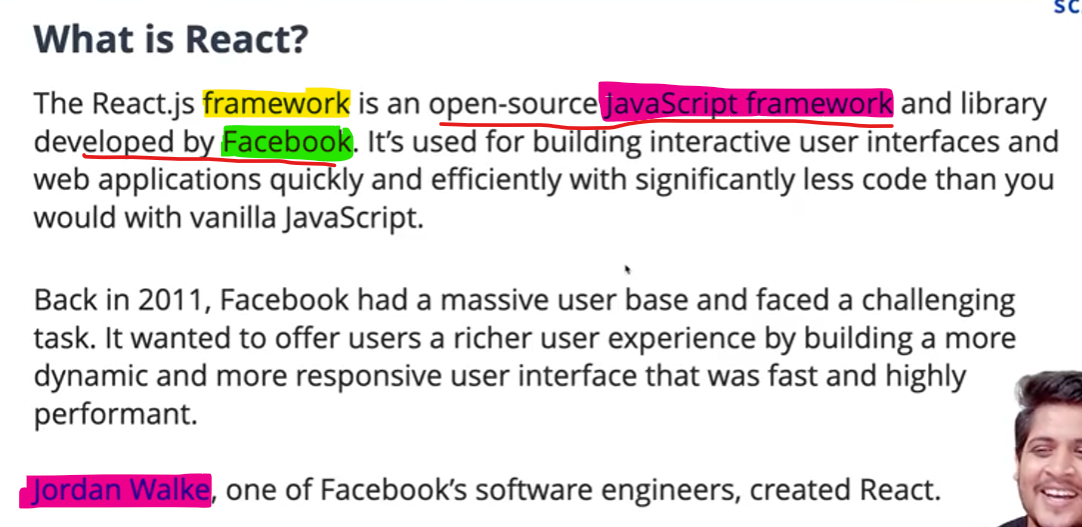
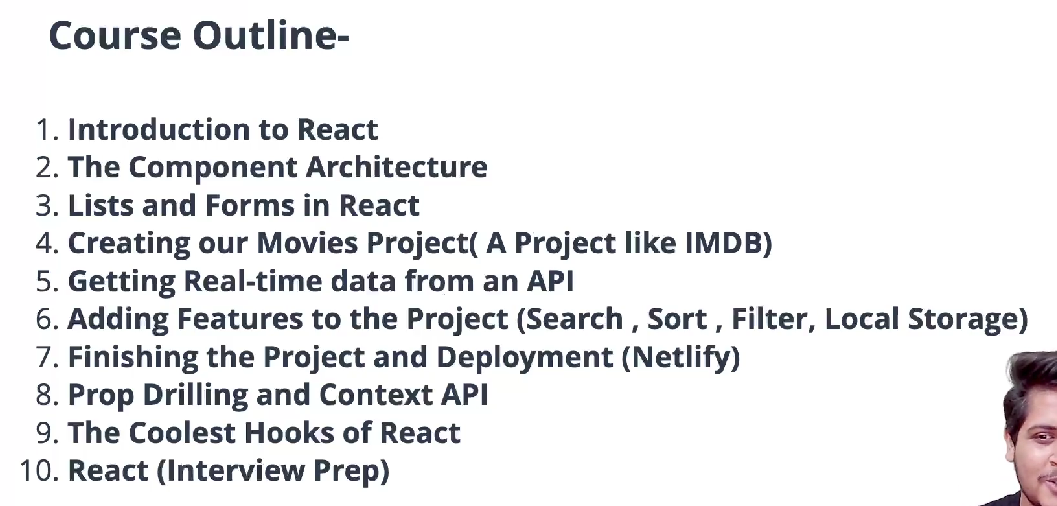
**REACT - SCALAR**

1. ***Introduction to React JS* :**
   1. **Introduction :**



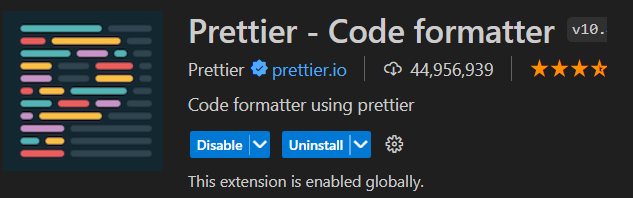




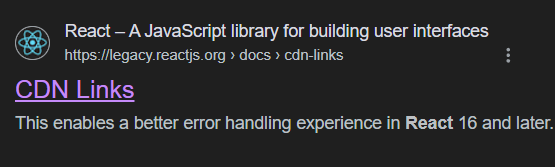
* 1. **Setting Necessary Tools for React :**
     1. **VS CODE.**
     2. **Install extension in VS Code:**



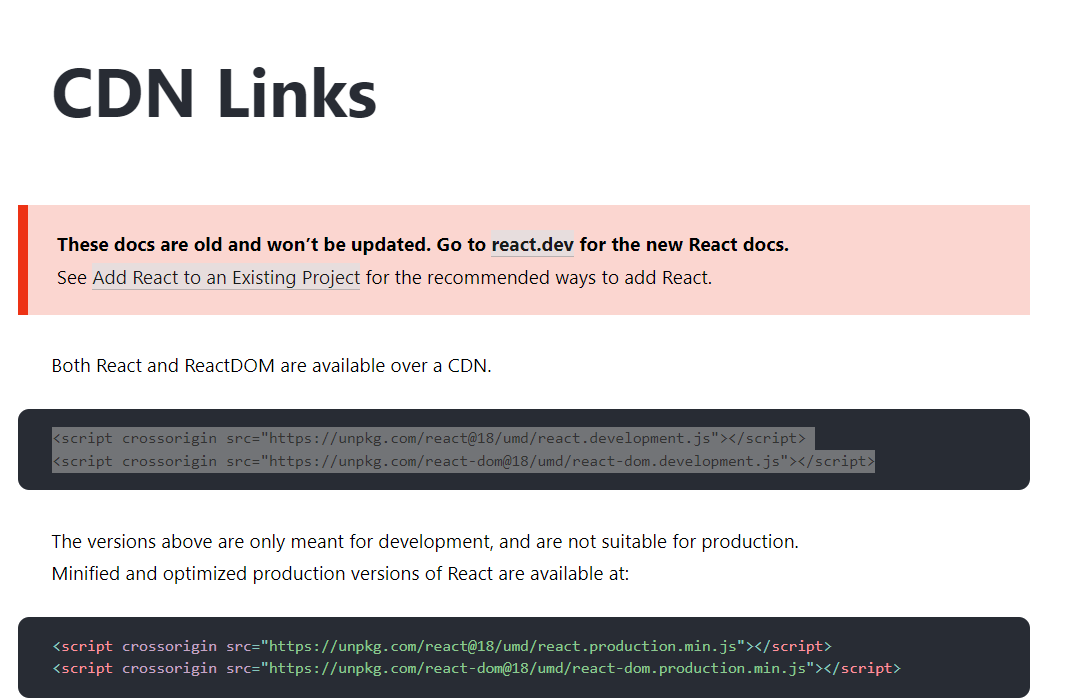
* + 1. **Install extension “prettier” in VS code :**



* + 1. **Install NodeJS . (React need support of Node JS).**
  1. **React with CDN :**
     1. Using CDN links in React gives quick and convenient way to integrate React into your projects without the need for a package manager.
     2. Type “cdn links for react” and go to the first link.



* + 1. Then copy and paste the two script links to your “index.html”(One link is react script and other is react-dom script).



* + 1. React is an External Library(JS Library).Our browser doesn’t know react exist or not.Our browser only knows the ‘DOM’.So the above two script links(1st link has all the properties and 2nd link helps React to communicate with DOM ) helps the the browser to get React .

“index.html” :

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

  </head>

  <body>

    <div id="root">

      <h1>Hello from HTML!</h1>

    </div>

    <!-- <script>

        //How to add h1 tag using JS:

        const message = document.createElement('h1');//h1 element created

        message.innerHTML = 'Hello From JavaScript';//message created

        const root = document.getElementById('root');//Getting the id of div tag to pass the message.

        root.appendChild(message);//message appended to div with id "root"

    </script> -->

    <!--CDN Links-->

    <script

      crossorigin

      src="https://unpkg.com/react@18/umd/react.development.js"

    ></script>

    <script

      crossorigin

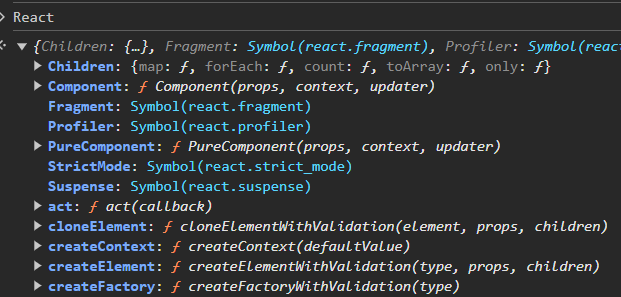
      src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"

    ></script>

  </body>

</html>

* 1. **Your First React code :**
     1. Goto inspect and select console and type ‘React’ ,then it will display all properties/methods of react :



* + 1. Method “createElement”:

<!--First React code-->

<script>

//step 1:create an element

const message = React.createElement('h1',{},'Hello From React.');

//step 2:setting root for the React.Here the root is the div with id='root'

      const root = ReactDOM.createRoot(document.getElementById('root'));

//inside the root the above specified 'h1' will be created.

      root.render(message);//whole message get rendered over here

</script>

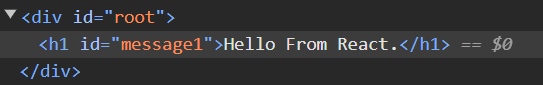
**OUTPUT:**



* 1. **Challenges with React CDN:**
     1. To Pass parameters (),specify it inside the “{ }” :

const message = React.createElement('h1',{id:'message1'},'Hello From React.');

**Inspect:**



* + 1. Now we have to build following in React:



* + 1. So comment div with id=’root’ and build React code to display like above.Previously when we passed ‘Hello from react’ there had only one div with id=’root’.But Here we have two divs ,i.e div with id=’child’ inside div with id=’parent’.So instead the message we have to put one more ‘React.createElement()’ :

**Body:**

<div id="parent">

        <div id="child">

          <h1 id="greeting"></h1>

          <h2 id="greeting2"></h2>

        </div>

      </div>

**Script :**

<script>

      const message = React.createElement(

        "div",

        { id: "parent" },

        React.createElement("div", { id: "child" }),[ //using array to store multiple element

        React.createElement("h1",{id:"greeting"},'Hello from react.'),

        React.createElement("h2",{id:"greeting2"},'Hello from react heading2.')

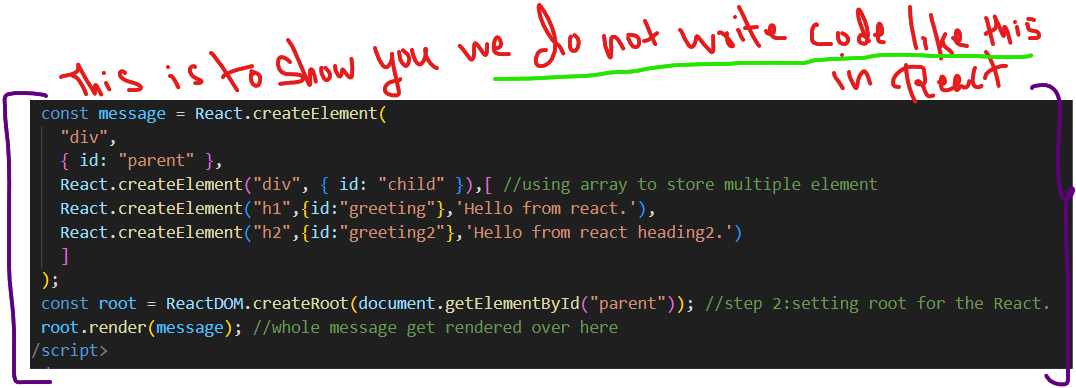
        ]

      );

      const root = ReactDOM.createRoot(document.getElementById("parent")); //step 2:setting root for the React.

      root.render(message); //whole message get rendered over here

    </script>



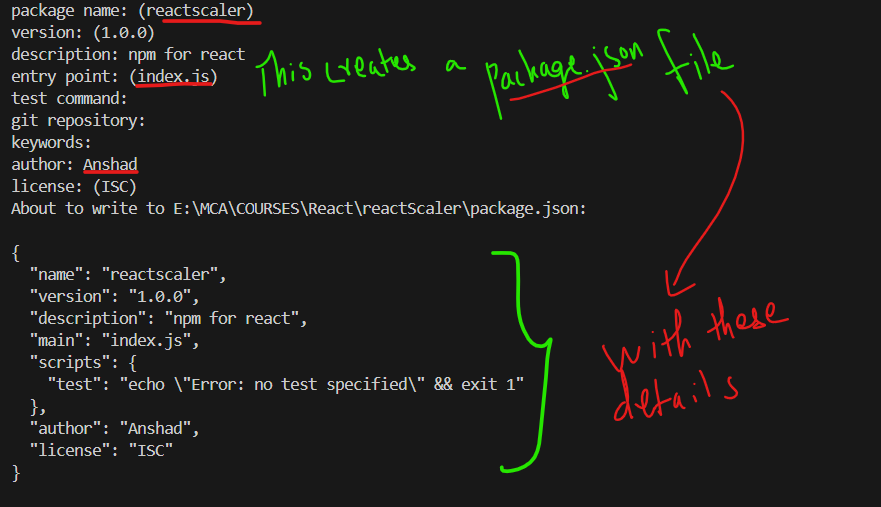
1. ***Setting up the Development Environment :***
   1. **Setting up NPM:**
      1. Since React CDN has some Challenges.Complex codes.So we make use of some Built Tools.
      2. Before moving to Built tools like (webpack,vite,parcel etc..) ,We need to understand about NPM(Node Package Manager).
      3. **NPM :** *How to Install and use an NPM Package* ?





* + 1. How to Initialize/ Use npm for your directory,Open certain folder in terminal and type:’npm init’ :

npm init



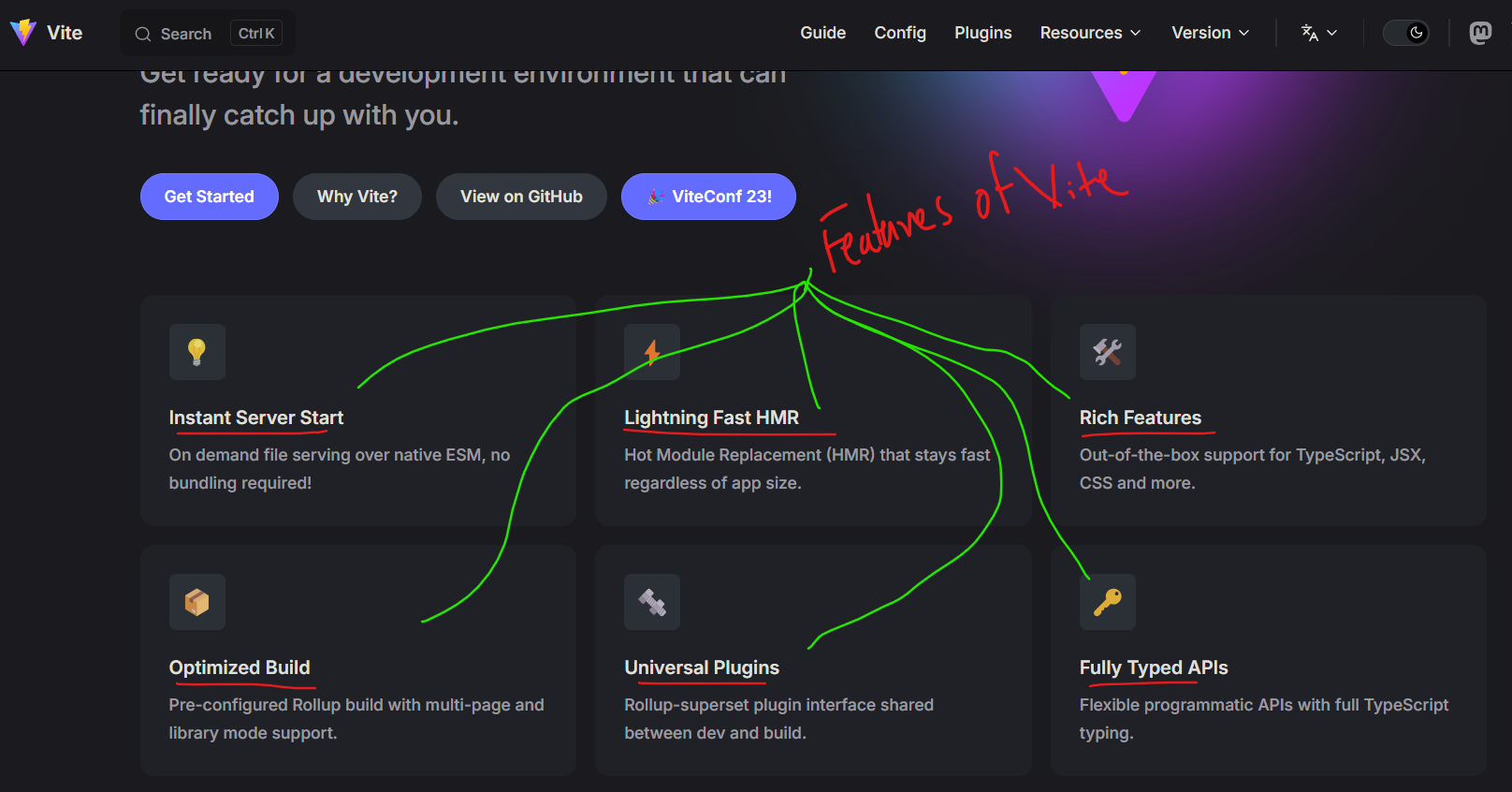
* 1. **Setup React App with Vite(a built in tool) :**
     1. What is Vite tool?

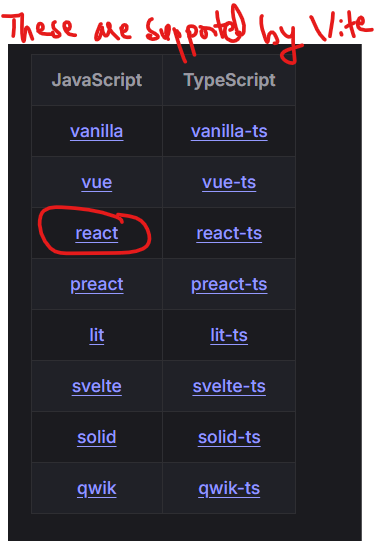
**Vite is a next-generation, front-end tool that focuses on speed and performance.**

* + 1. Why Vite is used?

