Proyecto de Investigación Formativa

**Título**

Tienda eCommerce con React

**Materias Integradas**

Fundamentos de la Investigación

**Autores**

Devin Fabricio Llerena Ballesteros

César Moisés Palma Quiñones

Febrero, 2023

**Proyecto de Investigación Formativa**

**Título:** Tienda eCommerce con React

César Palma Devin Llerena

**Programación Estructurada y Funcional Fundamento de la investigación,** PUCESE

Instructor: Mgt. Kléber Posligua Flores.

**Resumen.** El proyecto consiste en desarrollar una e-Commerce basada en el editor de texto Visual Studio Code, el lenguaje de programación JavaScript, Sanity Studio, Node.js, Stripe y GitHub. El objetivo principal del proyecto es crear una plataforma de comercio electrónico segura y confiable que permita a los usuarios realizar compras en línea de forma sencilla y segura. Esto se logra mediante el uso de código JavaScript para la creación de la interfaz de usuario, el uso de Node.js para la gestión de la información dinámica en la plataforma, Stripe para las transacciones de pago seguras, y GitHub para el control de versiones y el almacenamiento de los archivos del proyecto. El resultado es una plataforma de comercio electrónico moderna con una interfaz de usuario amigable, soluciones de seguridad eficaces, y una plataforma de pagos seguros para garantizar la satisfacción del usuario.

**Objetivos.**

Desarrollar una página de eCommerce con diseño atractivo y fácil de usar para los clientes que naveguen en la página.

Implementar funcionalidades avanzadas para que el sitio web sea más fácil de utilizar y puedan tener una buena experiencia al realizar compras online.

Integrar nuevas herramientas de marketing para atraer a nuevos clientes y aumentar la tasa de conversión.

1. **Fundamentación Teórica.**

**Concepto de eCommerce**

Para comenzar, el eCommerce es el proceso de compra y venta de bienes y servicios a través de Internet, donde los pagos y cobros se realizan electrónicamente. La creación de una tienda online implica la publicación de un catálogo de productos y la implementación de un sistema de pago que permita a los clientes hacer pedidos y pagar de manera segura. Además, en este sentido, el eCommerce puede ser una excelente oportunidad para las empresas que buscan expandir su alcance y llegar a un público más amplio a través de la web. [1]

**Ventajas principales de una página eCommerce:**

1. Siempre está abierto las 24 horas del día.
2. Tiene un alcance global y esto permitirá llegar a clientes de cualquier parte del mundo.
3. Su creación y mantenimiento tiene menores costos, debido a que es una tienda en línea y adicionando, no se requiere de tener un alquiler o comprar un espacio comercial.
4. Se pueden recopilar datos y realizar análisis detallados sobre el comportamiento de los clientes, tales como sus gustos, necesidades y/o preferencias. Esto puede ayudar a mejorar la estrategia de marketing y aumentar las ventas.

**Desventajas principales de una página eCommerce:**

1. Debido a la facilidad de crear una tienda en línea, la competencia puede ser mucho más fuerte que en una tienda física.
2. Aunque la creación de una tienda en línea puede ser más económica que una tienda física, aún existen costos iniciales significativos, como el desarrollo del sitio web y la implementación de un sistema de pago seguro.
3. La administración de una tienda en línea requiere habilidades técnicas y conocimientos, lo que puede representar un desafío para algunos propietarios de negocios.
4. A diferencia de una tienda física, una tienda en línea no permite la interacción personal con los clientes. Esto puede resultar en una falta de confianza y una disminución en la lealtad del cliente.

## JavaScript y Visual Studio Code

Después, se inició una investigación acerca de las herramientas que se necesitarán para crear la aplicación de la Tienda eCommerce., para ello se utilizó el lenguaje de programación JavaScript y el editor de texto Visual Studio Code. Agregando lo anterior, JavaScript es un lenguaje de programación que se utiliza en el desarrollo de páginas web para tareas como agregar interactividad, cambiar automáticamente la fecha de una página, también una página web puede ser programada para mostrar una ventana emergente cuando un usuario hace clic en un enlace específico. También es posible cambiar el texto o la imagen de un sitio web cuando el usuario mueve el ratón sobre ellos. Estas son características de interactividad que pueden ser implementadas en una página web utilizando programación en lenguajes como JavaScript. Asimismo, es compatible con todos los navegadores modernos sin necesidad de plugins y es una de las tres tecnologías esenciales en la producción mundial de sitios web, junto con HTML y CSS. Aunque tiene algunas similitudes con Java, no están relacionados y tienen semánticas muy diferentes. [2]

## Node.js

Node.js es un entorno de tiempo de ejecución de JavaScript que permite la ejecución de programas escritos en JavaScript como aplicaciones independientes en ordenadores. Su creación por los desarrolladores originales de JavaScript permite extender la capacidad de JavaScript más allá de los límites del navegador, lo que proporciona ventajas en la resolución de problemas y un mayor potencial de programación. El motor de tiempo de ejecución JavaScript V8 es utilizado por tanto por JavaScript como por Node.js para la conversión de código JavaScript a un código de máquina más rápido, optimizando así la velocidad de ejecución.

Node.js se destaca como una herramienta de programación de gran utilidad para el desarrollo de sitios web interactivos y para la creación de soluciones que anteriormente requerían lenguajes de secuencias de comandos como Python. Adquirir las bases, conceptos y habilidades necesarias en Node.js mediante la realización de un curso es altamente recomendado para profundizar en sus opciones y aprovechar al máximo su potencial en la programación. [3]

**ReactJS**

Con respecto a React, es una biblioteca de código abierto basada en JavaScript, desarrollada por Facebook, que se utiliza para la creación de interfaces de usuario. Esta biblioteca se basa en un modelo de programación declarativo que utiliza una eficiente manipulación del DOM para actualizar la interfaz de usuario.

ReactJS se utiliza principalmente para la creación de aplicaciones de página única (SPA) y es compatible con la creación de aplicaciones móviles. Asimismo, cuenta con un ecosistema de módulos y componentes que permiten a los desarrolladores abordar problemas más complejos de manera más rápida y efectiva.

Una de las principales ventajas de ReactJS es la mejora de la velocidad y eficiencia en el desarrollo de interfaces de usuario, gracias a la reducción de complejidad del código y a la mejora en la escalabilidad. Además, ReactJS ofrece una excelente capacidad de reutilización de código, permitiendo a los desarrolladores crear componentes personalizados y reutilizarlos en diferentes proyectos. [4]

**Sanity**

Sanity es una herramienta de gestión de contenidos (CMS) que puede ser utilizada tanto interna como externamente por los clientes. Con la capacidad de construir datos y editar imágenes directamente en la interfaz, Sanity reduce significativamente el tiempo dedicado al diseño de imágenes.

En cuanto a los valores de texto, la herramienta permite aplicar estilo o utilizar un cuadro de texto simple, lo que restringe el estilo. Estos cambios pueden ser implementados a través de esquemas en la aplicación Next.js, todo dentro de la interfaz llamada Sanity Studio.

