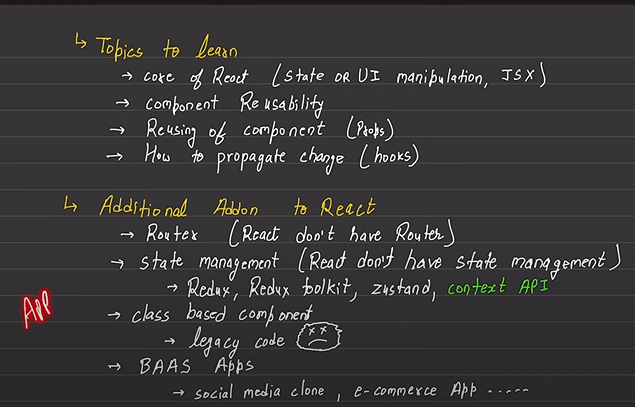
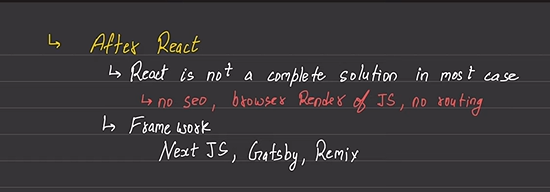


React=library--🡪so we can freedom to work on our role

In framework=-> ex jsx we want to work as given rules





8rh video

For counter if I said that in addCounter if

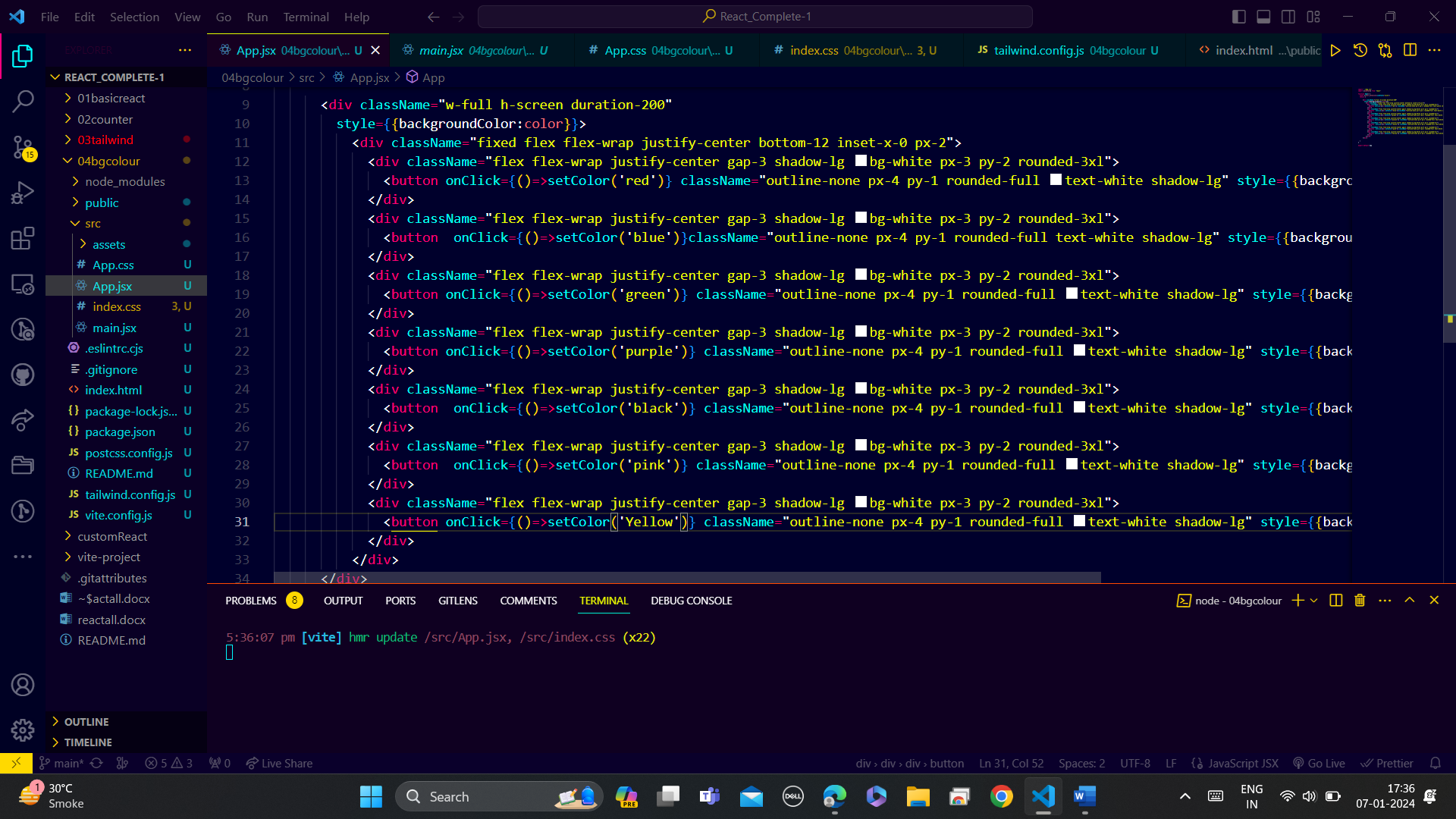
setCounter runs 5 times then what is the value



As this will not add the 15 to 19 it only ---but It will only add 1 value in it

Bez react fibre --- it will take all in bundle so only 15 will be go again in again in setCounter

9th bgcolour changer



Here onclickwants function as in the setColor we pass paramthessis but we can’t do link onClick={setColor(‘red’)} we want function to execeute the onclick

10th:

Here for increase the length use= useState(8)

For checkbox true and false will be there = useState(false) --- by default

useCallback

* If **useCallback** is not used, a new instance of **handleClick** would be created on each render of **ParentComponent**. This could lead to unnecessary re-renders of child components that use this callback, even if the function logic remains the same.
* By wrapping the callback with **useCallback**, the function is memoized and only recreated if the dependencies provided in the dependency array change. In this case, an empty dependency array **[]** indicates that the callback does not depend on any variables and will remain constant.

This callback done for rerender like something without refreshing the page depend on the dependencies if array [] is null then it will not depened upon dependencies and only run the function inside the useCallback

What done

1. Varibles will be initalize like password allowed , charallowed and all….
2. Password generator function str,pass,number and character … for loop loop pass+

useCallBack …. setPassword= optiize setPasswoord if I used password then it will go under loop

1. Input with onclick and defaultvalues
2. Copy to clipboard using useRef will be done--- using window
3. Then take reference like craete then pass that referech then use this passwordRef.current?.select() ---------------------do using use

Do by this useCallBack