**It Lets the browser take over part of the job of a bundler.[** Our project will have all type of files like html,css,images,js and so on.So this vite **bundles these all together** and **browser to understand** all these codes and for better performance.**]**

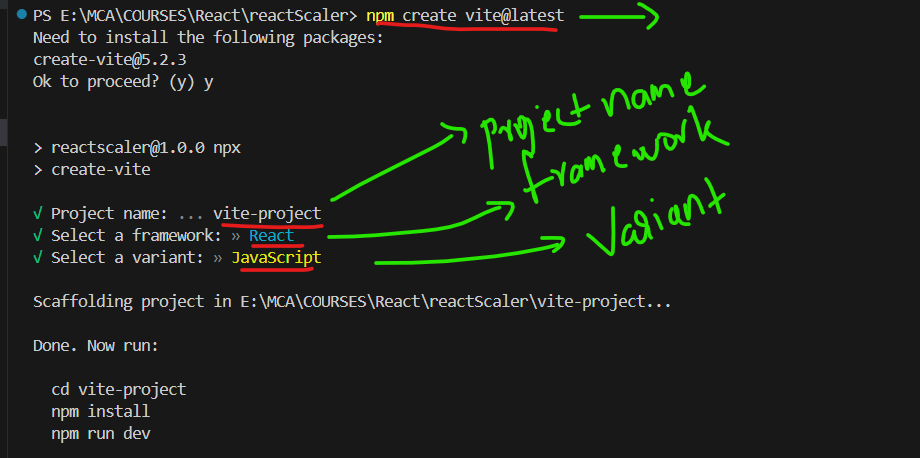
* + 1. Features of Vite(Goto vite website):



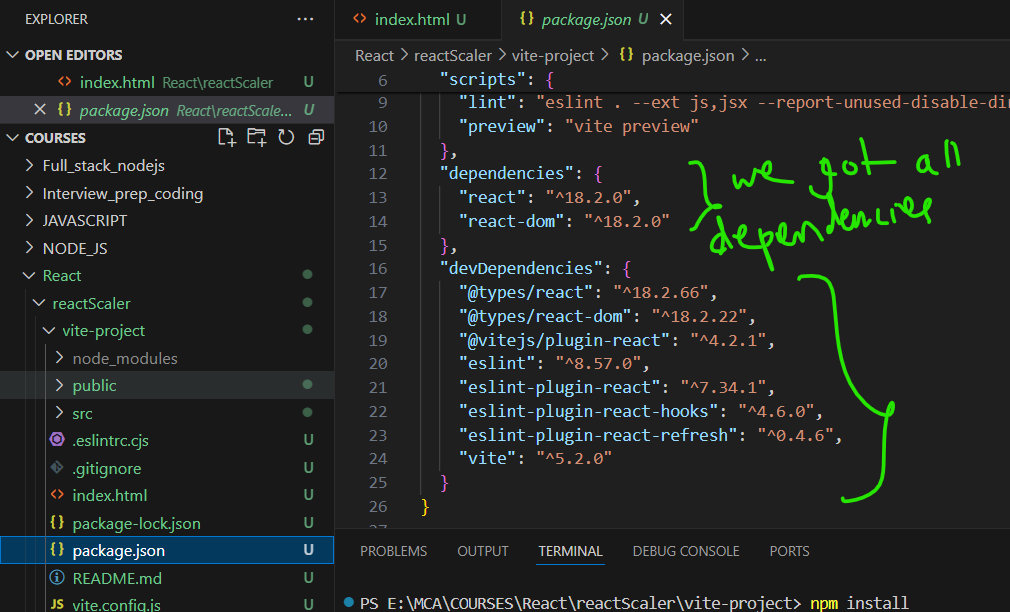


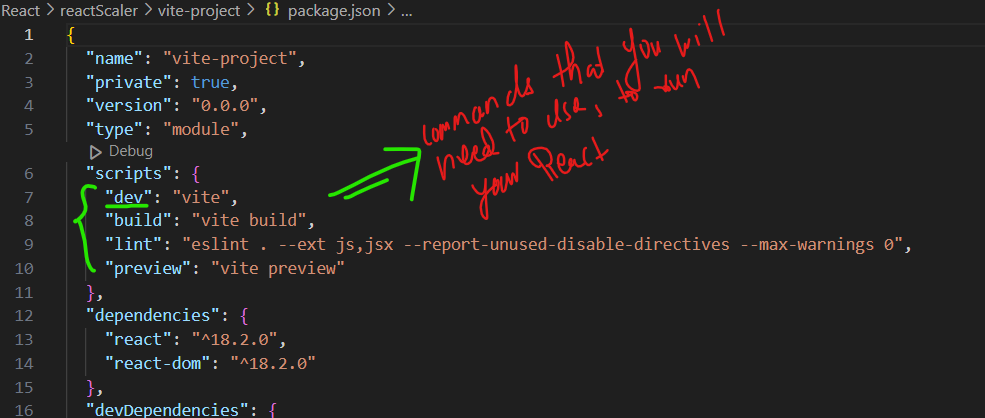
* + 1. How to use Use Vite inside your folder ?

1. First run command “**npm create vite@latest**”.
2. Give project name ->Eg:vite-project
3. Select Framework ->React.
4. Select variant-> JavaScript.

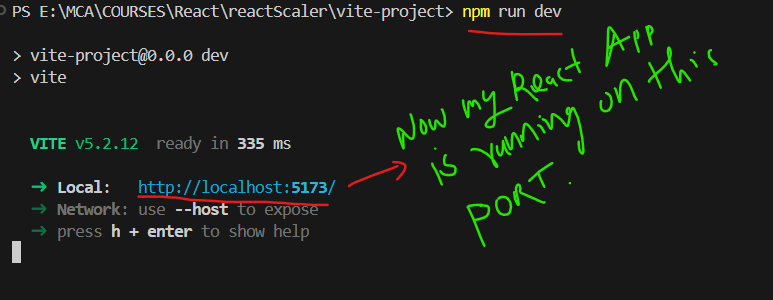


1. Now open that ‘vite-project’ folder in terminal and install npm : “npm install”.So that ‘vite-project’ will get all the dependencies .



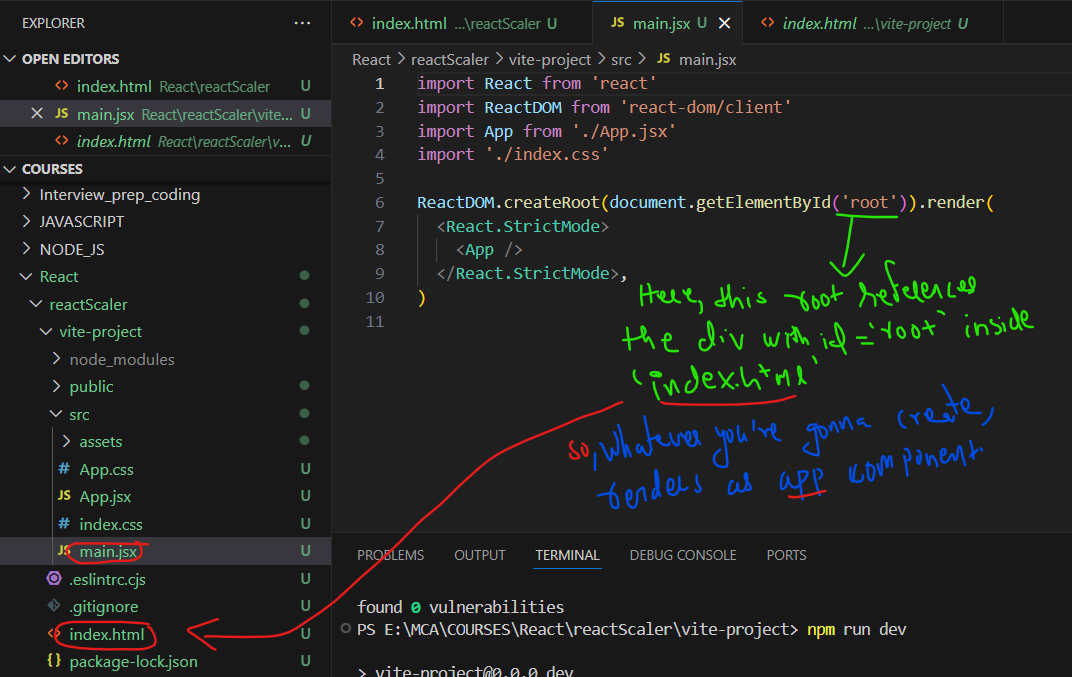


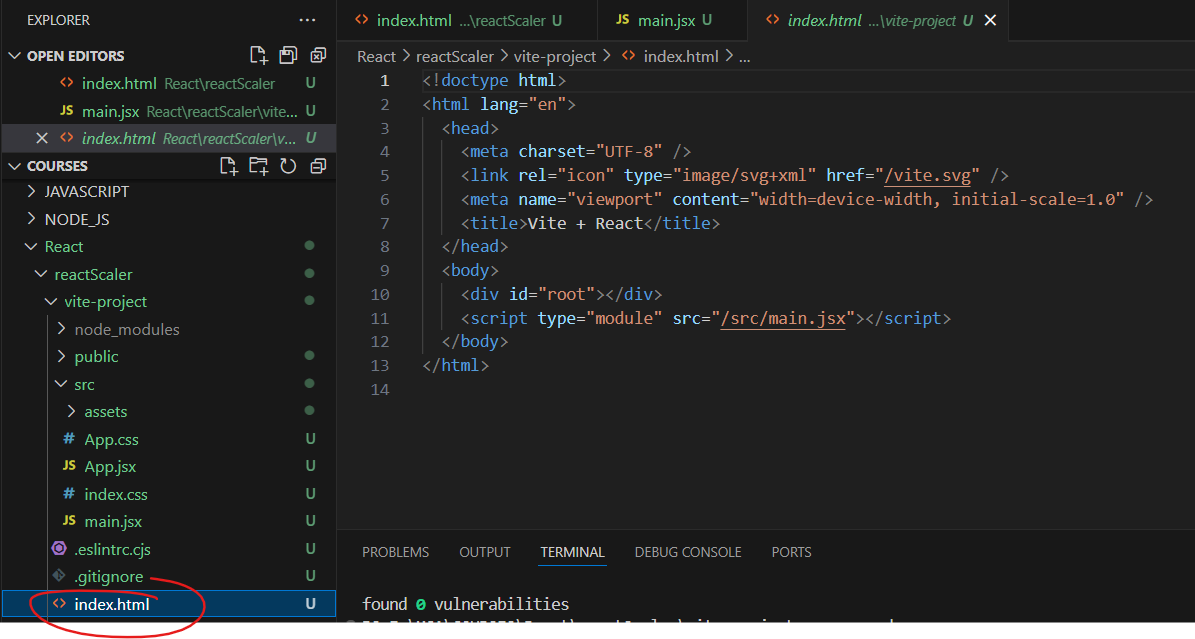
1. ‘dev’ command is used to RUN your React project: “npm run dev”.



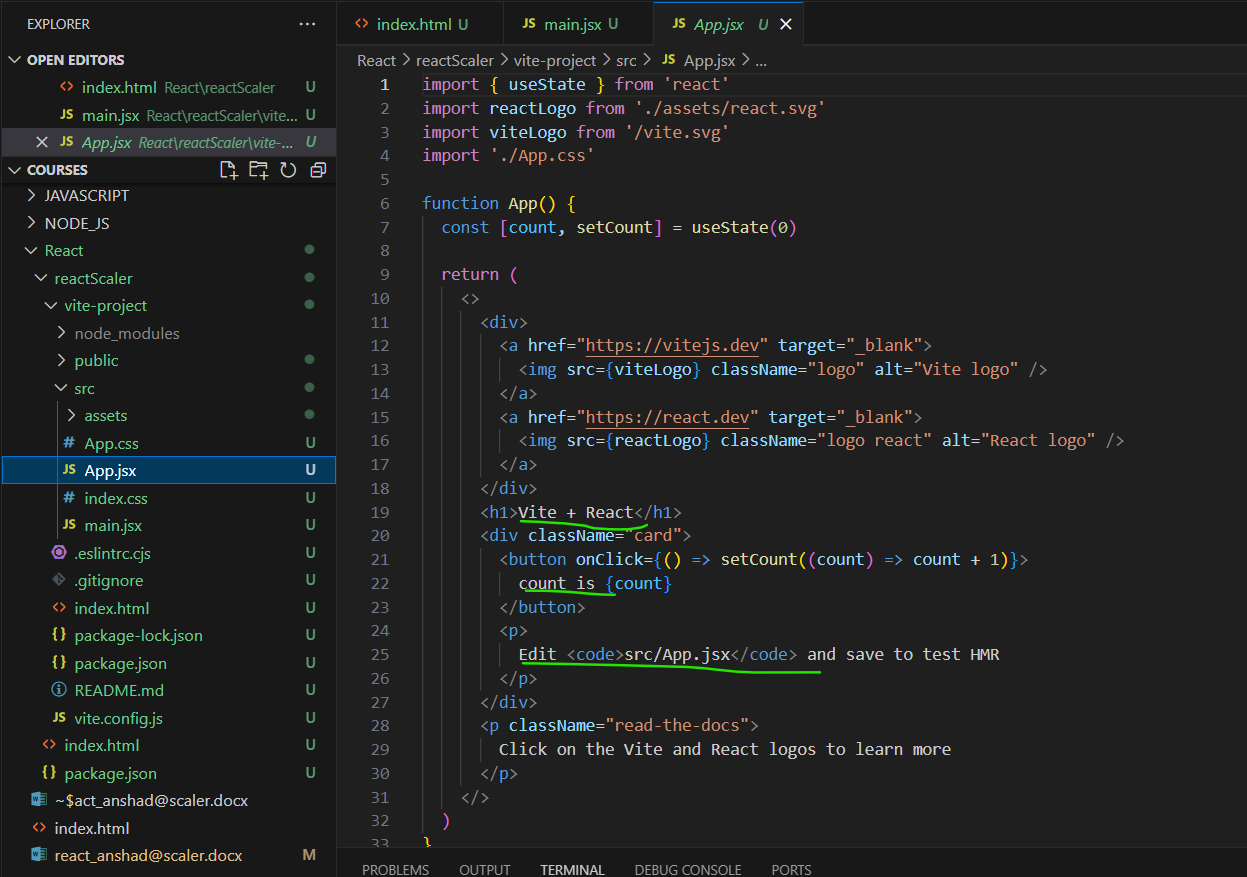
1. Now our React App is Ready:



* 1. **Folder Structure :**
     1. Open src>‘main.jsx’ file .
     2. ‘.jsx’ is the new way of writing JS files recommended by React.

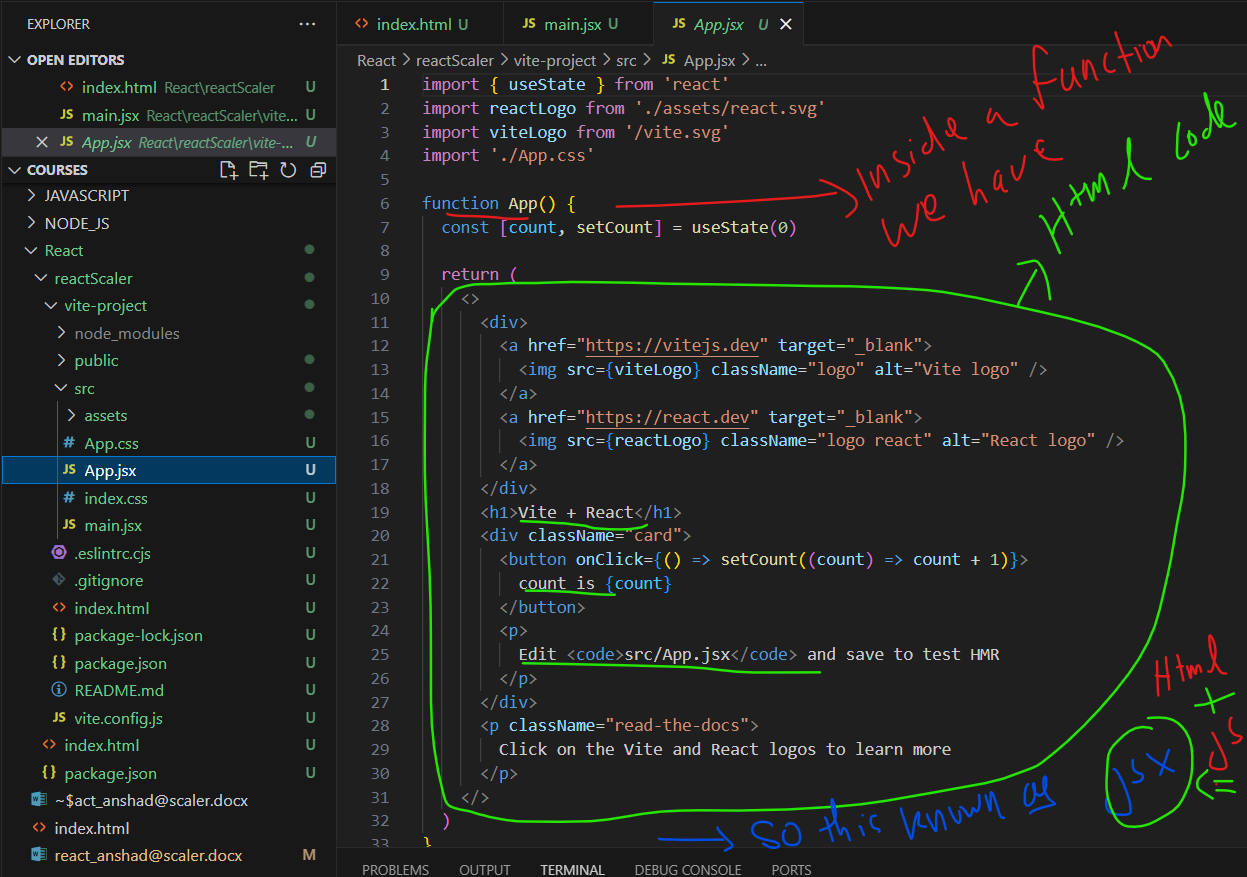


* + 1. So whatever we are gonna create is rendered as App component.
    2. What is Component ?=> Eg:-We can see different sections in Instagram like stories,feed and so on.To Manage our application in Efficient way.Components are just part of your Application.So this Component will also be inside one component right.That top most component is <App> .
    3. So “App.jsx” file is the component <App> file.What we saw previously in browser is the output of this file.



