Pontificia Universidad Católica del Ecuador

Facultad

Hábitat, Infraestructura y Creatividad

Tema

Memoria Técnica del Proyecto Integrador

Autores

James Arguello

Wismar Benítez

Ángel Loor

Índice

1 Resumen

El proyecto aborda el análisis y desarrollo de un sistema para una licorería cuyos procesos se realizaban de forma manual, lo que generaba dificultades en el control de inventarios, inconsistencias en la facturación y limitaciones para el uso de medios digitales. Frente a esta situación, el proyecto integrador BarBox plantea la construcción de una plataforma informática orientada al manejo de productos, clientes, proveedores, inventarios y facturación, integrando ventas presenciales y en línea dentro del alcance de un proyecto académico. Para su desarrollo se aplicó la metodología Team Software Process (TSP), utilizando sus fases y roles como guía para la planificación, organización y seguimiento del proyecto. El sistema fue implementado mediante tecnologías web, con un backend desarrollado en Node.js y Express, un frontend en React y la persistencia de datos gestionada mediante PostgreSQL en un entorno de nube. Como resultado, se obtuvo un sistema funcional a nivel académico que cumple con los requerimientos definidos y sirve como base para futuras mejoras y extensiones.

2 Introducción

2.1 Contexto académico del proyecto

El presente documento corresponde a la memoria técnica del desarrollo del proyecto integrador BarBox, elaborado como un proyecto académico universitario, con el propósito de aplicar conocimientos relacionados con el análisis, diseño, implementación y validación de sistemas de información.

2.2 Motivación

La motivación del proyecto surge a partir de la observación de procesos manuales en el manejo de información de una licorería, los cuales dificultaban el control de inventarios, la consistencia de los registros y la incorporación de herramientas digitales que apoyen dichos procesos dentro de un contexto académico.

2.3 Importancia del sistema

El desarrollo del proyecto integrador BarBox permitió explorar el uso de herramientas tecnológicas para la automatización y centralización de la información, así como la integración de distintos módulos funcionales en una sola plataforma, aportando una visión práctica sobre la construcción de sistemas de información aplicados a un escenario real.

2.4 Organización del documento

El documento se estructura en secciones que abarcan los objetivos del proyecto, su alcance, el marco teórico, la metodología de desarrollo, el análisis de requisitos, el diseño del sistema, la implementación, las pruebas realizadas, los resultados obtenidos, las conclusiones y el trabajo futuro.

3 Objetivos

3.1 Objetivo general

Desarrollar un proyecto integrador de software orientado al manejo integral de la información de una licorería, conforme a los requisitos funcionales definidos (F1–F6).

3.2 Objetivos específicos

- Diseñar e implementar un sistema informático que permita integrar el manejo de clientes, productos, facturación, proveedores, órdenes de compra y bodega dentro del alcance del proyecto integrador.
- Automatizar los procesos relacionados con la facturación y el control de inventarios, garantizando la consistencia de la información registrada en el sistema.
- Incorporar mecanismos de verificación de edad que aseguren el cumplimiento de la normativa legal vigente para la comercialización de bebidas alcohólicas, desde un enfoque académico.
- Desarrollar una arquitectura de software modular y escalable que facilite el mantenimiento, la evolución y la incorporación de nuevos módulos al sistema.
- Garantizar la seguridad, integridad y confidencialidad de la información mediante mecanismos de autenticación, control de accesos y validaciones de datos.
- Aplicar buenas prácticas durante el desarrollo del proyecto, utilizando la metodología Team Software Process (TSP) como marco de trabajo para el control y la calidad del proceso.

4 Alcance y limitaciones

El alcance del proyecto comprende el diseño y desarrollo de un sistema de información orientado a mejorar y automatizar los procesos operativos de una licorería, permitiendo la gestión eficiente de proveedores, órdenes de compra, bodega, clientes, productos y facturación. El sistema facilita el control de inventarios, el registro y seguimiento de

compras, la administración de datos de clientes y proveedores, así como la emisión de facturas de manera estructurada y consistente. Asimismo, integra las operaciones relacionadas con ventas presenciales y digitales, contribuyendo a la reducción de errores derivados de procesos manuales y al fortalecimiento de la toma de decisiones. El proyecto se desarrolla dentro de un contexto académico, por lo que su alcance se limita a la implementación funcional de los módulos definidos, sin considerar aspectos legales, fiscales o de operación en producción real.

5 Marco teórico

El marco teórico presenta los fundamentos conceptuales que respaldan el desarrollo del sistema.

5.1 Ingeniería de Software

La ingeniería de software proporciona principios y prácticas orientadas al desarrollo sistemático de soluciones informáticas, permitiendo estructurar las fases del proyecto y documentar los artefactos generados.

El proyecto se estructura con un enfoque de Arquitectura Empresarial (AE), organizando la información en los tres dominios solicitados, de la siguiente manera:

- **Arquitectura de Tecnología**: describe la infraestructura tecnológica que soporta el sistema, incluyendo plataformas, lenguajes, frameworks, herramientas de desarrollo, entorno de ejecución y consideraciones de seguridad y rendimiento.
- **Arquitectura de Información**: define cómo fluye la información dentro del sistema, los procesos que la generan o consumen, la interacción entre módulos y la forma en que los actores acceden y utilizan la información.
- **Arquitectura de Datos**: detalla la estructura y organización de los datos del sistema, incluyendo entidades, relaciones, estados, reglas de integridad y el soporte de la base de datos para las operaciones del negocio.

5.2 Metodología de desarrollo

La metodología TSP (Team Software Process) es un marco de trabajo orientado a la gestión disciplinada de proyectos de software en equipo, cuyo objetivo principal es ayudar a los grupos de desarrollo a planificar, medir, controlar y mejorar la calidad y productividad de sus proyectos. TSP fue creada por el Software Engineering Institute

(SEI) como una extensión del PSP (Personal Software Process), trasladando sus principios desde el nivel individual al trabajo colaborativo.

5.3 Arquitectura de software

La arquitectura de software define la estructura general del sistema y la forma en que interactúan sus distintos componentes, permitiendo organizar adecuadamente la lógica del negocio, la gestión de datos y la presentación de la información. En sistemas web, el uso de arquitecturas cliente-servidor y en capas facilita la escalabilidad, el mantenimiento y la evolución del sistema a lo largo del tiempo.

En el proyecto BarBox se adoptó la arquitectura MVC (Modelo-Vista-Controlador), la cual permite separar claramente las responsabilidades del sistema: el Modelo gestiona la lógica de negocio y el acceso a los datos, la Vista se encarga de la presentación de la información al usuario y el Controlador actúa como intermediario entre ambos, procesando las solicitudes y coordinando las respuestas. Este enfoque contribuye a una mejor organización del código, mayor reutilización de componentes y un desarrollo más controlado y mantenible.

5.4 Tecnologías clave

Las tecnologías clave empleadas en el proyecto BarBox se basan en el uso de Node.js y Express para la construcción de servicios REST que soportan operaciones CRUD, PostgreSQL como sistema gestor de base de datos relacional para el almacenamiento y consistencia de la información, React para el desarrollo de interfaces web dinámicas orientadas al usuario, TypeScript para mejorar la seguridad y mantenibilidad del código mediante tipado estático, y servicios en la nube para el despliegue y disponibilidad del sistema, destacando Neon como plataforma de base de datos PostgreSQL en la nube y Vercel como servicio de hosting para la publicación del frontend. Estas tecnologías permiten implementar soluciones web modernas orientadas a la gestión y comercialización digital de manera eficiente y escalable.

Las herramientas principales que se ocuparon para el desarrollo del proyecto integrador se encuentran compuestas en dos secciones backend y frontend:

Backend

- **Node.js:** entorno de ejecución que permite desarrollar aplicaciones del lado del servidor utilizando JavaScript.

