

Luisa Fernanda Triviño Gil

CC: 1030612775

Estudiante Sena ADSO

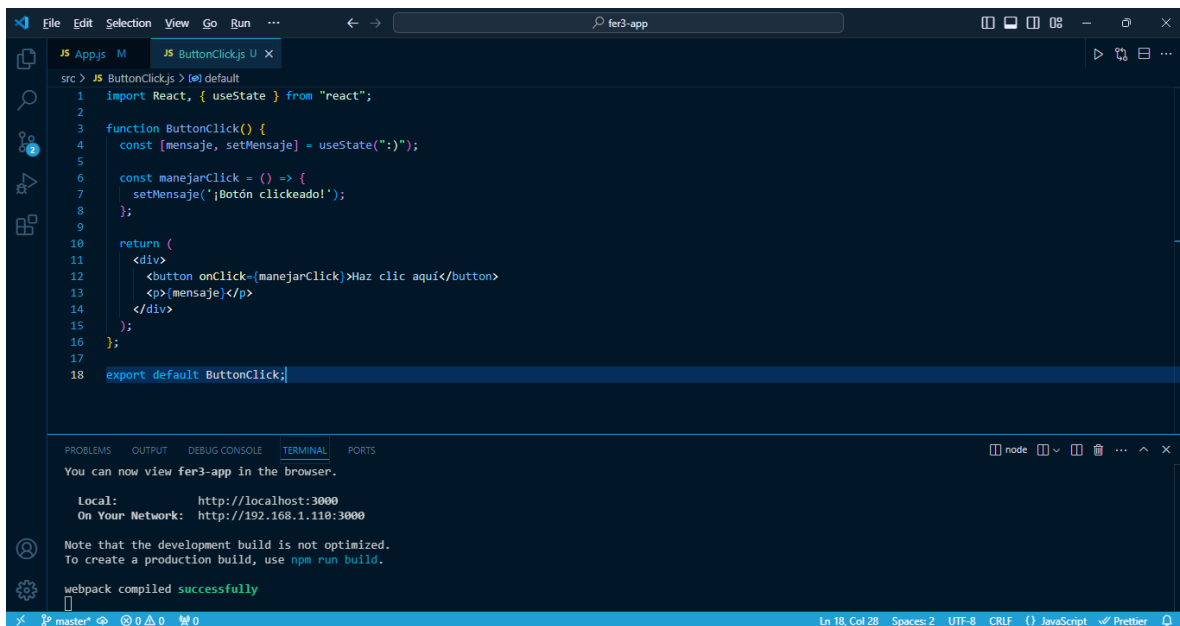
TALLER 4 REACT

Ejercicio 1: Manejo de eventos

En este ejercicio, vamos a trabajar con eventos como el clic en botones.

1. Crea un componente llamado ButtonClick.
2. En este componente, crea un botón que, al hacer clic, muestre una alerta que diga "¡Botón clickeado!".