Al igual que cualquier base de datos, la aplicación debe tener la capacidad de traducir los datos del backend (parte de la aplicación que se ejecuta en el servidor y es responsable del almacenamiento y procesamiento de datos) al frontend (parte de la aplicación que el usuario ve y utiliza directamente). Es aquí donde el lenguaje de consulta (GROQ) resulta útil. GROQ es un lenguaje declarativo diseñado para consultar colecciones de documentos JSON en gran medida sin esquema. Sus principales objetivos son el filtrado expresivo, la unión de varios documentos en una sola respuesta y la configuración de la respuesta para que se ajuste a la aplicación del cliente. [5]

**Stripe**

Stripe es un sistema de pago en línea que se puede integrar directamente en una tienda en línea para permitir que los clientes realicen pagos de forma segura y sencilla. Los comerciantes solo necesitan crear una cuenta gratuita en Stripe y vincular su cuenta bancaria. La integración en la web es fácil y se realiza a través de un formulario de pago seguro que no requiere grandes conocimientos técnicos. Desde el dashboard de Stripe, los comerciantes pueden gestionar los pagos recibidos, facturas, devoluciones y disputas, entre otras funciones. Además, Stripe cuenta con un modo de prueba para aprender a utilizar la plataforma antes de utilizarla en la tienda en línea. [6]

1. **Metodología**

En este proyecto se emplearon los siguientes materiales en su metodología:

* Editor de texto Visual Studio Code
* Lenguaje de programación JavaScript
* Sanity Studio
* Node.js
* Stripe
* GitHub

Lo primero que se realizó fue investigar diferentes proyectos de e-Commerce que posteriormente fue utilizado como muestra de este proyecto.

Se inicializa una aplicación usando Next.js con el comando ‘npx create-next-app’, se utiliza para crear una estructura de proyecto predeterminada y lista para usar con Next.js, incluyendo configuraciones de servidor y cliente, enrutamiento y herramientas de construcción preconfiguradas. Se examinó la estructura de carpetas creada por Next.js y se pudo observar que la carpeta "pages" es de gran importancia en la organización del proyecto. Esta carpeta es el lugar donde se crean los componentes y páginas de la aplicación. Es decir, cada archivo que se cree en esta carpeta se convierte en una página accesible mediante una URL única en el sitio web. Además, en la carpeta "pages" se pudieron crear subcarpetas para organizar las páginas por secciones o categorías. Esta estructura de carpetas permitió una fácil navegación y un manejo ordenado de los archivos, lo que facilitó el mantenimiento del proyecto a medida que creció en tamaño y complejidad.

Además de instalar las dependencias necesarias para el proyecto a través de un archivo de GitHub que se proporcionó en la descripción del tutorial. Este archivo de GitHub contuvo un listado de dependencias y sus respectivas versiones, que son necesarias para que el proyecto funcione correctamente. Al utilizar este archivo para instalar las dependencias, se asegura que todas las versiones sean compatibles y evita posibles conflictos entre ellas. Además, este método de instalación también permitió una configuración más rápida y sencilla del entorno de desarrollo del proyecto.

Además, se sugirió ejecutar "npm run dev" para iniciar la aplicación en el puerto 3000. Al ejecutar este comando, se inició el servidor de desarrollo de Next.js, que monitorea los cambios en los archivos y actualiza automáticamente la página web. Esto permitió visualizar los cambios en tiempo real mientras se trabajó en la aplicación. Asimismo, se procedió a abrir el enlace de Sanity en la descripción del tutorial y se realizó el registro para obtener una cuenta gratuita. Luego, se copió el comando proporcionado en la página de Sanity y se lo pegó en la terminal para instalar Sanity CLI globalmente.

Después de instalar Sanity CLI, se indicó que se debía iniciar sesión en la cuenta de Sanity usando Google. De esta manera, se pudo acceder a la consola de administración de Sanity y comenzar a crear un conjunto de datos para el proyecto. Finalmente, se eligió una plantilla de proyecto en Sanity de comercio electrónico.

Se ejecutó el comando "sanity start" para iniciar el "content studio" en localhost:3333. El "content studio" es una interfaz de usuario en la que se pueden administrar los datos del proyecto creado en Sanity. Al ejecutar el comando "sanity start", se inició el servidor local del "content studio", lo que permitió acceder a la interfaz en el navegador web. Además, se creó un archivo nuevo dentro de la carpeta "schemas" con el nombre "product.js". La carpeta "schemas" es una carpeta en el proyecto de Sanity que contiene los esquemas de los tipos de contenido que se utilizarán en el proyecto. Estos esquemas definen la estructura de los datos y los campos que se utilizarán en el proyecto. Al crear un archivo nuevo dentro de esta carpeta con el nombre "product.js", se está creando un nuevo tipo de contenido que se utilizó para almacenar información sobre los productos en el proyecto. Este archivo se puede editar posteriormente para ajustar la estructura de datos del tipo de contenido y agregar campos personalizados según sea necesario.



Objeto 1

Se creó una estructura JSX básica utilizando React, que incluyó un componente de banner principal, una sección de "productos más vendidos" y un bucle que finalmente mostró productos reales. Para crear la estructura JSX, se definió un componente principal en el archivo "index.js" dentro de la carpeta "pages" en el proyecto de Next.js. Este componente incluyó el componente de banner principal y la sección de "productos más vendidos". Además, se incluyó un bucle que recorrería los productos reales y mostraría la información correspondiente en la página. La sección de "productos más vendidos" y el bucle se creó utilizando la sintaxis de JSX para definir componentes secundarios y lógica de programación en JavaScript. Los productos se almacenaron en un archivo separado de datos dentro del proyecto, y se importaron a la página para su uso en el bucle.



Objeto 2

Después, se aplicaron nombres de clase a elementos HTML en una página utilizando React y CSS, con el objetivo de definir estilos CSS específicos para esos elementos. Los nombres de clase se aplicaron a través de la sintaxis JSX en React, y se definieron estilos CSS correspondientes en un archivo CSS global. Al aplicar estilos a elementos específicos mediante nombres de clase, se logra personalizar la apariencia visual de una página web de manera efectiva y consistente en todo el proyecto.

En el proyecto se creó un archivo llamado "index.js" dentro de la carpeta de componentes, desde donde se exportaron todos los componentes para su uso en las páginas del proyecto. Esto permitió reutilizar componentes en diferentes partes del proyecto y mantener un código más limpio y organizado.

Además, se importaron los componentes necesarios en la página "\_app.js" y se reemplazó el texto estático con componentes dinámicos. De esta manera, se logró generar contenido dinámico en la página web utilizando React y JSX, lo que permitió una mayor flexibilidad en la presentación de la información y un mejor rendimiento. También se evitó tener que repetir el mismo código en varias partes del proyecto, lo que resulta en un código más eficiente y fácil de mantener.

