

## 1) INTRODUCCIÓN Y PROPÓSITO

Este documento describe el proceso de testing realizado sobre los componentes principales del proyecto frontend en React: Navbar, Catálogo y Productos. El propósito del testing es asegurar que los componentes se rendericen correctamente, que las interacciones del usuario funcionen como se espera y que la comunicación con el backend se realice correctamente. Se buscó cubrir pruebas de renderizado, navegación y operaciones de la UI (ej. eliminación de productos).

## 2) HERRAMIENTAS DE CONFIGURACIÓN

- **React:** Librería para construcción de interfaces de usuario.
- **Vitest:** Framework de testing para JavaScript/TypeScript, similar a Jest.
- **@testing-library/react:** Librería para testing de componentes React, centrada en la interacción del usuario.
- **@testing-library/user-event:** Para simular eventos del usuario (click, type, etc.).
- **MemoryRouter de react-router-dom:** Para pruebas de navegación en componentes que usan rutas.
- **Mocking de fetch y useNavigate:** Para simular llamadas al backend y navegación sin depender del entorno real.

Configuración básica de mock para rutas:

```
import { vi } from "vitest";
const mockNavigate = vi.fn();

vi.mock("react-router-dom", async () => {
  const actual = await vi.importActual("react-router-dom");
  return {
    ...actual,
    useNavigate: () => mockNavigate,
  };
});
```

## 3) PLAN DE TESTING

**A) Navbar** Objetivo: Verificar que la marca y los links se muestren correctamente y que la navegación funcione.


```
import { render, screen } from "@testing-library/react";
import userEvent from "@testing-library/user-event";
import { Navbar } from "../componentes/Navbar";
```

```

test("Muestra marca y links con href correctos", () => {
  render(<Navbar />);
  expect(screen.getByText(/Tennis Court/i)).toBeInTheDocument();
  expect(screen.getByRole("link", { name: /Administrador/i })).toHaveAttribute("href", "/Administrador");
});

test("Navega a /Administrador al hacer click en el link", async () => {
  render(<Navbar />);
  await userEvent.click(screen.getByRole("link", { name: /Administrador/i }));
  expect(mockNavigate).toHaveBeenCalledWith("/Administrador");
});

```

**Resultado:**  Tests pasados, navegación correcta y links visibles.

**B) Catálogo** Objetivo: Verificar que los productos se muestren y que los botones de info redirijan correctamente.

```

import { render, screen } from "@testing-library/react";
import userEvent from "@testing-library/user-event";
import { Catalogo } from "../componentes/catalogo/catalogo";
import { vi } from "vitest";

test("Muestra los productos y permite navegar con + info", async () => {
  render(<Catalogo />);

  expect(screen.getByText(/Raquetas/i)).toBeInTheDocument();
  expect(screen.getByText(/Pelotas/i)).toBeInTheDocument();
  expect(screen.getByText(/Ropa deportiva/i)).toBeInTheDocument();

  await userEvent.click(screen.getByText("+ info"));
  expect(mockNavigate).toHaveBeenCalled();
});

```

**Resultado:**  Productos visibles, botones disparan navegación simulada.

**C) Productos** Objetivo: Verificar que se cargan los productos desde el backend y se renderizan en la tabla.

```

import { render, screen, waitFor } from "@testing-library/react";
import { Productos } from "../componentes/productos/productos";

beforeEach(() => {

```

```

global.fetch = vi.fn(() =>
  Promise.resolve({
    ok: true,
    json: () =>
      Promise.resolve([
        { id: 1, nombre: "Raqueta", descripcion: "Descripción 1", precio:
100, activo: true },
        { id: 2, nombre: "Pelota", descripcion: "Descripción 2", precio: 50,
activo: true },
      ]),
    })
  ) as unknown as typeof fetch;
});

test("muestra los productos cargados desde el backend", async () => {
  render(<Productos />);
  await waitFor(() => expect(screen.getByText(/Raqueta/i)).toBeInTheDocument());
  expect(screen.getByText(/Pelota/i)).toBeInTheDocument();
});

```

**Resultado:**  Productos correctamente cargados y visibles en la tabla.

## 4) CONCLUSIÓN

Se ha verificado que los componentes principales del frontend cumplen con las siguientes expectativas: - Navbar: correcta visualización de links y navegación. - Catálogo: renderizado de productos y botones de navegación funcionales. - Productos: conexión simulada con backend y renderizado correcto en tabla.

El uso de Vitest y React Testing Library permite validar el comportamiento de la UI de forma aislada, asegurando calidad en la interacción con el usuario y reduciendo riesgos de errores en producción.