

## **PRUEBA UNITARIA EN PROYECTO REACT**

**Elaborado por: Ubeimar Lizardo Yepes Portilla – Juan Pablo Lucero Morales**

- **Descripción General de la Actividad**

Como parte de la asignatura "Calidad de Software", esta actividad tuvo como propósito principal implementar una nueva funcionalidad en un proyecto React real y realizar su correspondiente prueba unitaria utilizando Jest y React Testing Library.

Se buscó promover la comprensión del flujo lógico de los componentes, el manejo de estados y el valor de las pruebas unitarias como herramienta de garantía de calidad.

- **Nueva Funcionalidad Implementada**

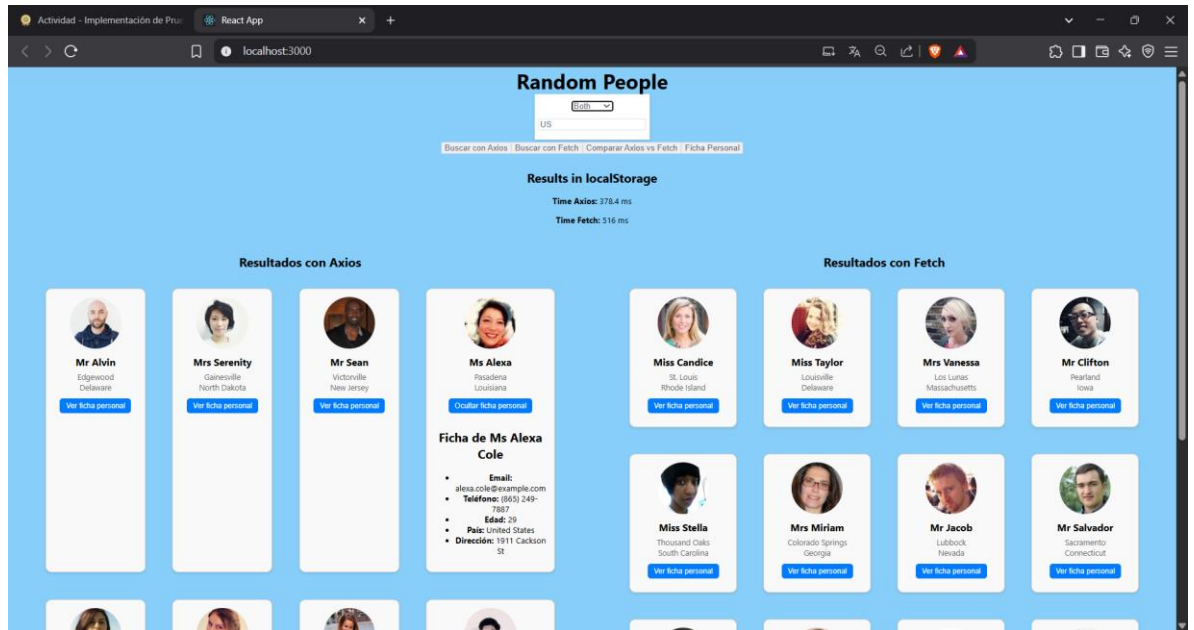
Se añadió un nuevo botón llamado “Ficha Personal” a la interfaz principal de la aplicación. Al hacer clic en él, se visualiza un componente llamado FichaPersonal, el cual despliega un mensaje personalizado y oculta los demás elementos de la interfaz relacionados con la búsqueda de personas.