- **Express**: framework web que facilita la creación de APIs REST mediante el manejo de rutas y middleware.
- **Prisma**: ORM moderno que permite el acceso seguro y tipado a bases de datos relacionales.
- **PostgreSQL**: sistema gestor de bases de datos relacional orientado a la confiabilidad y consistencia de los datos.
- **JWT (JSON Web Token):** mecanismo de autenticación basado en tokens para el control de acceso a la API.

Frontend

- **React**: librería de JavaScript utilizada para la construcción de interfaces de usuario basadas en componentes.
- **TypeScript**: lenguaje que extiende JavaScript incorporando tipado estático para reducir errores en el desarrollo.
- **React Router DOM**: librería que gestiona la navegación y el enrutamiento en aplicaciones web.
- **Axios**: cliente HTTP utilizado para la comunicación entre el frontend y el backend.
- **Tailwind CSS**: framework CSS basado en utilidades que permite diseñar interfaces de forma rápida y consistente.

Servicios Externos

- **PayPal**: plataforma de pagos electrónicos utilizada para procesar transacciones de forma segura en el sistema E-commerce.

Servicios de la nube

- **Neon**: servicio de base de datos PostgreSQL en la nube que proporciona alta disponibilidad y escalabilidad.
- **Vercel**: plataforma de hosting en la nube utilizada para el despliegue y publicación de aplicaciones web.

6 Metodología de desarrollo

Para el desarrollo del proyecto integrador BarBox se aplicó la metodología TSP (Team Software Process), permitiendo una gestión disciplinada del trabajo en equipo, orientada al control, medición y mejora continua del proceso de desarrollo de software.

6.1 Metodología aplicada

Se utilizó un ciclo de vida con metodología ágil basado en TSP, aplicado al desarrollo del proyecto integrador BarBox, el cual permitió trabajar de forma iterativa e incremental. El enfoque iterativo se reflejó en la construcción y validación de prototipos (Interacción Humano Computador, Desarrollo Basado en Plataformas e Ingeniería de Software), los cuales fueron refinados continuamente a partir de revisiones y retroalimentación, mientras que el enfoque incremental se evidenció en la implementación progresiva de funcionalidades organizadas en Ciclo 1 y Ciclo 2, ampliando el alcance del sistema de manera controlada.

6.2 Roles del proyecto

- **Líder del Equipo (Ángel Loor)**: coordina al equipo, facilita la comunicación y asegura el cumplimiento de los objetivos y compromisos del proyecto.
- **Administrador de Desarrollo (Ángel Loor)**: supervisa el progreso técnico del sistema y verifica que las actividades de diseño e implementación se ejecuten correctamente.
- **Administrador de Planeación (Wismar Benítez)**: elabora y mantiene las estimaciones de tiempo, esfuerzo y cronograma del proyecto, dando seguimiento al avance planificado.
- **Administrador de Calidad y Configuración (James Arguello)**: vela por el cumplimiento de la metodología TSP, los estándares definidos y la calidad del producto desarrollado.

6.3 Fases del desarrollo

- **Lanzamiento**: organización del equipo, asignación de roles y definición de objetivos y compromisos del proyecto.
- **Estrategia**: definición del enfoque general, alcance y prioridades que guían el desarrollo del sistema.
- **Planeación**: elaboración del plan del proyecto mediante la estimación de tiempos, esfuerzo y recursos.

- **Requerimientos**: identificación y documentación de las necesidades funcionales y no funcionales del sistema.
- **Diseño**: modelado de la solución mediante diagramas y especificaciones técnicas del sistema.
- **Implementación**: desarrollo e integración del software conforme al diseño aprobado.
- **Pruebas**: verificación del funcionamiento del sistema mediante pruebas funcionales y de integración.
- **Postmortem**: análisis de resultados, métricas y lecciones aprendidas del proyecto.

6.4 Herramientas utilizadas

- **Node.js**: se utilizó como entorno de ejecución del backend para implementar la lógica del servidor y el manejo de peticiones del sistema.
- **Express**: se empleó para desarrollar la API REST del sistema, gestionando rutas, controladores y servicios que soportan las operaciones CRUD.
- **PostgreSQL 15**: se utilizó esta versión del sistema gestor de base de datos relacional debido a que ofrece un alto nivel de estabilidad y madurez frente a versiones más recientes. PostgreSQL 15 cuenta con mejoras consolidadas en rendimiento, seguridad y manejo de concurrencia, lo que reduce riesgos asociados a cambios experimentales o incompatibilidades presentes en versiones más nuevas. Esta estabilidad resulta clave para garantizar la integridad de los datos y la confiabilidad del sistema durante su desarrollo y operación.
- **Neon**: se utilizó como servicio de base de datos PostgreSQL en la nube debido a su integración nativa con la plataforma de despliegue Vercel, lo que permitió una gestión más confiable de las variables de entorno durante los builds de producción. Durante las pruebas de despliegue, se identificaron fallos de conexión al utilizar Supabase, ocasionados por inconsistencias en la actualización de variables de entorno, lo que derivaba en errores internos del servidor (HTTP 500). Neon ofrece una arquitectura serverless, alta disponibilidad y compatibilidad optimizada con entornos de despliegue continuo, reduciendo significativamente los errores de conexión y simplificando el flujo de despliegue, por lo que se seleccionó como la alternativa más estable y alineada con la arquitectura del sistema.

- **React**: se empleó para el desarrollo de las interfaces web del E-commerce, POS y Backoffice, permitiendo una interacción dinámica con el usuario.
- **TypeScript**: se utilizó para mejorar la mantenibilidad y seguridad del código mediante tipado estático en el frontend.
- **Vercel**: se utilizó como plataforma de hosting en la nube para el despliegue y publicación de las aplicaciones web del proyecto. Se optó por su plan gratuito debido a que ofrece las funcionalidades necesarias para el despliegue, validación y pruebas del sistema dentro de un entorno académico, sin generar costos adicionales. Dado que el alcance del proyecto no contempla alta concurrencia ni tráfico intensivo, las capacidades del plan gratuito resultan suficientes y coherentes con una gestión eficiente de los recursos, manteniendo estabilidad y facilidad en el proceso de despliegue.
- **PayPal**: se integró como servicio externo para la gestión segura de pagos electrónicos en el sistema de E-commerce.
- **Git**: se utilizó como sistema de control de versiones para gestionar los cambios del código fuente durante el desarrollo del proyecto.
- **GitHub**: se empleó como plataforma de alojamiento del repositorio, facilitando la colaboración y el seguimiento del trabajo en equipo.
- **Visual Studio Code**: se utilizó como editor de código principal para el desarrollo del frontend y backend del sistema.

7 Análisis de requisitos

7.1 Requisitos funcionales

A continuación, los requisitos funcionales del proyecto integrador descritos en la siguiente tabla.

ID	Requisito	Descripción
RF-01	Gestión de Proveedores (F1)	Registro y administración de proveedores
RF-02	Gestión de Órdenes de Compra (F2)	Generación y control de compras
RF-03	Gestión de Bodega (F3)	Control de ingresos de inventario y stock
RF-04	Gestión de Clientes (F4)	Administración de clientes
RF-05	Gestión de Facturación (F5)	Emisión de facturas

RF-06	Gestión de Productos (F6)	Administración del catálogo

Tabla 1. Tablas de requisitos funcionales obtenidas del documento del Proyecto Integrador

7.2 Requisitos no funcionales

- **Rendimiento:** El sistema debe ofrecer tiempos de respuesta adecuados al entorno académico.
- **Seguridad:** Se debe garantizar la validación de datos y el control de acceso por roles.
- **Usabilidad:** La interfaz debe ser clara, intuitiva y consistente para el usuario.