**OUTPUT:**





* + 1. Here in ‘App.jsx’ we can see html code inside function.So that is why this is a jsx file [HTML + JavaScript =>jsx]
    2. How browser Understands this Html code inside a Javascript code ?=

=>With the Help of Tool Babel.It **Takes the JavaScript code and Transfile it to HTML** code .So that browser can understand it.

* + 1. ‘index.css’ and ‘App.css’ are default css files which gives a default style to the APP.

1. ***The Component Architecture and JSX :***
   1. **Create Your First React Component :**
      1. Before creating First Component,lets clear some deafult code in ‘App.jsx’(It is our Parent component):

function App() {

  return (

    <h1>Hello From React</h1>

  )

}

export default App

* + 1. First Create a **Folder** named ‘components’ inside ‘src’ .
    2. Now Create a **Component** inside folder ‘components’ say ‘Hello.jsx’(This component Prints Hello).
    3. [NOTE:**The first letter of component should be CAPS**] .
    4. Now we are gonna use this component inside the Parent component ‘App.jsx’.For that First we have to Export that ‘Hello.jsx’ :

function Hello(){

    return

    <h1>Hello from hello Component.</h1>

}

export default Hello

* + 1. Same like,we have to Import that exported component in ‘App.jsx’ :

import Hello from './components/Hello'

* + 1. Now we need to use this component ‘Hello’ :

import Hello from './components/Hello'

function App() {

  return (

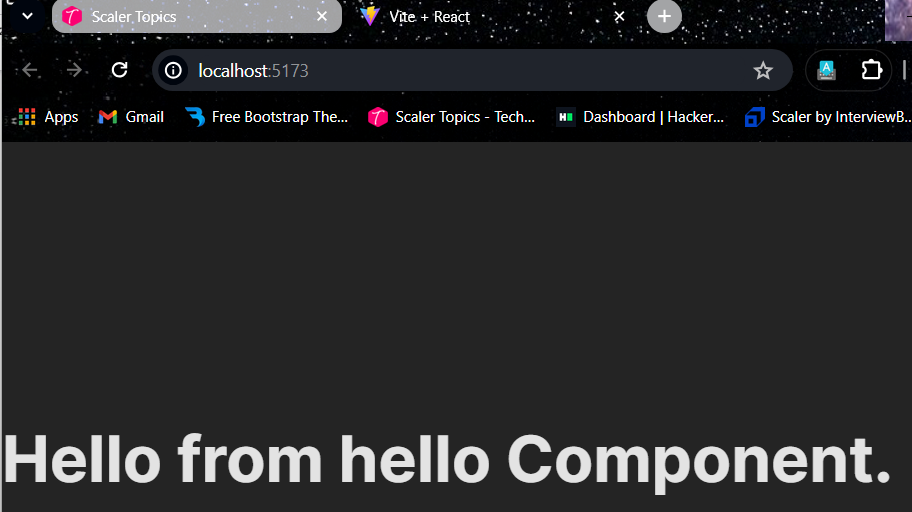
    <Hello/> //Here we  use the component 'Hello' which is imported.

  )

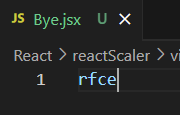
}

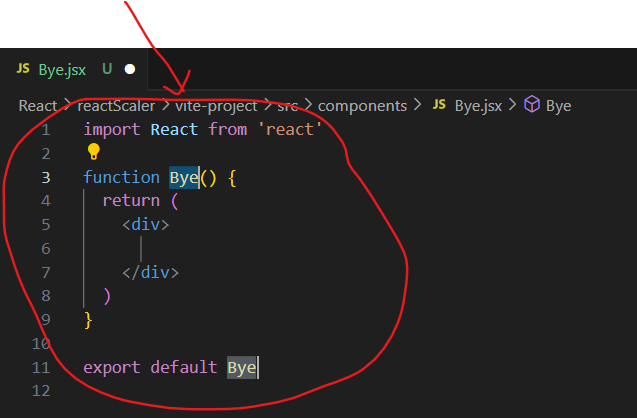
export default App

OUTPUT:



* + 1. Now lets create another component say ‘Bye.jsx’ .
    2. **[Note :** For Shortcut type ‘rfce’ and enter to get a structure of code . **]**





* + 1. Now Edit this code according your need:

import React from 'react'

function Bye() {

  return (

    <h1>Bye from Bye component</h1>

  )

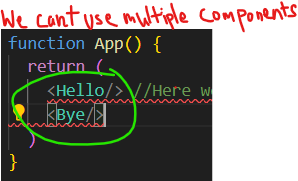
}

export default Bye

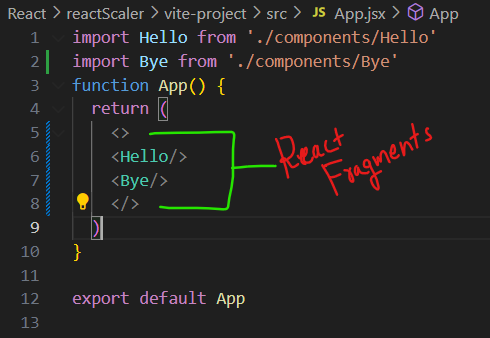
* + 1. Then import this Bye component inside ‘App.jsx’:

import Bye from './components/Bye'

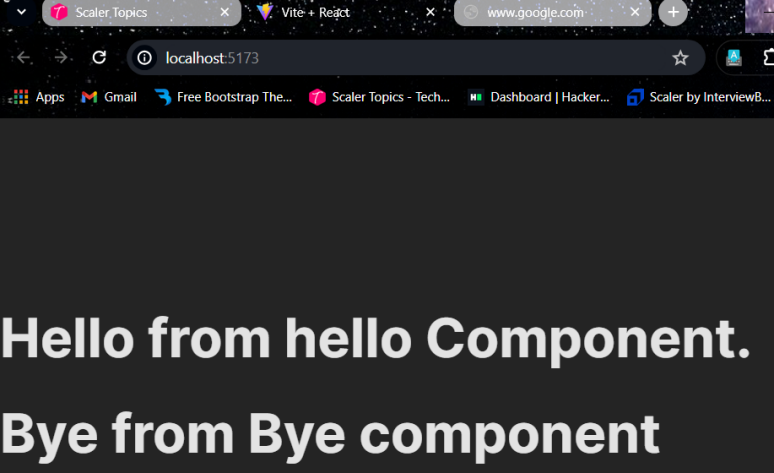
* + 1. Now we have to Use this component.But we already have used another component ‘Hello’ .



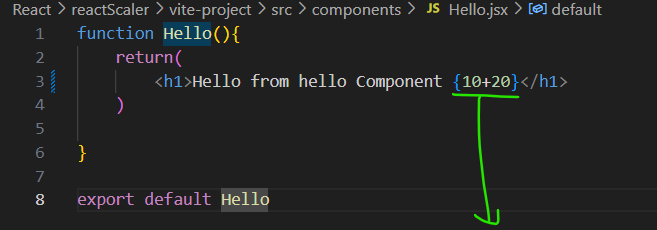
* + 1. To use multiple components together :We need to use ‘React Fragments’.
    2. HThat is we have to wrap your whole components inside ‘<> </>’ :

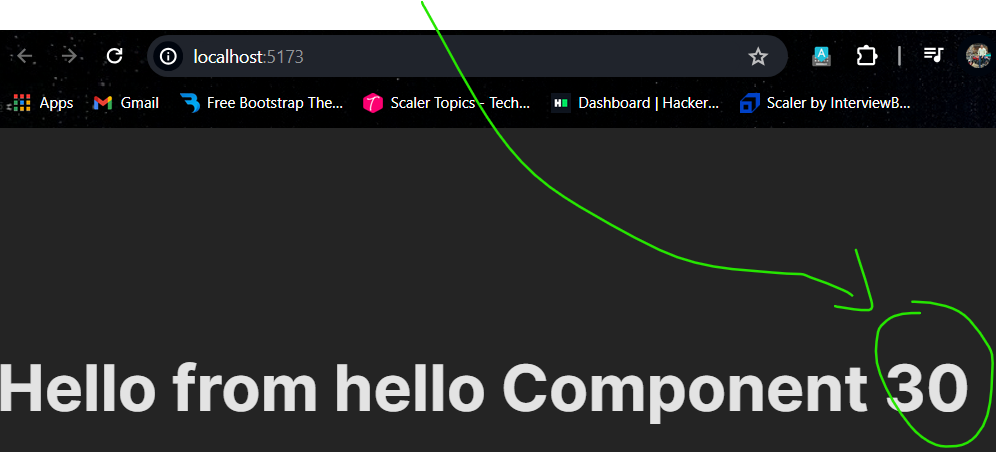


Output :

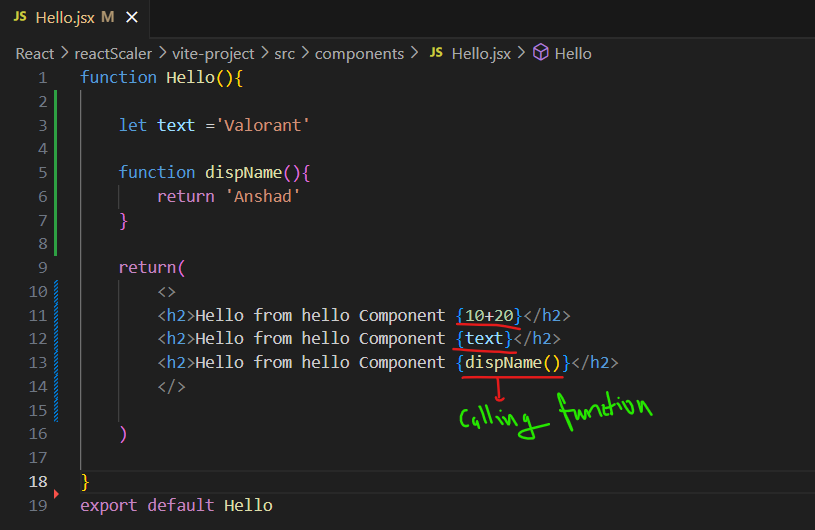


* 1. **Properties of JSX:**
     1. What if we write ‘{10 + 20}’ inside the html content like this :





* + 1. So whatever you write inside curly braces will be evaluated as JS.
    2. Lets try using variables and functions inside this :



**Output** :

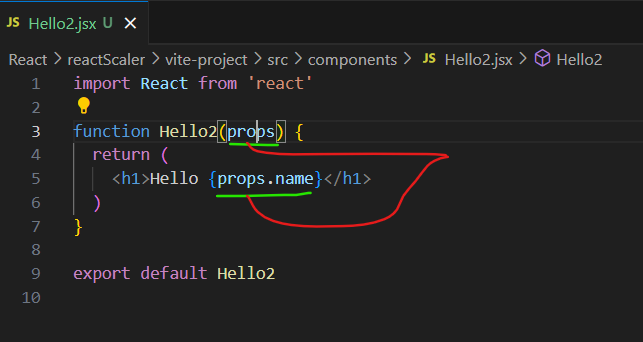


* 1. **Props and Reusing Components :**
     1. Props will help your component Go Dynamic.
     2. Until now we were used static data.So lets try passing input from App.
     3. For that lets create another component ‘Hello2.jsx’ :

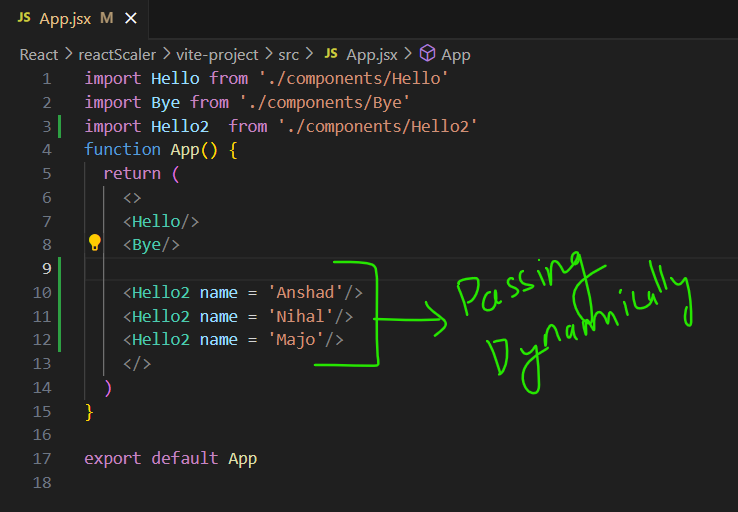
1. First import this newly added component to ‘App.jsx’:

import Hello2  from './components/Hello2'

1. Type ‘props’ inside function brackets.And ‘ {props.name} ‘ to get name property name from App.



1. Inside ‘App.jsx’ pass the name you want as much as you want with different names:



1. Similarly lets add phone property :

**‘Hello2.jsx’:**

import React from 'react'

function Hello2(props) {

  return (

    <h1>Hello {props.name} {props.phone}</h1>

  )

}

export default Hello2

**‘App.jsx’:**

import Hello from './components/Hello'

import Bye from './components/Bye'

import Hello2  from './components/Hello2'

function App() {

  return (

    <>

    <Hello/>

    <Bye/>

    <Hello2 name = 'Anshad' phone='8154564223'/>

    <Hello2 name = 'Nihal' phone='992564223'/>

    <Hello2 name = 'Majo' phone='124853325'/>

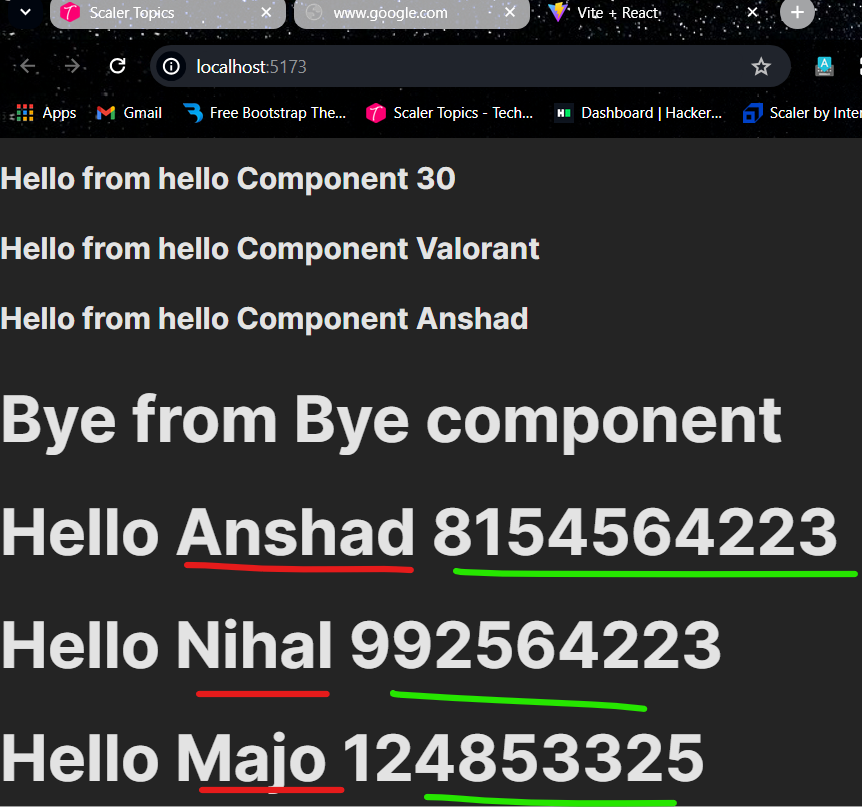
    </>

  )

}

export default App

**OUTPUT:**



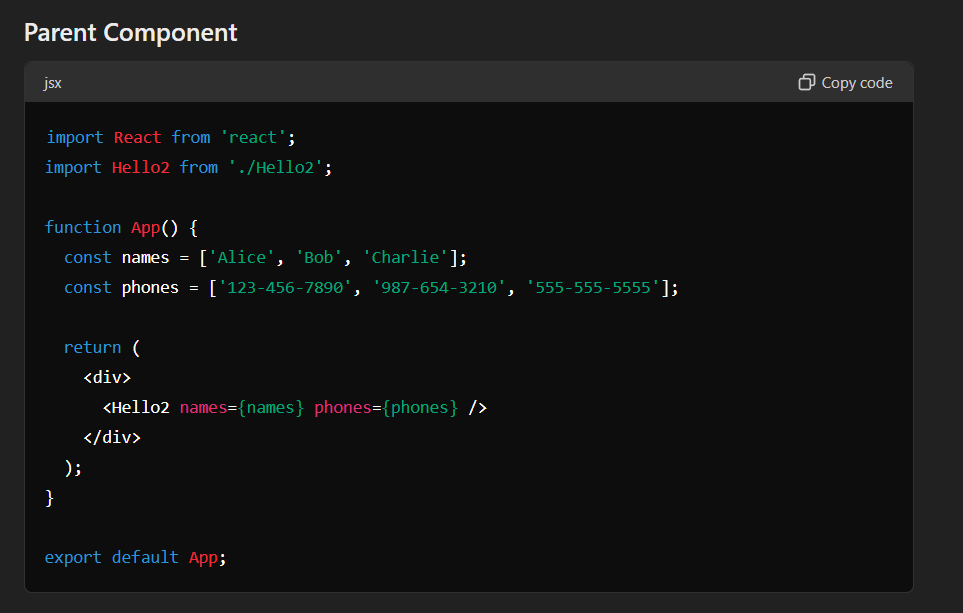
1. So this is called RESUING .That is Structure remains the Same but the values are changing.

