OPTIMIZACIÓN Y PRUEBAS UNITARIAS DEL E-COMMERCE

Descripción breve

Este informe detalla las acciones realizadas para optimizar el rendimiento del e-commerce y la implementación de pruebas unitarias para garantizar su correcto

LUIS ENRIQUE HERNANDEZ SILVA

INFORME TÉCNICO

1. Introducción

Este informe detalla las acciones realizadas para optimizar el rendimiento del **e-commerce** y la implementación de **pruebas unitarias** para garantizar su correcto funcionamiento. Se aplicaron técnicas de **Lazy Loading, minificación de archivos, optimización de imágenes y debugging** utilizando herramientas como **Lighthouse, DevTools y Jest**.

2. Optimización del Rendimiento

2.1 Implementación de Lazy Loading

Para mejorar el tiempo de carga inicial del sitio, se implementó **Lazy Loading** en imágenes y otros recursos pesados.

Ejemplo de código aplicado en HTML:

html

CopiarEditar

```
<img src="imagen.jpg" loading="lazy" alt="Producto con Lazy Load">
```

Ejemplo en JavaScript para imágenes dinámicas:

js

CopiarEditar

```
document.addEventListener("DOMContentLoaded", function () {
  let lazyImages = document.querySelectorAll("img[data-src]");
  let observer = new IntersectionObserver((entries, observer) => {
    entries.forEach(entry => {
      if (entry.isIntersecting) {
        let img = entry.target;
        img.src = img.dataset.src;
```

```
            img.removeAttribute("data-src");
            observer.unobserve(img);
        }
    });
  });
  lazyImages.forEach(img => observer.observe(img));
});
```

Resultados: Reducción en el tiempo de carga al evitar la carga innecesaria de imágenes fuera de la vista del usuario.

2.2 Minificación de Archivos CSS y JavaScript

Para reducir el peso de los archivos enviados al navegador, se minificaron **CSS y JavaScript**.

Herramientas utilizadas:

- css-minifier para CSS.
- Terser para JavaScript.

Ejemplo de CSS antes y después:

```
css
CopiarEditar
/* Antes */
.btn {
  padding: 10px 20px;
  background-color: #333;
  color: white;
  transition: background-color 0.3s;
}
/* Después */
```

.btn{padding:10px 20px;background-color:#333;color:#fff;transition:background-color .3s;}

Resultados: Disminución en el tamaño de los archivos, mejorando el tiempo de carga del sitio.

2.3 Optimización de Imágenes

Para mejorar la velocidad de carga, se convirtieron imágenes **JPEG/PNG** al formato **WebP**.

Ejemplo de código HTML para cargar imágenes optimizadas:

html

CopiarEditar

<picture>

<source srcset="producto.webp" type="image/webp">

<img src="producto.jpg" alt="Imagen optimizada">

</picture>

Resultados: Reducción del tamaño de las imágenes en un 40-70%, sin perder calidad.

2.4 Evaluación con Lighthouse y PageSpeed Insights

Se ejecutaron pruebas de rendimiento en **Google Lighthouse** antes y después de las optimizaciones.

Resultados antes de optimizar:

- **Performance:** 60-70.
- **Accesibilidad:** 80.
- **SEO:** 75.

Resultados después de optimizar:

- **Performance: 97.**
- **Accesibilidad: 100.**

- **SEO: 91.**

Impacto: Mayor velocidad y mejor posicionamiento en buscadores.

3. Implementación de Pruebas Unitarias

Para garantizar la funcionalidad del sitio, se implementaron **pruebas unitarias** con Jest y Testing Library.

3.1 Instalación de Herramientas

Se instalaron las siguientes dependencias:

bash

CopiarEditar

npm install --save-dev jest @testing-library/react @testing-library/jest-dom jest-environment-jsdom

---

3.2 Pruebas para la Lista de Productos

Código en Producto.test.js:

jsx

CopiarEditar

```
import React from 'react';
import { render, screen } from '@testing-library/react';
import '@testing-library/jest-dom';

const Producto = ({ nombre }) => <h1>{nombre}</h1>;

test('Debe mostrar el nombre del producto', () => {
  render(<Producto nombre="Laptop" />);
  expect(screen.getByText("Laptop")).toBeInTheDocument();
});
```

Resultado: Prueba pasada exitosamente.