8 Diseño del sistema

8.1 Arquitectura general

El sistema fue diseñado bajo una arquitectura cliente-servidor, seleccionada por su claridad conceptual y adecuación para sistemas web.

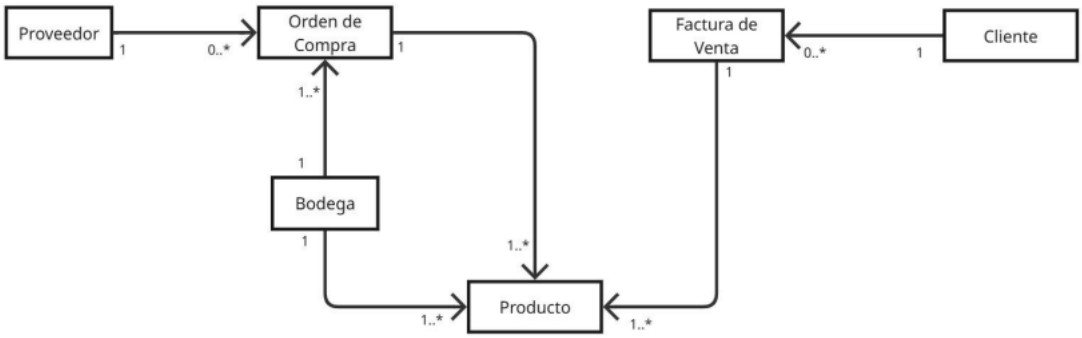

Figura 1. Arquitectura general del sistema

8.2 Diagrama de componentes

El diagrama de componentes describe la organización modular del sistema.

A continuación, se presentan los diferentes diagramas de componentes elaborados para cada sistema:

Diagrama de Componentes del E-commerce: se elaboró para representar la arquitectura del sistema orientado a las ventas en línea, mostrando los componentes encargados de la interacción con el cliente, la gestión de productos, pedidos y facturas, incluyendo también servicios externos como el de PayPal, así como su comunicación con el backend y la base de datos.

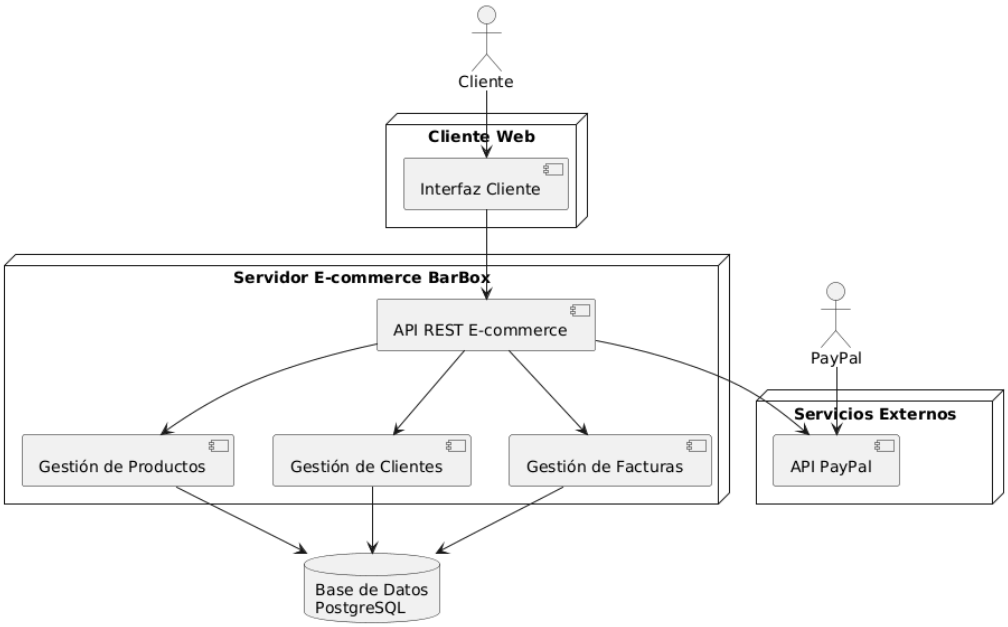

Figura 2. Diagrama de componentes del e-commerce

Diagrama de Componentes del Sistema POS: se desarrolló con el objetivo de modelar la arquitectura del sistema de ventas presenciales, detallando los componentes responsables del registro de clientes, facturas y actualización de inventarios (bodega) en tiempo real.

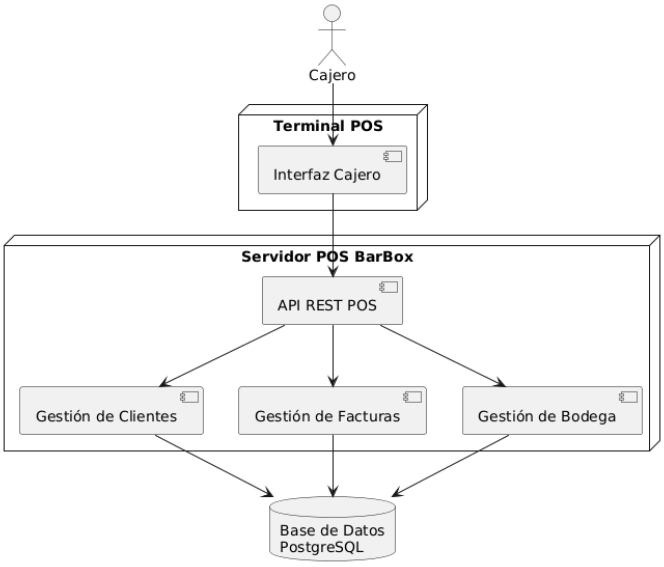

Figura 3. Diagrama de Componentes del sistema POS

Diagrama de Componentes del Back-office: se construyó para describir la arquitectura del sistema administrativo, evidenciando los componentes encargados de la gestión de proveedores, órdenes de compra, bodega, clientes, facturas y productos.

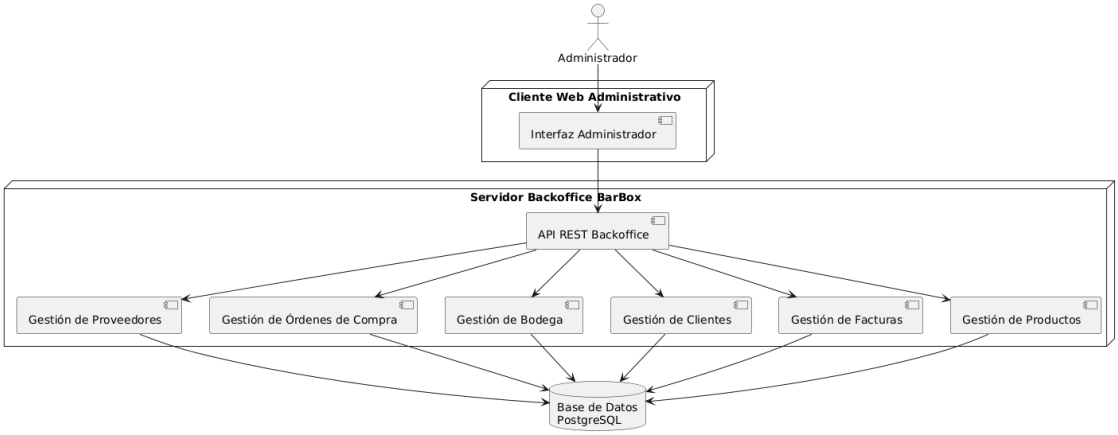

Figura 4. Diagramas de componentes del sistema back-office

8.3 Modelo de datos

El modelo de datos del sistema BarBox se diseñó utilizando un enfoque entidad–relación, permitiendo representar de manera estructurada las entidades del negocio, sus atributos y las relaciones existentes entre ellas. Este modelo entidad–relación es utilizado de forma transversal por todos los sistemas de BarBox, incluyendo el e-commerce, el backoffice administrativo y el sistema POS, garantizando coherencia y consistencia en la gestión de la información. Además, el modelo de datos se encuentra centralizado en la nube, lo que permite el acceso controlado y sincronizado a la información desde los distintos módulos del sistema, facilitando la integración y la escalabilidad de la plataforma.