* + 1. There is an Efficient/Shorter way to write this props: Destructuring.=>



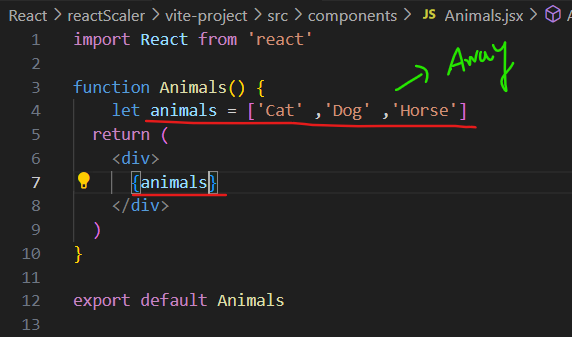
* 1. **Arrays and Objects as Props :**
     1. What if you want to pass an entire Array as a prop?

‘App.jsx’ :

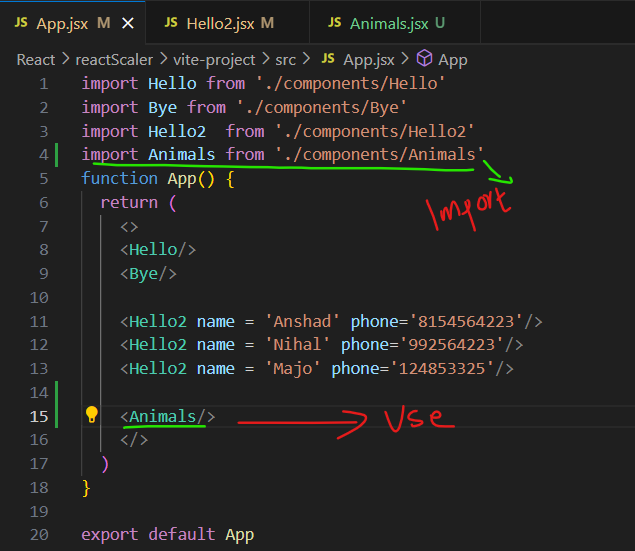


**‘Hello2.jsx’ :**

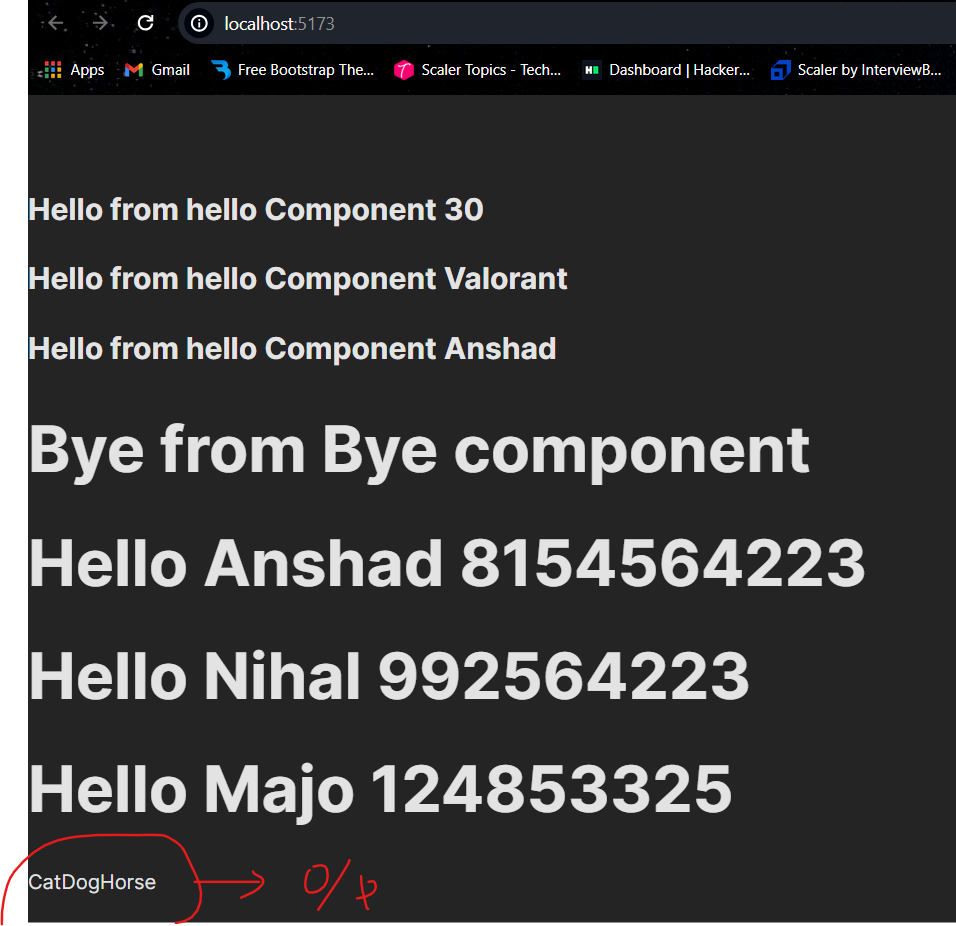
* + 1. e t
    2. asd
  1. **Rendering Array Elements :**
     1. Create a New componet ‘Animals.jsx’ :



* + 1. Then import and use this component inside ‘App.js’ :



**OUTPUT :**



* + 1. So If you try to directly access the Array like this we get output in **combined manner ,which is difficult to differentiate**.
    2. To avoid this we age going to use a **Higher Order Method.**



**OUTPUT:**



* + 1. If you want to make as List :

import React from 'react'

function Animals() {

    let animals = ['Cat' ,'Dog' ,'Horse']

  return (

    <div>

        <ul>

      {animals.map((animal) => (

        <li>{animal}</li>

      ))}

      </ul>

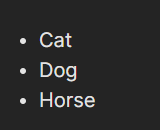
    </div>

  )

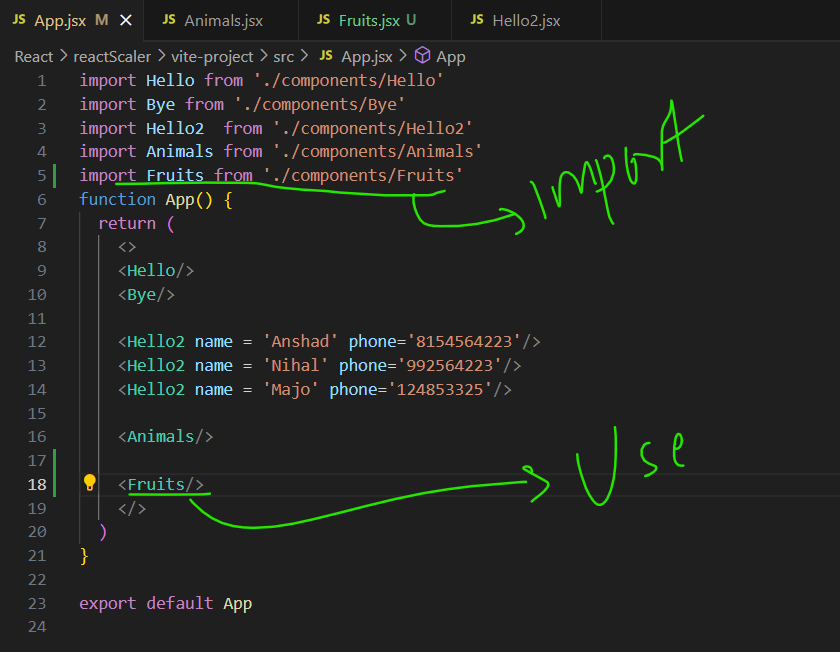
}

export default Animals

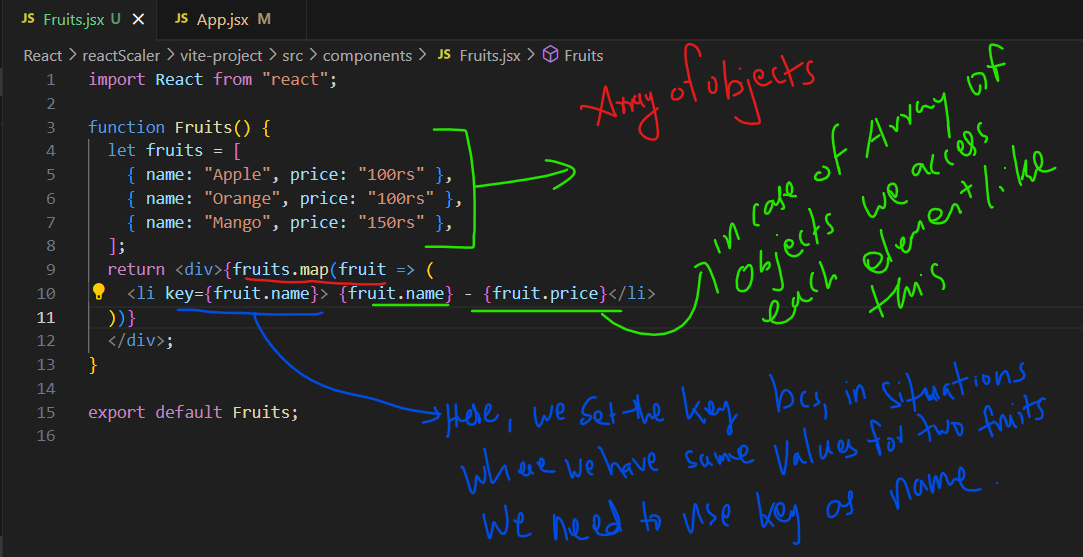
OUTPUT :



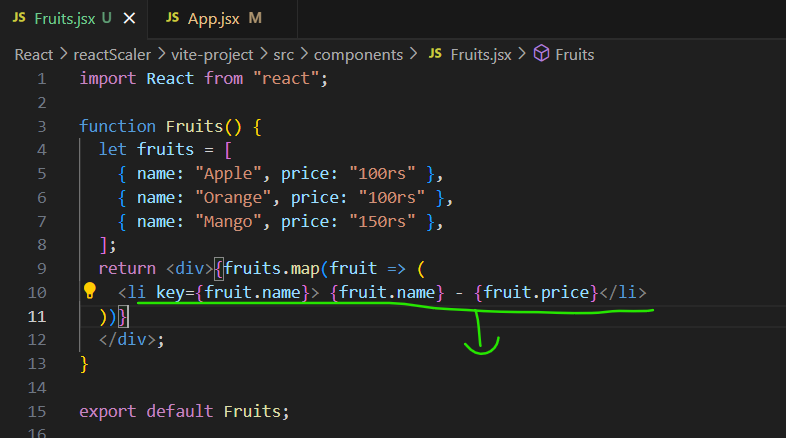
* 1. **Rendering Array of Objects :** 
     1. Create another component ‘Fruits.jsx’ .
     2. Then import and use this component inside ‘App.js’:



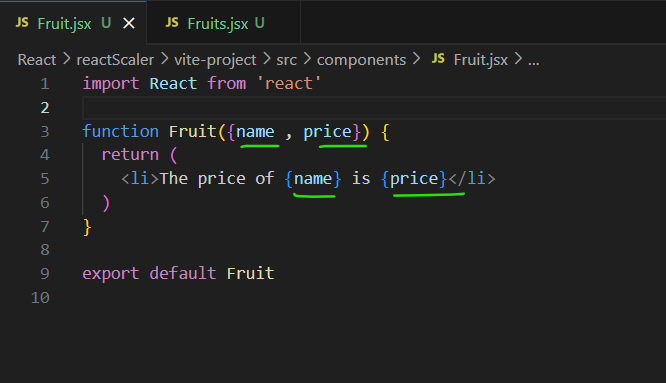
* + 1. ‘Fruits.jsx’:



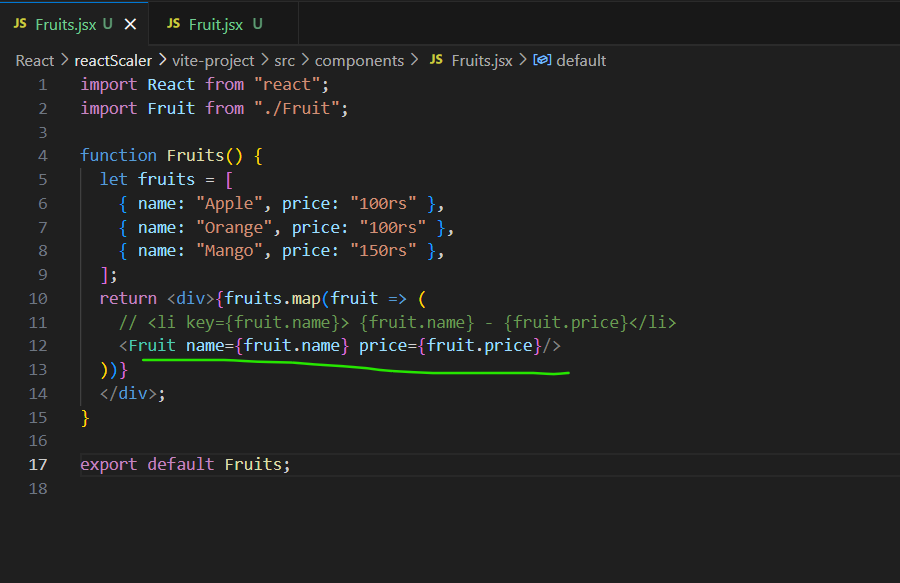
* 1. **Conditional Rendering :**
     1. In above code,We can handle this in much better way:



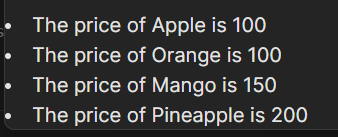
* + 1. React Suggests to take this line of code and place it in separate component file :’Fruit.jsx’ =>



’Fruits.jsx’ =>

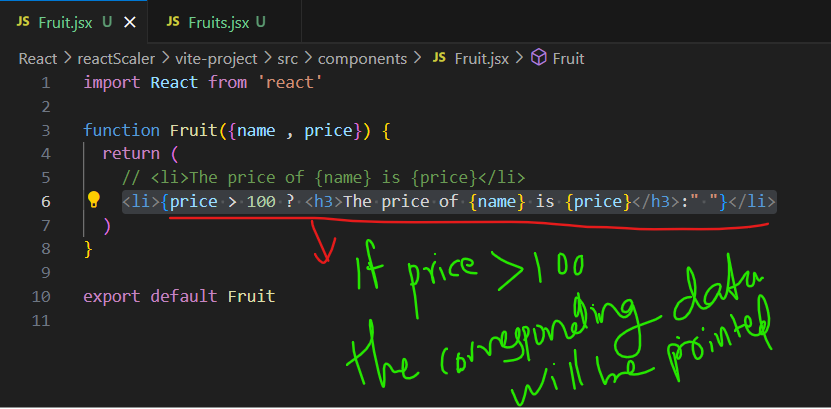
\

**OUTPUT :**

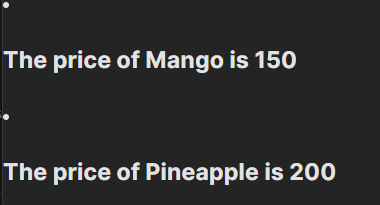


* + 1. Lets try The Technique of Ternary Operator .
    2. So lets add one more fruit and change price in number format for comparing:

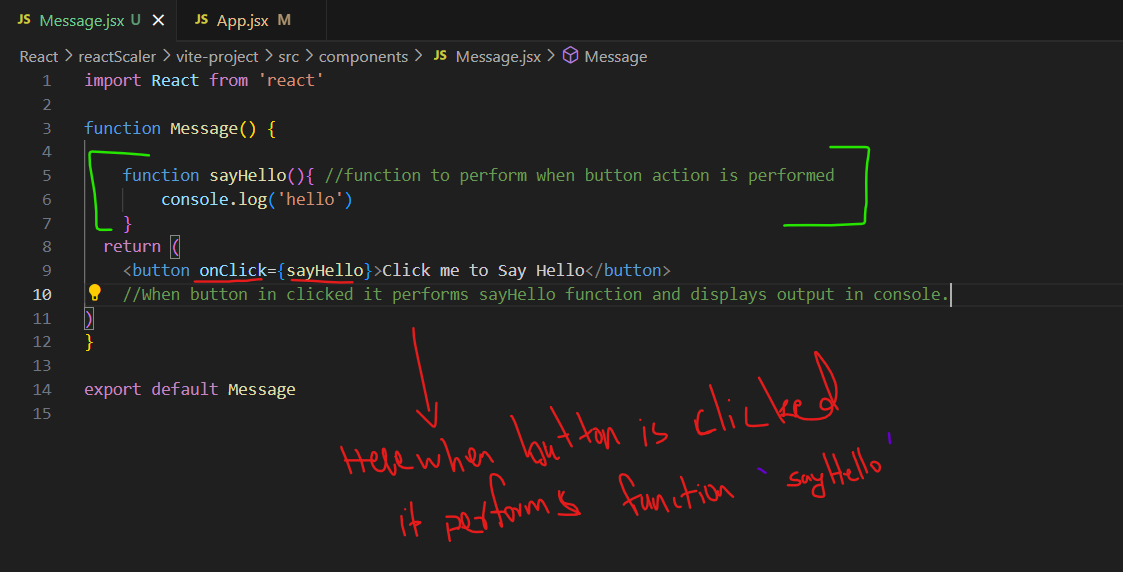


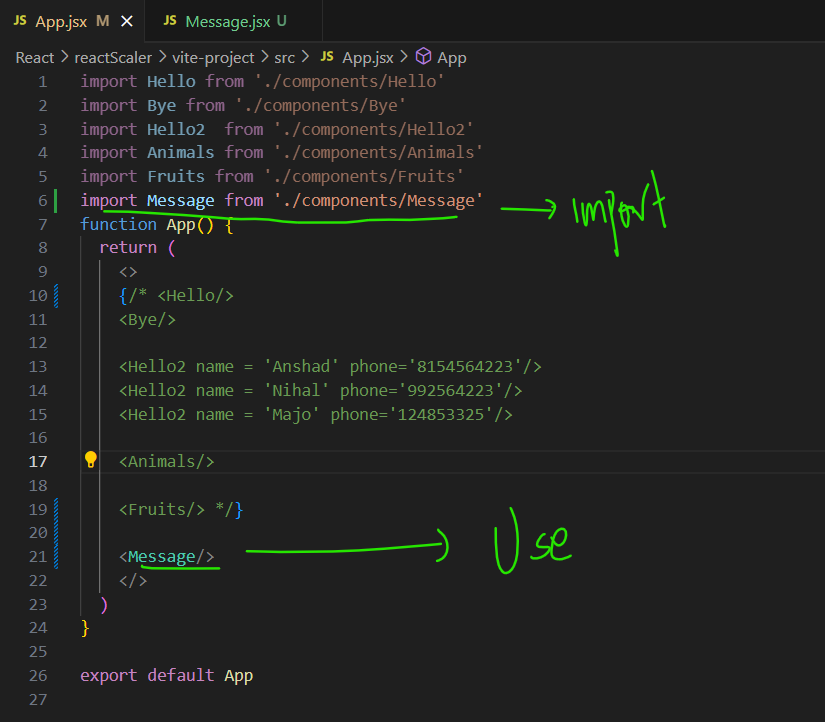


**OUTPUT:**

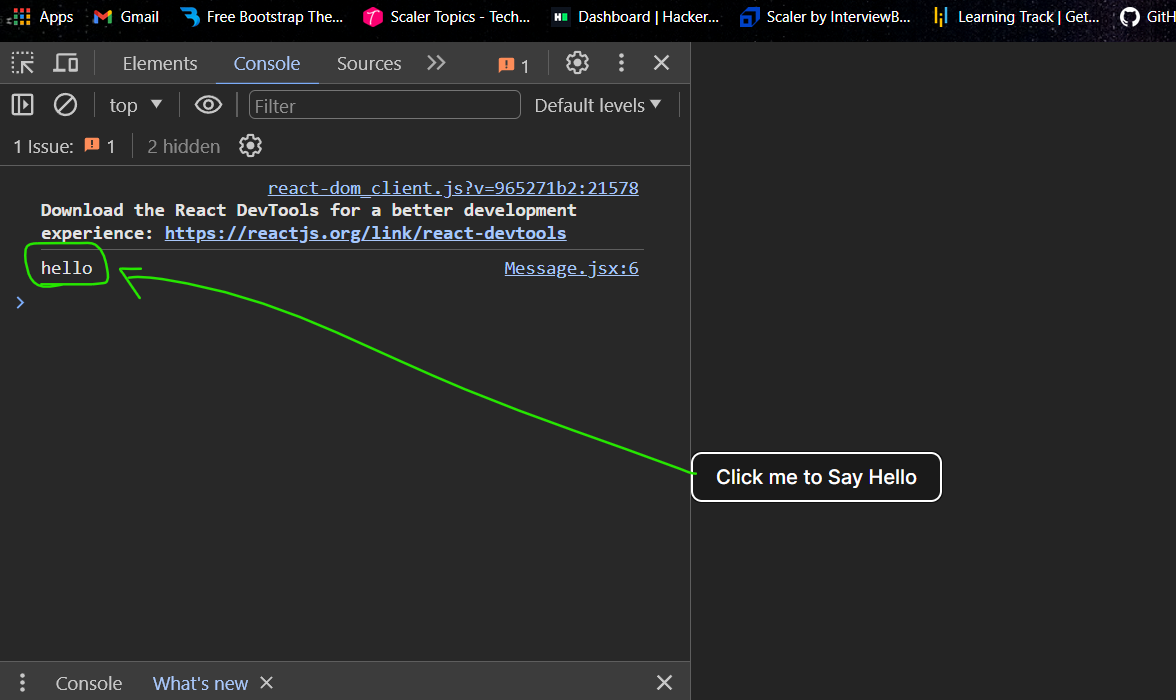


1. ***State Events and Forms :***
   1. **Handle an Event in React :**
      1. First Create a component ‘Message.jsx’ and import that in ‘App.jsx’:

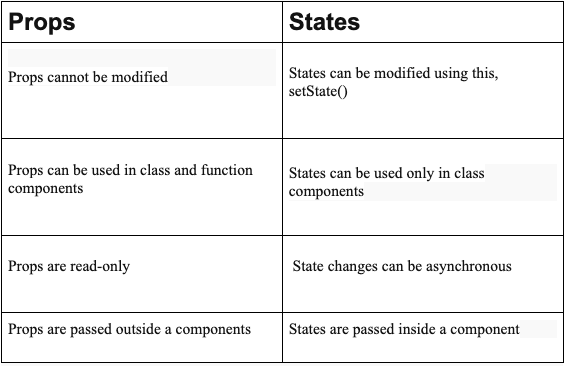


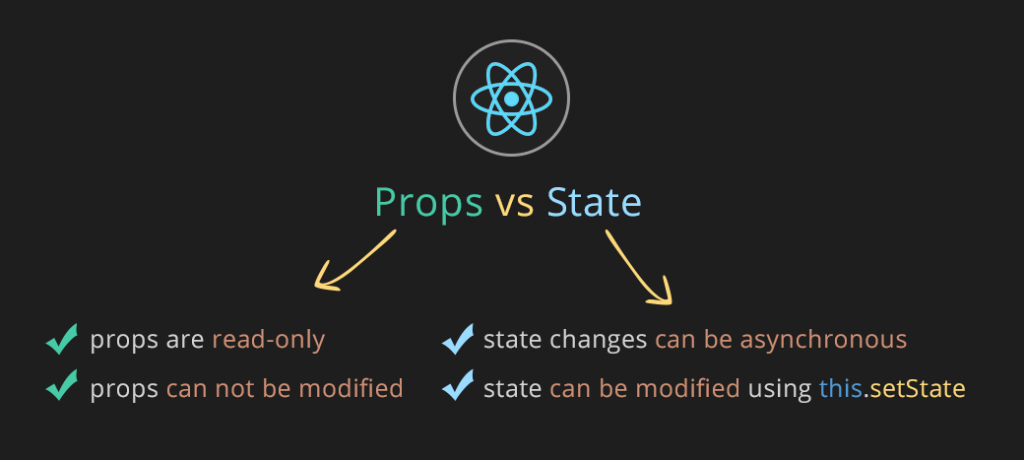


**OUTPUT** :



* 1. **What is a State in React :**
     1. You cannot Modify props,So we use ‘States’:





* + 1. Lets create a Counter app.
    2. So First create a component ‘Counter.jsx’ and link that in ‘App.jsx’ :

‘Counter.jsx’

import React from 'react'

const Counter = () => {

  return (

    <div>

      <h1>This is my Counter : 0

        <button>Increment</button>

        <button>Decrement</button>

      </h1>

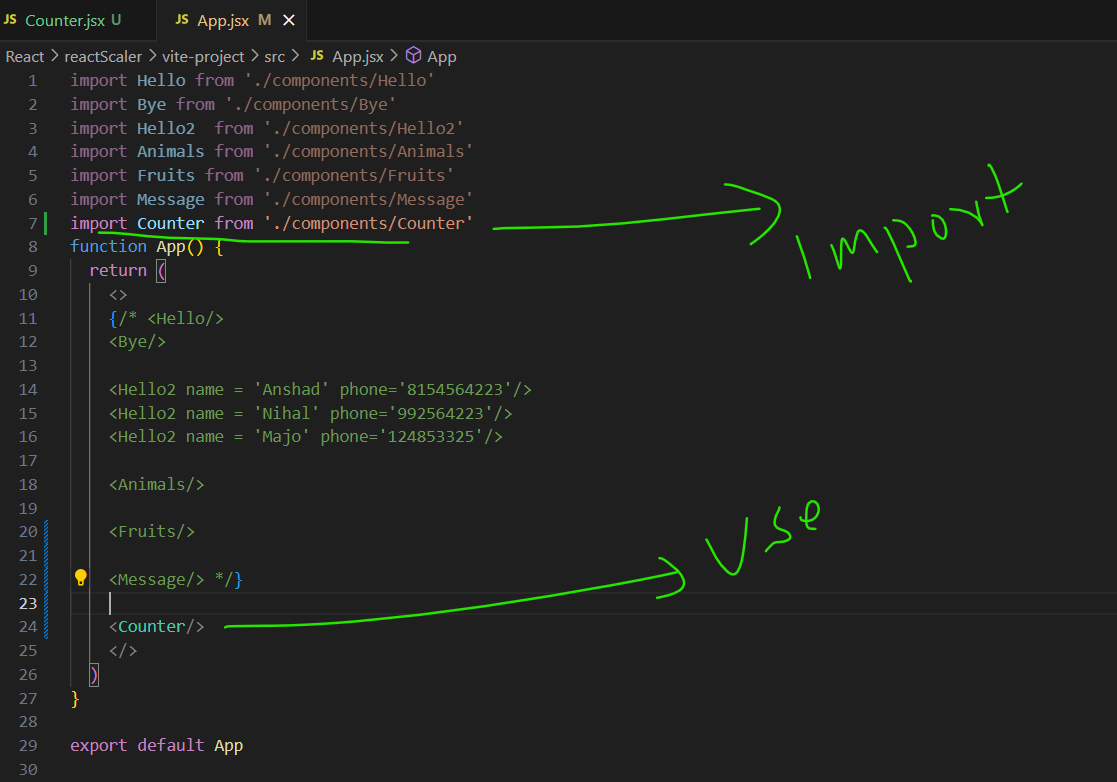
    </div>

  )

}

export default Counter

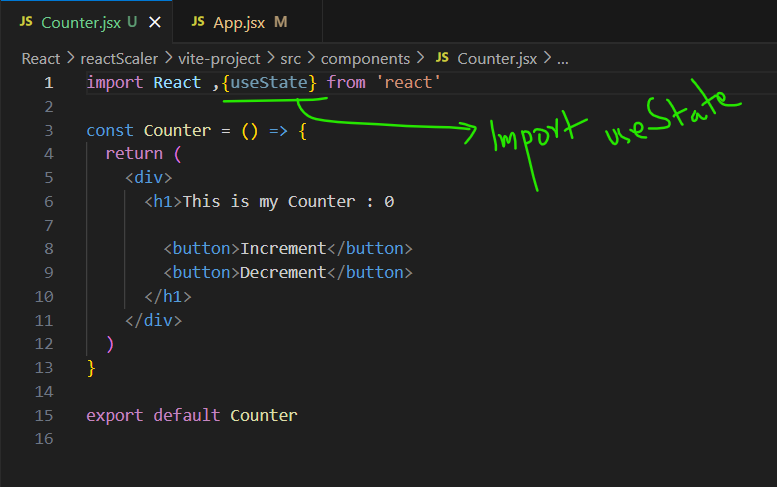
‘App.jsx’



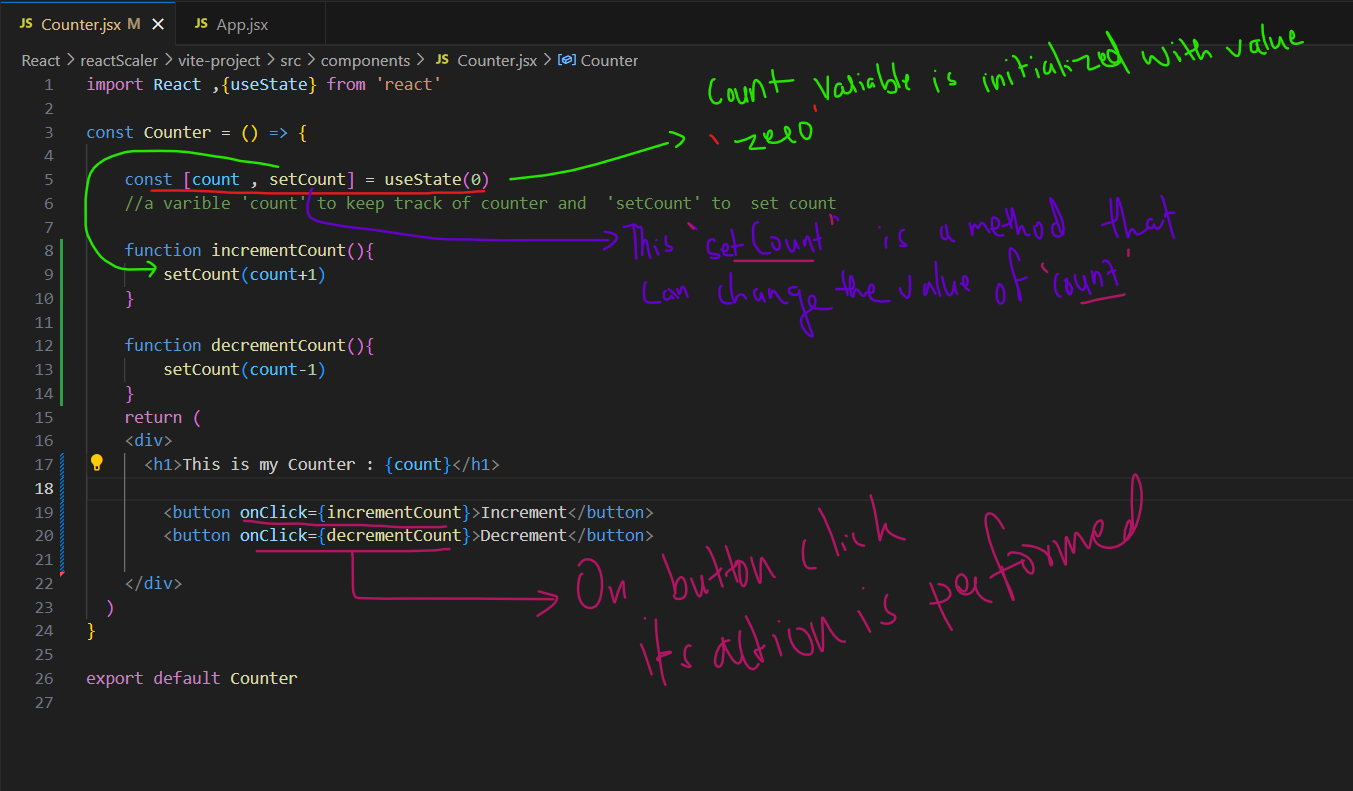
**OUTPUT:**



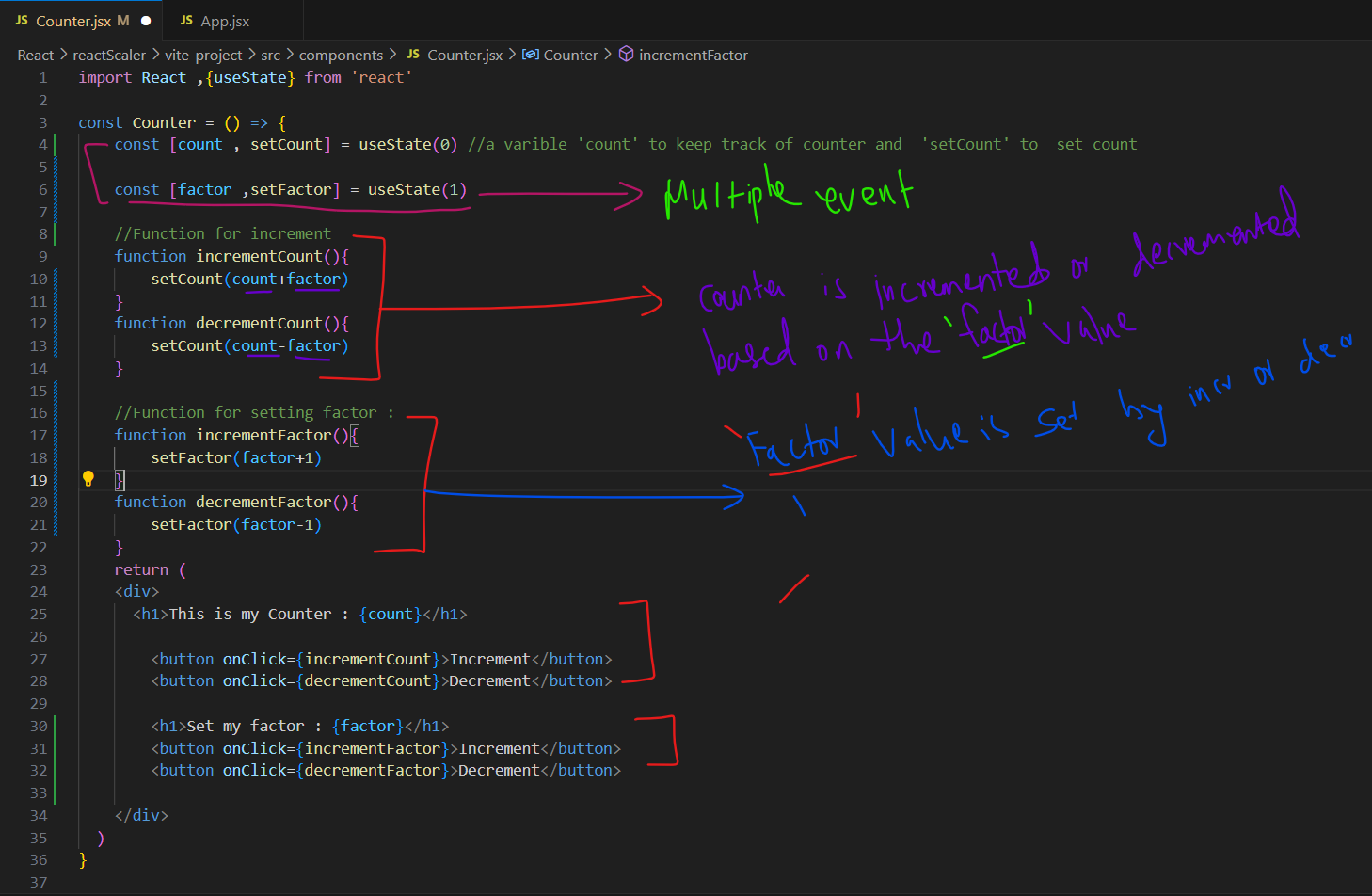
* + 1. Now we have two buttons to incerement and decrement values.But these wont perform any actions.
    2. So in order perform button actions we need to use ‘state’ in this situation.
    3. To use , First we need to import this ‘{useState}’ :



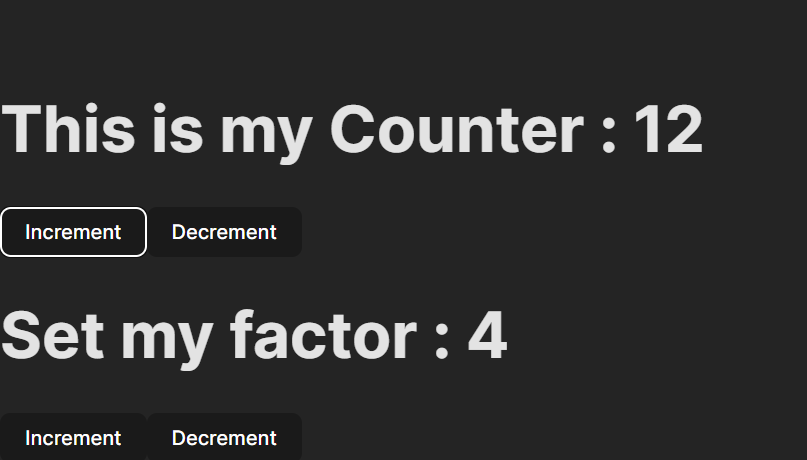
* + 1. Then set an array for storing counter values .