ButtonClick.js



```
src > JS ButtonClickjs > |@ default
1 import React, { useState } from "react";
2
3 function ButtonClick() {
4   const [mensaje, setMensaje] = useState("");
5
6   const manejarClick = () => {
7     setMensaje('¡Botón clickeado!');
8   };
9
10  return (
11    <div>
12      <button onClick={manejarClick}>Haz clic aquí</button>
13      <p>{mensaje}</p>
14    </div>
15  );
16 };
17
18 export default ButtonClick;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

You can now view fer3-app in the browser.

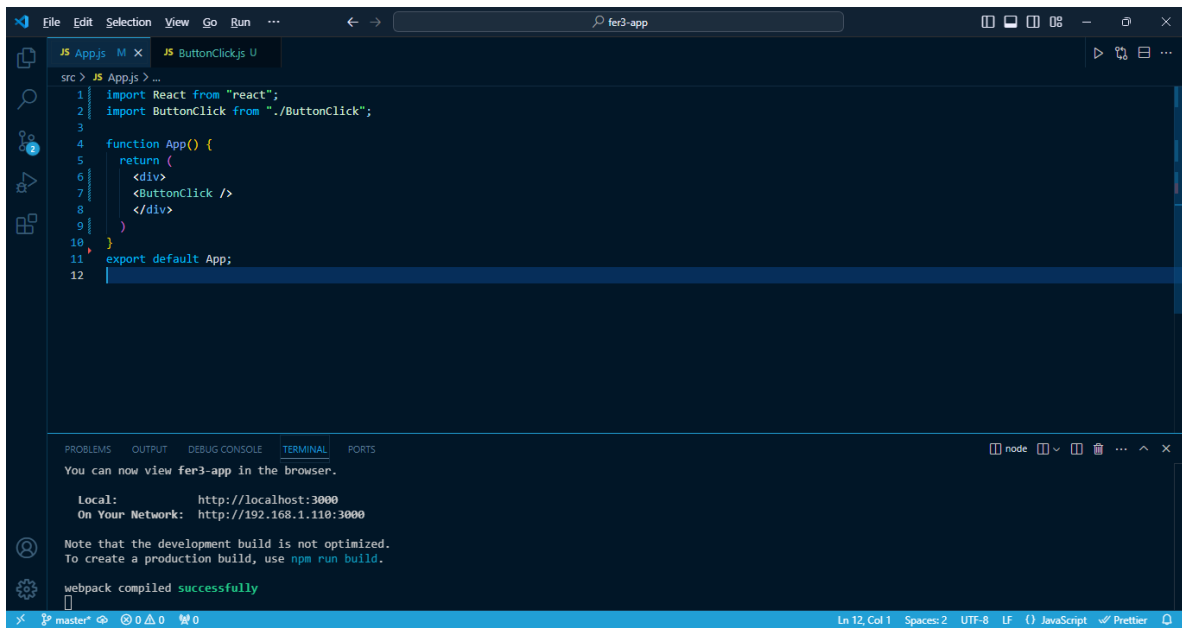
Local: http://localhost:3000
On Your Network: http://192.168.1.110:3000

Note that the development build is not optimized.
To create a production build, use `npm run build`.

webpack compiled successfully

Ln 18, Col 28 Spaces: 2 UTF-8 CRLF JavaScript Prettier

App.Js



The screenshot shows a VS Code editor with two files open: `App.js` and `ButtonClick.js`. The `App.js` file contains the following code:

```
1 import React from "react";
2 import ButtonClick from "../ButtonClick";
3
4 function App() {
5   return (
6     <div>
7       <ButtonClick />
8     </div>
9   )
10 }
11 export default App;
```

The terminal at the bottom shows the following output:

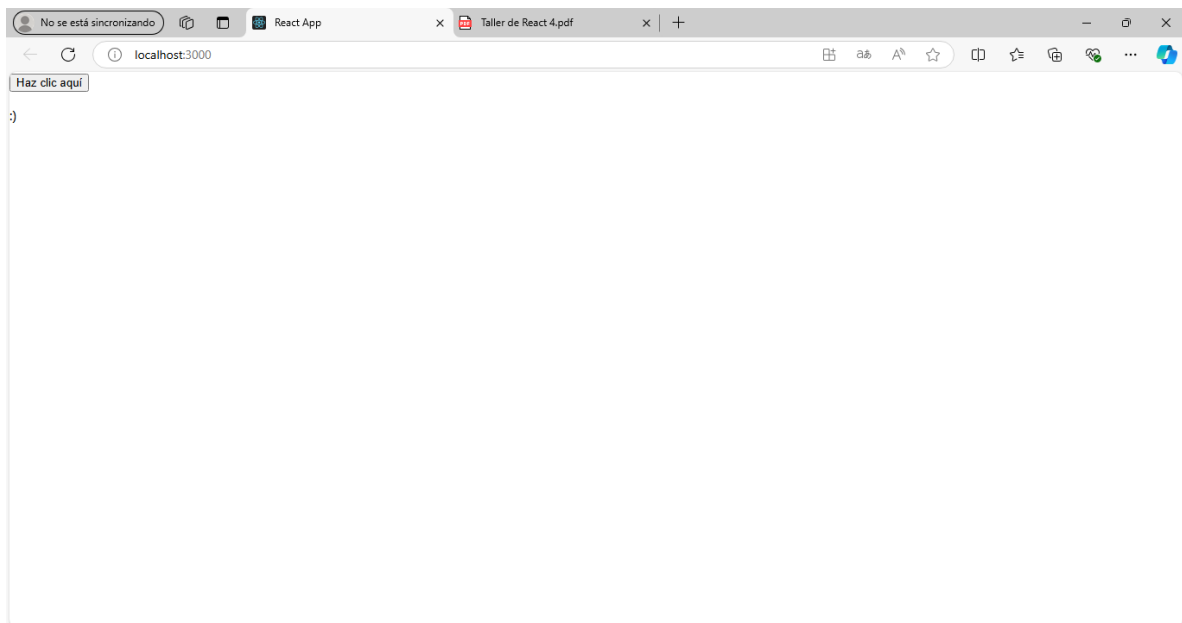
```
node
You can now view fer3-app in the browser.