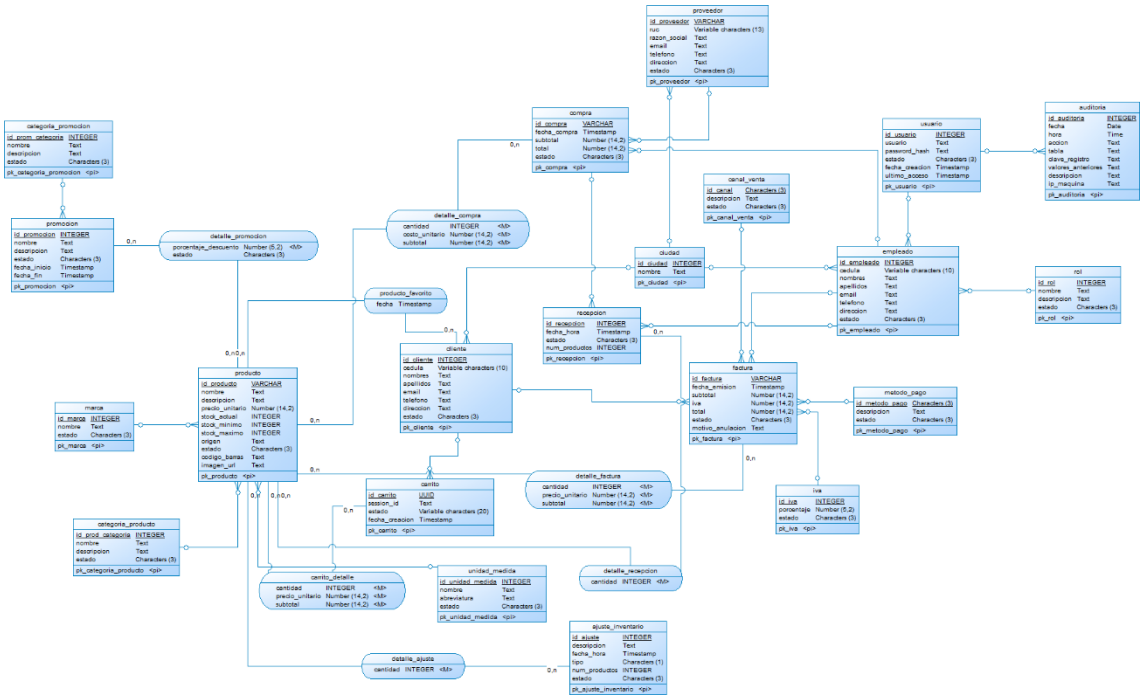

Figura 5. Modelo entidad–relación (ERD)

8.4 Diagramas UML relevantes

Como parte del diseño del sistema E-commerce BarBox, se desarrollaron diagramas UML con el fin de representar las funcionalidades del sistema y el flujo de los procesos principales. De acuerdo con el alcance definido para esta memoria técnica, se incluyen únicamente diagramas de casos de uso y diagramas de actividades, los cuales permiten describir el comportamiento del sistema desde una perspectiva funcional.

8.4.1 Diagramas de casos de uso

Los diagramas de casos de uso fueron elaborados para representar las funcionalidades del sistema desde la perspectiva de los actores que interactúan con él. Estos diagramas permiten identificar las acciones disponibles para cada tipo de usuario y su relación con los módulos funcionales definidos en el análisis de requisitos.

El diagrama general de casos de uso del sistema se presenta en la Figura 6, donde se identifican los actores principales y los casos de uso asociados a la gestión del sistema.

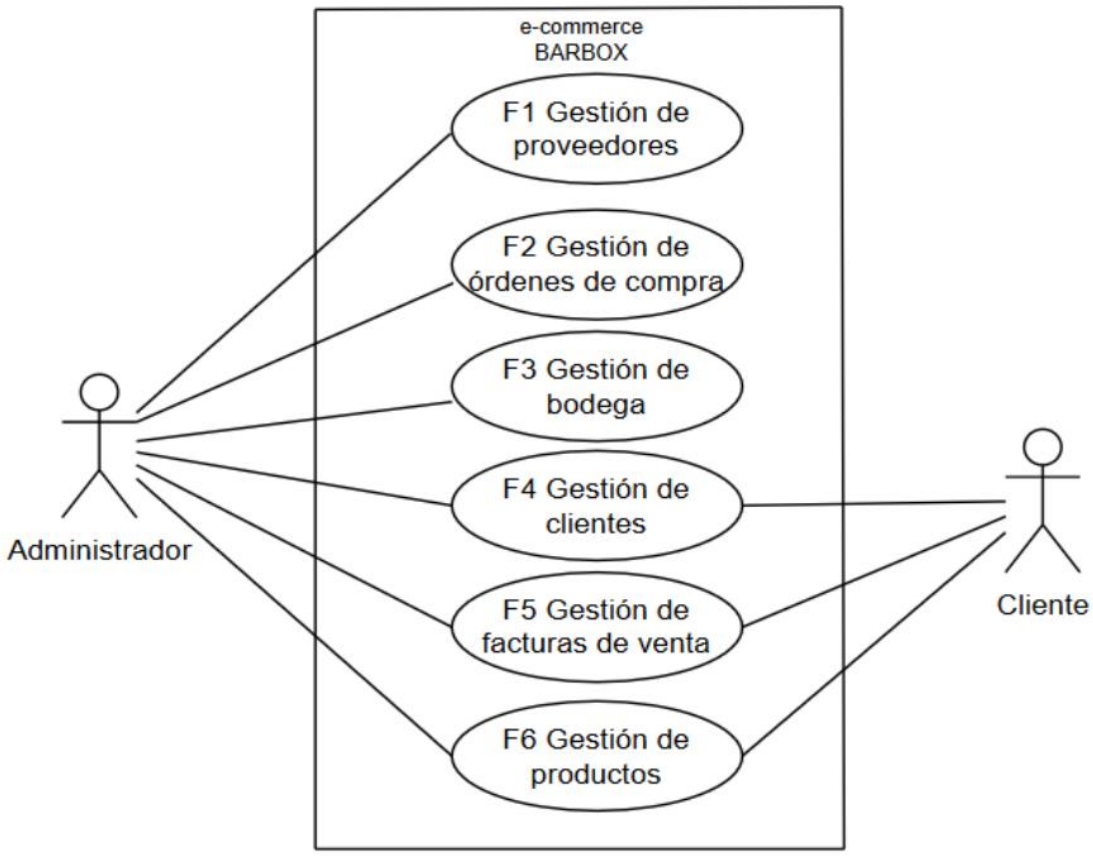

Figura 6. Diagrama general de casos de uso del sistema E-commerce BarBox

Este diagrama permite visualizar el alcance funcional del sistema y su correspondencia con los requisitos funcionales definidos (F1–F6).

8.5 Diagramas de actividades

Los diagramas de actividades fueron desarrollados para representar el flujo de los procesos principales del sistema, mostrando la secuencia de actividades y las decisiones involucradas en cada operación. Estos diagramas facilitan la comprensión del comportamiento general del sistema durante la ejecución de los procesos.