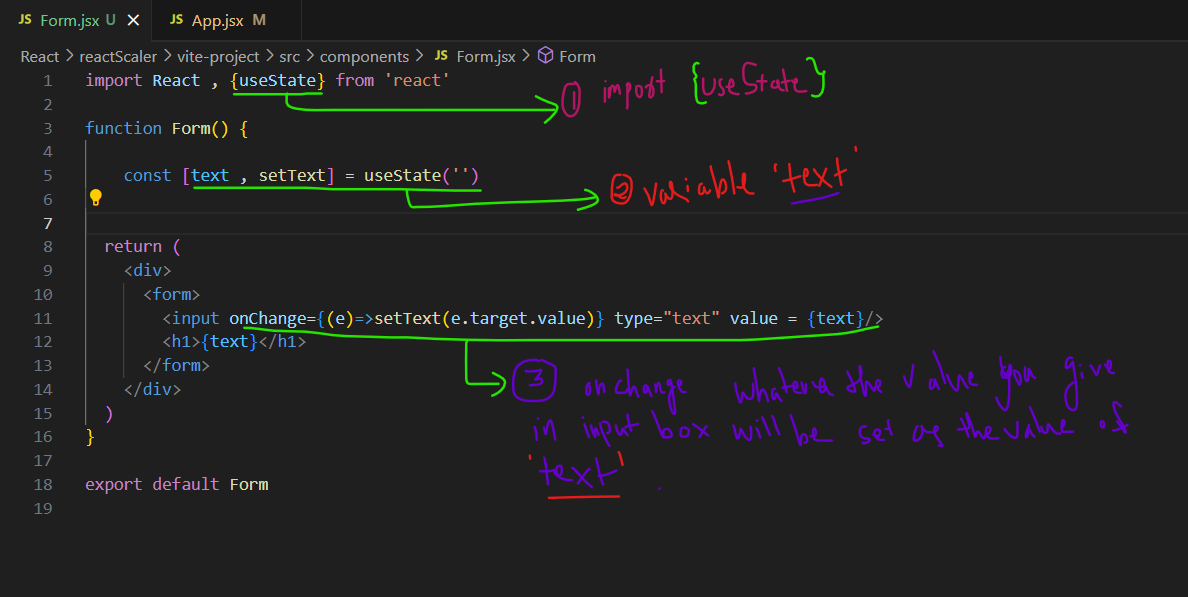
* + 1. Every time its ‘Re-rendering’.
  1. **Handle Multiple events in React :**
     1. Lets now add an even to increment and decrement the Factor so that based on that factor ,the counter is incremented or decremented:



**OUTPUT :**



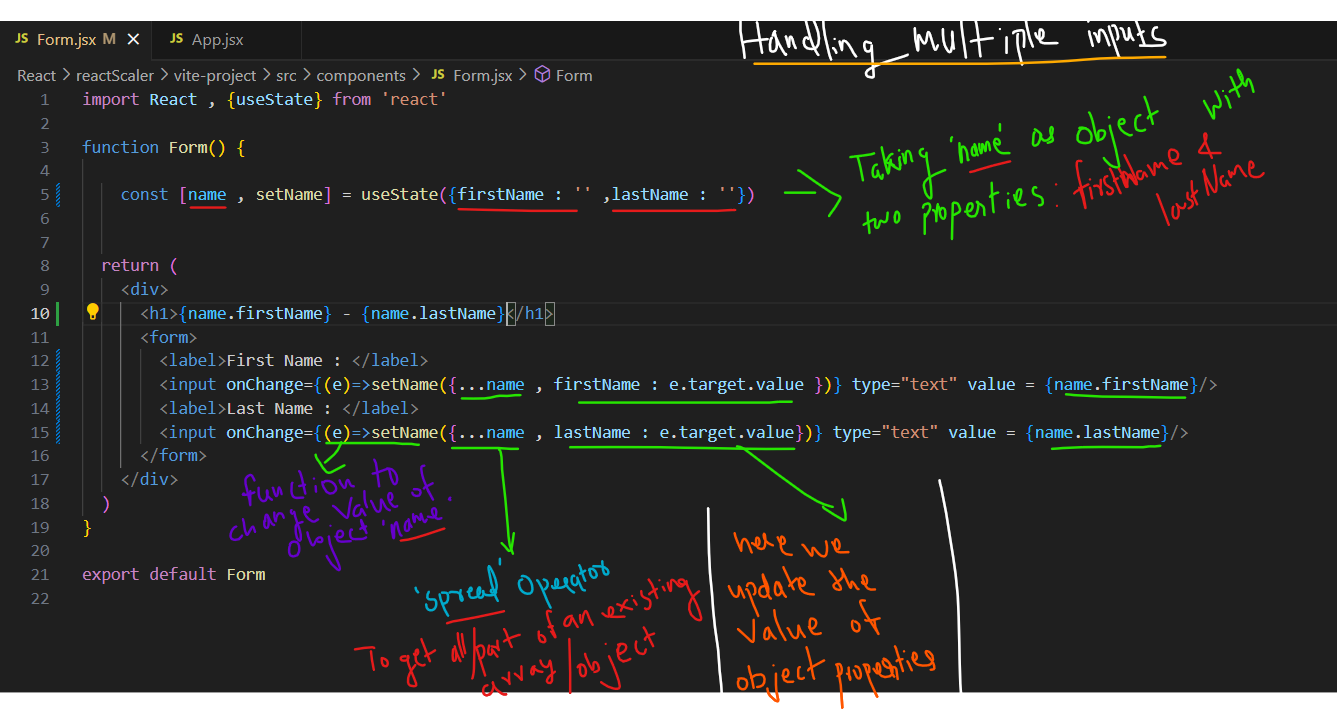
* 1. **Handle Input Element in React :**
     1. Input element with help of ‘state’:



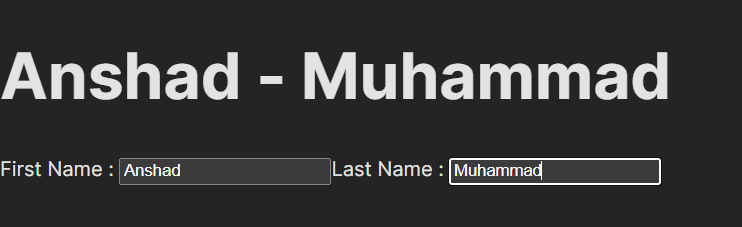
**OUTPUT:**



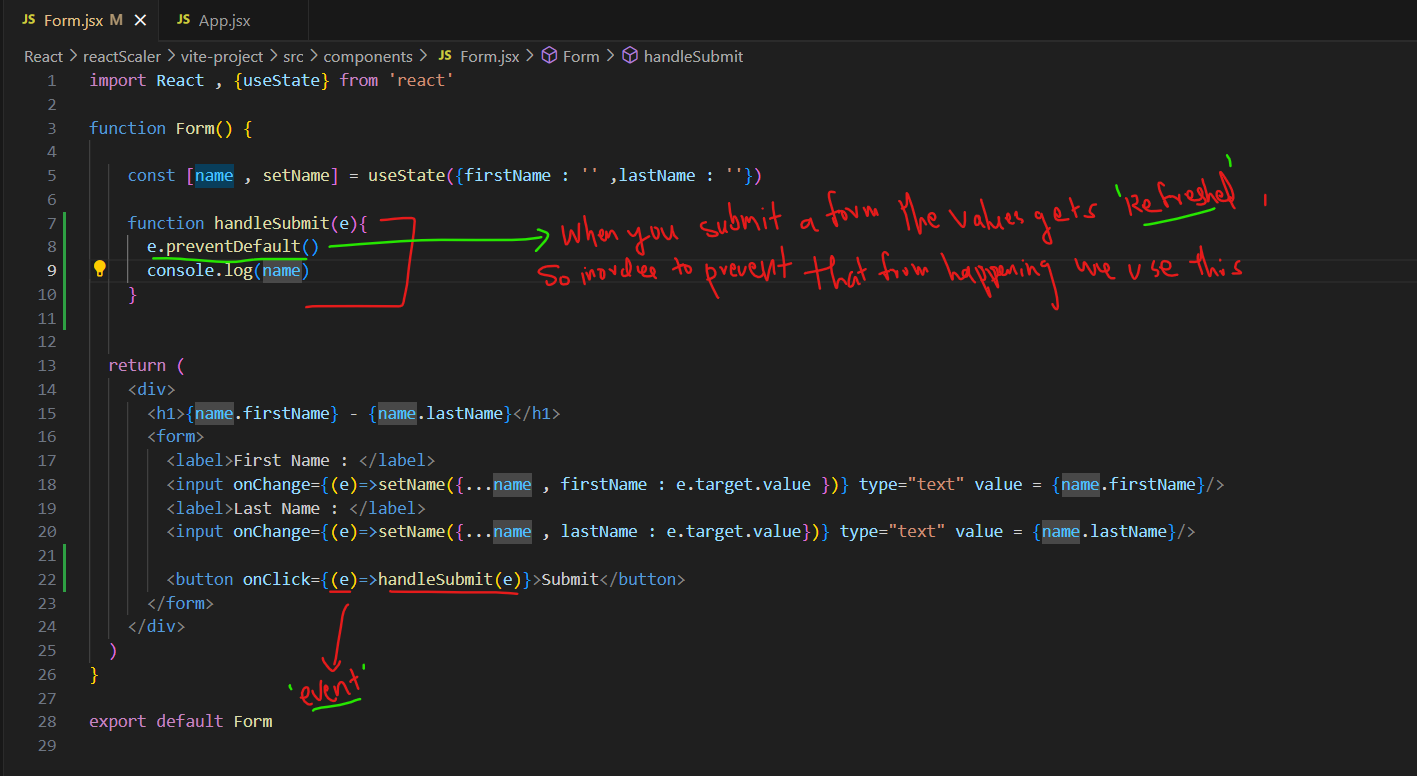
* 1. **Handle Multiple Inputs in React :**
     1. We can The JavaScript ‘spread’ operator (...) allows us to quickly copy all or part of an existing array or object into another array or object :



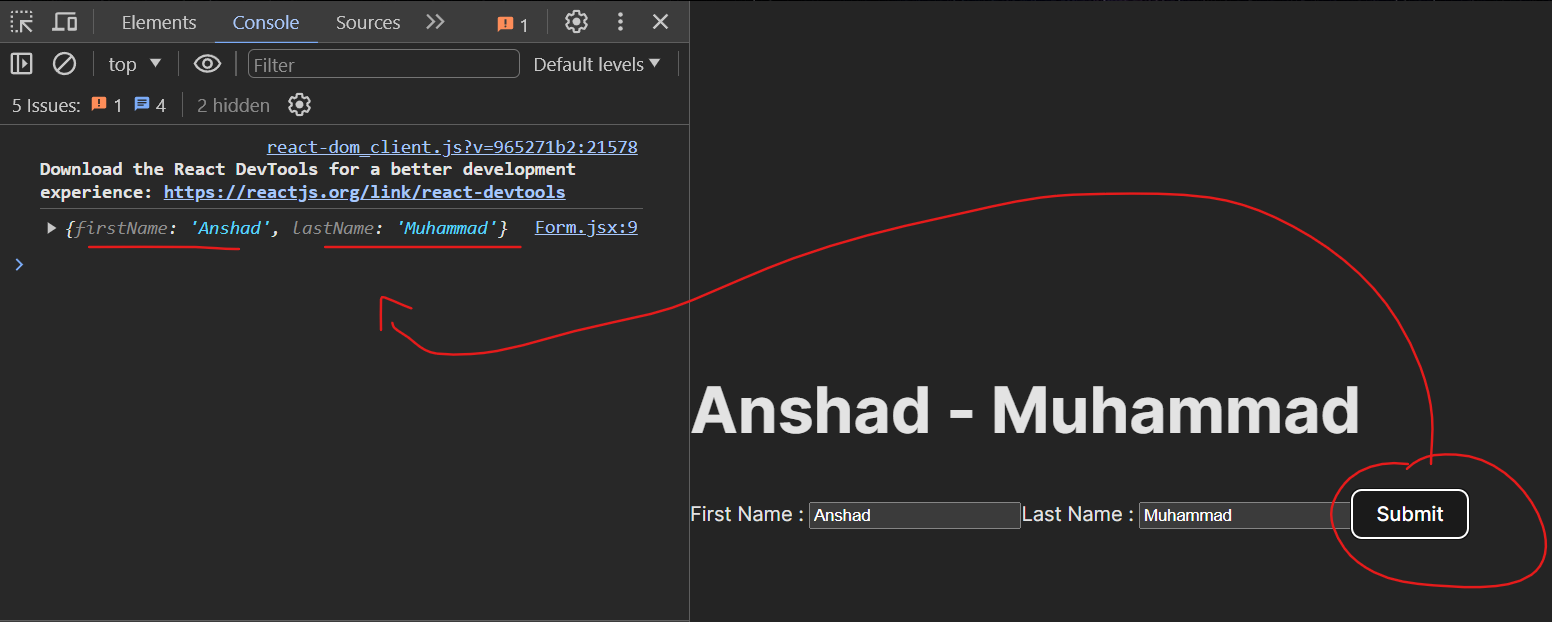
**OUTPUT:**



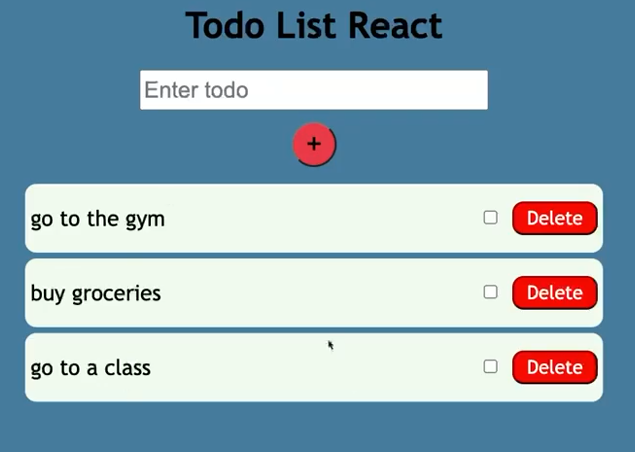
* 1. **Submit a Form in React :**
     1. Use ‘submit’ button.



OUTPUT :

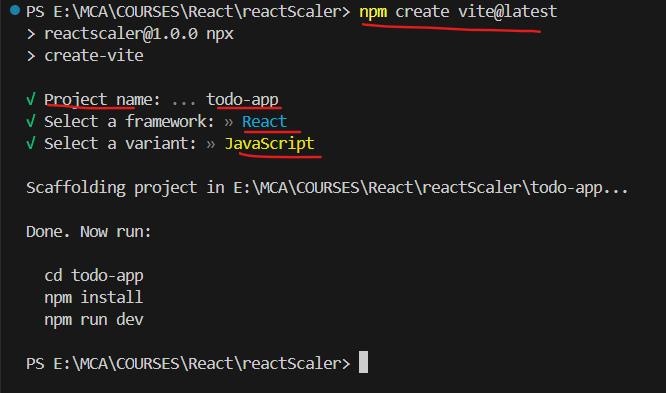


1. ***Buid a TODO List with React :*** 
   1. **TODO List Demo :**
      1. Below is the DEMO of TODO list app we are gonna craete.



* 1. **Design Your TODO List :**
     1. First Use Vite inside your folder ?