También se implementó el componente "HeroBanner", el cual es un componente React que representa una sección de la página con un banner principal. Se agregaron las clases CSS necesarias y se definió el texto básico que se mostrará en el banner.

Luego, se creó una función asíncrona llamada "getServerSideProps", la cual utiliza la constante "sanityClient" para obtener los datos de los productos y los datos del banner. Esta función es específica de Next.js y se utiliza para generar contenido dinámico en el servidor antes de enviar la página al cliente.

Al utilizar "getServerSideProps" en el proyecto, se logra una mayor eficiencia en la carga de la página, ya que se obtienen los datos necesarios antes de enviar la página al cliente. Además, al utilizar "sanityClient" para obtener los datos de los productos y los datos del banner, se asegura que la información sea precisa y actualizada.

También, creó un archivo para el componente "Layout", el cual contiene la estructura general de la página, incluyendo la etiqueta "head". Esto permite definir los elementos que aparecerán en la sección "head" de la página y su configuración.

Además, se creó un componente de navegación con un enlace a la página principal y un icono de carrito de compras. Este componente es reutilizable y puede ser utilizado en varias páginas del proyecto. La inclusión de un componente de navegación facilita la navegación del usuario y mejora la experiencia de usuario.

De paso, se llevó a cabo una serie de pasos, como la creación de componentes, funciones asíncronas y archivos necesarios para la estructura general de la página. También se solucionaron errores y se agregaron secciones de detalle del producto, revisión y precio, así como una sección de imágenes en miniatura con funcionalidad de selección al pasar el ratón por encima. Todo esto se logró mediante el uso de Next.js y herramientas como la constante sanity client.

Para permitir a los usuarios seleccionar una imagen en miniatura en la página de detalles del producto, se agregó un estado a la aplicación para manejar la imagen seleccionada. Se importó la función useState de React y se creó un nuevo estado const index y setIndex con un valor inicial de cero. Luego, se reemplazó el valor estático de cero en la sección de código correspondiente con el estado index.

Se creó un div de carrusel para mostrar las imágenes en miniatura y se estableció una función onmouseenter con un callback que establecía el valor de index al índice de la imagen seleccionada. Se proporcionó un nombre de clase dinámico para las imágenes en miniatura basado en el valor de index para resaltar visualmente la imagen seleccionada.

Se necesitó agregar la funcionalidad de agregar productos al carrito y ver la cantidad de productos en el mismo. Para esto se creó un estado "cart items" vacío que se llenaría con los datos de almacenamiento local y se creó un proveedor de contexto para compartir los valores del estado entre los componentes. En la página de detalles del producto se implementaron funciones para incrementar, disminuir y agregar productos al carrito usando botones específicos. De esta manera, los usuarios podían agregar productos a su carrito y ver la cantidad de productos en el mismo.

En este proyecto, se necesitó agregar la funcionalidad de actualizar la cantidad y el precio de los artículos en el carrito de compras. Para hacerlo, se actualizaron los estados del total de precios y la cantidad total.

Cuando se agregaba un producto al carrito, se verificaba si ya existía en el carrito. Si era así, se actualizaba la cantidad del producto y se mostraba una notificación de éxito. Si no existía, se agregaba el producto con la cantidad especificada y se actualizaban los estados del total de precios y la cantidad total.

Se agregó la función "onAdd" al botón de "Agregar al carrito" y se implementó el componente del carrito en la barra de navegación. Se utilizó el estado global "show cart" para mostrar u ocultar el carrito en función de si se había hecho clic en el botón de "Ver carrito" o no.

Se implementó el componente del carrito, importando los elementos necesarios, como los iconos y la función "toast" de React. Se utilizó la función "map" para recorrer todos los elementos del carrito y mostrarlos en la interfaz de usuario. Se agregaron botones para aumentar o disminuir la cantidad de cada elemento en el carrito y para eliminar el elemento del carrito. Se actualizaron los estados del total de precios y la cantidad total cuando se cambiaba la cantidad de un elemento o se eliminaba un elemento del carrito.

Con estas funcionalidades implementadas, los usuarios podían actualizar la cantidad y el precio de los artículos en su carrito de compras, y ver la cantidad total y el precio total de los productos en su carrito en todo momento.

Se creó un componente de carrito de compras que se muestra cuando se hace clic en un botón en la barra de navegación. El botón tiene una propiedad llamada "show cart", que se inicializa en "false" y se utiliza para mostrar u ocultar el carrito.

Si hay uno o más productos en el carrito, se utiliza el método "map" para generar un elemento "div" para cada producto en el carrito. Para cada producto, se muestra su nombre, cantidad, precio y una opción para eliminar el producto del carrito.

Se añadió la funcionalidad para actualizar la cantidad de un producto en el carrito al hacer clic en los botones "+" y "-" que se muestran junto a cada producto. Sin embargo, se encontró un problema con la identificación del producto al que se le debe actualizar la cantidad debido al alcance de la función. Este problema se dejó pendiente para una futura implementación.

Para implementar la funcionalidad de eliminación de elementos del carrito, se agregó un botón de "Eliminar" para cada producto en el carrito. Al hacer clic en el botón de "Eliminar", se ejecuta una función que actualiza el estado global de "cart items", eliminando el producto correspondiente. También se actualizan los estados de la cantidad total y el precio total del carrito para reflejar la eliminación del producto.

Para hacer esto, se utilizó el método "filter" de JavaScript para crear una nueva matriz de elementos en el carrito que excluya el elemento que se está eliminando. Luego, se actualizó el estado global de "cart items" con la nueva matriz. Se actualizaron también los estados de la cantidad total y el precio total del carrito para reflejar la eliminación del producto.

Con esto, se logra implementar la funcionalidad de eliminación del elemento en el contexto del estado y se decrementa el precio y la cantidad total de producto correspondientes al elemento eliminado.

