Informe página web

Rafael Patiño Diaz

1000049622

Universidad Manuela Beltrán

Diana Marcela Toquica Rodríguez

Ingeniería de software

19 de Abril de 2025

**1. Introducción**

Este informe detalla la implementación del diseño responsive en la tienda online, explicando las herramientas utilizadas, las pruebas realizadas y adjuntando capturas de pantalla de la adaptabilidad en distintos dispositivos desde las DevTools de Microsoft Edge.

**2. Implementación del Diseño**

Se han utilizado técnicas como CSS Grid, Flexbox y Media Queries para garantizar una correcta adaptabilidad a diferentes tamaños de pantalla.

**2.1 Uso de CSS Grid**

* Se empleó CSS Grid para estructurar la maquetación principal de la página.
* La clase. wrapper define una cuadrícula con grid-template-columns: 1fr 4fr;, asegurando una correcta distribución entre la barra lateral y el contenido principal.
* En dispositivos pequeños, se desactiva esta estructura y se cambia a un diseño en columna con display: flex; flex-direction: column;.

**2.2 Uso de Flexbox**

* Se ha utilizado en elementos como. menu-list y. producto para organizar los elementos de manera flexible.
* En. producto, se usa display: flex; flex-direction: column; para mantener la estructura ordenada.
* En. carro-acciones, se utiliza display: flex; justify-content: space-between; para distribuir los elementos correctamente.

**2.3 Uso de Media Queries**

* Se implementaron Media Queries para adaptar el diseño a distintos tamaños de pantalla:
* Para tablets (max-width: 900px): Se reduce la cantidad de columnas en .container-producto a grid-template-columns: 1fr 1fr 1fr;.
* Para móviles (max-width: 600px): Se modifica. wrapper para una estructura de columna y se hace el sidebar oculto por defecto.
* Para pantallas pequeñas (max-width: 400px): Se ajustan los elementos a grid-template-columns: 1fr; para mostrar productos en una sola columna.

**3. Implementación de JavaScript**

**3.1 app.js**

Este archivo gestiona la tienda en línea, organizando y mostrando los productos. Sus funciones principales incluyen:

* Cargar productos en la página y permitir filtrado por categoría.
* Agregar productos al carrito y almacenar la información en localStorage.
* Actualizar el número de productos en el carrito en tiempo real.

**3.2 compras.js**

Maneja el carrito de compras, asegurando que los usuarios puedan:

* Ver los productos agregados con sus respectivas cantidades.
* Modificar cantidades, eliminar productos o vaciar el carrito.
* Calcular el total de la compra y finalizar la transacción.
* Guardar los datos del carrito en localStorage para persistencia.

**3.3 menu.js**

Controla la interactividad del menú lateral:

* Abre y cierra el aside del menú cuando el usuario lo requiera.

Cierra automáticamente el menú cuando se selecciona una categoría de productos

**3.4 perfil.js**

Pagina del perfil del usuario donde se puede manejar la información personal

**3.5 auth.js**

Registro inicio de sesión:

* obliga al usuario a registrarse antes de iniciar su compra, lo que garantiza seguridad y controlo sobre.
* de momento guarda los datos en el localStorage.
* sirve para la seguridad del sitio.

**4. Pruebas Realizadas**

Para verificar la adaptabilidad del diseño, se realizaron las siguientes pruebas:

* Prueba en navegador: Uso de las herramientas de desarrollo (DevTools) en Microsoft Edge para simular distintos dispositivos.
* Prueba de interacción: Se verificó el funcionamiento del carrito de compras y la navegación del menú en distintos tamaños de pantalla.

**5. Optimización**

**5.1 Implementación Loading=”lazy”**

Se implemento loading a las imágenes, esto asegura que mientras el usuario no esté en la sección de la pagina donde se muestran las imágenes, estas no estén cargadas consumiendo menos recursos, ya que al fin y al cabo solo se van a mostrar cuando el usuario cargue esa sección de la página.