En el proyecto se elaboraron diagramas de actividades para los procesos de Gestión de Facturación y Gestión de Bodega, correspondientes a los ciclos definidos durante el desarrollo del sistema. Estos diagramas se presentan en las Figuras 7 y 8, respectivamente.

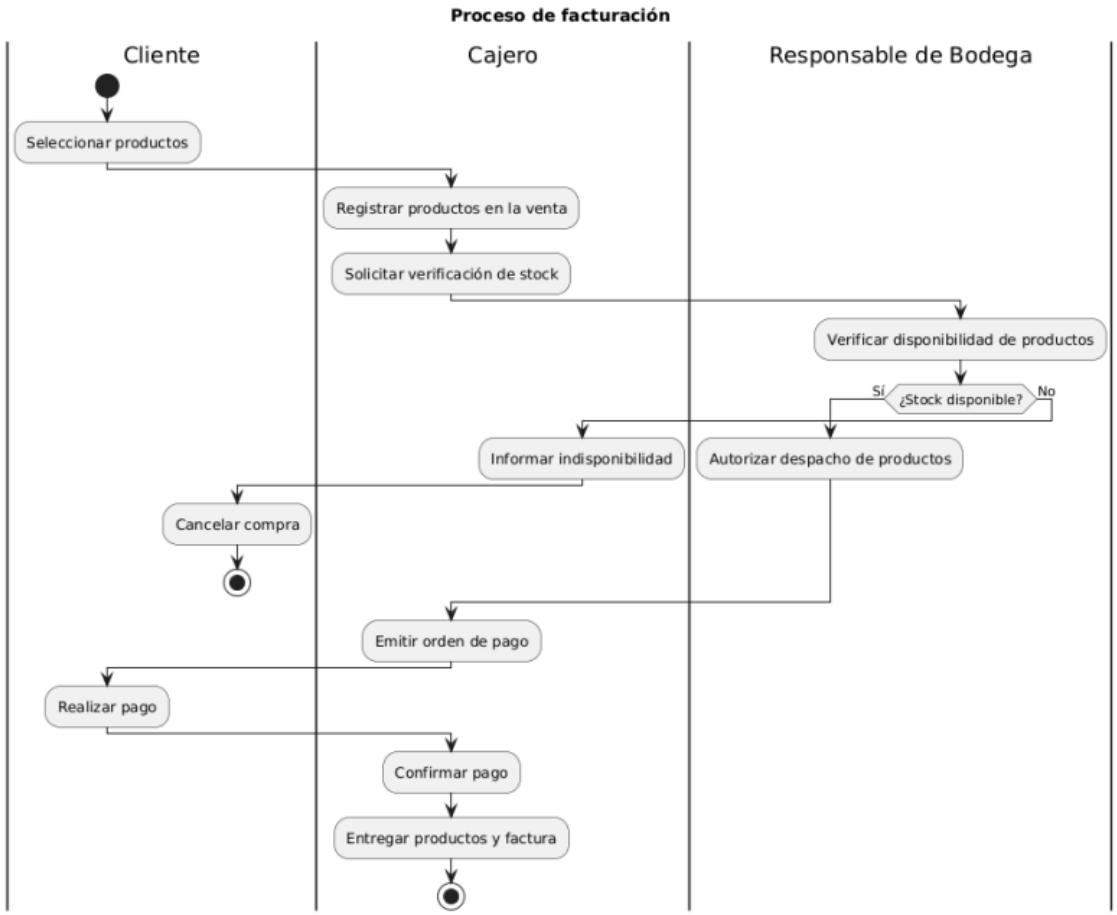

Figura 7. Diagrama de actividades – Gestión de facturación (Ciclo 1)

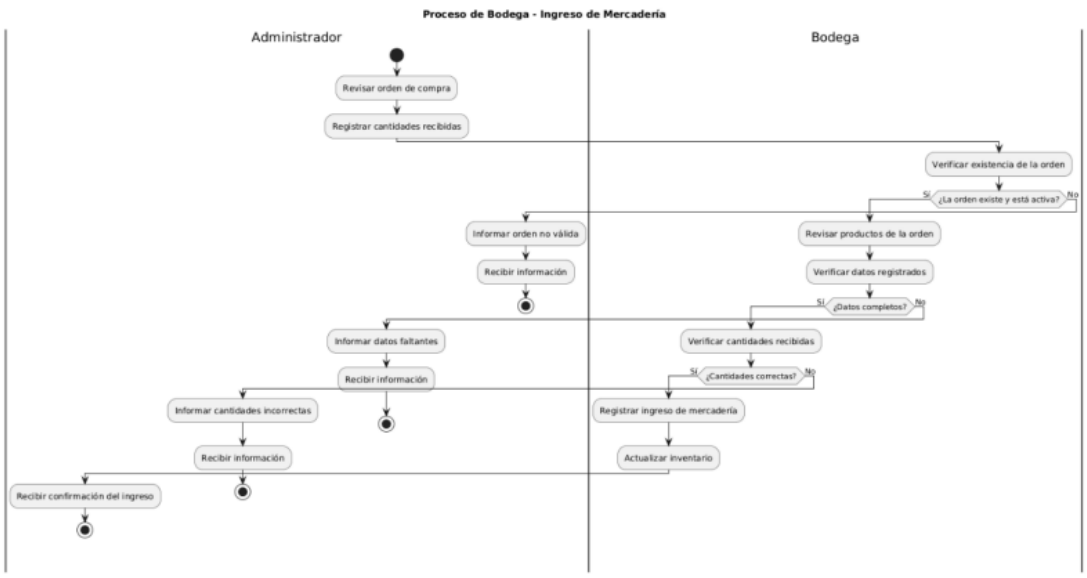

Figura 8. Diagrama de actividades – Gestión de bodega (Ciclo 2)

Los diagramas de actividades permiten identificar las transiciones entre acciones y el flujo general de los procesos analizados.

9 Implementación

El sistema BarBox fue implementado utilizando JavaScript como lenguaje base, aplicando una arquitectura web moderna orientada a servicios. El backend se desarrolló con Node.js y Express, mediante la construcción de una API REST encargada de gestionar las operaciones CRUD de los distintos módulos del sistema, mientras que el frontend se implementó con React, incorporando TypeScript para mejorar la mantenibilidad y seguridad del código a través de tipado estático. La persistencia de datos se realizó utilizando PostgreSQL, alojado en la nube mediante Neon, garantizando disponibilidad y acceso remoto a la información. El despliegue de las aplicaciones web se llevó a cabo utilizando Vercel como plataforma de hosting, permitiendo la publicación y pruebas del sistema en un entorno en línea. La integración de pagos electrónicos se realizó mediante el uso de PayPal como servicio externo. La organización del código se estructuró de forma modular y alineada con la arquitectura definida, utilizando Git y GitHub para el control de versiones y Visual Studio Code como entorno de desarrollo, lo que facilitó el trabajo colaborativo y la gestión del proyecto.

10 Pruebas y validación

La fase de pruebas y validación tuvo como finalidad analizar el comportamiento del sistema E-commerce BarBox frente a distintos escenarios de uso, considerando tanto aspectos funcionales como no funcionales. Para ello, se contemplaron diferentes tipos de pruebas, las cuales permitieron evaluar el funcionamiento de los módulos implementados y la interacción entre ellos.

10.1 Pruebas unitarias

Las pruebas unitarias se enfocaron en el análisis individual de los módulos del sistema, evaluando el comportamiento de cada funcionalidad de manera aislada. Estas pruebas permitieron revisar operaciones como el registro de productos, la gestión de clientes y el control de inventario, verificando que cada módulo respondiera conforme a lo definido en los requisitos funcionales.