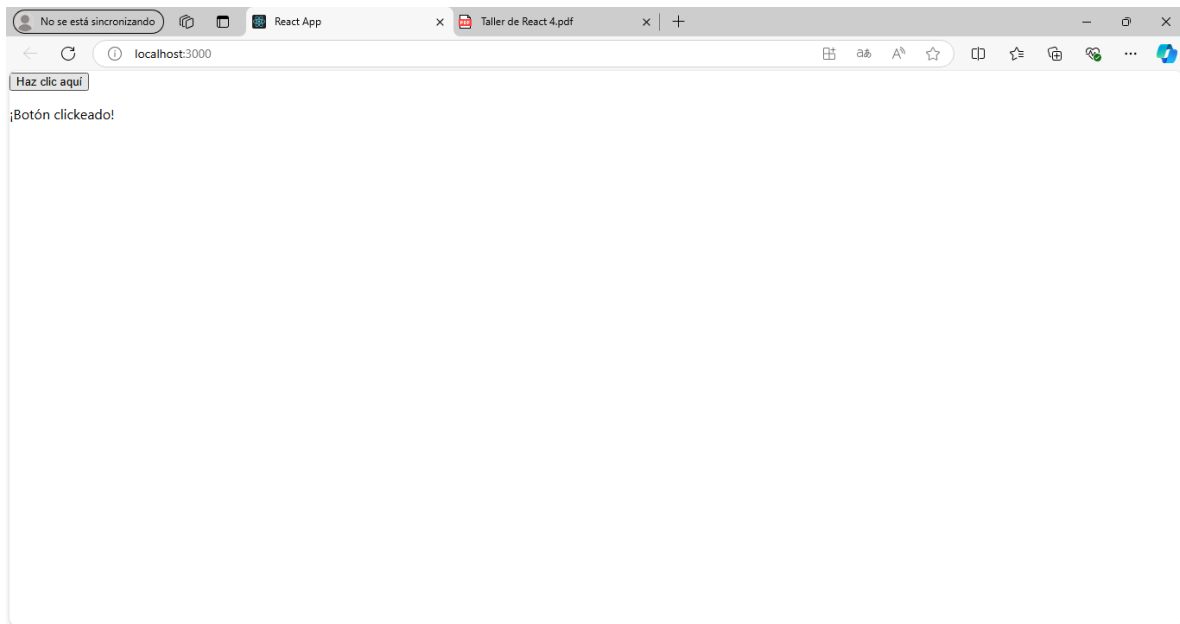
Local:      http://localhost:3000
On Your Network: http://192.168.1.110:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