- **Componentes creados o modificados:**

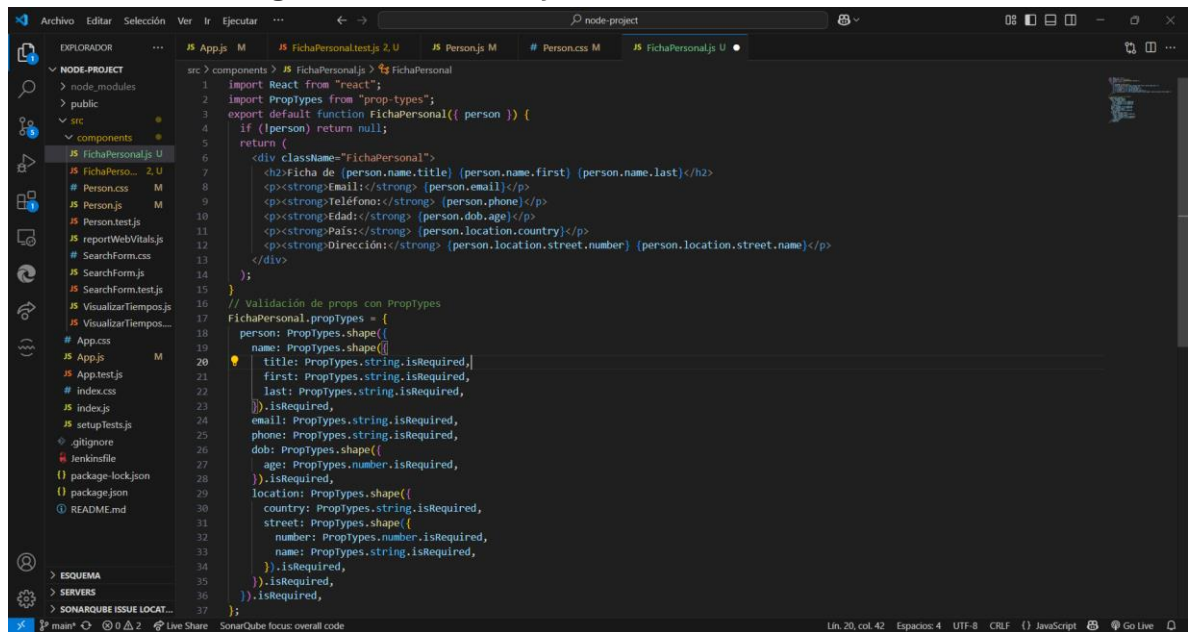
1. App.js: Se incorporó el nuevo botón y se implementó el estado showFicha para manejar el renderizado condicional del componente FichaPersonal.
2. components/FichaPersonal.js: Nuevo componente que muestra un mensaje simple en pantalla.