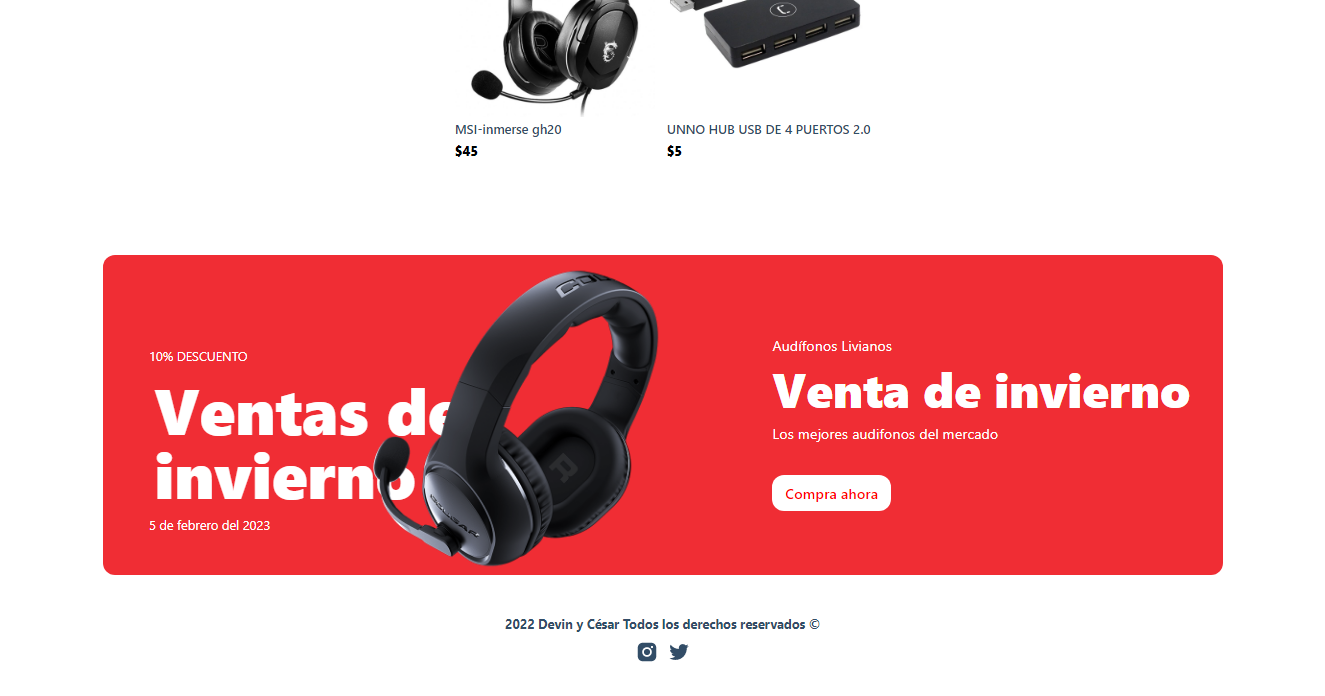
**Resultados.**

Estos fueron los siguientes resultados que se obtuvo en nuestra página de e-Commerce:

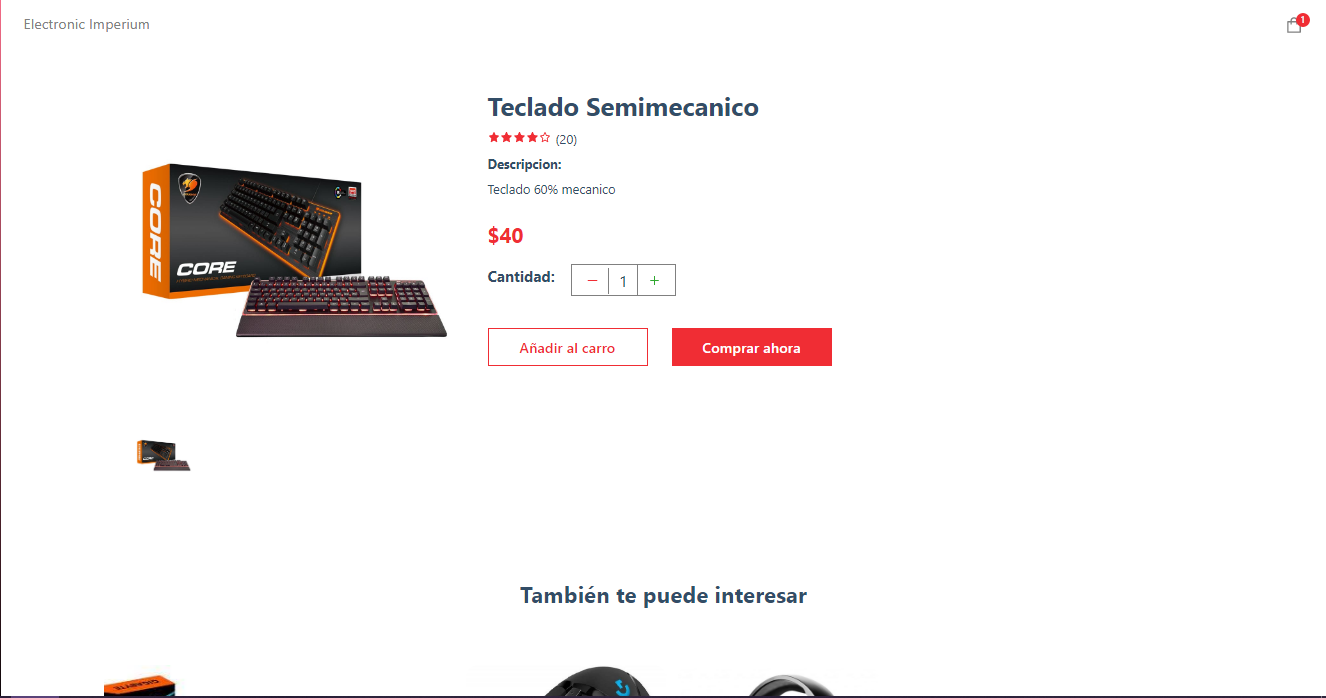
Las siguientes imágenes son una vista general cuando se ingresa a nuestra página de e-Commerce.