Prueba	**Resultado esperado**	**Resultado obtenido**
Registro de producto	El producto se registra correctamente con sus datos completos	El producto se registró correctamente
Actualización de producto	Los cambios se guardan sin errores	La información se actualizó correctamente
Registro de cliente	El cliente se registra con datos válidos	Registro exitoso del cliente
Consulta de cliente	Se muestra la información correcta del cliente	Información visualizada correctamente
Control de inventario	El stock se actualiza según la operación realizada	Stock actualizado correctamente

Tabla 2. Casos de prueba correspondientes a las pruebas unitarias del sistema E-commerce BarBox extraídas del documento del Proyecto Integrador.

10.2 Pruebas de integración

Las pruebas de integración se orientaron a observar la interacción entre los distintos módulos del sistema. En este tipo de pruebas se analizó, por ejemplo, la relación entre el

módulo de facturación y el módulo de bodega, observando cómo una operación de venta se refleja en la actualización del inventario. Estas pruebas permitieron identificar el comportamiento del sistema cuando los componentes funcionan de manera conjunta.

Prueba	**Resultado esperado**	**Resultado obtenido**
Facturación y bodega	Al generar una factura, el stock disminuye	El inventario se actualizó correctamente
Anulación de factura	Al anular una factura, el stock se restablece	El stock se restauró correctamente
Órdenes de compra y bodega	Al aprobar una orden, el stock aumenta	Inventario actualizado correctamente
Cliente y facturación	La factura se asocia al cliente correcto	Asociación realizada correctamente
Productos y promociones	Se aplican promociones al producto	Promoción aplicada correctamente

Tabla 3. Casos de prueba correspondientes a las pruebas de integración entre los módulos del sistema E-commerce BarBox extraídas del documento del Proyecto Integrador.

10.3 Pruebas de aceptación

Las pruebas de aceptación se realizaron tomando como referencia los casos de uso definidos durante la fase de análisis. Estas pruebas permitieron evaluar si el sistema responde de manera adecuada a los escenarios previstos desde la perspectiva del usuario, considerando flujos completos como la gestión de clientes, la emisión de facturas y la consulta de información.

Prueba	**Resultado esperado**	**Resultado obtenido**
Gestión de clientes	El usuario puede registrar y consultar clientes	Flujo ejecutado correctamente
Emisión de facturas	El sistema permite emitir facturas completas	Factura generada correctamente

Consulta de inventario	El usuario visualiza el stock actualizado	Información mostrada correctamente
Gestión de productos	El usuario administra productos sin errores	Operación realizada con éxito
Flujo de venta completo	El proceso se completa sin inconsistencias	Flujo ejecutado correctamente

Tabla 4. Casos de prueba correspondientes a las pruebas de aceptación, basadas en los casos de uso definidos para el sistema E-commerce BarBox extraídas del documento del Proyecto Integrador.

10.4 Pruebas de usabilidad

Adicionalmente, se consideraron pruebas orientadas a la usabilidad del sistema, enfocadas en analizar la facilidad de uso, la claridad de la interfaz y la comprensión de las funcionalidades por parte de los usuarios. Estas pruebas se apoyaron en principios de usabilidad ampliamente aceptados, como las heurísticas de Jakob Nielsen, permitiendo identificar aspectos relacionados con la navegación, consistencia visual y retroalimentación del sistema ante las acciones del usuario.

Prueba	**Resultado esperado**	**Resultado obtenido**
Navegación del sistema	El usuario navega de forma intuitiva	Navegación clara y comprensible
Claridad de la interfaz	Los elementos visuales son comprensibles	Interfaz clara y consistente
Retroalimentación del sistema	El sistema muestra mensajes claros	Mensajes visualizados correctamente
Consistencia visual	Uso coherente de colores y estilos	Consistencia mantenida
Comprensión de funciones	El usuario entiende las acciones disponibles	Funcionalidades comprendidas

Tabla 5. Casos de prueba correspondientes a las pruebas de usabilidad del sistema E-commerce BarBox, extraídas de los resultados de Google Lighthouse.

10.5 Pruebas de Rendimiento

Las pruebas de rendimiento tuvieron como objetivo evaluar el comportamiento del sistema E-commerce BarBox en términos de tiempos de carga, fluidez de la interfaz y estabilidad durante la interacción del usuario. Para ello, se utilizó la herramienta Google Lighthouse, la cual permitió medir métricas claves asociadas a la experiencia del usuario en entorno web.

Los resultados obtenidos evidencian un rendimiento general de 77/100, considerado aceptable para un entorno académico, con tiempos de carga inicial adecuados y sin bloqueos en la ejecución del sistema. Métricas como First Contentful Paint, Speed Index y Total Blocking Time reflejan una respuesta rápida del sistema, mientras que se identificó como aspecto de mejora el valor de Cumulative Layout Shift, relacionado con cambios visuales durante la carga de la página.

Prueba	**Resultado esperado**	**Resultado obtenido**
Carga inicial del sitio web	Visualización del contenido principal en un tiempo aceptable	Contenido visible en ~1.0 s (First Contentful Paint)
Fluidez de la carga	Navegación sin bloqueos durante la carga	Tiempo de bloqueo total de 0 ms
Renderizado del contenido principal	Renderizado rápido del elemento principal	Renderizado en ~1.3 s (Largest Contentful Paint)
Estabilidad visual	Mínimos cambios de diseño durante la carga	CLS de 0.388, se identifican ajustes visuales
Índice de velocidad	Carga perceptible rápida para el usuario	Speed Index de ~1.0 s

Tabla 6. Casos de prueba correspondientes a las pruebas de rendimiento, evaluadas mediante métricas de Google Lighthouse (FCP, LCP, TBT, Speed Index y CLS) para el sistema E-commerce BarBox.

10.6 Pruebas de Seguridad

Las pruebas de seguridad se orientaron a evaluar el cumplimiento de buenas prácticas básicas de protección del sistema web, apoyándose en las auditorías automáticas de Google Lighthouse. Estas pruebas permitieron analizar aspectos relacionados con la configuración del entorno, la prevención de ataques comunes y el uso de políticas de seguridad recomendadas.

Los resultados muestran que el sistema cumple con las prácticas recomendadas de seguridad, obteniendo una calificación de 100/100 en este apartado. Lighthouse no detectó vulnerabilidades críticas durante la evaluación, y se identificaron recomendaciones adicionales que pueden ser validadas manualmente para fortalecer aún más la seguridad del sistema.

Prueba	**Resultado esperado**	**Resultado obtenido**
Protección contra XSS	Prevención de inyección de scripts maliciosos	Cumple prácticas recomendadas
Aislamiento de orígenes	Configuración adecuada de políticas de aislamiento	No se detectan problemas críticos
Mitigación de clickjacking	Protección frente a secuestro de clics	Cumple recomendaciones Lighthouse
Seguridad del DOM	Prevención de XSS basado en DOM	Sin vulnerabilidades detectadas
Configuración general de seguridad	Cumplimiento de auditorías de seguridad web	Calificación 100/100

Tabla 7. Casos de prueba correspondientes a las pruebas de seguridad, evaluadas mediante auditorías de buenas prácticas y controles recomendados por Google Lighthouse para el sistema E-commerce BarBox.

11 Resultados

Durante el desarrollo del proyecto se implementaron los módulos funcionales definidos dentro del alcance del sistema, los cuales permiten gestionar información relacionada con proveedores, órdenes de compra, bodega, clientes, productos y facturación desde una

única plataforma. Estos módulos fueron desarrollados de manera progresiva y coherente con los objetivos planteados para el sistema E-commerce BarBox.

A partir de la ejecución de las pruebas unitarias, de integración y de aceptación, fue posible observar el comportamiento del sistema frente a los escenarios establecidos en los requisitos funcionales. Dichas pruebas permitieron contrastar las funcionalidades implementadas con los objetivos definidos, evidenciando correspondencia entre ambos dentro del contexto del proyecto académico.