- **Capturas de Pantalla**

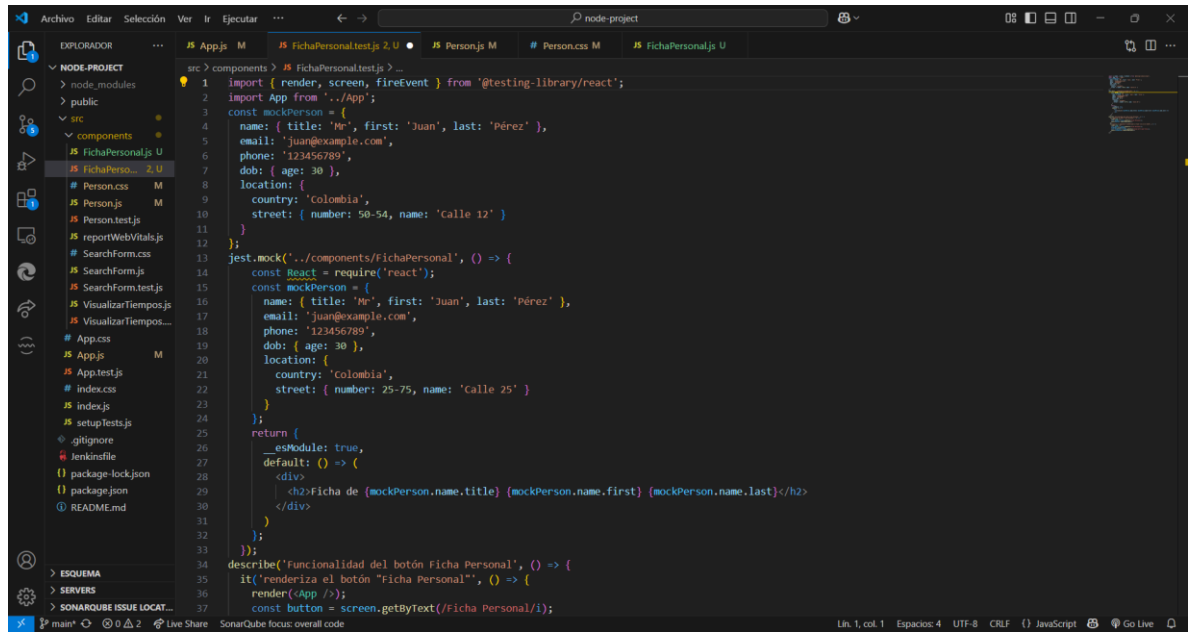
1. **Evidencia 1 – Interfaz**



## 2. Evidencia 2 –Codigo “FichaPersonal.js”

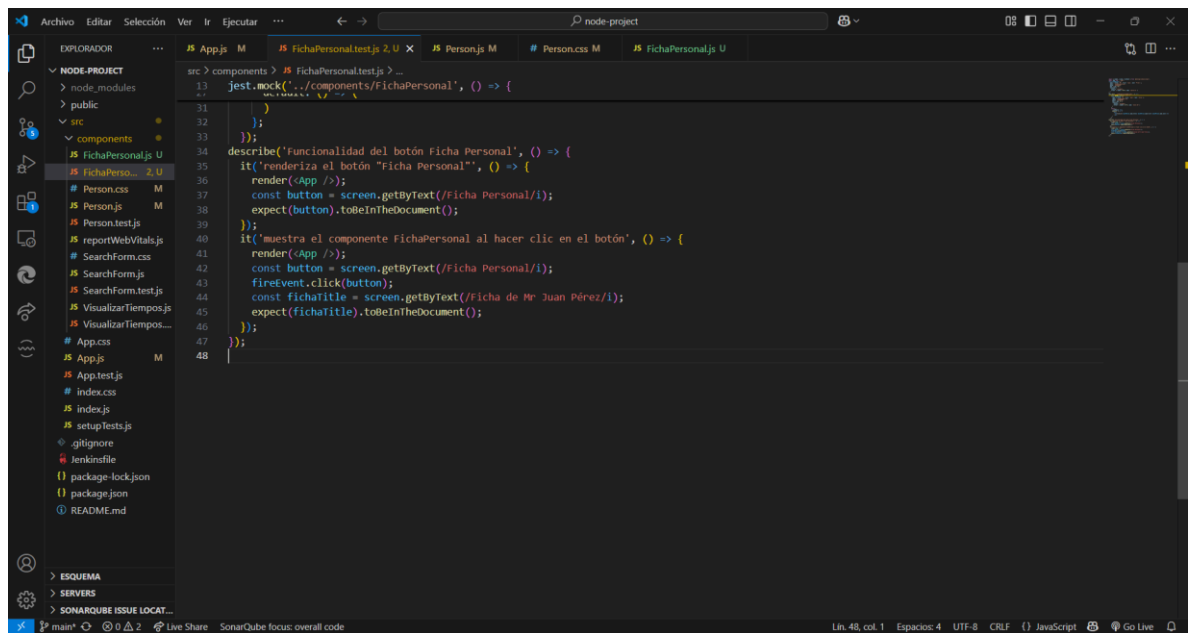


## 3. Evidencia 3 –Codigo “FichaPersonal.test.js”



The screenshot shows the VS Code editor with the file explorer on the left. The active file is `FichaPersonal.test.js` in the `src > components` directory. The code defines a mock for the `FichaPersonal` component using `jest.mock`. The mock function returns a React element that renders a heading with the mock person's name. The code includes imports for `render`, `screen`, and `fireEvent` from `@testing-library/react`, and `App` from `../App`. A `mockPerson` object is defined with properties like `name`, `email`, `phone`, `dobi`, and `location`. The mock function uses these properties to construct the rendered HTML. The code also includes a `describe` block for the mock's functionality and an `it` block for a test case.

```
1 import { render, screen, fireEvent } from '@testing-library/react';
2 import App from '../App';
3 const mockPerson = {
4   name: { title: 'Mr', first: 'Juan', last: 'Pérez' },
5   email: 'juan@example.com',
6   phone: '123456789',
7   dobi: { age: 30 },
8   location: {
9     country: 'Colombia',
10    street: { number: 50-54, name: 'Calle 12' }
11  }
12 };
13 jest.mock('../components/FichaPersonal', () => {
14   const React = require('react');
15   const mockPerson = {
16     name: { title: 'Mr', first: 'Juan', last: 'Pérez' },
17     email: 'juan@example.com',
18     phone: '123456789',
19     dobi: { age: 30 },
20     location: {
21       country: 'Colombia',
22       street: { number: 25-75, name: 'Calle 25' }
23     }
24   };
25   return (
26     <div>
27       <h2>Ficha de {mockPerson.name.title} {mockPerson.name.first} {mockPerson.name.last}</h2>
28     </div>
29   );
30 });
31
32 describe('Funcionalidad del botón Ficha Personal', () => {
33   it('renderiza el botón "Ficha Personal"', () => {
34     render(<App />);
35     const button = screen.getByText(/Ficha Personal/i);
36   });
37 });
```