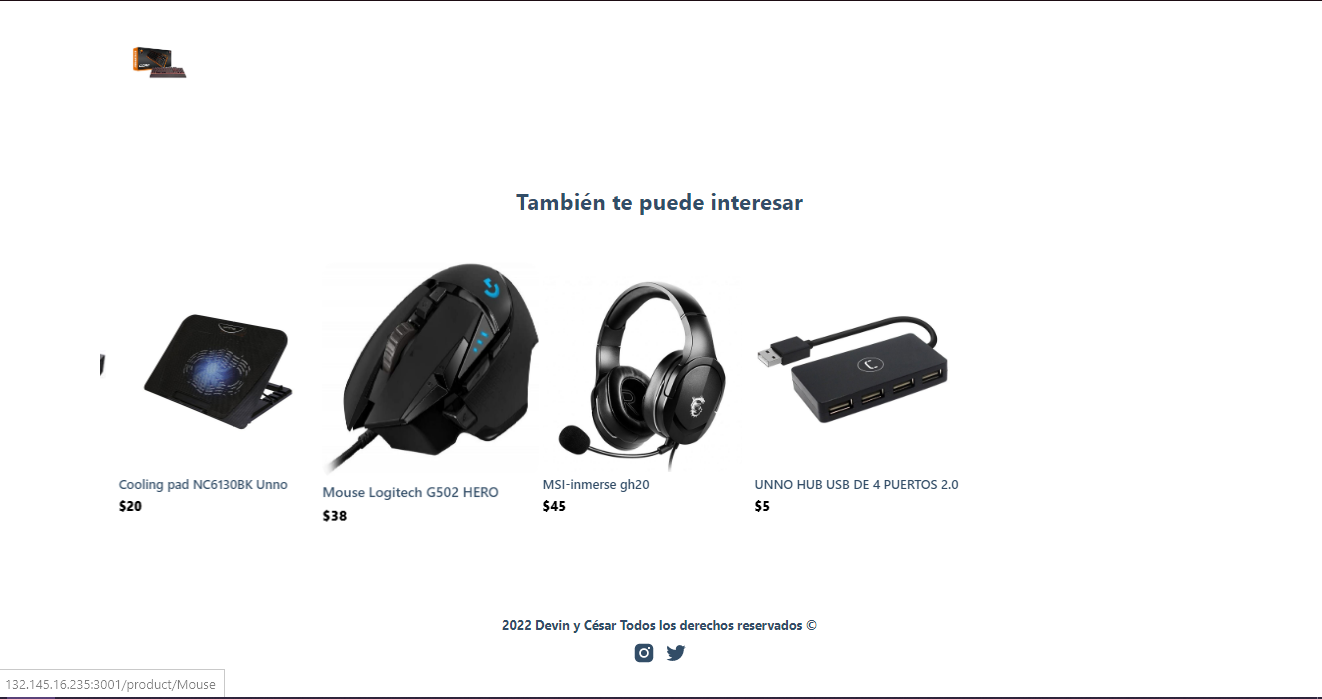




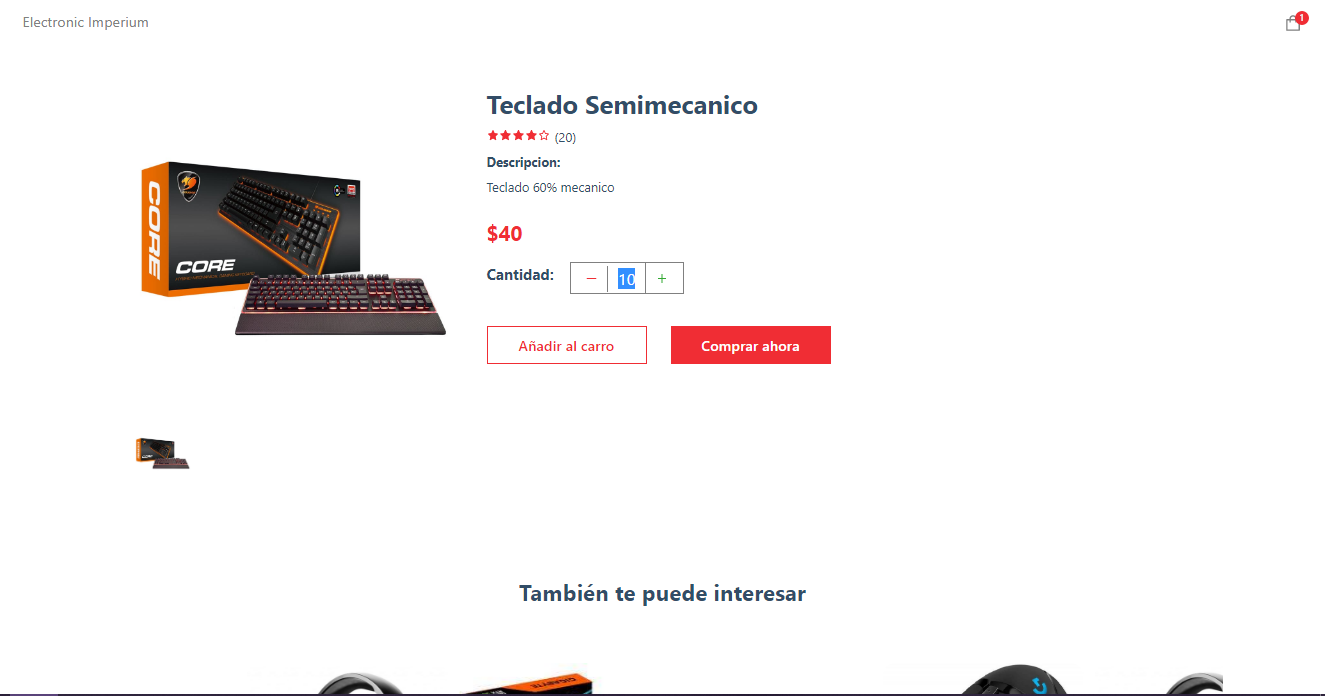
La siguiente captura de pantalla fue tomada cuando se selecciona algún producto, este apartado será la misma interfaz para los demás productos.



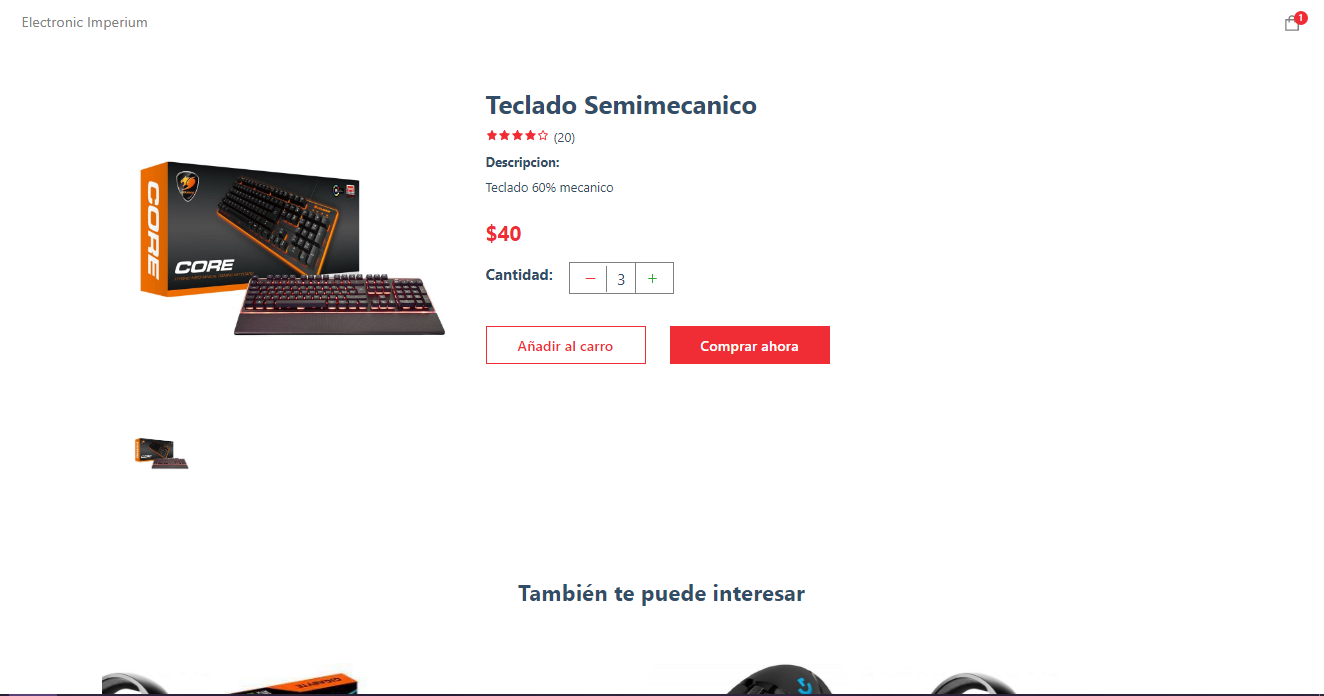
En esta imagen, se puede apreciar de que hay una animación corrida mostrando la lista de productos que contiene nuestra tienda e-Commerce