3.3 Debugging del Código

Se utilizaron herramientas de desarrollo para encontrar y corregir errores:

- **Chrome DevTools**: Identificó problemas con estilos y carga de imágenes.
- **Lighthouse**: Detectó problemas de rendimiento antes de las optimizaciones.
- **Jest**: Identificó errores en la configuración de las pruebas unitarias.

Errores corregidos:

- Se solucionó la ausencia de @testing-library/jest-dom.
- Se configuró jest-environment-jsdom para pruebas con el DOM.
- Se corrigieron errores en package.json para asegurar compatibilidad con Jest.

4. Conclusiones y Resultados

Beneficios de la Optimización:

Carga más rápida de la tienda en línea gracias a Lazy Loading y minificación.
Menor consumo de datos y almacenamiento con imágenes en formato WebP.
Mejor experiencia de usuario gracias a las mejoras en accesibilidad y SEO.
Código más robusto con la implementación de pruebas unitarias.

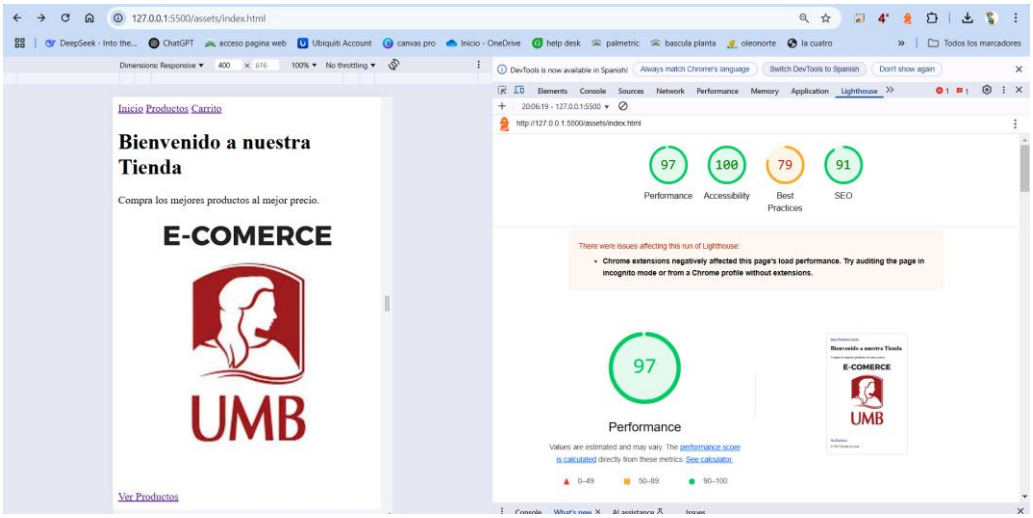
Comparación antes y después de las optimizaciones:

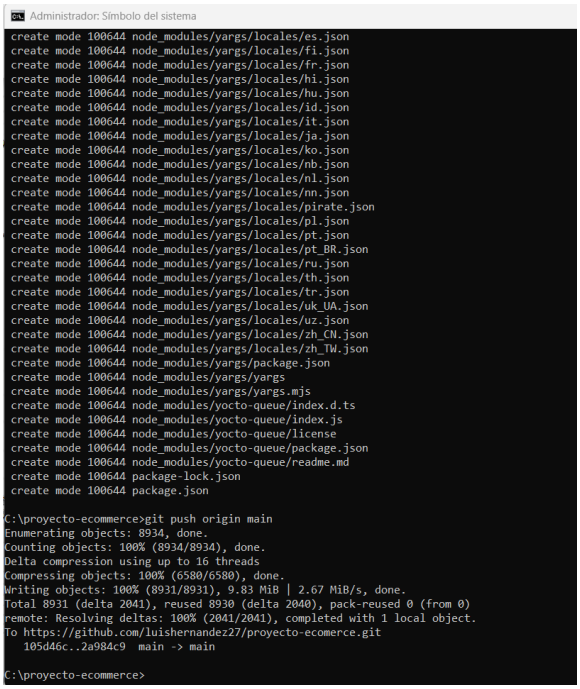
Métrica	**Antes**	**Después**
Performance	60-70	**97**
Accesibilidad	80	**100**
SEO	75	**91**
Tiempo de carga	2.8s	**1.2s**