Resultado





Ejercicio 2: Creación y gestión de formularios

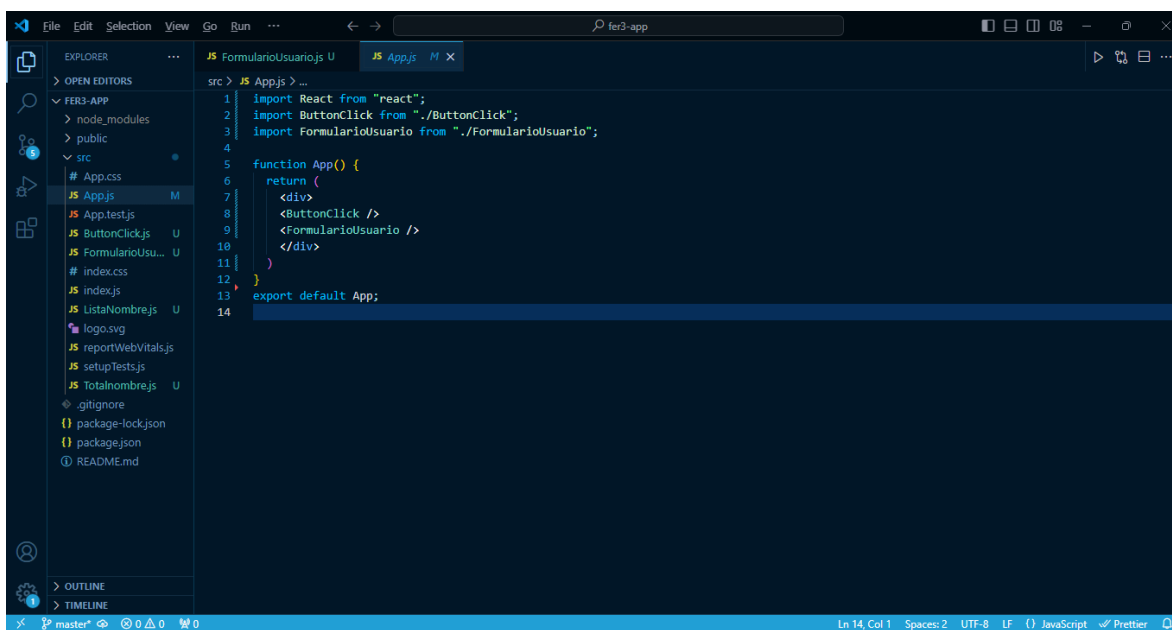
Ahora, vamos a crear un formulario simple para capturar el nombre de usuario.

1. Crea un componente llamado FormularioUsuario.
2. En este componente, agrega un campo de texto (input) y un botón.

FormularioUsuario.js

```
File Edit Selection View Go Run ...
src > JS FormularioUsuario.js > [0] default
1 import React, { useState } from "react";
2
3 //Funcion para almacenar nombre//
4 function FormularioUsuario () {
5   const[nombre, setNombre]= useState("")
6
7   const manejarCambio = (evento) => {
8     setNombre(evento.target.value);
9   };
10
11 //funcion para mostrar mensaje con el nombre ingresado//
12 const manejarEnvio = (evento) => {
13   evento.preventDefault();
14   alert("Usuario ingresado: ${nombre}")
15
16   setNombre(""); //Resetea el formulario//
17 };
18 return(
19   <form onSubmit={manejarEnvio}>
20     <label>
21       Nombre usuario:
22       <input type="text" value={nombre} onChange={manejarCambio} />
23     </label>
24     <button type="submit">Enviar </button>
25   </form>
26 )
27
28 }
29
30 export default FormularioUsuario;
```

