




INFORME DE TEST UNITARIOS FRONTEND – SPRINT III

Los tests unitarios en frontend se están realizando con base en el framework Jest. Para este sprint se inició con su implementación en los Components que se han creado hasta el momento, por lo que a continuación detallaré lo realizado.

- **Navbar**

En esta parte se realizará la validación del componente Navbar. El objetivo es verificar los diferentes estados de acuerdo con el estado de autenticación del usuario.

Nombre y descripción	Código del Test	Resultado
Renders Navbar componente: Valida el renderizado inicial verificando si el id que debe ser asignado en el código, está en el documento luego de renderizar.	<pre>test('renders Navbar component', () => { render(<Navbar />); const navbarElement = screen.getByTestId('navbar'); expect(navbarElement).toBeInTheDocument(); });</pre>	 Pendiente de ejecución
Renders Sign In and Sign UP buttons when not authenticated: valida el renderizado sin autenticación de los botones Iniciar sesión y Registrarse.	<pre>test('renders Sign In and Sign Up buttons when not authenticated', () => { localStorage.setItem('auth', JSON.stringify(false)); render(<Navbar />); const signInButton = screen.getByText('Iniciar Sesión'); const signUpButton = screen.getByText('Registrarse'); expect(signInButton).toBeInTheDocument(); expect(signUpButton).toBeInTheDocument(); });</pre>	 Pendiente de ejecución
Renders Log Out button and Avatar when authenticated: valida el renderizado del navbar con autenticación, en el cual deben aparecer el Avatar y el botón de Cerrar Sesión.	<pre>test('renders Log Out button and Avatar when authenticated', () => { localStorage.setItem('auth', JSON.stringify(true)); render(<Navbar />); const logOutButton = screen.getByText('Cerrar Sesión'); const avatarElement = screen.getByTestId('avatar'); expect(logOutButton).toBeInTheDocument(); expect(avatarElement).toBeInTheDocument(); });</pre>	 Pendiente de ejecución

- **Slider**


En esta parte se realizará la validación del componente Slider. El objetivo es verificar el renderizado de la imagen que está predeterminada en el código:

Nombre y descripción	Código del Test	Resultado
renders Slider component with the correct image: valida el renderizado del slider con la imagen predeterminada que está en el código.	<pre>test('renders Slider component with the correct image', () => { render(<Slider />); const sliderElement = screen.getByTestId('slider'); const imageElement = screen.getByRole('img'); expect(sliderElement).toBeInTheDocument(); expect(imageElement).toBeInTheDocument(); expect(imageElement).toHaveAttribute('src', '/images/fotoCanchaFutbol1banner.jpg'); });</pre>	⚠ Pendiente de ejecución

- **Avatar**


En esta parte se realizará la validación del componente Avatar. El objetivo es verificar el renderizado del componente, incluyendo las iniciales que debe contener de acuerdo con el nombre del usuario.

Nombre y descripción	Código del Test	Resultado
renders Avatar component with initials when image is not loaded: valida el renderizado del avatar con las iniciales de acuerdo con el nombre del usuario que se haya registrado. Además valida que no haya una imagen cargada con el <code>not.toBeInTheDocument()</code>	<pre>test('renders Avatar component with initials when image is not loaded', () => { render(<Avatar name="Ana Garcia" image="/path/to/image.jpg" />); const avatarElement = screen.getByTestId('avatar'); const initialsElement = screen.getByText('AG'); const imageElement = screen.queryByRole('img'); expect(avatarElement).toBeInTheDocument(); expect(initialsElement).toBeInTheDocument(); expect(imageElement).not.toBeInTheDocument(); });</pre>	⚠ Pendiente de ejecución

renders Avatar component with image when image is loaded : verifica el renderizado del avatar con una imagen cargada en el registro del usuario.	<pre>test('renders Avatar component with image when image is loaded', () => { render(<Avatar name="Ana Garcia" image="/path/to/image.jpg" />); const avatarElement = screen.getByTestId('avatar'); const initialsElement = screen.getByText('AG'); const imageElement = screen.getByRole('img'); expect(avatarElement).toBeInTheDocument(); expect(initialsElement).not.toBeInTheDocument(); expect(imageElement).toBeInTheDocument(); expect(imageElement).toHaveAttribute('src', '/path/to/image.jpg'); });</pre>	 Pendiente de ejecución
---------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------

- **PoliticCards**

En esta parte se realizará la validación del componente Avatar. El objetivo es verificar el renderizado del componente, incluyendo las iniciales que debe contener de acuerdo con el nombre del usuario.

Nombre y descripción	Código del Test	Resultado
renders PoliticCards component with the correct content: verifica el renderizado de las políticas de cada cancha, validando que se generen las 3 columnas y que estas tengan la información que se pasa por props al componente.	<pre>test('renders PoliticCards component with the correct content', () => { const fieldRules = 'No pets allowed. No smoking.'; const safety = 'Masks are mandatory. Hand sanitizers available.'; const cancelation = 'Free cancellation up to 24 hours before the reservation.'; render(<PoliticCards fieldRules={fieldRules} safety={safety} cancelation={cancelation} />); });</pre>	 Pendiente de ejecución

```
const fieldRulesElement = screen.getByText('Reglas de la
cancha');
const safetyElement = screen.getByText('Salud y
seguridad');
const cancelationElement = screen.getByText('Política de
cancelación');

expect(fieldRulesElement).toBeInTheDocument();
expect(safetyElement).toBeInTheDocument();
expect(cancelationElement).toBeInTheDocument();

const fieldRulesContent = screen.getByText(fieldRules);
const safetyContent = screen.getByText(safety);
const cancelationContent = screen.getByText(cancelation);

expect(fieldRulesContent).toBeInTheDocument();
expect(safetyContent).toBeInTheDocument();
expect(cancelationContent).toBeInTheDocument();
});
```