**<img class="producto-img" src="${productos.imagen}" alt="${productos.titulo} loading="lazy">**

Con esto nos aseguramos de añadirle elloading="lazy**"** a cada imagen

**5.2 Cambio de formato de Jpg a WebP**

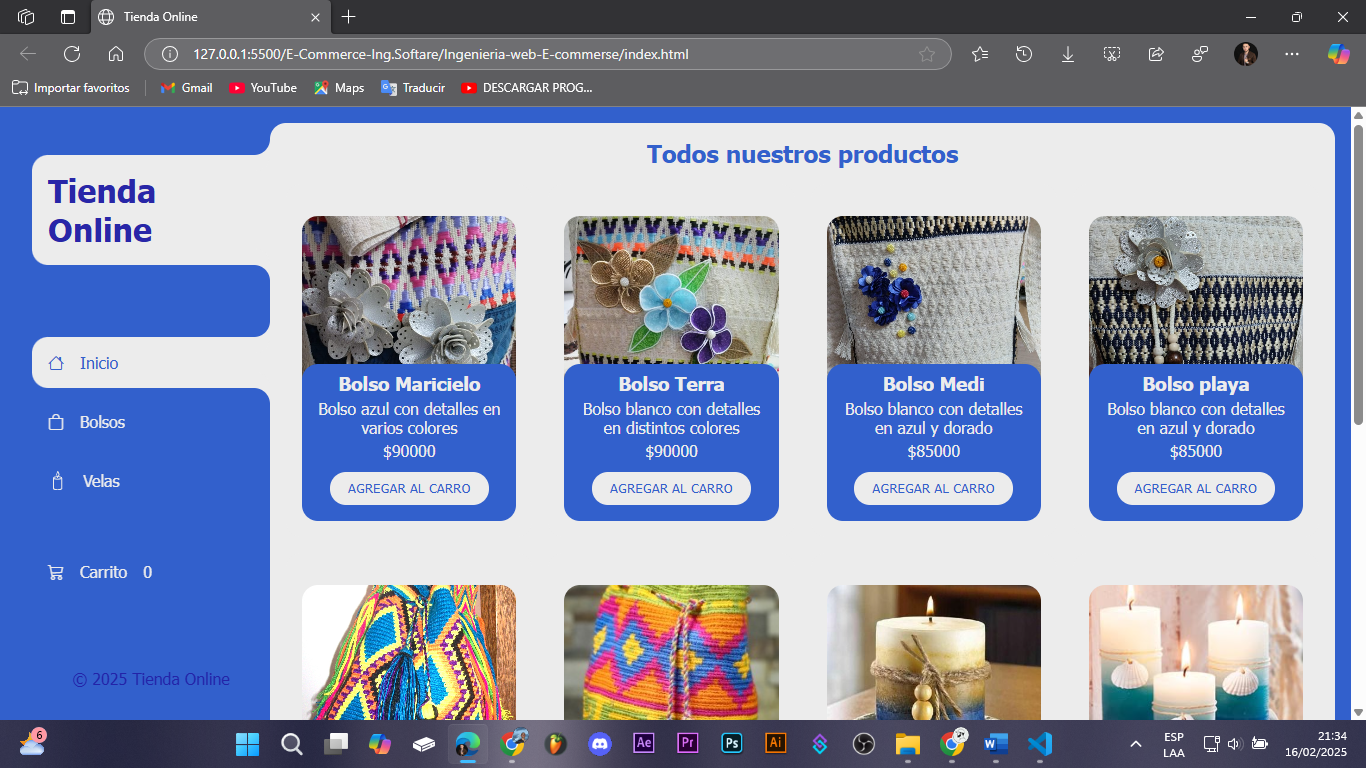
Mediante el uso de la aplicación Web llamada [**Squoosh**](https://squoosh.app/) se comprimieron las imágenes por lo que se redujo el tamaño de las imágenes para optimizar así los recursos que consumen estas.

**5.3 Pruebas con PageSpeed Insights**

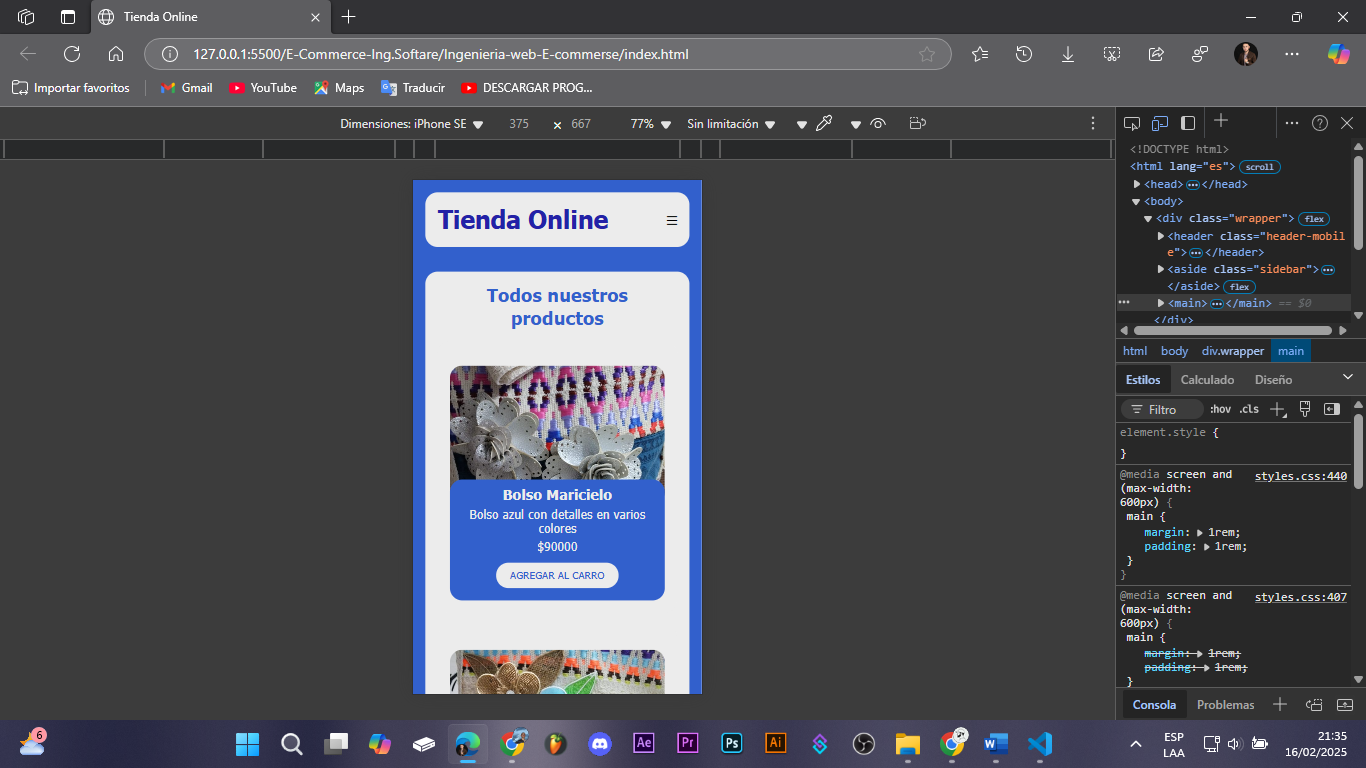
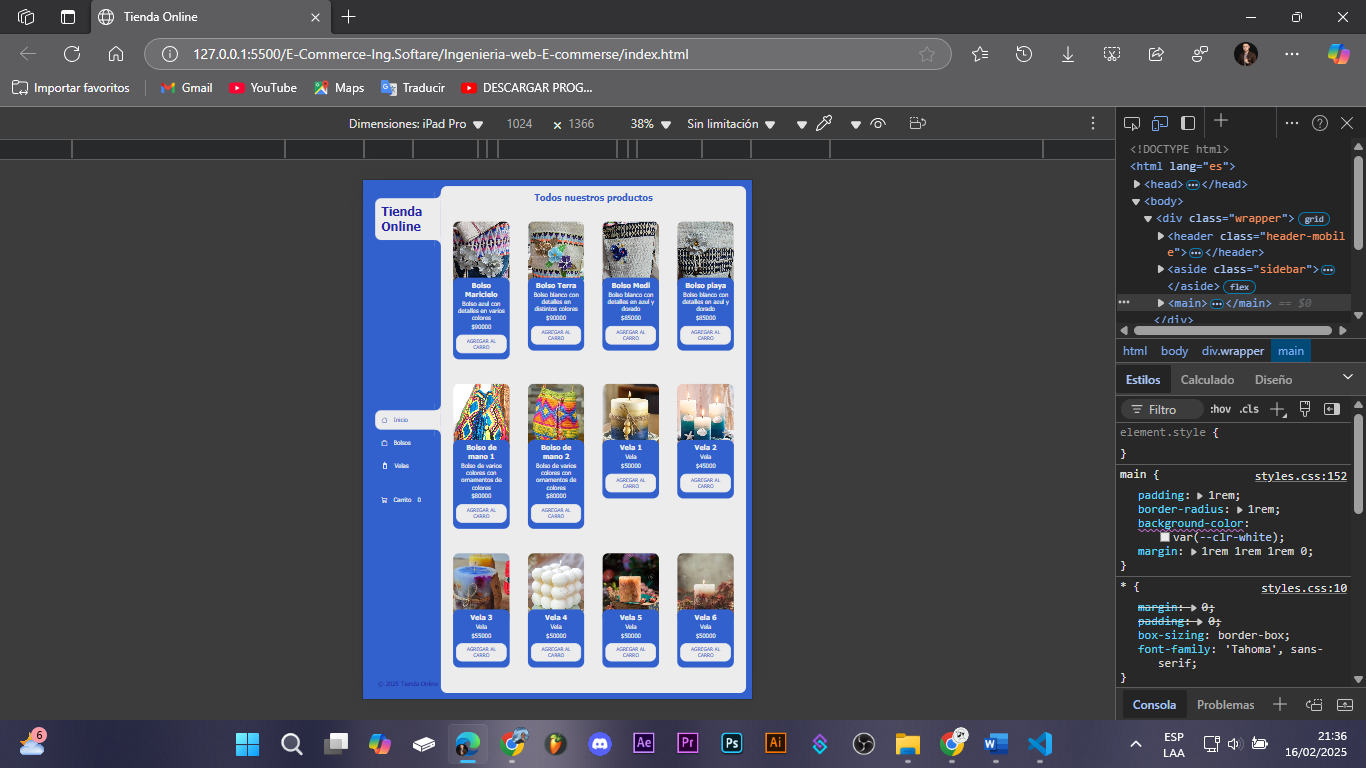
Usando la extensión[**PageSpeed Insights**](https://pagespeed.web.dev/?hl=es)se realizaron pruebas antes de la optimización optimización obteniendo antes un rendimiento en edge de 99 y en chrome 88 esto en desktop. Luego en movile en edge de 95 en edge y 94 en chrome

