

# Frontend Development

## useState Hook:

The useState hook in React is a function that allows functional components to manage local state. It takes an initial state value as an argument and returns an array containing the current state value and a function to update that value.

```
const[count, func]=useState(0);
```

0 = initial data;      count = current data      func = updated data function



```
1 import React, { useState } from 'react';
2 import './App.css';
3 function App() {
4   const [Num, SetNum] = useState(0);
5   const UP = () => { SetNum(prev => prev + 1) }
6   const DN = () => { SetNum(prev => prev - 1) }
7   return (
8     <>
9       <div className='DIV'>
10        <button onClick={UP}>+</button>
11        <h1>{Num}</h1>
12        <button onClick={DN}>-</button>
13      </div>
14    </>
15  );
16 }
17 export default App;
```

Atish Kumar Sahu

Click!

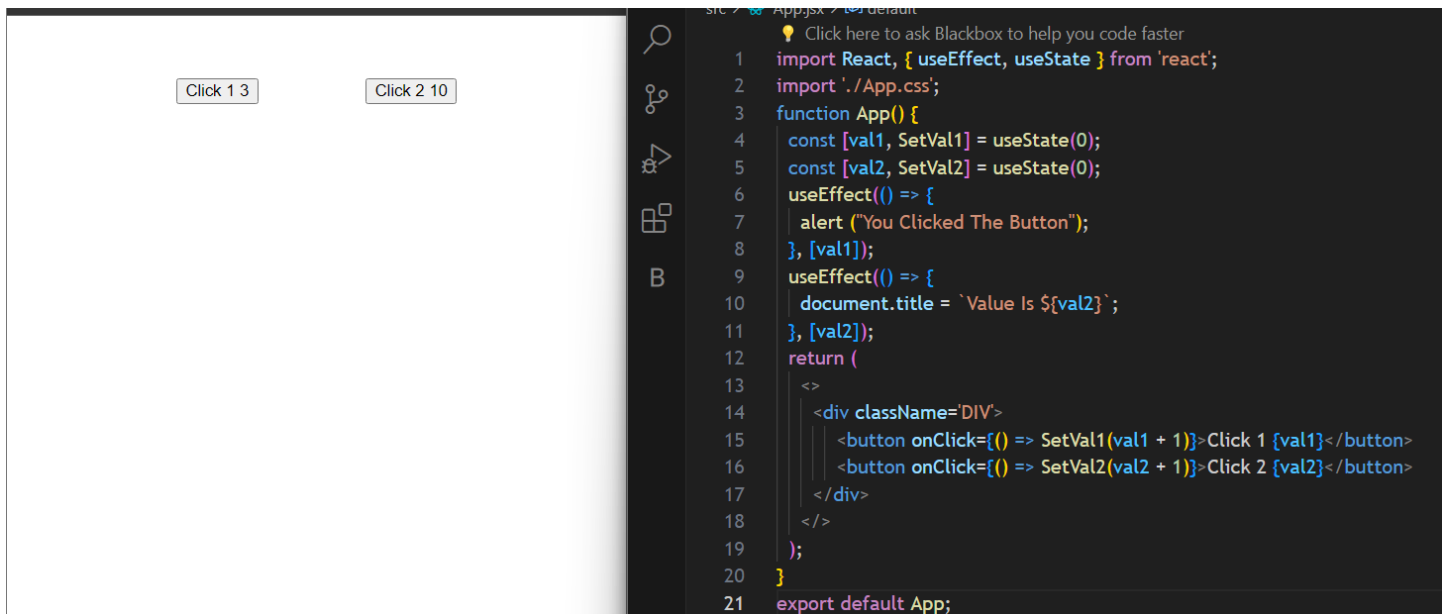
```
Click here to ask Blackbox to help you code faster
1 import React, { useState } from 'react';
2 import './App.css';
3 function App() {
4   const [Name, SetName] = useState("Atish Kumar Sahu");
5   const Click = () => {
6     let val1 = Name;
7     val1 === "Mritunjay Atish Kumar Sahu" ?
8     SetName("Atish Kumar Sahu") : SetName("Mritunjay Atish Kumar Sahu");
9   }
10  return (
11    <>
12      <div className='DIV'>
13        <p>{Name}</p>
14        <button onClick={Click}>Click!</button>
15      </div>
16    </>
17  );
18 }
19 export default App;
```

## useEffect Hook:

`useEffect` is a hook provided by React, you tell React that your component needs to do something after render. React will remember the function you passed we'll refer to it as our effect, and call it later after performing the DOM updated. By default it runs both after the first render and after every update.

The `useEffect` hook in React is a special addition to the component lifecycle. instead of thinking in terms of "mounting" and "updating" you might find it easier to think that effects happen after render.

React guarantees the DOM has been updated by the time it runs the effects. If you are familiar with react class lifecycle method you can think of `useEffect` hook as `componentDidMount`, `componentDidUpdate`, and `componentWillUnmount` combined.



```
src / App.jsx / default
Click here to ask Blackbox to help you code faster
1 import React, { useEffect, useState } from 'react';
2 import './App.css';
3 function App() {
4   const [val1, SetVal1] = useState(0);
5   const [val2, SetVal2] = useState(0);
6   useEffect(() => {
7     alert("You Clicked The Button");
8   }, [val1]);
9   useEffect(() => {
10    document.title = `Value Is ${val2}`;
11  }, [val2]);
12  return (
13    <>
14      <div className="DIV">
15        <button onClick={() => SetVal1(val1 + 1)}>Click 1 {val1}</button>
16        <button onClick={() => SetVal2(val2 + 1)}>Click 2 {val2}</button>
17      </div>
18    </>
19  );
20 }
21 export default App;
```