**React.js**

1. React uses your keys to know what happened if you later insert, delete, or reorder the items.
2. When a function is declared in a useEffect(()=>{handleClick},[]) that function is now a stale variable.
3. A stale variable is when the variable points to the memory address of pre-render value, which is containing old value and if the value is changed the new value acquire new memory address but the stale variable(handleClick()) points to old memory and return old value.
4. In JS functions are treated as values like string or number when allotting memory space.
5. cleanUp function of hooks() is called every re-render(in first render React saves the reference of cleanUp function and call just before second render of hooks()) just before the hook which is now obligated to re-render with the component, makes the hook run as new.
6. *Learning useEffect(setup, dependencies)*
7. After your component is removed from the DOM, React will run your cleanup function.
8. React will compare each dependency with its previous value using the Object.is comparison.
9. useEffect returns undefined.
10. If some of your dependencies are objects or functions defined inside the component, there is a risk that they will cause the Effect to re-run more often than needed.
11. If your Effect is doing something visual (for example, positioning a tooltip even you want to re-paint the screen after then a click will trigger the useEffect), and the delay is noticeable (for example, it flickers), replace useEffect with **useLayoutEffect.**