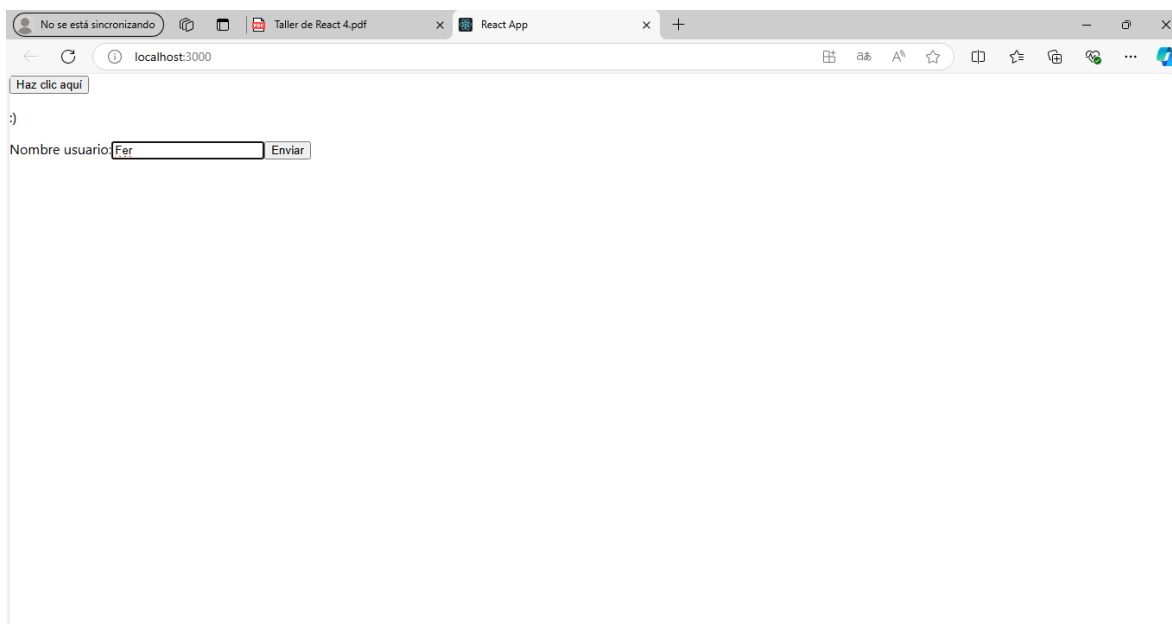
App.js

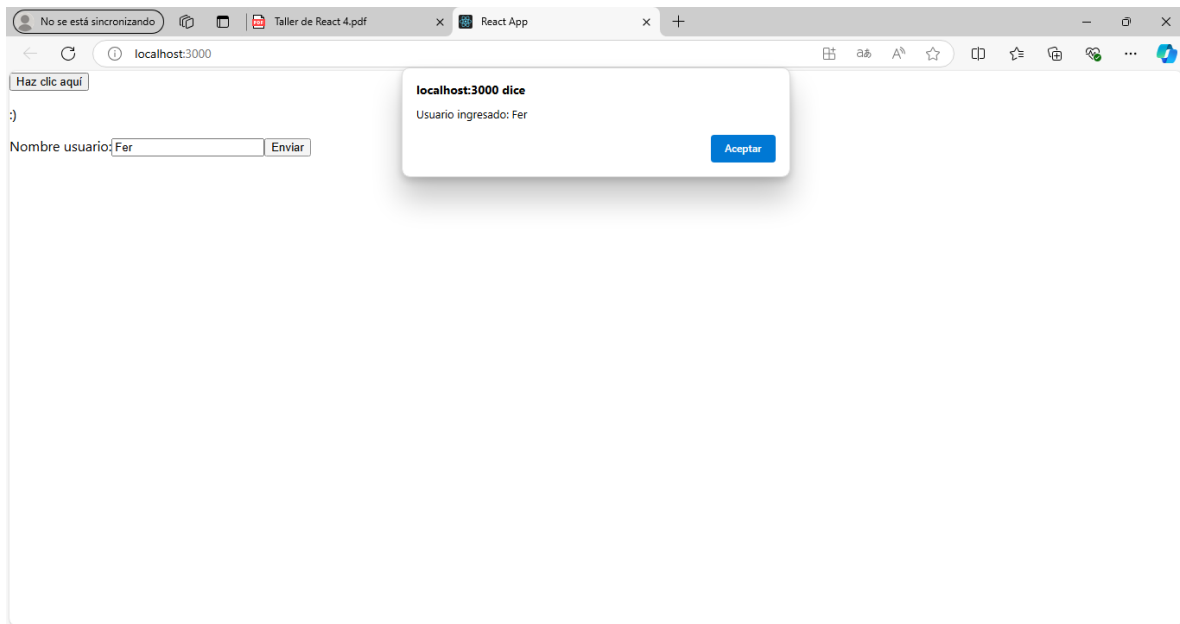


The screenshot shows the Visual Studio Code editor with a project named 'fer3-app'. The Explorer sidebar on the left shows the file structure: 'src' contains 'App.css', 'App.js', 'App.test.js', 'ButtonClick.js', 'FormularioUsuar...', 'index.css', 'index.js', 'ListaNombres.js', 'logo.svg', 'reportWebVitals.js', 'setupTests.js', and 'TotalNombres.js'. The main editor area displays the content of 'App.js'.

```
src > JS Appjs > ...
1  import React from "react";
2  import ButtonClick from "../ButtonClick";
3  import FormularioUsuario from "../FormularioUsuario";
4
5  function App() {
6    return (
7      <div>
8        <ButtonClick />
9        <FormularioUsuario />
10      </div>
11    )
12  }
13  export default App;
14
```