1. First run command “**npm create vite@latest**”.
2. Give project name ->Eg:todo-app
3. Select Framework ->React.
4. Select variant-> JavaScript.
5. Now open that ‘todo-app’ folder in terminal and install npm : “npm install”.So that ‘todo-app’ will get all the dependencies.
6. Now RUN the project.‘dev’ command is used to run your React project: “npm run dev”



* + 1. Then clean the codes inside ‘index.css’, ‘App.css’ and ‘App.jsx’.
    2. Start with <main> tag.All my codes will be inside this:

‘App.jsx’:

import './App.css'

function App() {

  const [count, setCount] = useState(0)

  return (

    <main>

      <h1>TODO List</h1>

      <div className='input-container'>

        <input type="text"/>

        <button>+</button>

      </div>

    </main>

  )

}

export default App

‘App.css’:

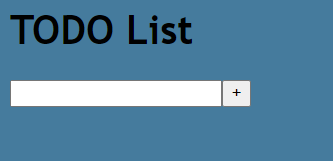
body{

  font-family:'Trebuchet MS', 'Lucida Sans Unicode', 'Lucida Grande', 'Lucida Sans', Arial, sans-serif;

  background-color: #457B9D;

}

OUTPUT:



* + 1. Add css to align <h1> to center:

h1{

  text-align: center;

}

* + 1. Add style to div with id =’input-container’ in ‘App.css’:

.input-container{

  display: flex;

  flex-direction: column;

  align-items : center;

}

* + 1. Add style to the input we are gonna give :

.input-container > input{

  width: 300px;

  height: 30px;

  font-size: 1.25rem;

}

* + 1. Samelike do style for button:

.input-container > button{

  width: 40px;

  height: 40px;

  border-radius: 20px;

  font-size: 1.5rem ;

  background-color: rgb(235, 54, 54);

  margin-top: 10px;

}

* + 1. Add another <div> to display the list items(for now static):

‘App.jsx’:

      <div className='container'>

        <div className='todo'>

          <p>go to the class</p>

        </div>

      </div>

* + 1. Then , Add style for this div with className :‘container’=>

.container{

  margin-top: 15px;

  display: flex;

  flex-direction: column;

  align-items: center;

  gap: 5px;

}

* + 1. Now, Add style for div with className:‘todo’ =>

.todo{

  width : 500px;

  font-size: 1.2rem;

  display: flex;

  justify-content: space-between;

  align-items: center;

  border-radius : 10px;

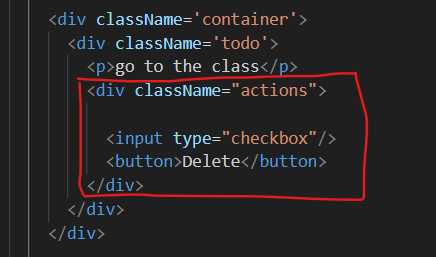
  background-color: #F1FAEE;

  padding: 0px 5px;

}

* + 1. Now Add another <div> with className:’actions’ inside the <div> with className:’todo’ :

‘App.jsx’:



* + 1. Now Add style for this button :

.actions > button {

  background-color: #E63946;

  color : white;

  height : 30px;

  width : 75px;

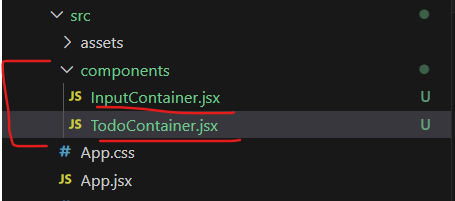
  border-radius: 10px;

  font-size: 1.05rem;

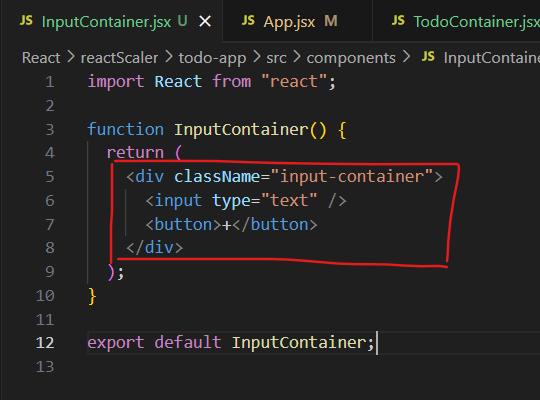
  margin-left: 10px;

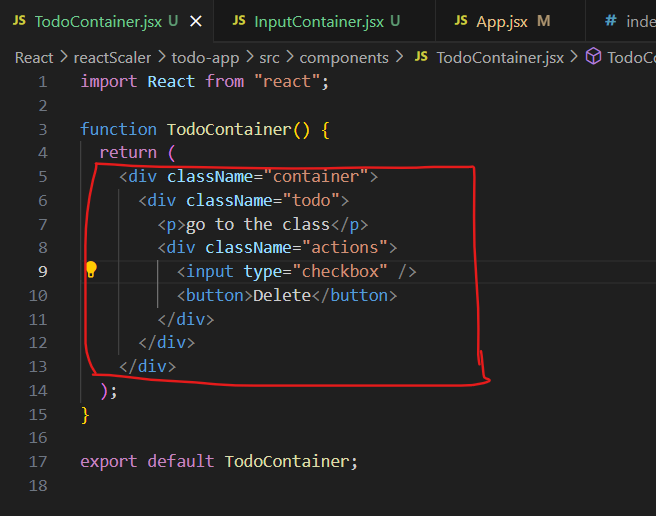
}

* 1. **Add Features to the App :**
     1. Lets Create two separate Components , One for the **input textfield** and another for **Todo lists**.
     2. For that first create a folder inside ‘src’ named ‘components’ and inside that create two components ‘InputContainer.jsx’ and ‘TodoContainer.jsx’:

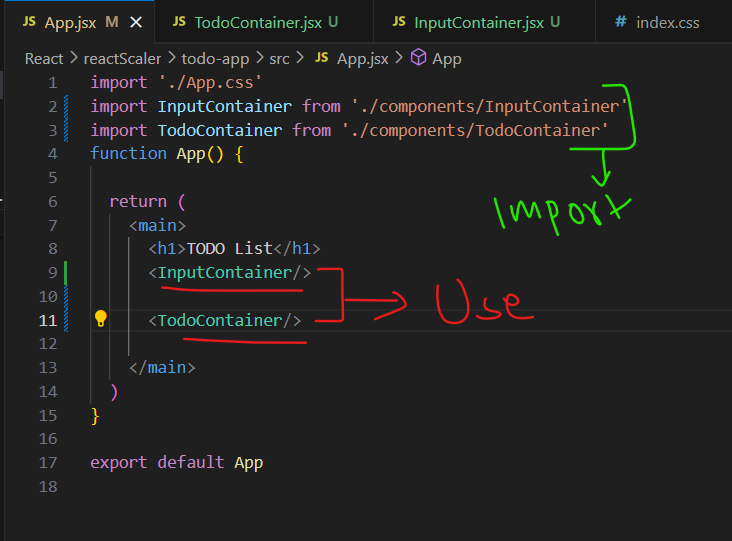


* + 1. Now cut and paste **input textfield** code inside ‘InputContainer.jsx’ and **Todo lists** code inside ‘TodoContainer.jsx’ :

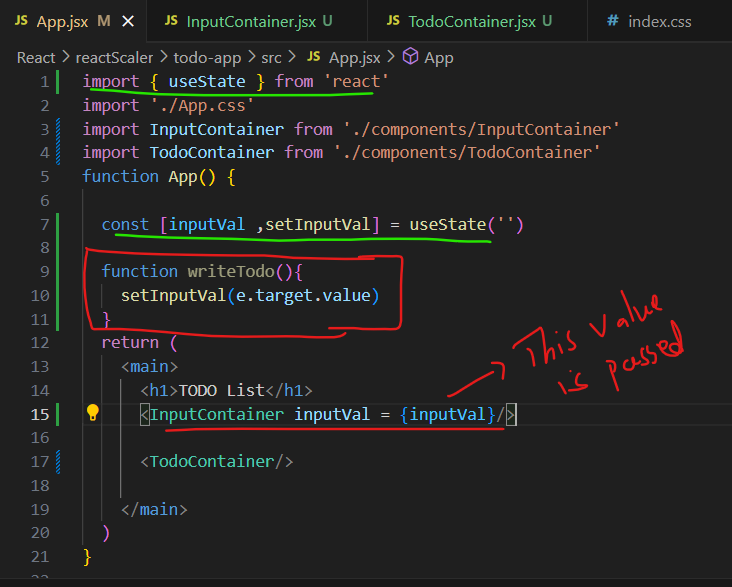




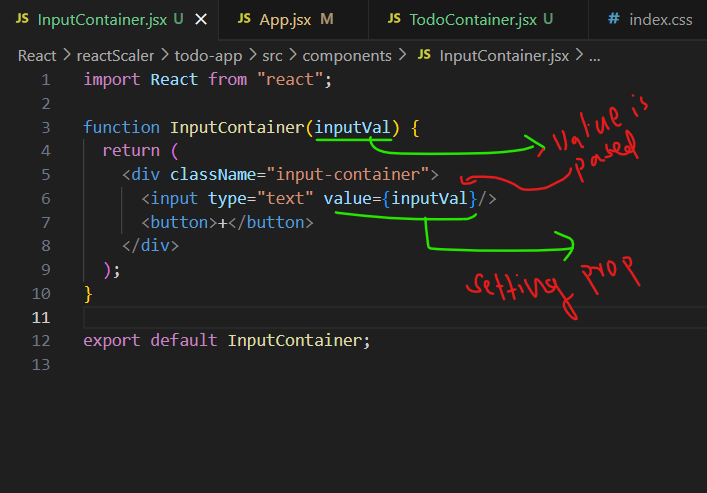
* + 1. Then import these two components inside ‘App.jsx’ :



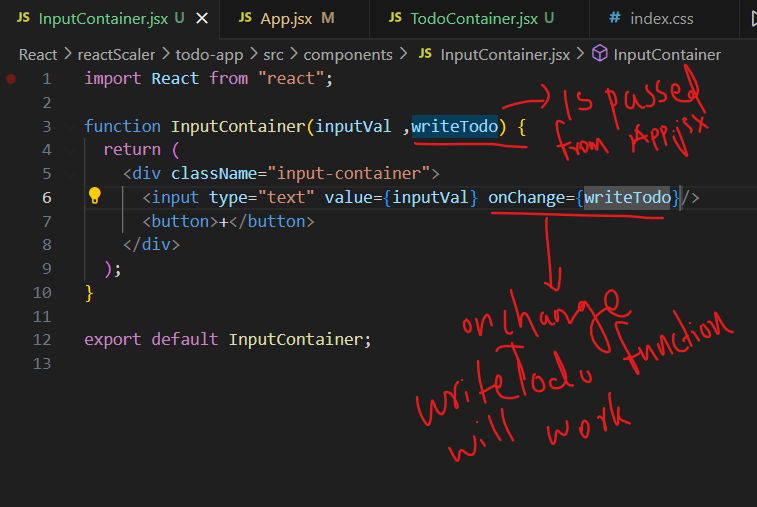
* + 1. So,Lets first start with handling input .
    2. For that lets maintain a ‘state’ and pass the value of textfield.



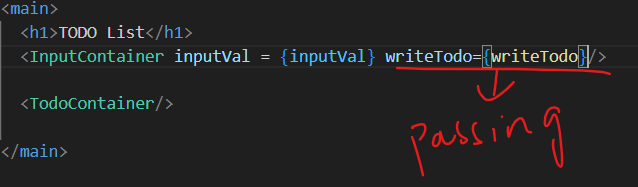
* + 1. Getting passed value ‘inputVal’ in ‘InputContainer.jsx’ :

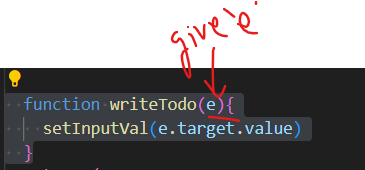


* + 1. Now make writeTodo function work in <input> tag as value to ‘onChange={writeTodo} ‘ :

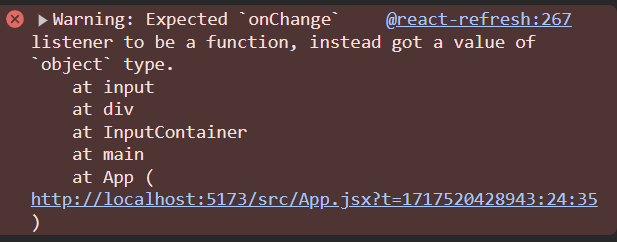


**‘App.jsx’:**

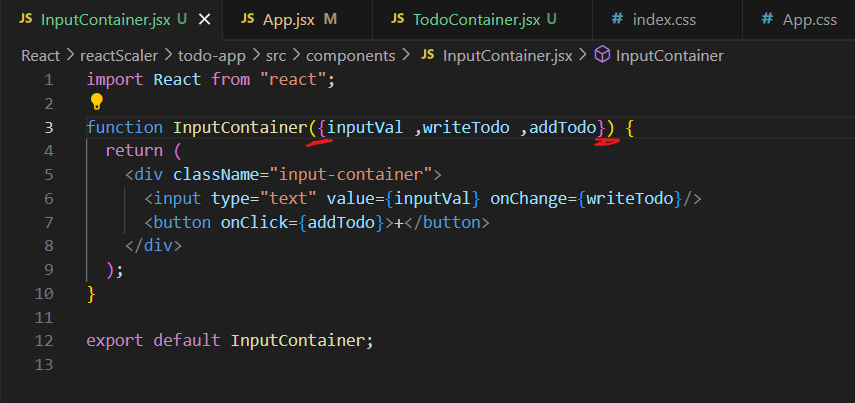


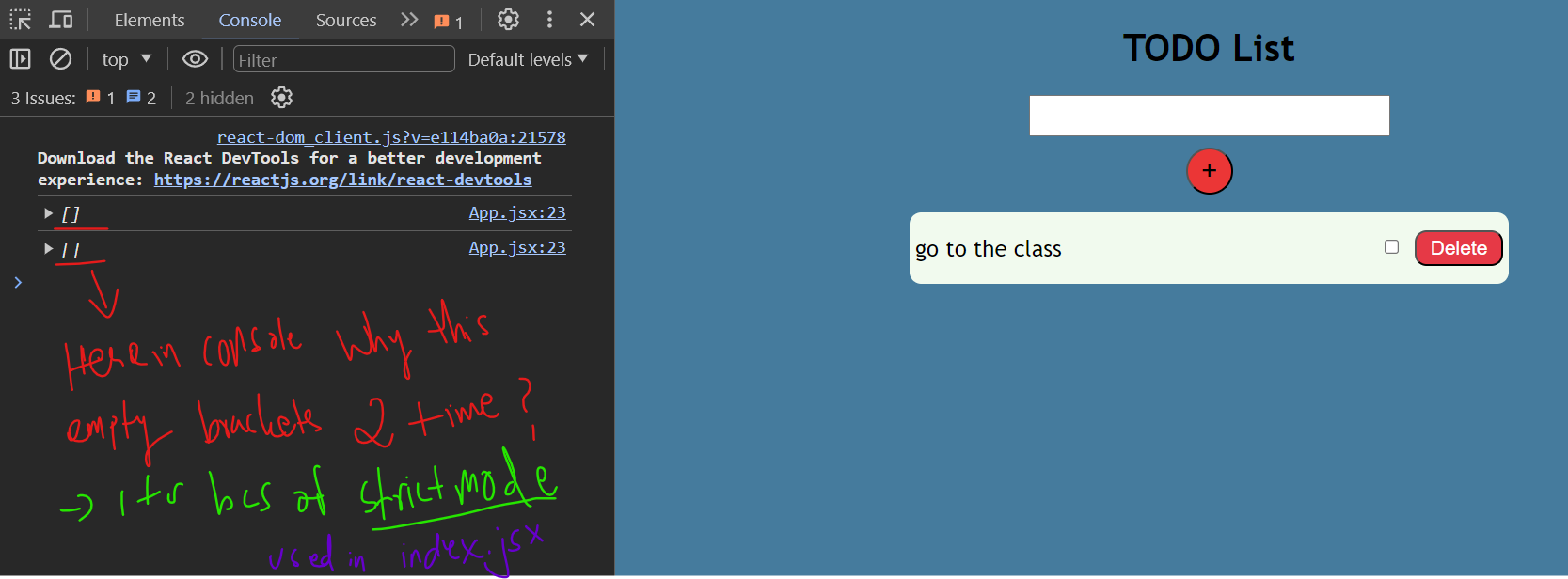


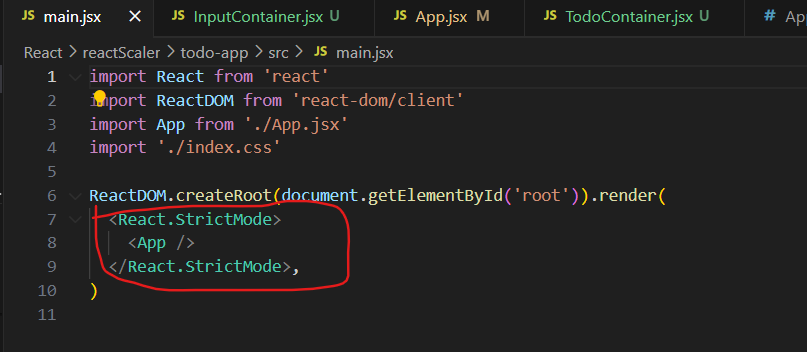
* + 1. **I got a Warning while running :**



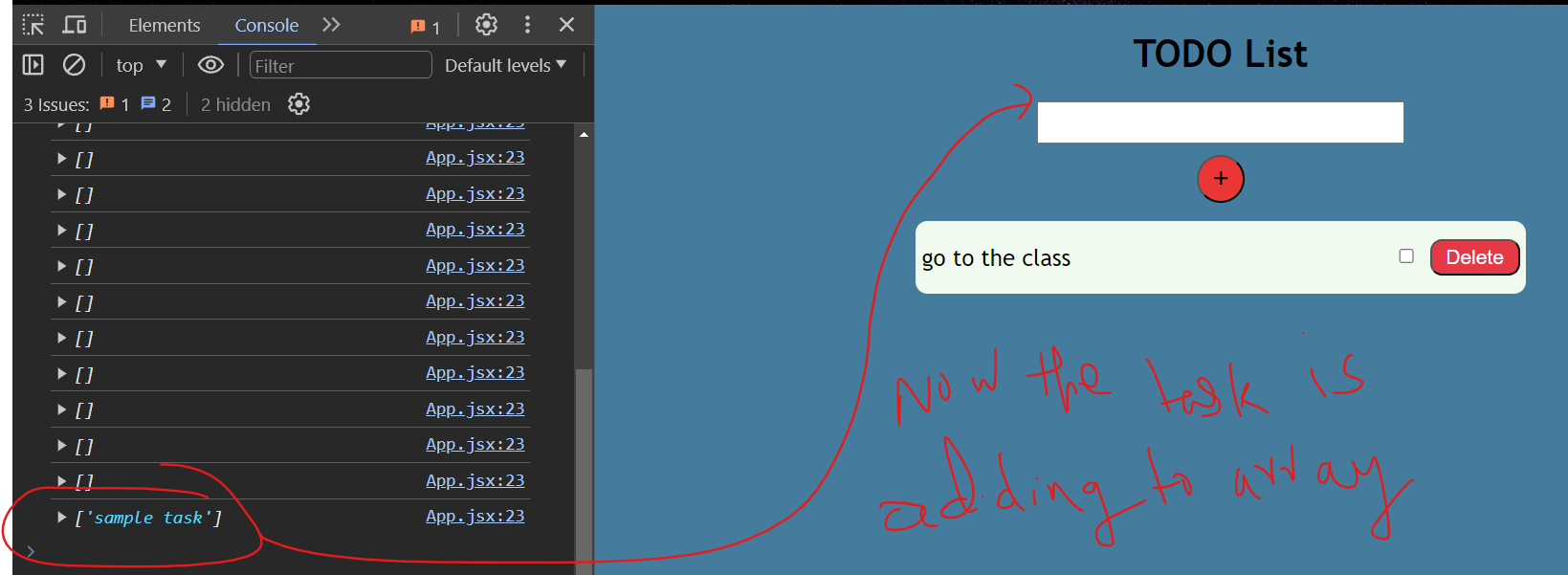
**This happened bcs I forgot to put ‘{}’ while passing :**