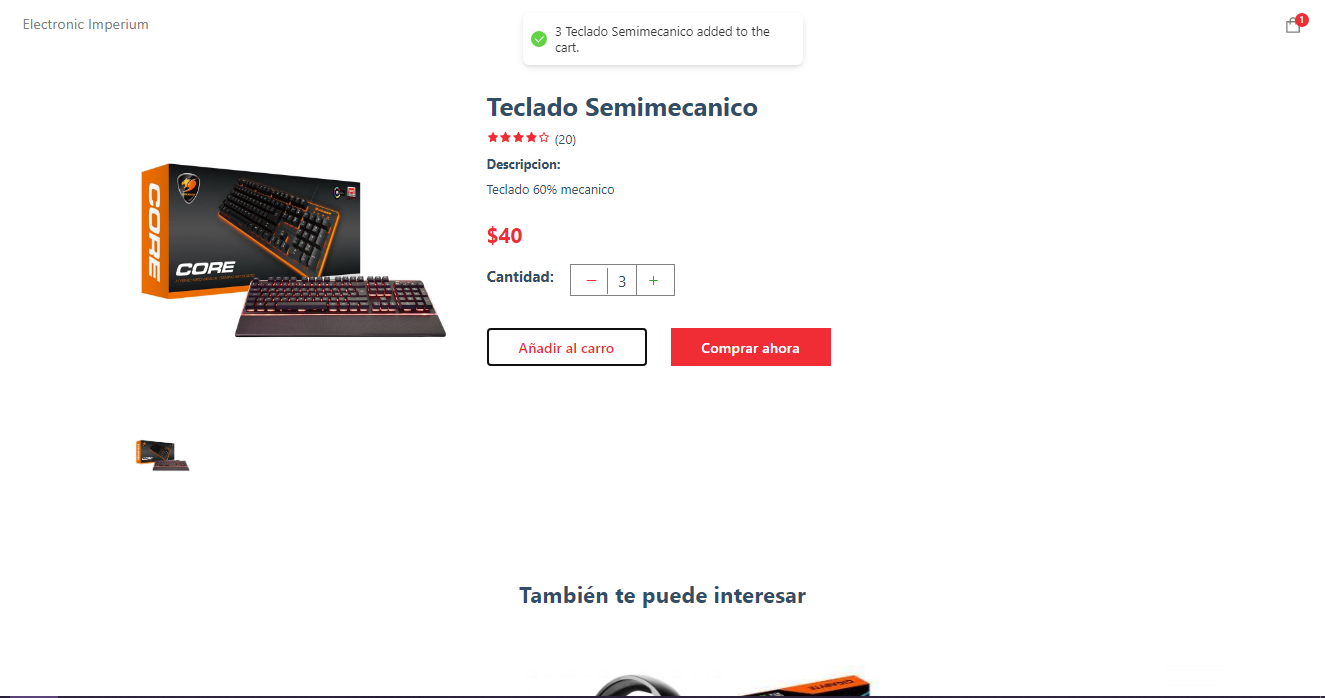
En la siguiente imagen, se están añadiendo varios productos al carrito.



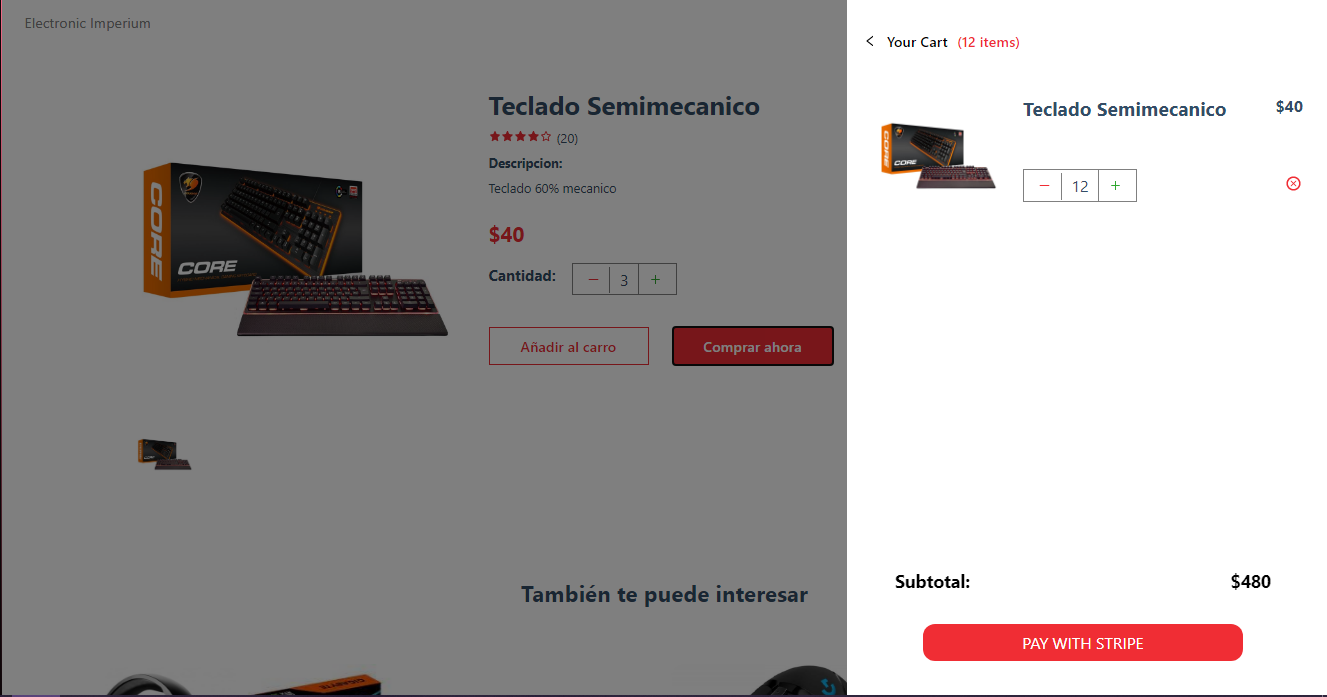
En la siguiente imagen, se está mermando la cantidad del producto.