Los resultados obtenidos a partir de las pruebas de integración permitieron analizar cómo las acciones realizadas en un módulo influyen en otros componentes del sistema, especialmente en procesos críticos como la facturación y el control de inventarios. Asimismo, las pruebas de aceptación facilitaron la evaluación del funcionamiento del sistema desde una perspectiva cercana al uso real, considerando flujos completos de operación.

En cuanto a los aspectos no funcionales, se realizaron pruebas de rendimiento, accesibilidad, buenas prácticas y SEO utilizando la herramienta Google Lighthouse. Los resultados obtenidos evidencian un comportamiento adecuado del sistema para el entorno académico, reflejando tiempos de carga aceptables, cumplimiento de buenas prácticas y ausencia de problemas críticos de seguridad, conforme a los valores mostrados en las capturas presentadas en esta sección.

Adicionalmente, se llevaron a cabo pruebas de usabilidad con usuarios reales, las cuales permitieron identificar el nivel de comprensión de la interfaz y la facilidad de interacción con el sistema. Estas observaciones aportaron información relevante sobre la experiencia de uso, sin emitir afirmaciones categóricas sobre eficiencia o satisfacción, manteniendo un enfoque descriptivo acorde al alcance del proyecto.

Finalmente, la evidencia del funcionamiento del sistema se encuentra respaldada por la documentación de los casos de prueba ejecutados, los resultados obtenidos mediante Google Lighthouse y la elaboración de un video de evidencia, en el cual se corrobora el correcto funcionamiento del proyecto BarBox. Dicho material, junto con los enlaces a los repositorios GitHub correspondientes a los sistemas Backoffice, POS y E-commerce, se encuentra disponible en la sección de Referencias del presente documento.

11.1 Evidencias

Video: https://youtu.be/9dUCYQn1WeE?si=6zOfdn18s4VDFcL2

Enlaces de Producción:

Backoffice: https://pos-sc6p.vercel.app/login

POS: https://pos-opal-three.vercel.app/

E-commerce: https://e-commerce-sepia-iota-99.vercel.app/

Enlaces de GitHub:

Backoffice: https://github.com/chuchobck/BackofficeFinalBarbox.git

POS: https://github.com/chuchobck/POS.git

Backend: https://github.com/chuchobck/backoffice-.git

E-commerce: https://github.com/chuchobck/E-commerce.git

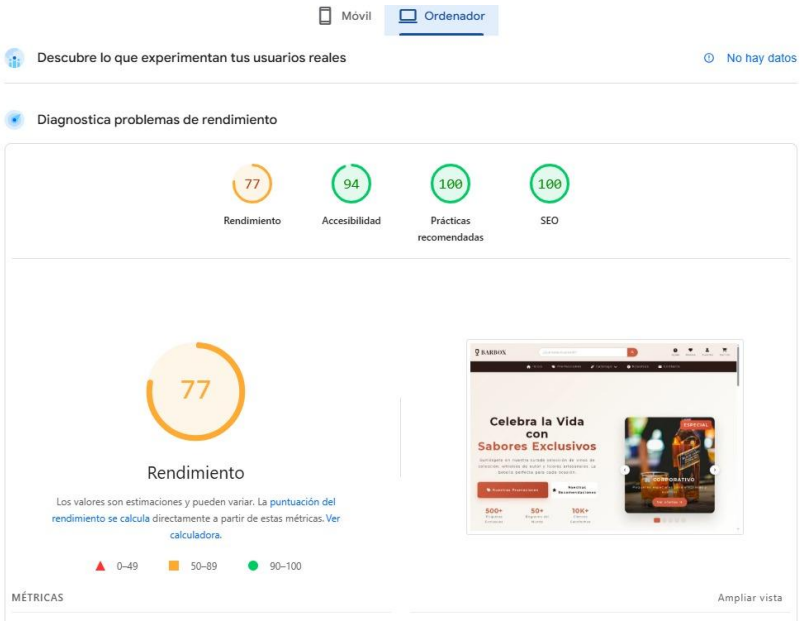

Figura 8. Resultados generales de rendimiento, accesibilidad, buenas prácticas y SEO del sistema E-commerce BarBox obtenidos mediante Google Lighthouse.

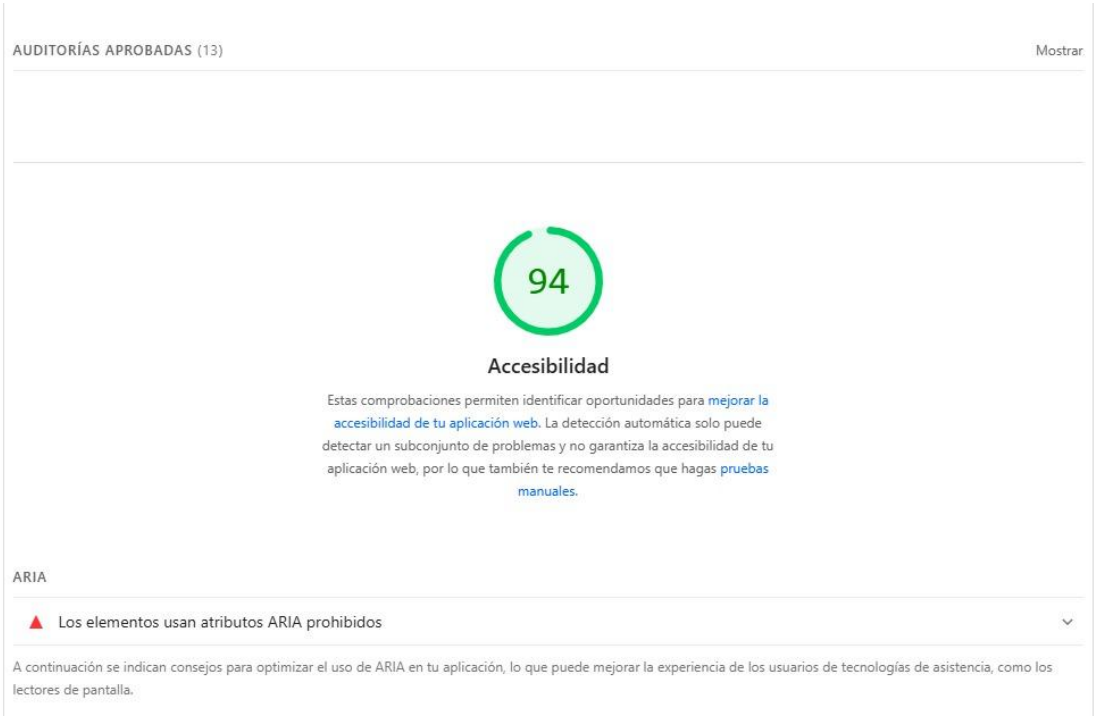

Figura 9. Resultado de la evaluación de accesibilidad del sistema E-commerce BarBox mediante Google Lighthouse.