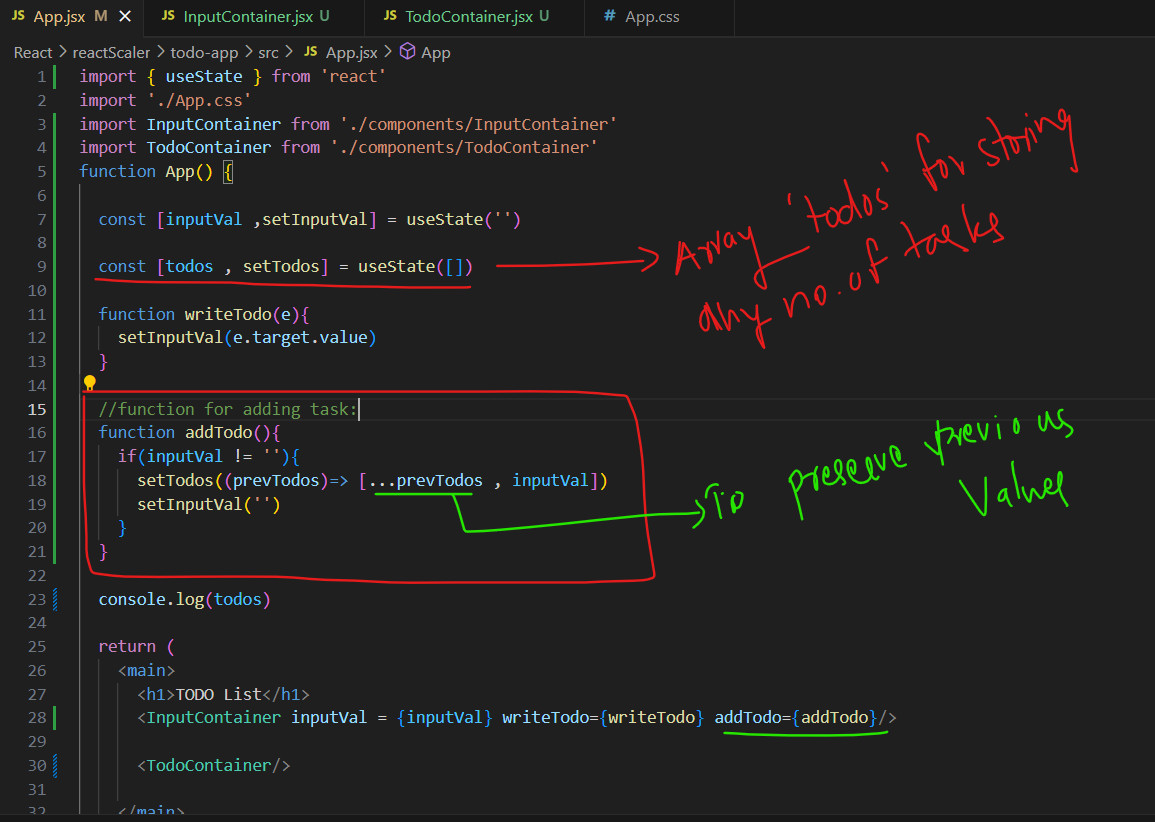




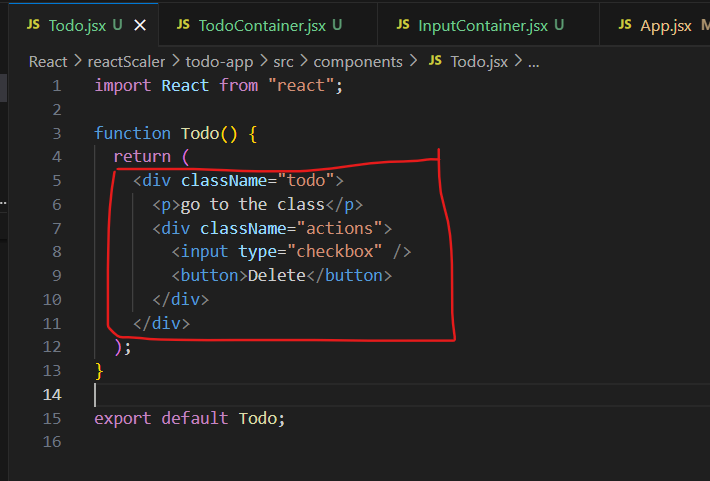
**FOR now lets remove that strict mode.**



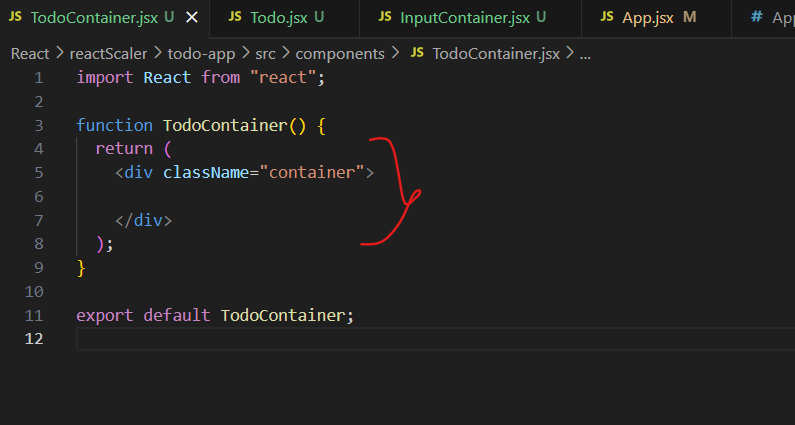
* + 1. Now create another function for add task .



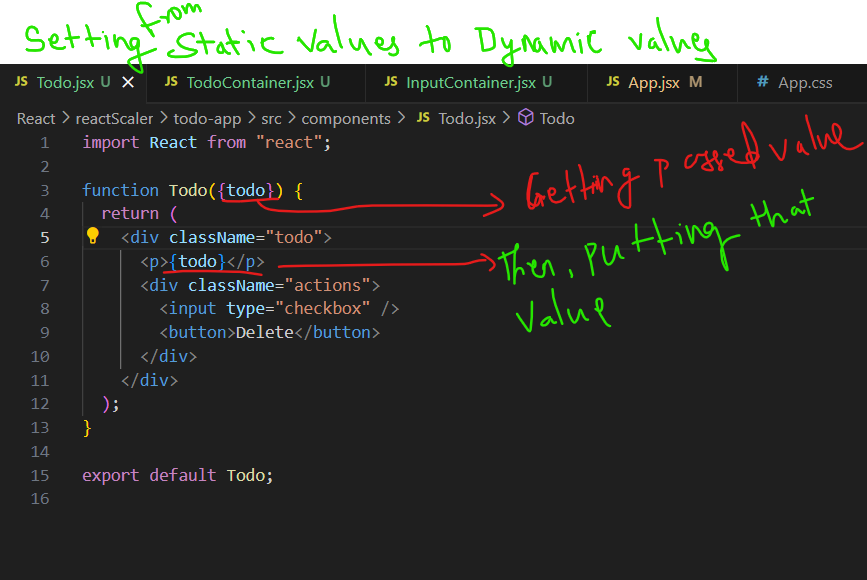
* + 1. Now the added tasks are storing in array.
    2. The next step is to display these added tasks inside this component.
    3. Create a separate component ‘Todo.jsx’ .
    4. And cut and paste contents inside <div className="container"> from ‘TodoContainer.jsx’ to ‘Todo.jsx’ :



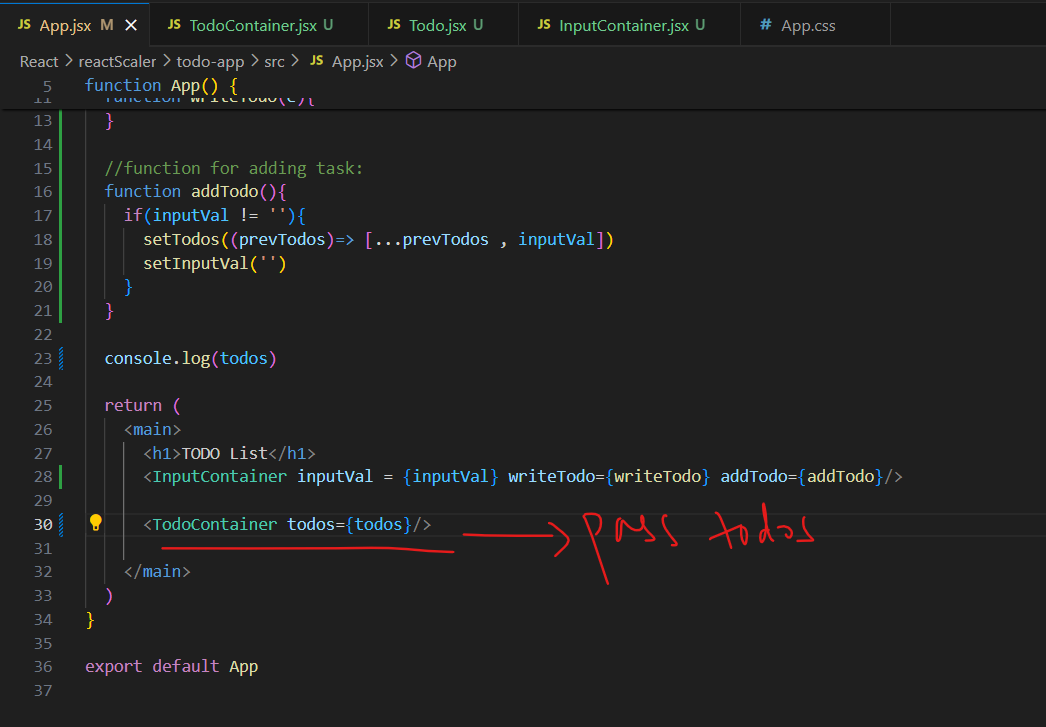
‘TodoContainer.jsx’:



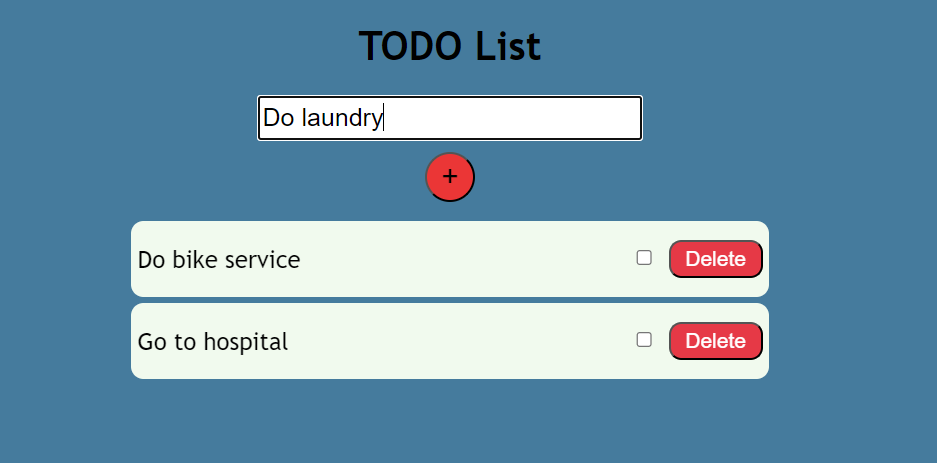
* + 1. Now inside this div = ‘container’ we can pass this component ‘Todo.list’ one by one :



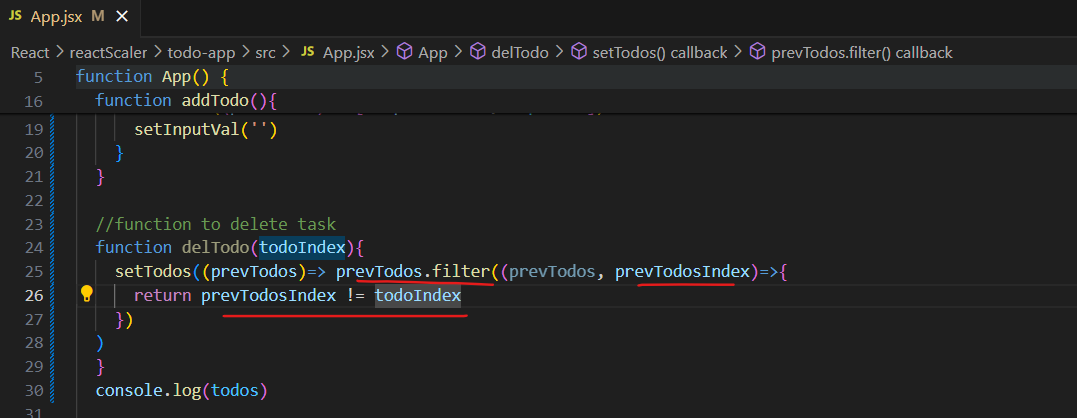


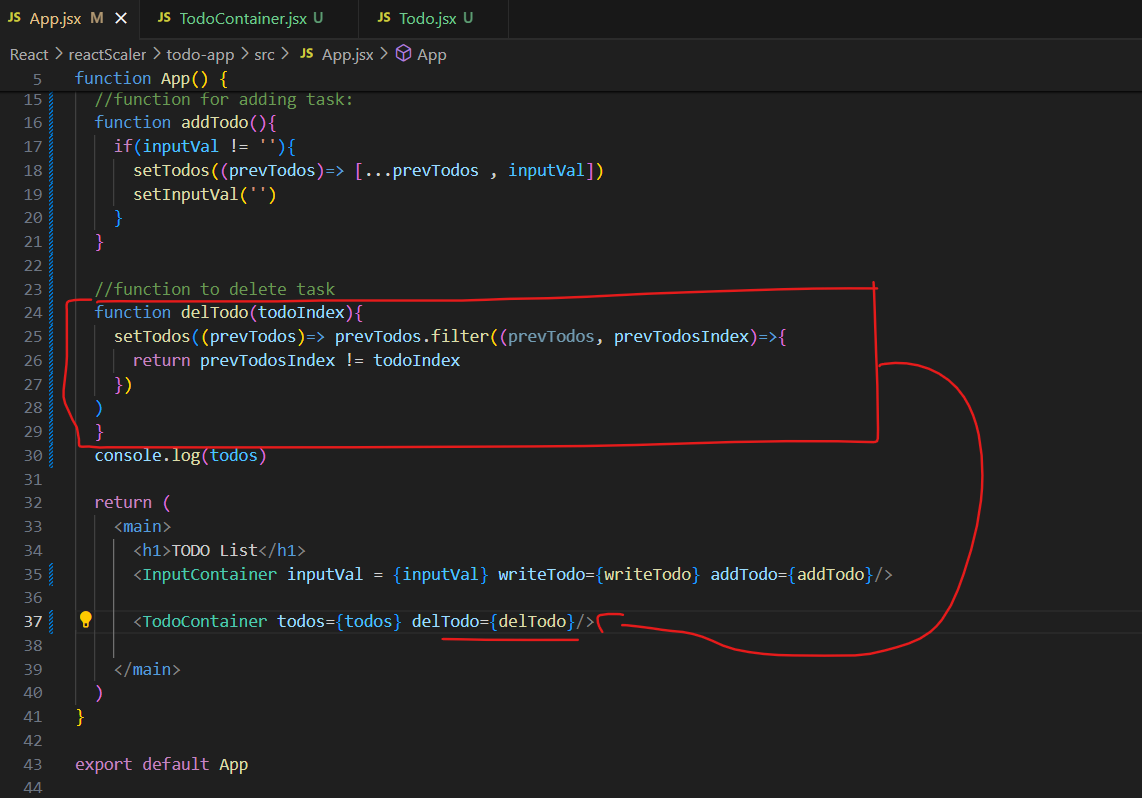


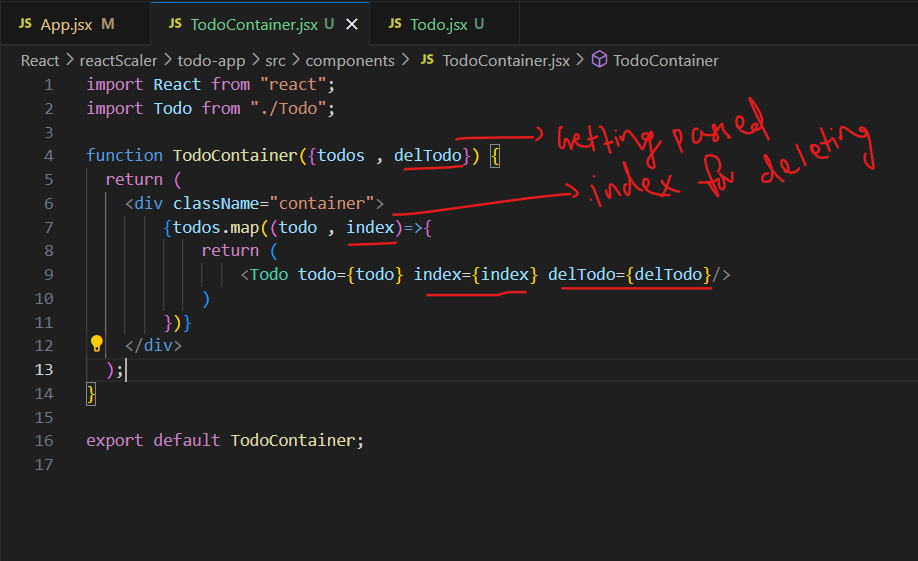
* + 1. Now we are able to Add tasks.

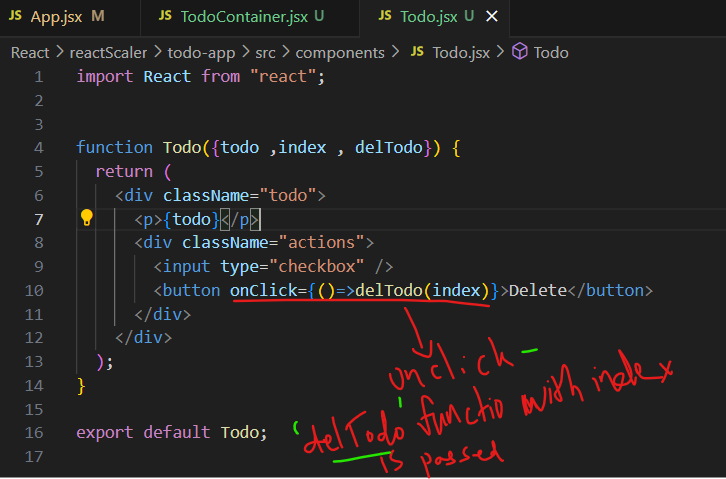


* 1. **Delete a Task from List :**
     1. First create a function inside ‘App.jsx’ for delete todo :









* + 1. Asdsad