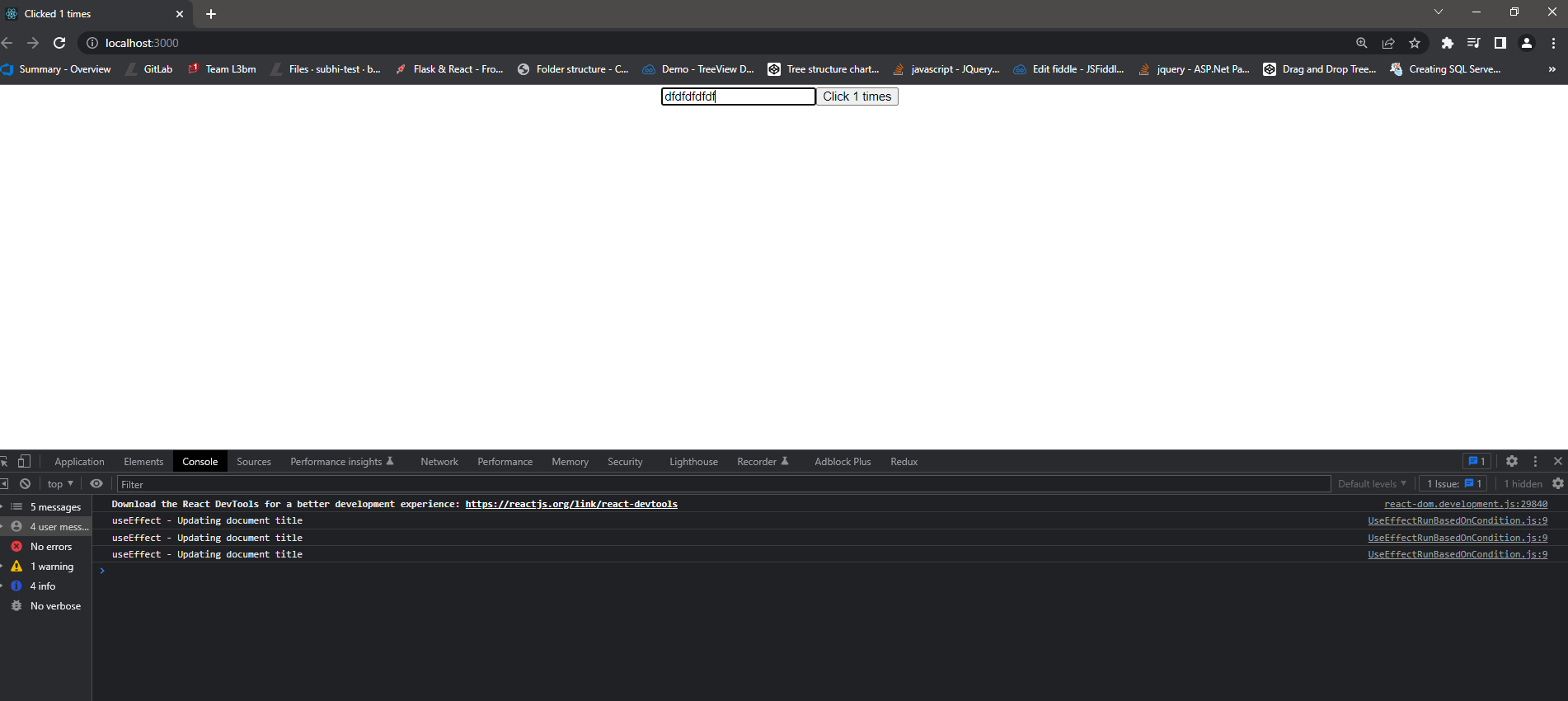
  </div>

  )