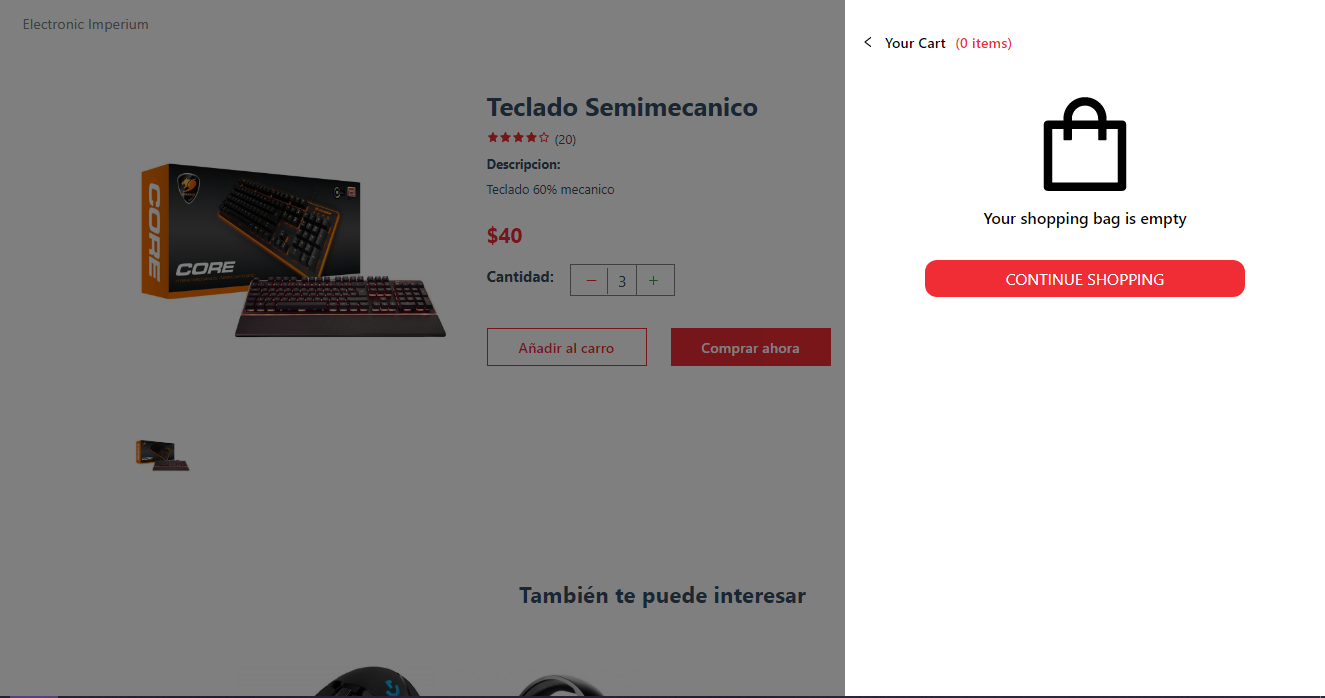
En la siguiente imagen, cuando se selecciona el botón de “Añadir al carro”, aparece un mensaje en la parte superior de la página, de que el producto seleccionado ha sido añadido al carro de compras.



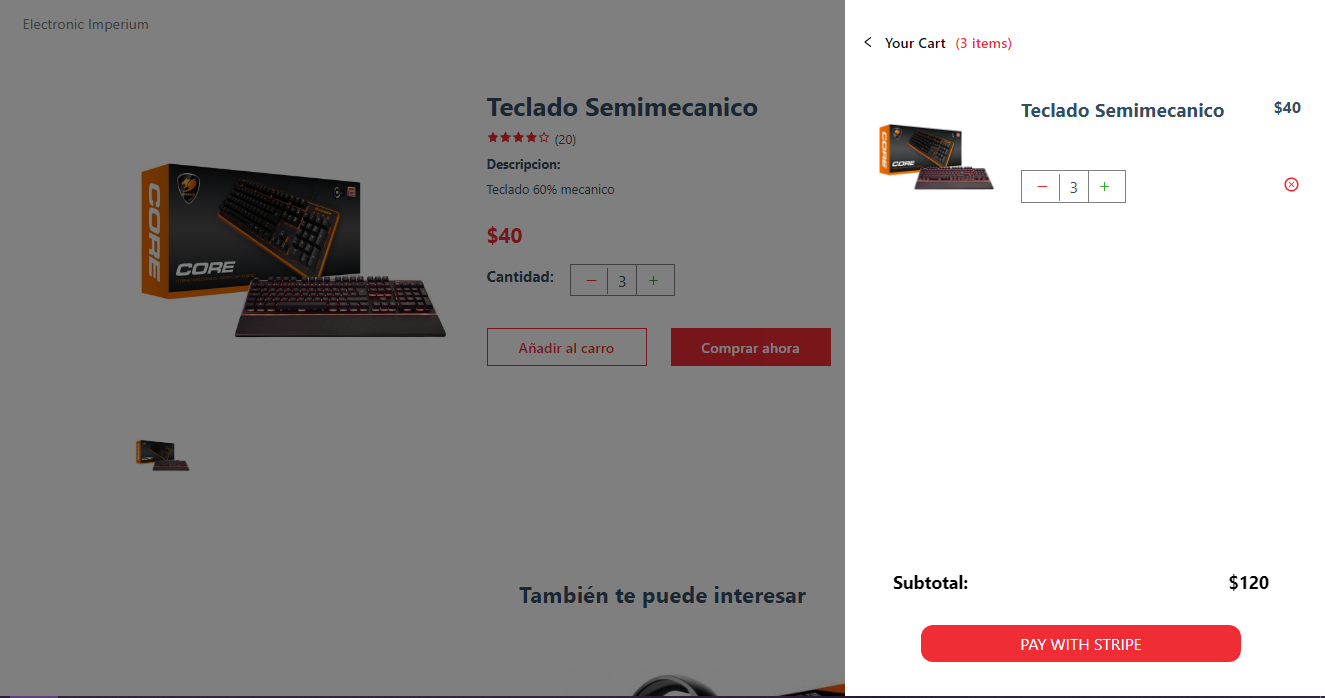
En la siguiente imagen, se logra ver qué es lo que pasa cuando se selecciona el botón de comprar ahora. En la parte superior derecha aparece cuál es la cantidad del producto que se ha añadido.