Resultado

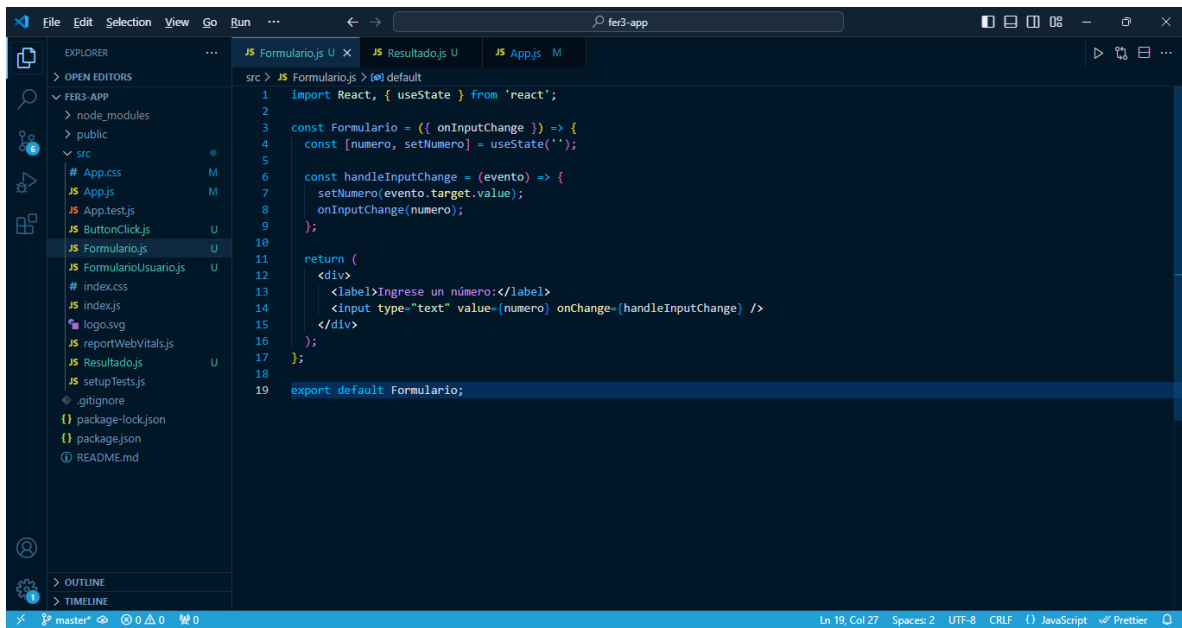




Ejercicio 3: Lifting State Up El lifting state up es un concepto clave para compartir el estado entre componentes. Vamos a practicarlo con un pequeño ejercicio.

1. Crea dos componentes: Formulario y Resultado.
2. Formulario contendrá un campo de texto donde el usuario podrá escribir un número.
3. El número que escriba el usuario debe mostrarse en el componente Resultado.