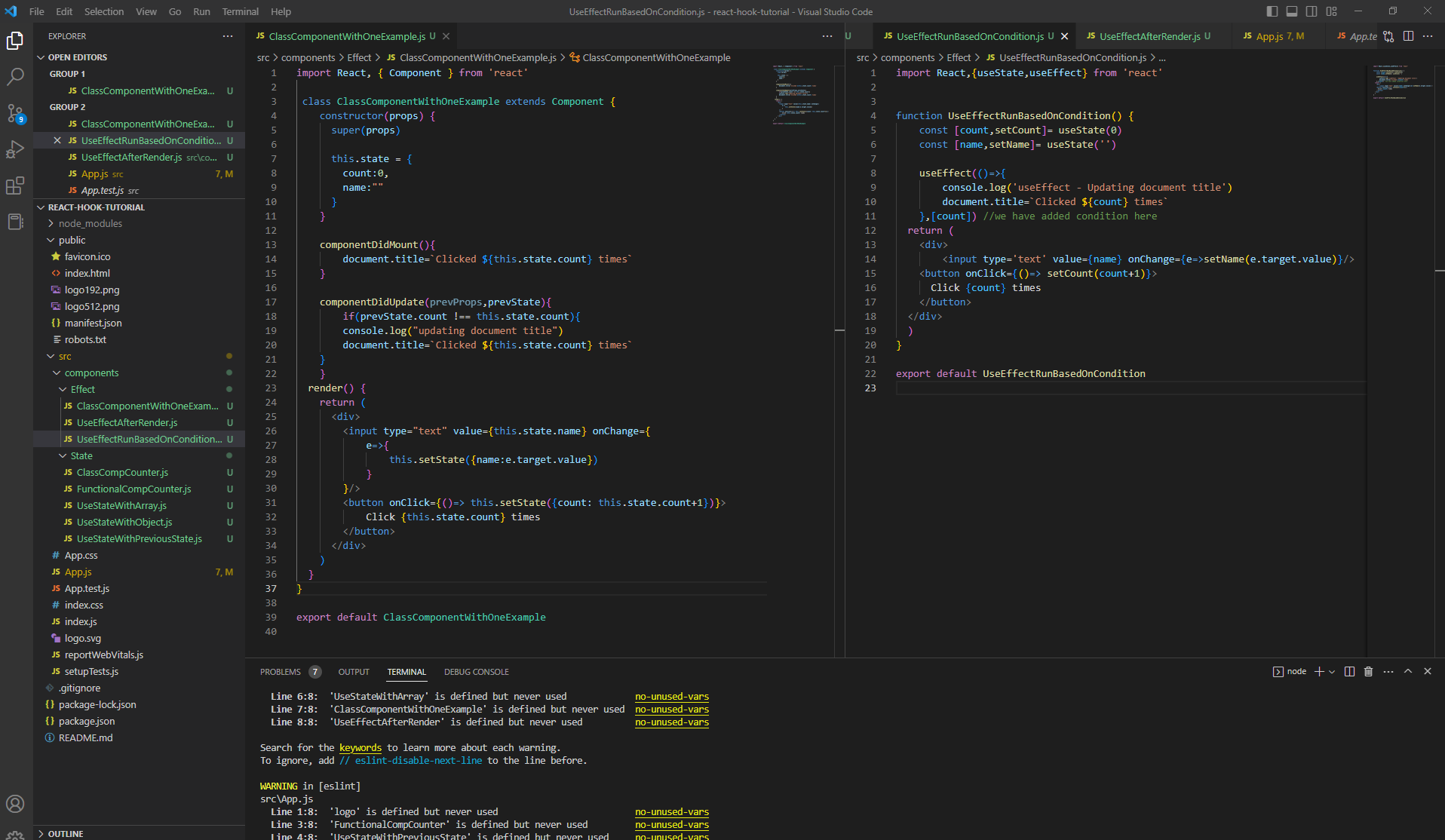
}

export default UseEffectRunBasedOnCondition

When count is change then it will fire it will not fire whwn u changing input



Comparison Hook code and normal Class Component code :



Example 3: useEffect Called once

   useEffect(()=>{

        console.log('useEffect Called')

        window.addEventListener('mousemove',logMousePosition)

    },

    []) //[] this sign tell to react render once in 2nd parameter

import React,{useState,useEffect} from 'react'

function UseEffectRunOnce() {

    const [x,setX]= useState(0)

    const [y,setY]=useState(0)

    const logMousePosition =e=> {

        console.log('Mouse Event')

        setX(e.clientX)

        setY(e.clientY)