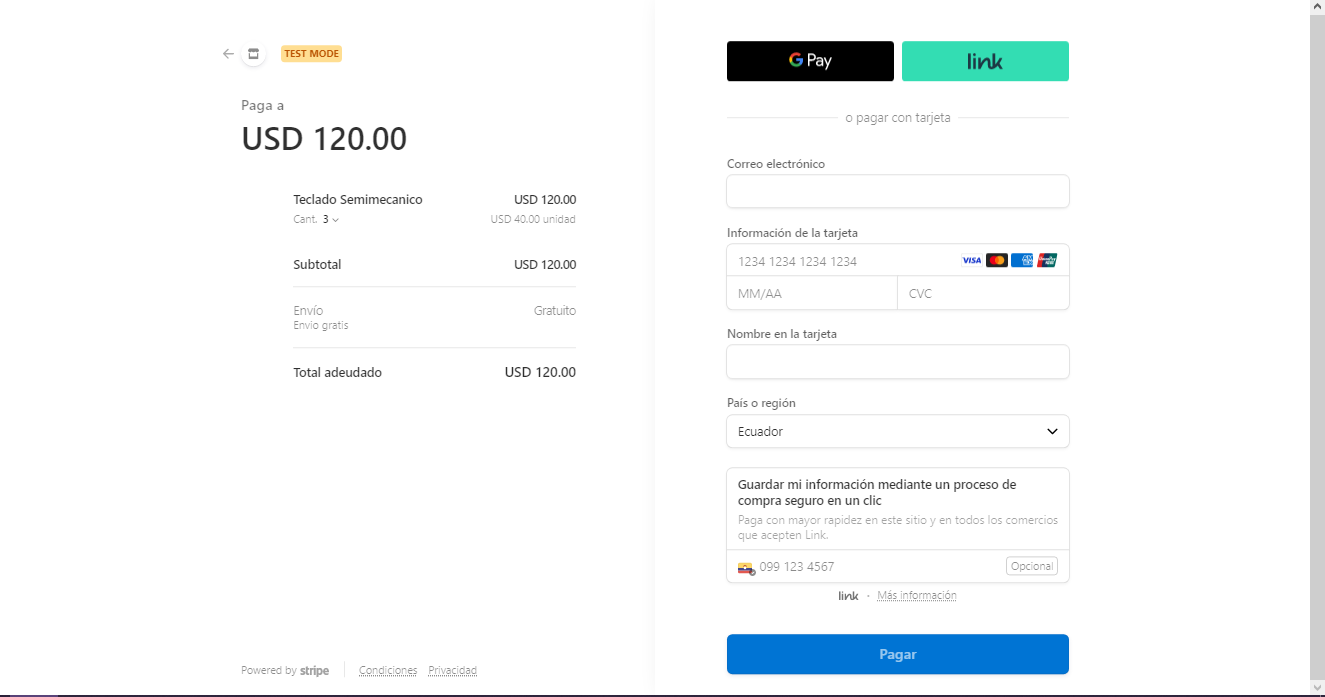


La siguiente imagen demuestra qué es lo que sucede cuando se quita el producto del carrito de compras.



Las siguientes imágenes demuestran qué es lo que pasa cuando se selecciona el botón “Pay with Stripe”. Cabe recalcar que la página de pago está en modo de prueba.





La siguiente imagen demuestra qué es lo que pasa cuando se decide cancelar la compra.



1. **Plan de acción / Propuesta.**

De manera general, se ha pensado en tres ideas que podrían ser tomadas en cuenta para mejorar nuestro proyecto de Tienda e-Commerce.

Como propuesta principal, se tiene “Subir nuestra página e-Commerce a un servidor dedicado”, estos son las ventajas para ello: Mayor rendimiento: se puede mejorar la velocidad y la experiencia de usuario al no tener que compartir recursos con otros sitios web.

Mayor seguridad: se pueden implementar medidas de seguridad avanzadas y personalizadas para proteger la información confidencial del cliente.

Control total: se tiene un control total sobre la configuración y personalización del servidor para ajustar la configuración y mejorar el rendimiento y la seguridad del sitio.

Escalabilidad: se pueden agregar recursos adicionales para satisfacer las necesidades cambiantes del sitio de e-eCommerce y permitir el crecimiento y la expansión del sitio.

Soporte técnico: se cuenta con el soporte técnico de un equipo de expertos en servidores para resolver cualquier problema técnico que pueda surgir.

Como segunda propuesta, se tiene “Mejorar la interfaz gráfica”, es importante para mejorar la experiencia del usuario, aumentar la credibilidad, la tasa de conversión y destacar sobre la competencia.

Como tercera propuesta, se tiene “Traducir las palabras que están en inglés a español”, esto es muy importante ya que las personas que visitan nuestra página eCommerce son de habla español, y a ciertas personas se les dificulta el idioma inglés. Aunque, tiene pensado en un futuro hacer una sección para cambiar de idioma, y así los usuarios tengan una mejor experiencia

1. **Conclusiones**

En este proyecto se utilizaron diferentes tecnologías y herramientas para desarrollar una aplicación de e-commerce. Se utilizó Visual Studio Code como editor de texto, JavaScript como lenguaje de programación, Next.js como framework de desarrollo web, Sanity Studio para la gestión de contenido, Node.js para el manejo del servidor, Stripe para el procesamiento de pagos y GitHub para el control de versiones.

El proyecto se inició con la creación de una aplicación de Next.js, que proporciona una estructura de proyecto predeterminada con configuraciones de servidor y cliente, enrutamiento y herramientas de construcción preconfiguradas. Se crearon componentes y páginas dentro de la carpeta "pages", que se convierten en páginas web accesibles mediante una URL única. Además, se utilizó un archivo de GitHub para instalar las dependencias necesarias y asegurarla compatibilidad entre versiones.

Se utilizó Sanity Studio para la gestión de contenido del proyecto, creando un conjunto de datos para el comercio electrónico y una plantilla de proyecto. También se utilizó Stripe para el procesamiento de pagos y Node.js para el manejo del servidor. El proyecto incluyó un componente de banner principal y una sección de "productos más vendidos" con un bucle que mostraba productos reales.

Se aplicaron nombres de clase a elementos HTML en una página utilizando React y CSS, lo que permitió definir estilos CSS específicos para esos elementos. Los nombres de clase se aplicaron mediante la sintaxis JSX en React y se definieron estilos CSS correspondientes en un archivo CSS global.

**5. Referencias Bibliográficas.**

|  |  |
| --- | --- |
| [1] | N. Gamella, «Doofinder,» 03 02 2023. [En línea]. Available: https://www.doofinder.com/es/blog/que-es-e-commerce. [Último acceso: 15 02 2023]. |
| [2] | D. (. Urrutia, «Arimetrics,» 05 08 2021. [En línea]. Available: https://www.arimetrics.com/glosario-digital/javascript. [Último acceso: 15 02 2023]. |
| [3] | J. Lucas, «OpenWebinars.net,» 04 09 2019. [En línea]. Available: https://openwebinars.net/blog/que-es-nodejs/. [Último acceso: 15 02 2023]. |
| [4] | Equipo Geek, «Ifgeekthen,» 26 03 2020. [En línea]. Available: https://ifgeekthen.nttdata.com/es/que-es-reactjs-conceptos-basicos. [Último acceso: 15 02 2023]. |
| [5] | K. (K.), «Medium,» 29 08 2021. [En línea]. Available: https://javascript.plainenglish.io/what-is-sanity-io-c0a58d66342f. [Último acceso: 15 02 2023]. |
| [6] | H. (. Ramírez, «Grupo Atico34,» 13 05 2022. [En línea]. Available: https://protecciondatos-lopd.com/empresas/stripe-metodo-de-pago/. [Último acceso: 15 02 2023]. |