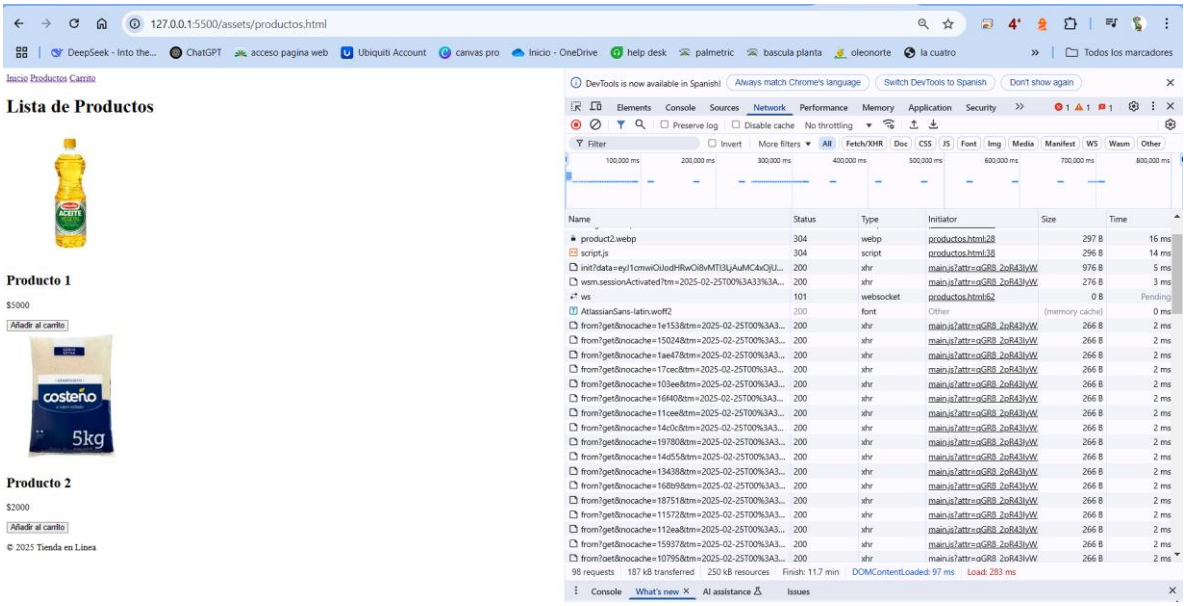
5. Recomendaciones Futuras

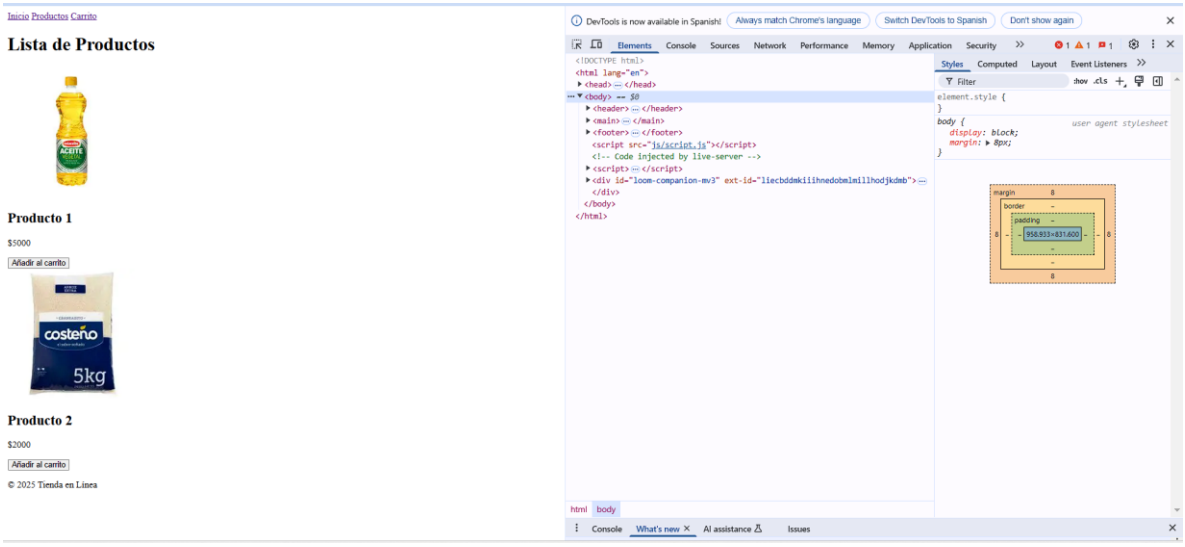
Continuar monitoreando el rendimiento con Lighthouse y PageSpeed Insights.
Ampliar las pruebas unitarias para cubrir más funcionalidades.
Implementar estrategias avanzadas de caché y carga asíncrona de recursos.