    }

    useEffect(()=>{

        console.log('useEffect Called')

        window.addEventListener('mousemove',logMousePosition)

    },

    []) //[] this sign tell to react render once in 2nd parameter

  return (

    <div>

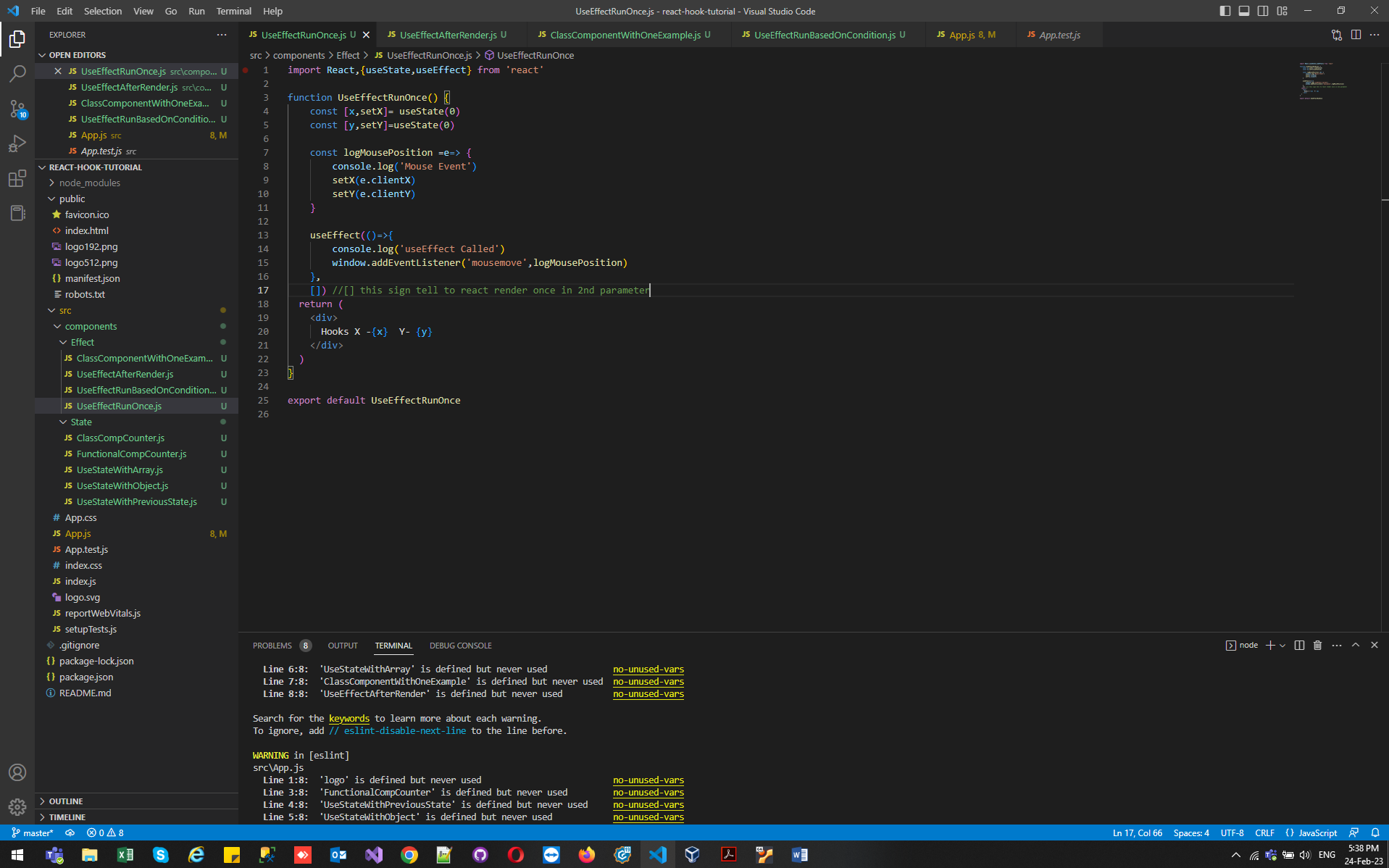