**6. Capturas de Pantalla**

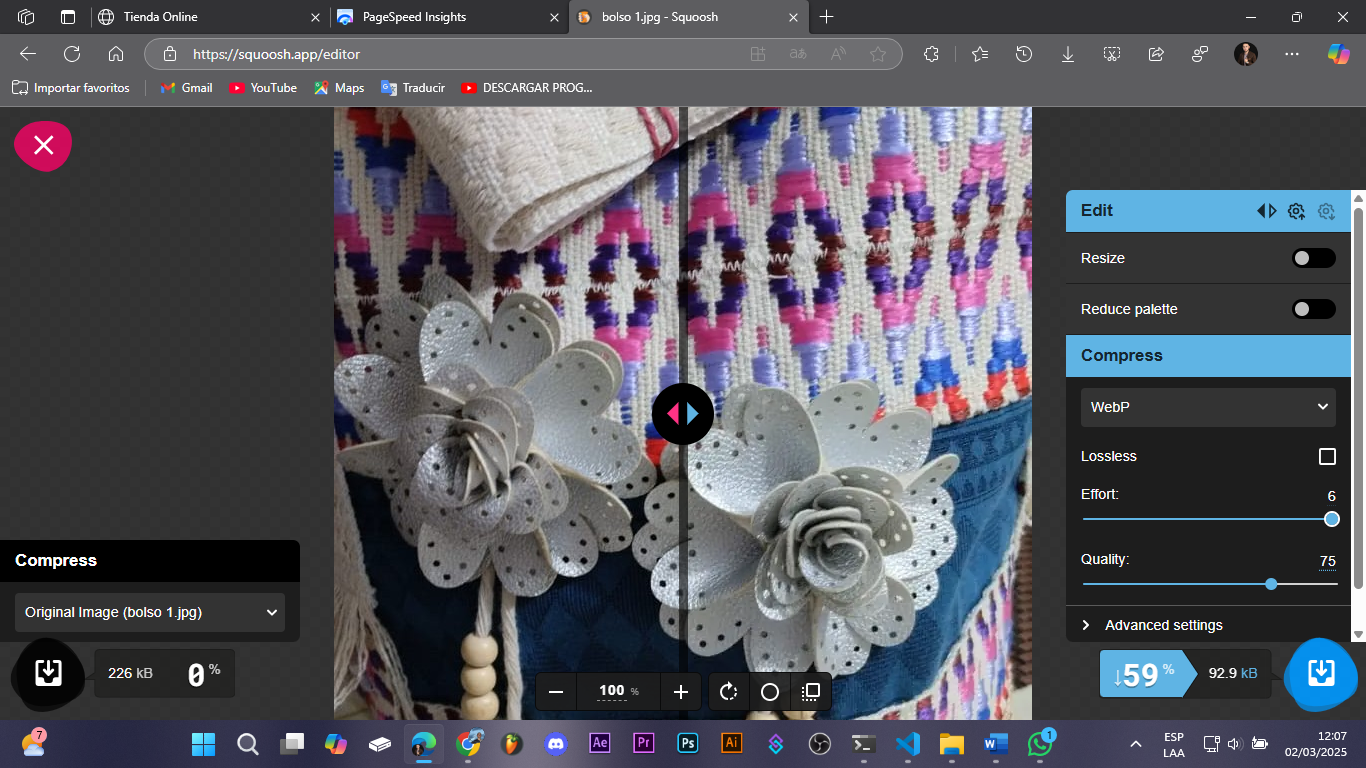
Vista en escritorio:



Vista en tableta:

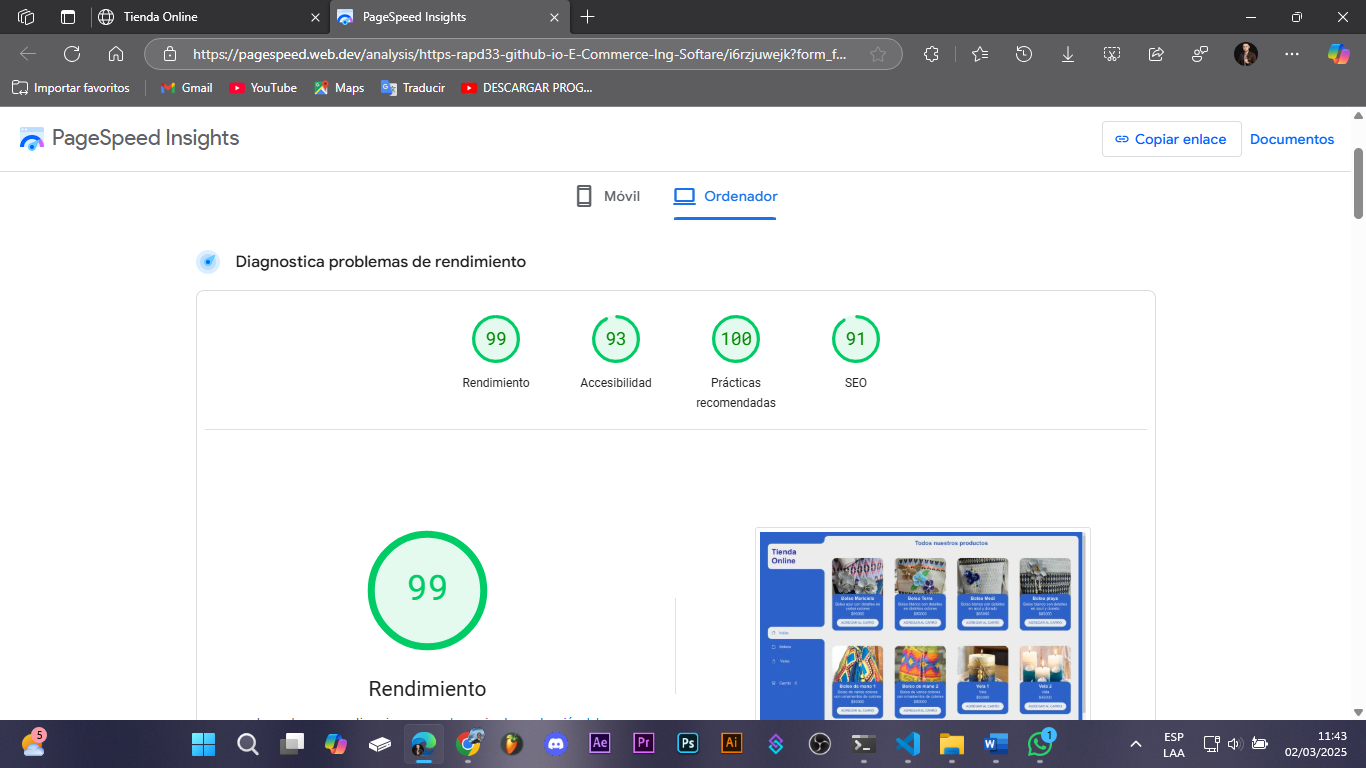


Compresión en la herramienta Squuosh:

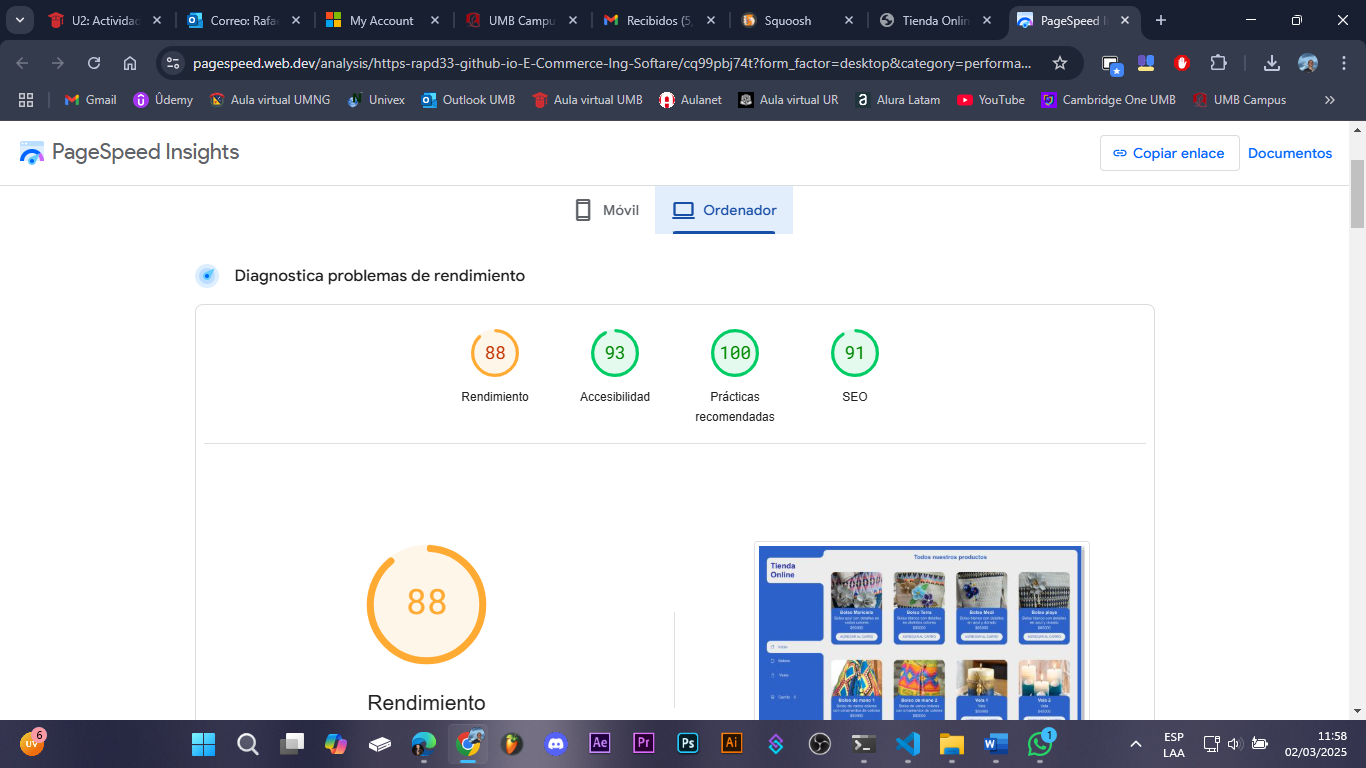


Pruebas PageSpeed Insights antes de la optimización completa (solo con el lazy loading):

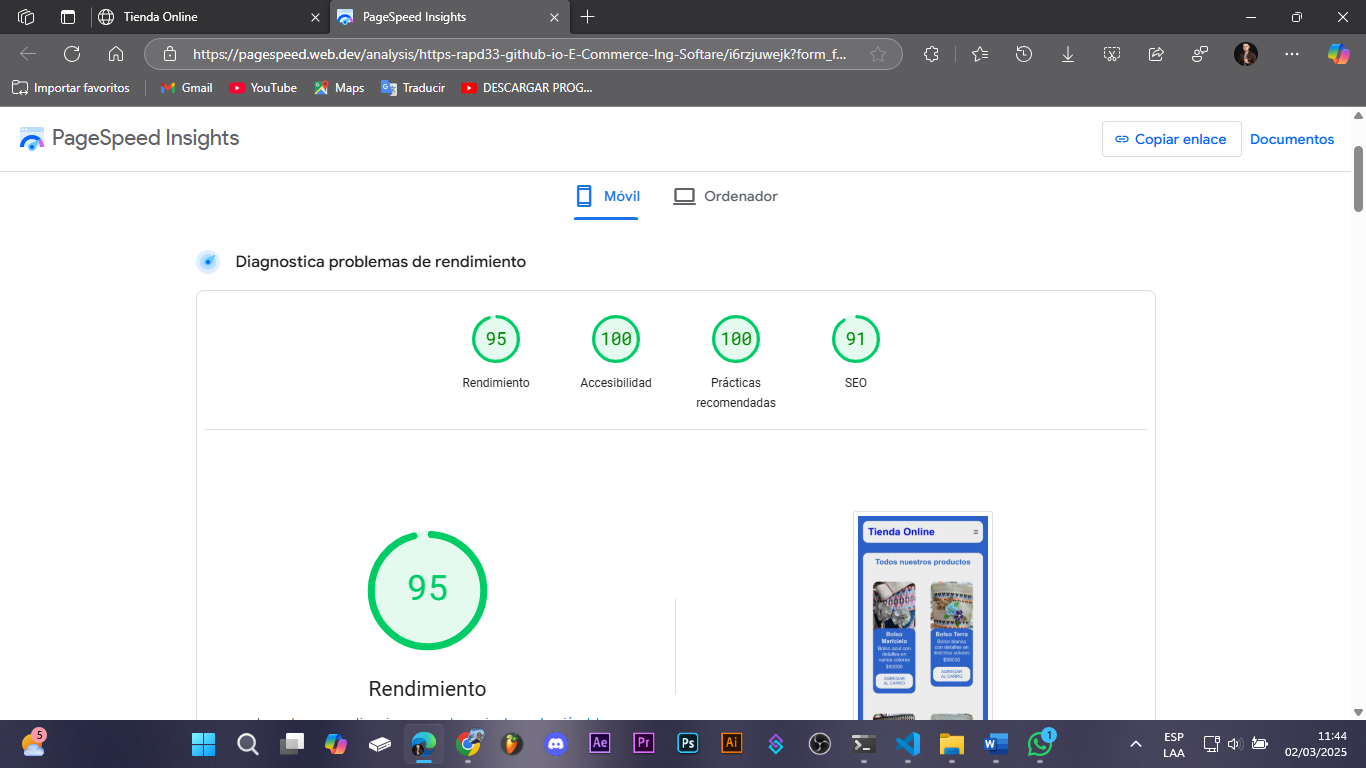
Desktop Edge:



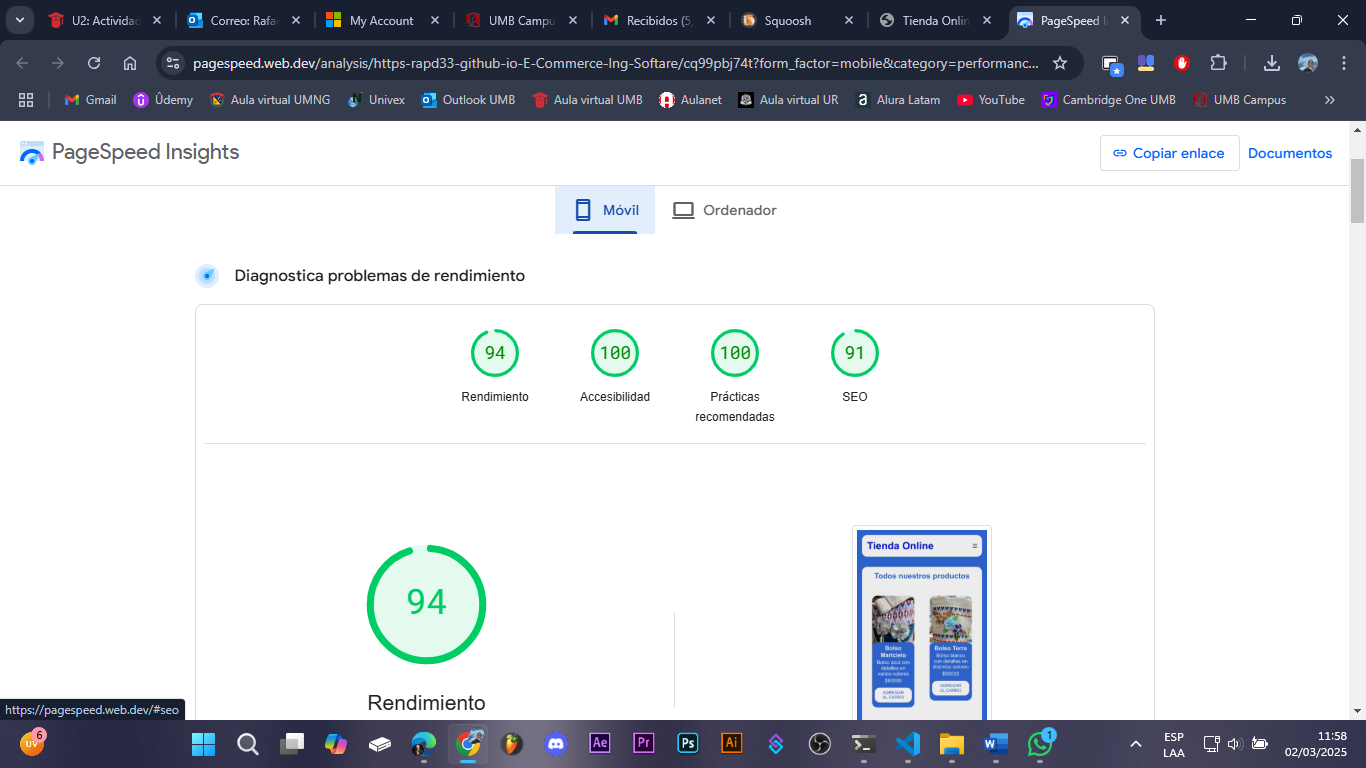
Desktop Chrome:



Mobile Edge:

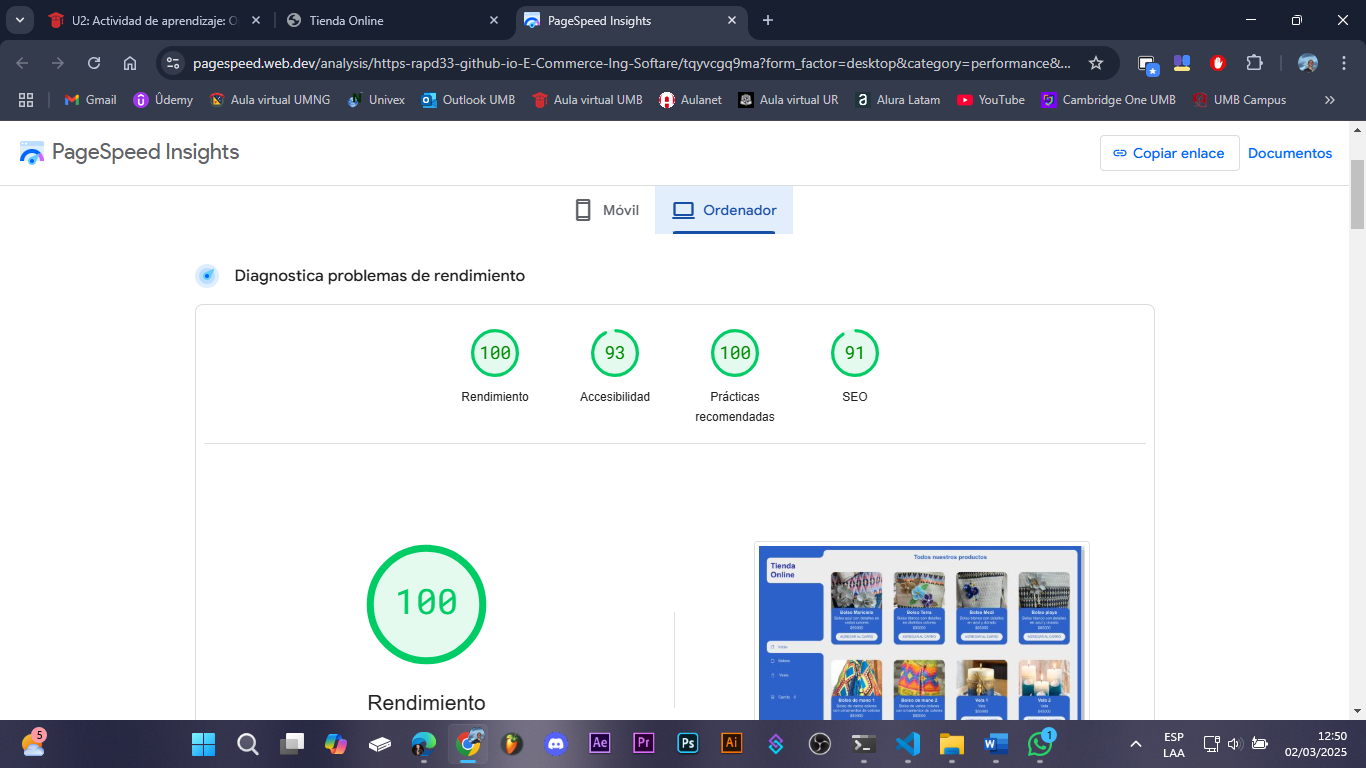


Mobile Chrome:

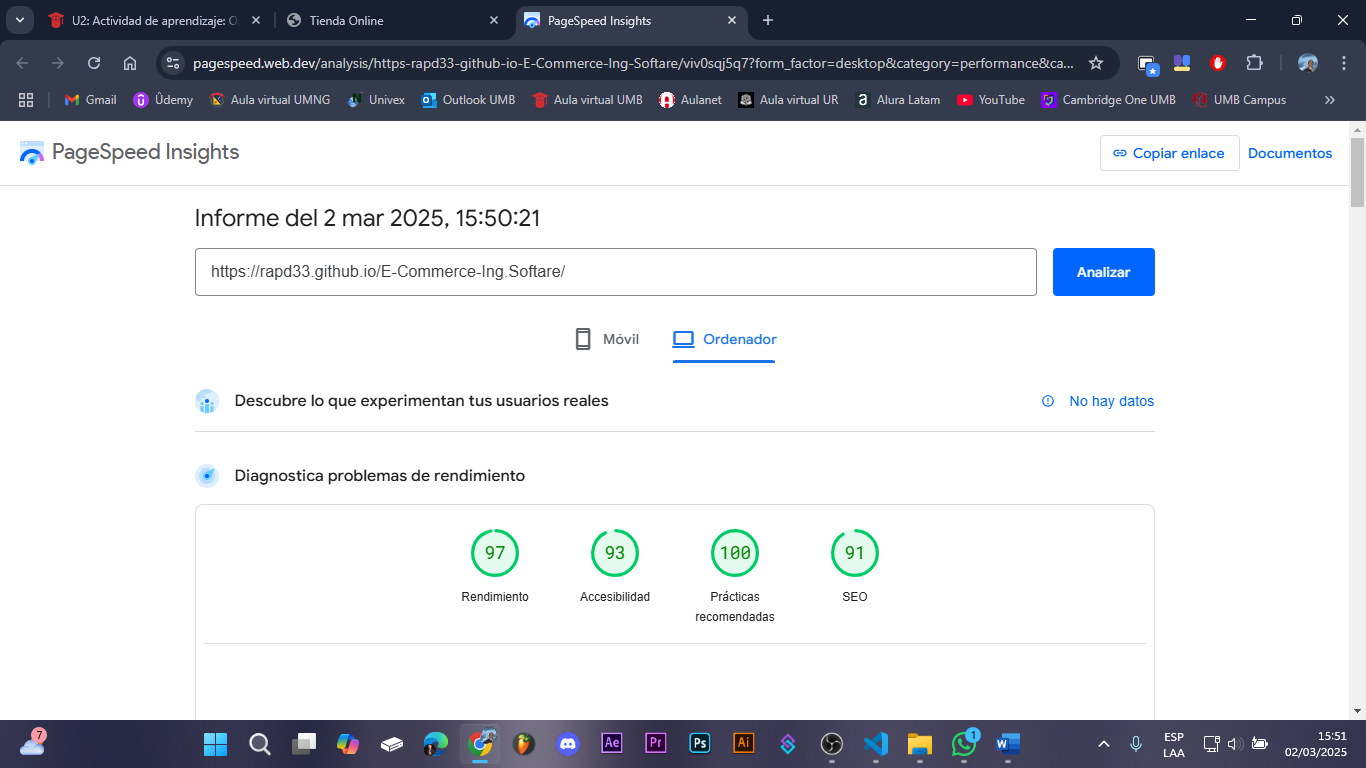


Pruebas PageSpeed Insights después:

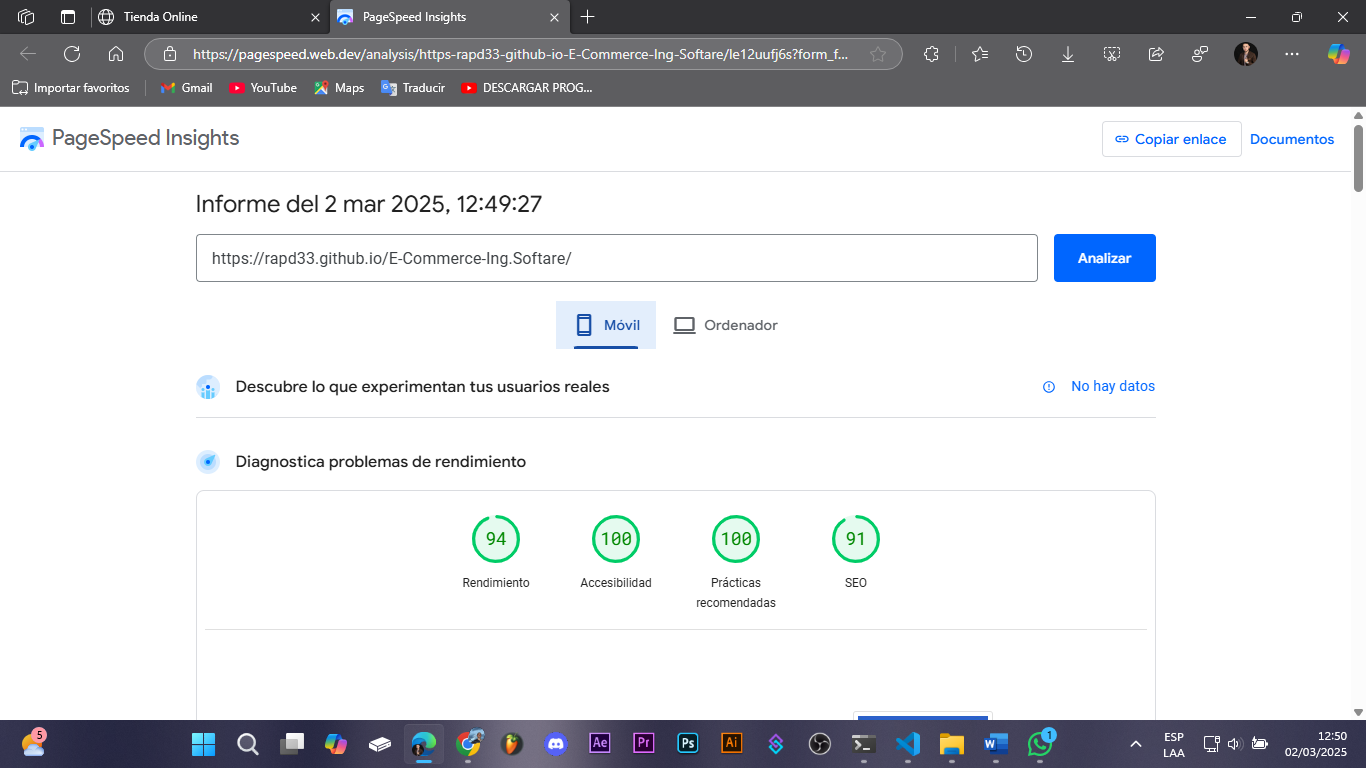
Desktop Edge:



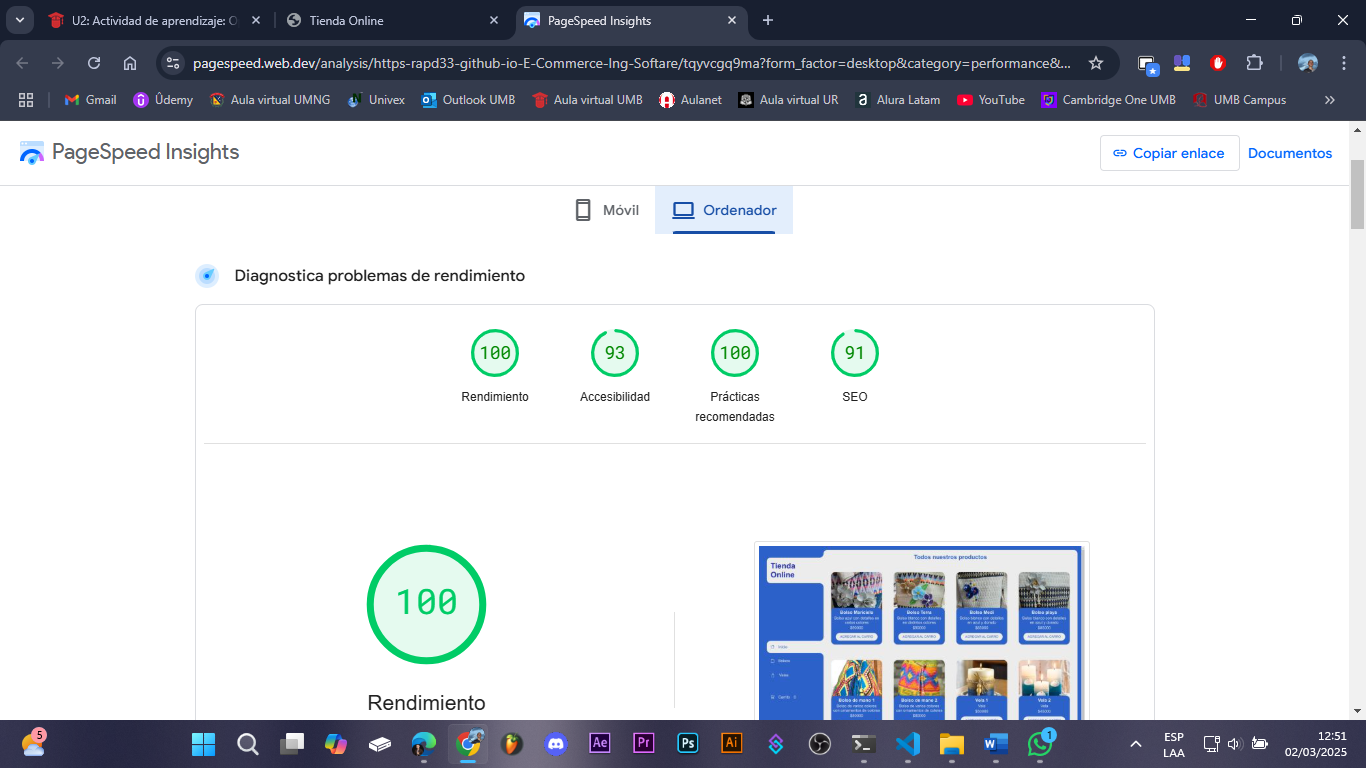
Desktop Chrome:



Mobile Edge:



Mobile Chrome:



**6. Conclusión**

La implementación de JavaScript permitió mejorar la funcionalidad de la tienda en línea, brindando una experiencia interactiva y dinámica para los usuarios. Además, través del diseño responsive con CSS Grid, Flexbox y Media Queries, aseguramos que la tienda se adapte correctamente a distintos dispositivos, facilitando la navegación y la interacción. Mediante métodos de validación, como lo es el registro e inicio de sesión, aseguramos que la página sea más segura y que el E-commerce sea más seguro. Las pruebas realizadas confirman que la tienda ofrece una experiencia fluida y eficiente en móviles, tabletas y escritorios, garantizando accesibilidad y usabilidad en cualquier entorno digital.