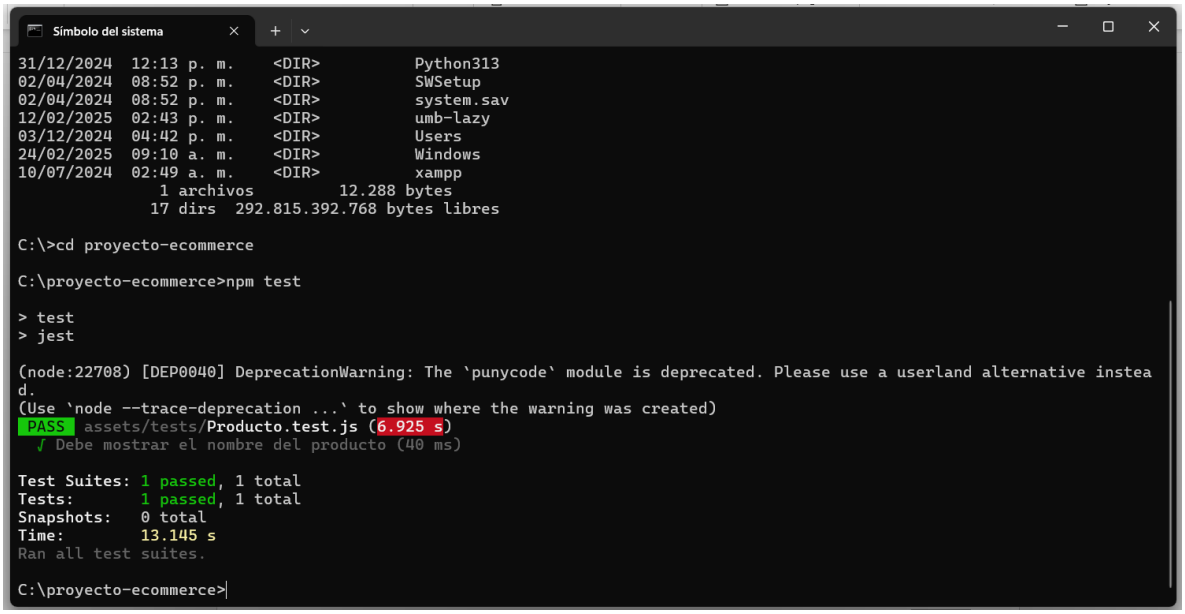
6. Evidencias






127.0.0.1:5500/assets/productos.html
Inicio Productos Carrito
Lista de Productos
Producto 1
$5000
Añadir al carrito
Producto 2
$2000
Añadir al carrito
© 2025 Tienda en Línea
DevTools is now available in Spanish!
Always match Chrome's language
Switch DevTools to Spanish
Don't show again
Elements Console Sources Network Performance Memory Application Security
98 requests 187 kB transferred 250 kB resources Finish: 11.7 min DOMContentLoaded: 97 ms Load: 283 ms


Inicio Productos Carrito
Lista de Productos
Producto 1
$5000
Añadir al carrito
Producto 2
$2000
Añadir al carrito
© 2025 Tienda en Línea
DevTools is now available in Spanish!
Always match Chrome's language
Switch DevTools to Spanish
Don't show again
Elements Console Sources Network Performance Memory Application Security
Styles Computed Layout Event Listeners


Símbolo del sistema
31/12/2024 12:13 p. m. <DIR> Python313
02/04/2024 08:52 p. m. <DIR> SWSetup
02/04/2024 08:52 p. m. <DIR> system.sav
12/02/2025 02:43 p. m. <DIR> umb-lazy
03/12/2024 04:42 p. m. <DIR> Users
24/02/2025 09:10 a. m. <DIR> Windows
10/07/2024 02:49 a. m. <DIR> xampp
1 archivos 12.288 bytes
17 dirs 292.815.392.768 bytes libres
C:\>cd proyecto-ecommerce
C:\proyecto-ecommerce>npm test
> test
> jest
(node:22708) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Please use a userland alternative instead.
(Use `node --trace-deprecation ...` to show where the warning was created)
PASS assets/tests/Producto.test.js (6.925 s)
✓ Debe mostrar el nombre del producto (40 ms)
Test Suites: 1 passed, 1 total
Tests: 1 passed, 1 total
Snapshots: 0 total
Time: 13.145 s
Ran all test suites.
C:\proyecto-ecommerce>