      Hooks X -{x}  Y- {y}

    </div>

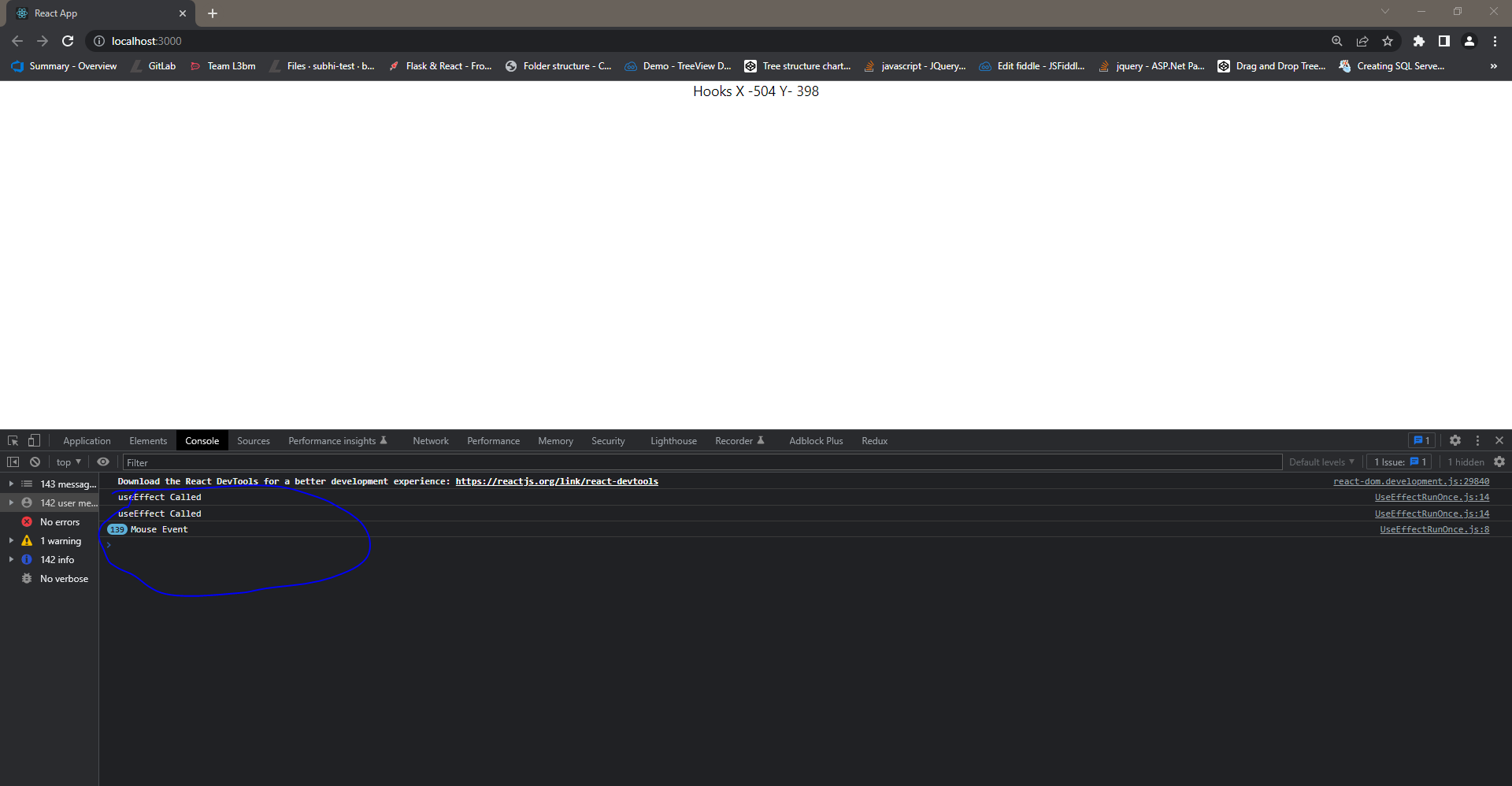
  )

}

export default UseEffectRunOnce



If you see output useeffect call during render and after render once and our mouse event call multiple times :



useEffect With cleanup :