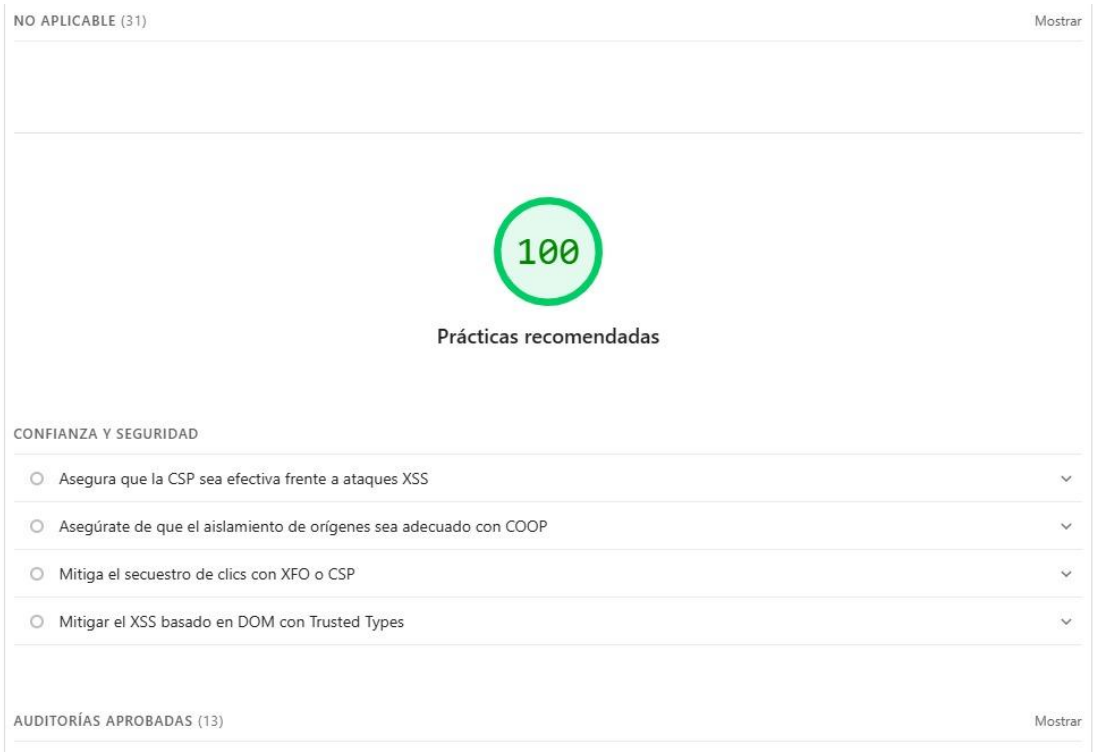

Figura 10. Resultado de la evaluación de buenas prácticas del sistema E-commerce BarBox mediante Google Lighthouse.

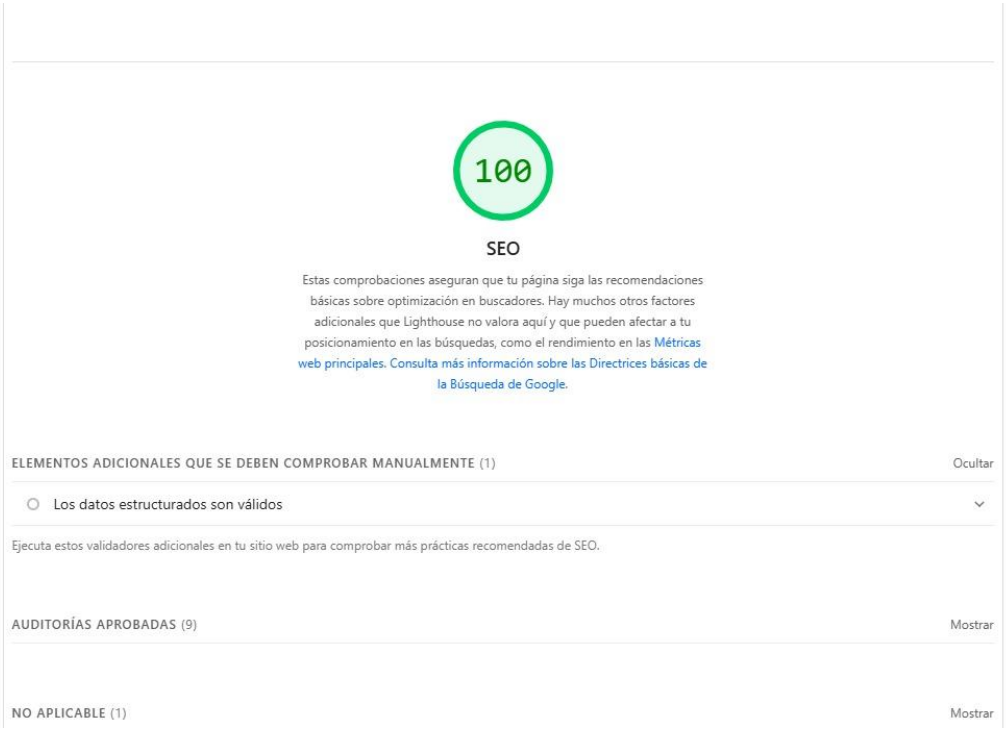

Figura 11. Resultado de la evaluación de SEO del sistema E-commerce BarBox mediante Google Lighthouse.

Figura 12. Métricas de rendimiento del sistema E-commerce BarBox obtenidas mediante Google Lighthouse (FCP, LCP, TBT, Speed Index y CLS).

12 Conclusiones

El desarrollo del sistema E-commerce BarBox permitió aplicar de forma práctica conceptos relacionados con el análisis, diseño, implementación y validación de un sistema de información, integrando módulos funcionales que responden a los procesos comerciales definidos dentro del alcance del proyecto académico.

La ejecución de pruebas unitarias, de integración, de aceptación y de usabilidad facilitó la evaluación del comportamiento del sistema frente a distintos escenarios de uso, permitiendo analizar la interacción entre los módulos y verificar su funcionamiento

conforme a los requisitos establecidos. De manera complementaria, la aplicación de pruebas no funcionales mediante Google Lighthouse permitió observar el desempeño del sistema en términos de rendimiento, accesibilidad, buenas prácticas y SEO dentro de un entorno controlado.

Desde el punto de vista académico, el proyecto permitió organizar el desarrollo del sistema siguiendo un proceso estructurado, aplicando la metodología TSP, documentando las fases del ciclo de vida y respaldando los resultados obtenidos mediante evidencia técnica y repositorios de código, lo que contribuyó a consolidar los conocimientos adquiridos durante el semestre.

13 Trabajo futuro

Como trabajo futuro se consideran posibles extensiones del sistema, tales como la integración de pasarelas de pago, funcionalidades de analítica y mejoras en la experiencia de usuario, las cuales se encuentran documentadas como requerimientos deseables en la documentación de la materia correspondiente.

14 Referencias

Angel Loor. (2026, 25 enero). *BarBox funcionamiento general* [Vídeo]. YouTube. https://www.youtube.com/watch?v=9dUCYQn1WeE

BARBOX - Panel administrativo. (s. f.). https://pos-sc6p.vercel.app/login

BARBOX - punto de venta. (s. f.). https://pos-opal-three.vercel.app/

Barbox - tienda online de bebidas premium. (s. f.). https://e-commerce-sepia-iota-99.vercel.app/

Chuchobck. (s. f.). *GitHub - chuchobck/backoffice-: Parte administrativa de BARBOX*. GitHub. https://github.com/chuchobck/backoffice-.git

Chuchobck. (s. f.). *GitHub - chuchobck/BackofficeFinalBarbox: Backoffice – parte administrativa de BARBOX*. GitHub. https://github.com/chuchobck/BackofficeFinalBarbox.git

Chuchobck. (s. f.-b). *GitHub - chuchobck/E-commerce: E-commerce del Proyecto Integrador*. GitHub. https://github.com/chuchobck/E-commerce.git

Chuchobck. (s. f.-b). *GitHub - chuchobck/POS: Punto de venta física de BARBOX*. GitHub. https://github.com/chuchobck/POS.git

Express.js. (2024). *Express documentation*. https://expressjs.com

Node.js Foundation. (2024). *Node.js documentation*. https://nodejs.org

PostgreSQL Global Development Group. (2024). *PostgreSQL documentation*. https://www.postgresql.org

Prisma. (2024). *Prisma ORM documentation*. https://www.prisma.io

React Team. (2024). *React documentation*. https://react.dev