Formulario.js

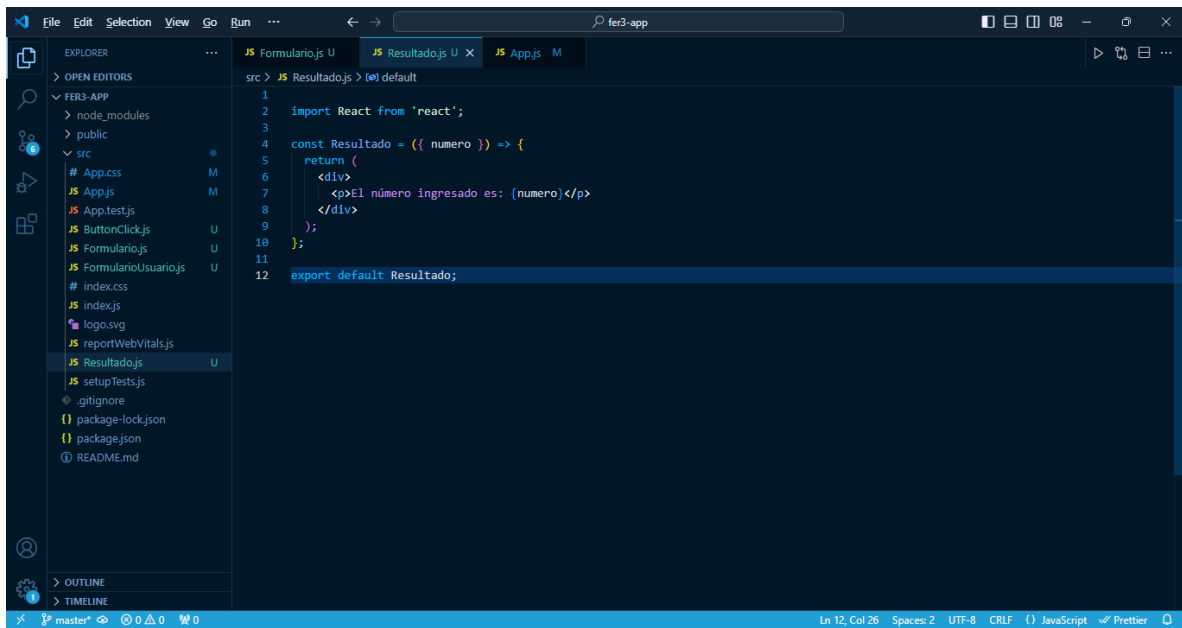


The screenshot shows the VS Code editor with the 'Formulario.js' file open. The Explorer sidebar on the left shows the project structure with files like App.css, App.js, App.test.js, ButtonClick.js, Formulario.js, FormularioUsuario.js, index.css, index.js, logo.svg, reportWebVitals.js, Resultado.js, setupTests.js, .gitignore, package-lock.json, package.json, and README.md. The main editor area displays the following code:

```
src > JS Formulario.js U X JS Resultado.js U JS App.js M
1  import React, { useState } from 'react';
2
3  const Formulario = ({ onChange }) => {
4    const [numero, setNumero] = useState('');
5
6    const handleInputChange = (evento) => {
7      setNumero(evento.target.value);
8      onChange(numero);
9    };
10
11   return (
12     <div>
13       <label>Ingresa un número:</label>
14       <input type="text" value={numero} onChange={handleInputChange} />
15     </div>
16   );
17 };
18
19 export default Formulario;
```

The status bar at the bottom indicates 'Ln 19, Col 27', 'Spaces: 2', 'UTF-8', 'CRLF', 'JavaScript', and 'Prettier'.