The screenshot shows the continuation of the `FichaPersonal.test.js` file. The code continues the `describe` block with an `it` block that tests the rendering of the button. It uses `render` to render the `App` component, then `screen.getByText` to find the button. The `expect` function is used to assert that the button is in the document. The code also includes a `fireEvent.click` call to simulate a click on the button. The `it` block is followed by a `describe` block for the mock's functionality and an `it` block for a test case.

```
33   it('renderiza el botón "Ficha Personal"', () => {
34     render(<App />);
35     const button = screen.getByText(/Ficha Personal/i);
36     expect(button).toBeInTheDocument();
37   });
38
39   it('muestra el componente FichaPersonal al hacer clic en el botón', () => {
40     render(<App />);
41     const button = screen.getByText(/Ficha Personal/i);
42     fireEvent.click(button);
43     const fichaTitle = screen.getByText(/Ficha de Mr Juan Pérez/i);
44     expect(fichaTitle).toBeInTheDocument();
45   });
46 });
47
48
```

#### 4. Evidencia 4 – Resultado CMD

```
C:\WINDOWS\system32\cmd. X + v
PASS src/components/FichaPersonal.test.js
  Funcionalidad del botón Ficha Personal
    / renderiza el botón "Ficha Personal" (33 ms)
    / muestra el componente FichaPersonal al hacer clic en el botón (16 ms)

Test Suites: 1 passed, 1 total
Tests: 2 passed, 2 total
Snapshots: 0 total
Time: 4.865 s
Ran all test suites matching /FichaPersonal.test.js/i.
Watch Usage: Press w to show more.
```

- **Prueba Unitaria Desarrollada**

Se creó el archivo **components/FichaPersonal.test.js** con el siguiente contenido:

```
import { render, screen, fireEvent } from '@testing-library/react';
import App from '../App';
const mockPerson = {
  name: { title: 'Mr', first: 'Juan', last: 'Pérez' },
  email: 'juan@example.com',
  phone: '123456789',
  dob: { age: 30 },
  location: {
    country: 'Colombia',
    street: { number: 50-54, name: 'Calle 12' }
  }
};
jest.mock('../components/FichaPersonal', () => {
  const React = require('react');
  const mockPerson = {
    name: { title: 'Mr', first: 'Juan', last: 'Pérez' },
    email: 'juan@example.com',
    phone: '123456789',
    dob: { age: 30 },
    location: {
      country: 'Colombia',
```

```

    street: { number: 25-75, name: 'Calle 25' }
  }
};
return {
  __esModule: true,
  default: () => (
    <div>
      <h2>Ficha de {mockPerson.name.title} {mockPerson.name.first}
{mockPerson.name.last}</h2>
    </div>
  )
};
});
describe('Funcionalidad del botón Ficha Personal', () => {
  it('renderiza el botón "Ficha Personal", () => {
    render(<App />);
    const button = screen.getByText(/Ficha Personal/i);
    expect(button).toBeInTheDocument();
  });
  it('muestra el componente FichaPersonal al hacer clic en el botón', () => {
    render(<App />);
    const button = screen.getByText(/Ficha Personal/i);
    fireEvent.click(button);
    const fichaTitle = screen.getByText(/Ficha de Mr Juan Pérez/i);
    expect(fichaTitle).toBeInTheDocument();
  });
});
});

```

- **Ejecución de la Prueba**

Para validar el correcto funcionamiento de la nueva funcionalidad, se ejecutó el siguiente comando:

*Npm test FichaPersonal.test.js*

Donde el resultado de la prueba fue:

PASS src/components/FichaPersonal.test.js

Funcionalidad del botón Ficha Personal

✓ renderiza el botón "Ficha Personal" (33 ms)

✓ muestra el componente FichaPersonal al hacer clic en el botón (16 ms)

Test Suites: 1 passed, 1 total

Tests: 2 passed, 2 total

Snapshots: 0 total

Time: 4.865 s

Ran all test suites matching /FichaPersonal.test.js/i.

- **Conclusion**

Esta actividad permitió aplicar de forma práctica los conocimientos sobre pruebas unitarias en React, reforzando habilidades como:

- Comprender la estructura de un proyecto React real.
- Añadir nuevas funcionalidades mediante componentes reutilizables.
- Validar componentes de forma automática con pruebas unitarias.
- Reconocer la importancia de la calidad del software a través del testing.

El uso de Jest y React Testing Library hizo posible implementar pruebas claras, específicas y enfocadas, alineadas con buenas prácticas del desarrollo profesional.