Resultado.js

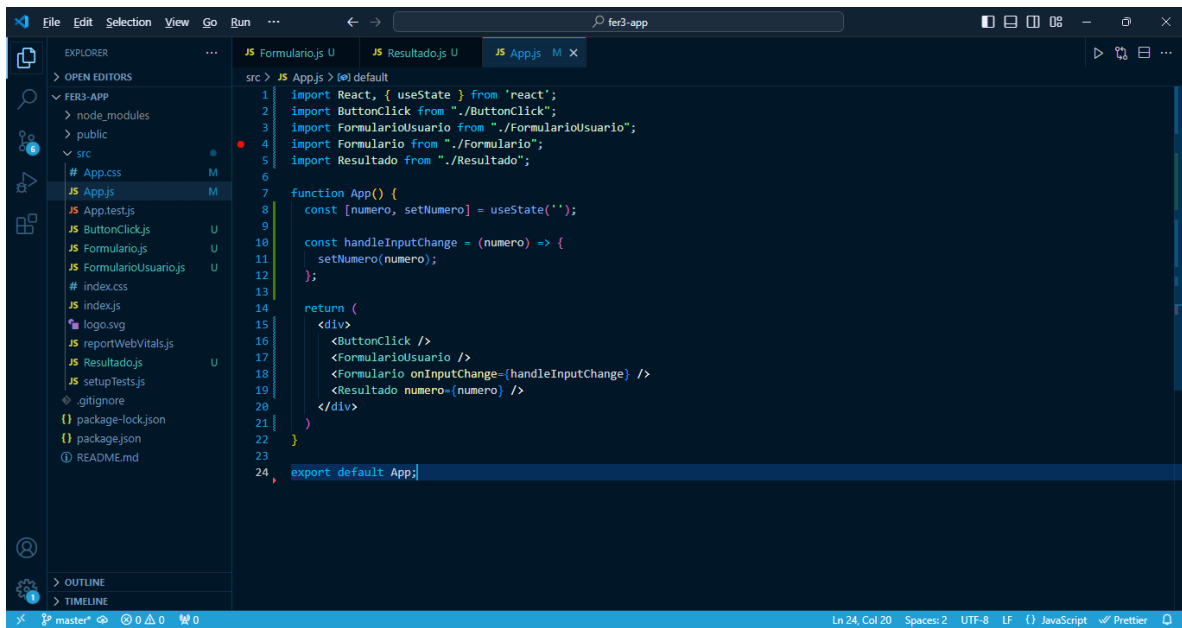


The screenshot shows the VS Code editor with the 'Resultado.js' file open. The Explorer sidebar on the left shows the project structure. The main editor area displays the following code:

```
src > JS Formulario.js U JS Resultado.js U X JS App.js M
1
2  import React from 'react';
3
4  const Resultado = ({ numero }) => {
5    return (
6      <div>
7        <p>El número ingresado es: {numero}</p>
8      </div>
9    );
10 };
11
12 export default Resultado;
```

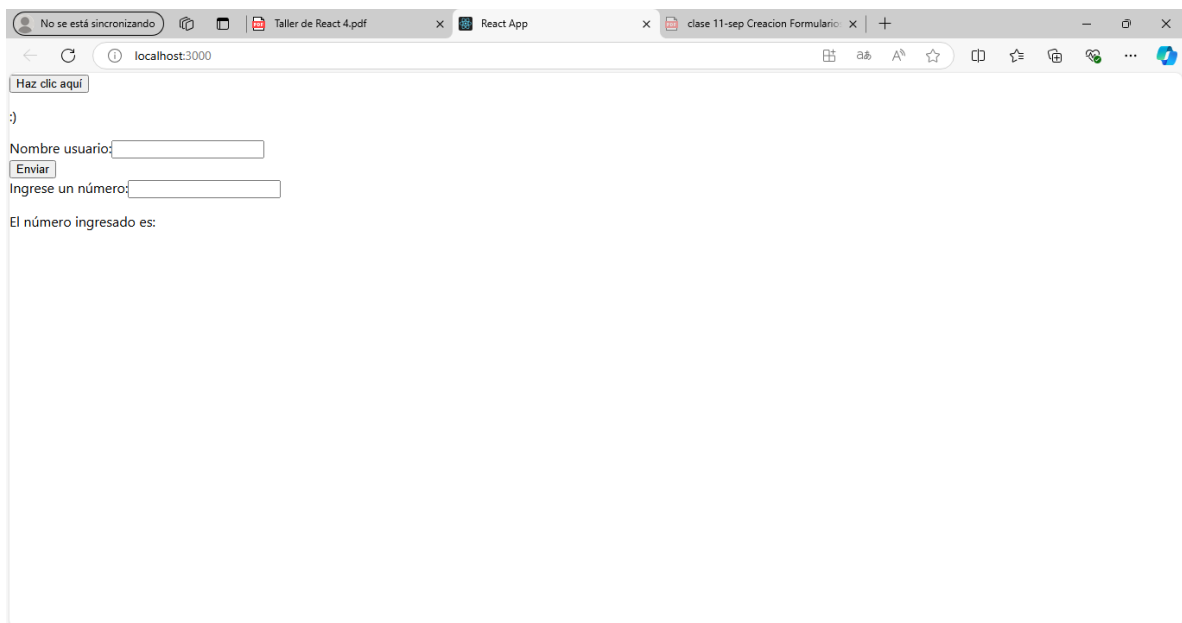
The status bar at the bottom indicates 'Ln 12, Col 26', 'Spaces: 2', 'UTF-8', 'CRLF', 'JavaScript', and 'Prettier'.

App.js



```
src > JS Appjs > | default
1 import React, { useState } from 'react';
2 import ButtonClick from "../ButtonClick";
3 import FormularioUsuario from "../FormularioUsuario";
4 import Formulario from "../Formulario";
5 import Resultado from "../Resultado";
6
7 function App() {
8   const [numero, setNumero] = useState('');
9
10  const handleInputChange = (numero) => {
11    setNumero(numero);
12  };
13
14  return (
15    <div>
16      <ButtonClick />
17      <FormularioUsuario />
18      <Formulario onChange={handleInputChange} />
19      <Resultado numero={numero} />
20    </div>
21  )
22 }
23
24 export default App;
```

Resultado



Haz clic aquí

;)

Nombre usuario:

Enviar

Ingresa un número:

El número ingresado es:

