



UNIVERSIDAD
CATÓLICA
DE CUENCA

UNIVERSIDAD CATÓLICA DE CUENCA

Comunidad Educativa al Servicio del Pueblo

**FACULTAD DE INFORMÁTICA, CIENCIAS DE LA
COMPUTACIÓN E INNOVACIÓN TECNOLÓGICA
CARRERA DE SOFTWARE**

**Implementación de un sistema de inventario y catálogo web responsivo
para pequeñas empresas comerciales**

**PROYECTO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL
TÍTULO DE INGENIERO DE SOFTWARE**

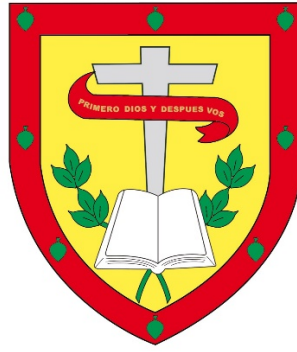
AUTOR: KEVIN ANDRES TARQUI TAPIA

DIRECTORA: ING. DIANA XIMENA POMA JAPÓN, MSC.

CUENCA - ECUADOR

2026

DIOS, PATRIA, CULTURA Y DESARROLLO



UNIVERSIDAD CATÓLICA DE CUENCA

Comunidad Educativa al Servicio del Pueblo

FACULTAD DE INFORMÁTICA, CIENCIAS DE LA COMPUTACIÓN E INNOVACIÓN TECNOLÓGICA

CARRERA DE SOFTWARE

**Implementación de un sistema de inventario y catálogo web responsivo
para pequeñas empresas comerciales**

**PROYECTO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL
TÍTULO DE INGENIERO DE SOFTWARE**

AUTOR: KEVIN ANDRES TARQUI TAPIA

DIRECTORA: ING. DIANA XIMENA POMA JAPÓN, MSC.

CUENCA - ECUADOR

2026

DIOS, PATRIA, CULTURA Y DESARROLLO



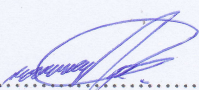
Universidad
Católica
de Cuenca

DECLARATORIA DE AUTORÍA

Declaratoria de Autoría y Responsabilidad

Kevin Andres Tarqui Tapia portador(a) de la cédula de ciudadanía N° **0150943934**. Declaro ser el autor de la obra: **“Implementación de un sistema de inventario y catálogo web responsivo para pequeñas empresas comerciales”**, sobre la cual me hago responsable sobre las opiniones, versiones e ideas expresadas. Declaro que la misma ha sido elaborada respetando los derechos de propiedad intelectual de terceros y eximo a la Universidad Católica de Cuenca sobre cualquier reclamación que pudiera existir al respecto. Declaro finalmente que mi obra ha sido realizada cumpliendo con todos los requisitos legales, éticos y bioéticos de investigación, que la misma no incumple con la normativa nacional e internacional en el área específica de investigación, sobre la que también me responsabilizo y eximo a la Universidad Católica de Cuenca de toda reclamación al respecto.

Cuenca, **04 de agosto de 2026**

F: 

Kevin Andres Tarqui Tapia

C.I. 0150943934

 Universidad Católica de Cuenca	CERTIFICADO DEL TUTOR
---	------------------------------

Yo, Diana Ximena Poma Japón, certifico que el presente proyecto de titulación, con el título **“Implementación de un sistema de inventario y catálogo web responsivo para pequeñas empresas comerciales”**, fue desarrollado por Kevin Andres Tarqui Tapia, con número de cédula 0150943934, bajo mi supervisión.



F: _____

Ing. Diana Ximena Poma Japón, MSc.

C.I. 1104101710

Dedicatoria

Dedico este trabajo, en primer lugar, a mis padres, quienes, de distintas maneras, han sido fundamentales en mi formación y en la culminación de esta etapa.

A mi madre, por acompañarme desde el hogar, sostenerme con su cariño, paciencia y apoyo constante, y por estar presente en los momentos en que necesité fortaleza para continuar.

A mi padre, por haber confiado en mí, brindarme la oportunidad de iniciar mis estudios universitarios y sostenerme con su apoyo a lo largo de esta etapa, permitiéndome avanzar hacia una de las metas más importantes de mi vida.

A ambos, gracias por sus esfuerzos, por todo lo que han hecho por mí y por contribuir, cada uno a su manera, a que hoy pueda alcanzar este logro. Esta meta también les pertenece.

Dedico también este trabajo al amor de mi vida, Gabriela Patiño, mi rayito de luz, quien, con su amor, apoyo y compañía, me motivó a seguir adelante. Gracias a ti comprendí que este esfuerzo ya no lo realizaba únicamente por mí, sino por los dos y por el futuro que anhelamos construir juntos. Junto a ti deseo compartir mi vida y formar una familia; ese anhelo me ha enseñado que, por más difícil, agotador o doloroso que pueda ser cada paso, no debo rendirme, porque los sueños que compartimos dan sentido a cada esfuerzo. Tu luz me brinda las fuerzas necesarias para continuar.

Qué bendición ha sido tu llegada a mi vida. Gracias por tu cariño, tu paciencia, cada momento compartido y todo el amor que me has brindado. Te amo.

Kevin Andres Tarqui Tapia

Agradecimiento

Agradezco, en primer lugar, a Dios, por iluminar mi camino, guiar mis decisiones y sostenerme cuando sentí que las fuerzas se agotaban. Gracias por acompañarme con su protección y permitirme culminar esta etapa de mi formación profesional.

A la Ing. Diana Ximena Poma Japón, directora de este trabajo de titulación, por su orientación, paciencia y disposición para compartir sus conocimientos. Sus observaciones, recomendaciones y acompañamiento fueron fundamentales tanto para fortalecer mi aprendizaje como para el desarrollo de esta tesis.

De manera especial, a los docentes Milton Campoverde, David Chicaiza, Jorge Carrión, Junior Wachapa y Bryan Vega, quienes marcaron distintas etapas de mi formación al guiarme desde los fundamentos de la disciplina hasta los contenidos técnicos más avanzados y las buenas prácticas de la Ingeniería de Software. Sus consejos, orientaciones y metodologías de trabajo fueron valiosos para consolidar mis capacidades y prepararme para el ejercicio profesional.

A los demás docentes de la carrera de Software, por compartir sus conocimientos y experiencias en las distintas áreas que integran la profesión. Cada asignatura aportó una perspectiva necesaria y complementó los aprendizajes que hicieron posible mi formación integral.

A la Abg. Marieta Zúñiga, Mgs., por su apoyo y acompañamiento durante los momentos difíciles de mi formación universitaria, por confiar en mis capacidades y por animarme a continuar con fortaleza. Su cercanía y sus palabras de aliento me ayudaron a perseverar y a seguir esforzándome hasta alcanzar esta meta.

A mis abuelitas, quienes, de distintas maneras, han dejado una huella profunda en mi vida. A mi abuelita con quien viví desde mi nacimiento, gracias por acompañarme de cerca durante mi crecimiento y enseñarme a sonreír aun en los momentos difíciles, así como a afrontar cada circunstancia con esperanza, serenidad y una actitud positiva. A mi otra abuelita, a quien visitaba y con quien compartí momentos que conservo con cariño, la recuerdo con profunda gratitud; sus enseñanzas continúan presentes en mi vida. Gracias a ambas por inculcarme valores como la humildad, el respeto, la honradez, la empatía, la generosidad y la perseverancia. Procuraré llevar siempre estos principios conmigo, tanto en mi vida personal como en mi carrera profesional.

Kevin Andres Tarqui Tapia

Resumen

En las pequeñas empresas comerciales, el control de productos, existencias y ventas suele apoyarse en registros manuales o dispersos, mientras la información ofrecida a los clientes se distribuye entre mensajes, redes sociales y consultas presenciales. Por ello, esta investigación tuvo como objetivo desarrollar un sistema web responsivo que integrara la gestión administrativa del inventario con un catálogo público de productos. Se aplicó un enfoque tecnológico y evaluativo, con desarrollo incremental mediante una adaptación de Scrum y una arquitectura cliente–servidor. La solución se construyó con React, TypeScript, NestJS y PostgreSQL, y se desplegó mediante contenedores en ambientes separados. Los resultados comprenden la definición y el cumplimiento de requerimientos funcionales y no funcionales, el diseño de un modelo relacional, la implementación de módulos para usuarios, estructura del negocio, catálogo, inventario, ventas y trazabilidad, así como una interfaz pública con búsqueda, filtros, disponibilidad y personalización. El funcionamiento se comprobó mediante pruebas automatizadas, validaciones operativas y evaluaciones de usabilidad en los contextos administrativo y público. Se concluye que Stockify centraliza la información operativa, mantiene la trazabilidad de las existencias y proporciona a los clientes un catálogo organizado y responsivo para consultar los productos disponibles.

Palabras clave: *gestión de inventario; catálogo digital; aplicación web; pequeñas empresas; usabilidad.*

Abstract

In small retail businesses, tracking products, inventory, and sales often relies on manual or scattered records, while the information provided to customers is disseminated through messages, social media, and in-person inquiries. Therefore, this research aimed to develop a responsive web system that integrates administrative inventory management with a public product catalog. A technological and evaluative approach was applied, with incremental development based on an adaptation of the Scrum framework and a client–server architecture. The solution was developed using React, TypeScript, NestJS, and PostgreSQL, and deployed using containerized services in separate environments. The results include the definition and fulfillment of functional and non-functional requirements, the design of a relational model, and the implementation of modules for users, business structure, catalog, inventory, sales, and traceability, as well as a public interface with search, filters, availability, and customization features. The system’s operation was validated through automated testing, operational validations, and usability evaluations in both administrative and public contexts. It is concluded that Stockify centralizes operational information, maintains inventory traceability, and provides customers with an organized and responsive catalog for browsing available products.

Keywords: *inventory management, digital catalog, web application, small businesses, usability*

Índice

Introducción	1
Capítulo I	2
Marco referencial	2
1.1. Planteamiento del problema	2
1.2. Justificación	3
1.3. Objetivos.....	4
1.4. Preguntas de investigación	5
Capítulo II	6
Revisión de la literatura	6
2.1. Estado del arte	6
2.1.1. Sistemas web para gestión de inventario	6
2.1.2. Catálogos digitales en pequeñas empresas	6
2.1.3. Sistemas web comerciales con gestión de ventas e inventario	7
2.1.4. Aplicaciones web orientadas a pequeñas empresas comerciales	7
2.1.5. Plataformas existentes de gestión de inventario y catálogo	9
2.1.6. Síntesis del estado del arte y delimitación del proyecto	10
2.2. Marco teórico referencial	11
2.2.1. Gestión de inventario y control de existencias	11
2.2.2. Sistema de información	11
2.2.3. Catálogo digital	11
2.2.4. Base de datos relacional.....	12
2.2.5. Arquitectura cliente-servidor	12
2.2.6. API REST y servicios web	12
2.2.7. Aplicaciones web responsivas	13
2.2.8. Seguridad, autenticación y control de acceso	13
2.2.9. Marco Scrum	13
2.2.10. Despliegue de aplicaciones web en servidor	14
2.2.11. Síntesis crítica de los fundamentos seleccionados.....	14
Capítulo III.....	16
Metodología	16
3.1. Enfoque metodológico y tipo de investigación	16
3.1.1. Negocios participantes	17
3.2. Metodología de desarrollo de software.....	17
3.2.1. Justificación de la metodología seleccionada	18
3.2.2. Aplicación de Scrum.....	18
3.2.3. Herramientas de apoyo y trazabilidad	19

3.2.4. Planificación y ejecución de sprints	19
3.3. Levantamiento y análisis de requerimientos	20
3.3.1. Técnicas y fuentes	20
3.3.2. Actores y perfiles	22
3.3.3. Requerimientos funcionales.....	22
3.3.4. Requerimientos no funcionales.....	24
3.3.5. Historias de usuario y criterios de aceptación	25
3.4. Diseño de la solución	26
3.4.1. Arquitectura lógica general.....	26
3.4.2. Diseño del modelo de datos	27
3.4.3. Diseño del backend.....	28
3.4.4. Diseño del frontend	29
3.4.5. Diseño del almacenamiento de imágenes	30
3.4.6. Seguridad y control de acceso	31
3.5. Estrategia de desarrollo e integración	32
3.5.1. Construcción incremental	33
3.5.2. Organización de repositorios	33
3.5.3. Ramas y merge requests.....	33
3.5.4. Integración continua y puertas de calidad	33
3.6. Estrategia de pruebas y validación	34
3.6.1. Pruebas técnicas	34
3.6.2. Evaluación de usabilidad	35
3.7. Estrategia de despliegue y entrega continua	36
3.7.1. Contenerización, construcción y Registry	36
3.7.2. Separación de QA y producción	36
3.7.3. Dominios, enrutamiento y multiinstancia.....	37
3.7.4. Automatización, aprobación productiva y recuperación	38
Capítulo IV	39
Desarrollo e implementación	39
4.1. Introducción	39
4.2. Entorno técnico y tecnologías	39
4.2.1. Base de datos y persistencia.....	39
4.2.2. Backend	40
4.2.3. Frontend	42
4.2.4. Infraestructura y DevOps	43
4.3. Arquitectura y fundamentos técnicos implementados	44
4.3.1. Arquitectura cliente–servidor y despliegue multiinstancia	44

4.3.2.	Modelo relacional y dominios de datos	45
4.3.3.	Persistencia, migraciones y seeds	51
4.3.4.	Backend y contratos API.....	53
4.3.5.	Frontend: rutas y gestión de estado.....	55
4.3.6.	Integración entre frontend y backend	57
4.3.7.	Almacenamiento y exposición de imágenes.....	58
4.4.	Área administrativa	58
4.4.1.	Usuarios, roles y permisos	58
4.4.2.	Biblioteca y selección de imágenes	60
4.4.3.	Negocio y estructura operativa	61
4.4.4.	Marcas, categorías, atributos, productos y variantes	64
4.4.5.	Inventario, movimientos, operaciones y transferencias.....	69
4.4.6.	Ventas, cambios, devoluciones y ajustes	73
4.5.	Landing institucional y catálogo público	77
4.5.1.	Landing institucional	77
4.5.2.	Estructura, rutas y navegación.....	77
4.5.3.	Inicio, catálogo, búsqueda y filtros	78
4.5.4.	Ficha, variantes e imágenes	79
4.5.5.	Disponibilidad.....	80
4.5.6.	Cuenta pública y lista de deseos	80
4.5.7.	Información comercial y contacto	82
4.5.8.	Diseño responsivo y personalización	82
4.6.	Seguridad y trazabilidad transversal	84
4.6.1.	Sesiones y control de acceso.....	84
4.6.2.	Auditoría y trazabilidad	85
4.7.	Integración, entrega continua y operación.....	86
4.7.1.	Canalizaciones y artefactos versionados	86
4.7.2.	Despliegue y provisionamiento controlados	88
4.7.3.	Exposición mediante Nginx y Cloudflare Tunnel.....	89
4.7.4.	Backups, restauración y verificación.....	89
Capítulo V		91
Resultados y análisis		91
5.1.	Introducción	91
5.2.	Cumplimiento de los requerimientos funcionales	91
5.3.	Pruebas automatizadas	92
5.3.1.	Backend	92
5.3.2.	Frontend	93

5.3.3. Pipelines CI	93
5.3.4. Verificaciones técnicas de infraestructura	94
5.4. Pruebas funcionales automatizadas	94
5.4.1. Área administrativa	94
5.4.2. Catálogo público	95
5.4.3. Controles transversales	95
5.5. Cumplimiento de los requerimientos no funcionales	96
5.5.1. Seguridad y contratos HTTP	97
5.5.2. Responsividad, persistencia y operación	97
5.6. Usabilidad	97
5.6.1. Participantes y condiciones	97
5.6.2. Resultados administrativos	98
5.6.3. Resultados del catálogo público	100
5.6.4. SUS y valoraciones complementarias	100
5.6.5. Dificultades y mejoras	101
5.6.6. Síntesis	101
5.7. Resultados del despliegue y la validación operativa	102
5.7.1. Promoción y provisionamiento	102
5.7.2. Servicios y rutas	102
5.7.3. Separación de recursos y respaldos	103
5.8. Cumplimiento de los objetivos	103
5.9. Discusión	104
5.10. Limitaciones de la evaluación	105
Conclusiones y recomendaciones	107
Conclusiones	107
Contribuciones del trabajo	108
Recomendaciones	110
Trabajos futuros	110
Referencias	112
Anexos	118

Lista de figuras

Fig. 1:	Arquitectura lógica general de Stockify	27
Fig. 2:	Flujo lógico de una operación en el backend	29
Fig. 3:	Flujo lógico general del frontend.....	30
Fig. 4:	Flujo lógico del almacenamiento de imágenes.....	31
Fig. 5:	Flujo de autenticación, renovación y autorización	32
Fig. 6:	Flujo documental de cada incremento	33
Fig. 7:	Flujo de despliegue y entrega continua de Stockify	38
Fig. 8:	Arquitectura cliente–servidor implementada	45
Fig. 9:	Relaciones del dominio de identidad y acceso	46
Fig. 10:	Relaciones del dominio de catálogo	47
Fig. 11:	Relaciones del dominio de estructura operativa	47
Fig. 12:	Relaciones del dominio de inventario	48
Fig. 13:	Relaciones del dominio de ventas	49
Fig. 14:	Relaciones del dominio de medios y presentación	50
Fig. 15:	Relaciones de las funciones públicas.....	51
Fig. 16:	Migraciones pendientes identificadas mediante TypeORM.....	52
Fig. 17:	Inicio de la ejecución de migraciones pendientes	52
Fig. 18:	Migraciones aplicadas después de la actualización del esquema.....	53
Fig. 19:	Ejecución del seed de datos iniciales.....	53
Fig. 20:	Configuración global de validación y normalización de errores.....	54
Fig. 21:	Documentación agrupada por contexto funcional.....	54
Fig. 22:	Contrato HTTP documentado para el registro administrativo de productos .	55
Fig. 23:	Proveedores globales que componen la aplicación de página única	55
Fig. 24:	Enrutamiento de las áreas pública y administrativa de la aplicación	56
Fig. 25:	Panel administrativo de gestión de usuarios	59
Fig. 26:	Panel administrativo de roles y permisos.....	60
Fig. 27:	Formulario con acceso reutilizable a la galería de imágenes	60
Fig. 28:	Biblioteca administrativa de imágenes y consulta de usos	61
Fig. 29:	Panel de información del negocio	62
Fig. 30:	Panel de configuraciones del negocio	62
Fig. 31:	Jerarquías admitidas por la estructura operativa	63
Fig. 32:	Panel administrativo de estructura operativa	64
Fig. 33:	Panel administrativo de marcas	65
Fig. 34:	Panel administrativo de categorías	65
Fig. 35:	Panel administrativo de atributos.....	66
Fig. 36:	Configuración de atributos aplicables a una categoría	67

Fig. 37: Panel administrativo de productos y variantes	68
Fig. 38: Listado de variantes asociado a un producto	68
Fig. 39: Modal y panel administrativo de colecciones	69
Fig. 40: Persistencia coordinada del saldo y su movimiento.....	70
Fig. 41: Consulta de existencias por producto, variante y ubicación	71
Fig. 42: Registro de una operación manual de entrada de stock	72
Fig. 43: Kardex detallado de movimientos de inventario.....	73
Fig. 44: Registro administrativo de una nueva venta	74
Fig. 45: Historial de ventas realizadas	75
Fig. 46: Confirmación de un cambio con cálculo del saldo comercial	76
Fig. 47: Historial de cambios y devoluciones	76
Fig. 48: Landing institucional de Stockify	77
Fig. 49: Página de inicio del catálogo público en vista móvil	78
Fig. 50: Catálogo público en vista móvil con búsqueda y filtros	79
Fig. 51: Ficha pública de producto en vista móvil	80
Fig. 52: Acceso de usuarios al catálogo público en vista móvil	81
Fig. 53: Cuenta pública en vista móvil.....	81
Fig. 54: Lista de deseos en vista móvil	82
Fig. 55: Catálogo público en vista de escritorio	83
Fig. 56: Mecanismos de auditoría y trazabilidad	86
Fig. 57: Canalización del frontend hacia QA.....	86
Fig. 58: Canalización del frontend hacia producción.....	87
Fig. 59: Imágenes versionadas del backend en GitLab Container Registry	87
Fig. 60: Imágenes versionadas del frontend en GitLab Container Registry	87
Fig. 61: Ejecución manual del trabajo de provisionamiento productivo	88
Fig. 62: Servicios de producción asociados a clientes provisionados	89
Fig. 63: Túneles configurados para los ambientes de QA y producción	89
Fig. 64: Verificación de restauración de respaldos mediante bases temporales	90

Lista de tablas

Tabla 1:	Síntesis crítica de trabajos académicos previos	8
Tabla 2:	Comparación de plataformas existentes	9
Tabla 3:	Síntesis del enfoque metodológico	16
Tabla 4:	Negocios participantes en el proyecto	17
Tabla 5:	Aplicación de los elementos de Scrum en Stockify	18
Tabla 6:	Herramientas de apoyo y trazabilidad	19
Tabla 7:	Planificación de fechas de los sprints	19
Tabla 8:	Fuentes empleadas para definir y refinar requerimientos	21
Tabla 9:	Actores e interacción principal en el sistema	22
Tabla 10:	Requerimientos funcionales.....	22
Tabla 11:	Requerimientos no funcionales	24
Tabla 12:	Historias de usuario representativas y criterios de aceptación.....	25
Tabla 13:	Dominios y entidades principales del modelo de datos	28
Tabla 14:	Controles de seguridad previstos en el diseño	31
Tabla 15:	Puertas de calidad definidas en integración continua	34
Tabla 16:	Niveles de prueba.....	35
Tabla 17:	Estrategia de despliegue por ambiente	36
Tabla 18:	Elementos aislados y compartidos en la multiinstancia	37
Tabla 19:	Componentes de la capa de persistencia implementada.....	40
Tabla 20:	Tecnologías principales del backend implementado	40
Tabla 21:	Responsabilidades principales del backend	41
Tabla 22:	Tecnologías principales del frontend implementado	42
Tabla 23:	Responsabilidades principales del frontend	42
Tabla 24:	Componentes de infraestructura y su función	43
Tabla 25:	Distribución de recursos en la arquitectura multiinstancia	45
Tabla 26:	Mecanismos de estado utilizados en el frontend	56
Tabla 27:	Variantes de imagen generadas por el backend.....	58
Tabla 28:	Reglas verificadas sobre la estructura operativa.....	63
Tabla 29:	Operaciones manuales y transferencias de inventario	70
Tabla 30:	Tratamiento de ventas y ajustes posteriores	74
Tabla 31:	Adaptaciones responsivas observadas en el catálogo	83
Tabla 32:	Elementos de autenticación y sesión.....	84
Tabla 33:	Capas de auditoría y trazabilidad	85
Tabla 34:	Artefactos incluidos y comprobaciones del respaldo	90
Tabla 35:	Requerimientos con evidencia automatizada ejecutada	91
Tabla 36:	Cumplimiento consolidado de los requerimientos funcionales.....	92

Tabla 37:	Resultados locales de pruebas y controles del backend	93
Tabla 38:	Resultados locales de pruebas y controles del frontend	93
Tabla 39:	Resultados de verificaciones técnicas de infraestructura.....	94
Tabla 40:	Recorridos automatizados ejecutados del área administrativa.....	95
Tabla 41:	Recorridos automatizados ejecutados del catálogo público	95
Tabla 42:	Controles transversales verificados mediante E2E	96
Tabla 43:	Cumplimiento de los requerimientos no funcionales	96
Tabla 44:	Distribución de participantes por negocio	98
Tabla 45:	Resultados de las tareas administrativas	99
Tabla 46:	Resultados de las tareas del catálogo público.....	100
Tabla 47:	Resultados de la escala SUS	101
Tabla 48:	Síntesis de los resultados de la validación operativa	103
Tabla 49:	Cumplimiento de los objetivos del proyecto	104
Tabla 50:	Actividades del formulario administrativo	119
Tabla 51:	Escala SUS del formulario administrativo.....	119
Tabla 52:	Valoración general del área administrativa	120
Tabla 53:	Registro de las tareas administrativas.....	121
Tabla 54:	Actividades del formulario del catálogo público.....	123
Tabla 55:	Escala SUS del formulario de catálogo público.....	123
Tabla 56:	Valoración general del catálogo público	124
Tabla 57:	Registro de las tareas del catálogo público	125
Tabla 58:	Repositorios y commits de la versión evaluada	126
Tabla 59:	Tecnologías, versiones y magnitudes de la versión evaluada	126
Tabla 60:	Trazabilidad individual de los requerimientos funcionales.....	127
Tabla 61:	Capacidades administrativas y control efectivo del backend.....	129
Tabla 62:	Disponibilidad pública derivada de una variante	129
Tabla 63:	Riesgos funcionales y condiciones verificables para su resolución	130
Tabla 64:	Pipelines y downstream relevantes verificados	131
Tabla 65:	Detalle sanitizado del VPS inspeccionado	131
Tabla 66:	Backups QA y producción inspeccionados.....	132
Tabla 67:	Detalle reproducible de las pruebas backend	132
Tabla 68:	Detalle reproducible de las pruebas frontend	132
Tabla 69:	Verificaciones técnicas de infraestructura	133

Introducción

Las pequeñas empresas comerciales necesitan gestionar las existencias de sus productos, conocer el stock disponible y consultar los movimientos de inventario, además de comunicar la oferta a sus clientes. Cuando esta información se mantiene en cuadernos, anotaciones aisladas, mensajes o publicaciones dispersas, resulta difícil consultar de forma ordenada las cantidades, precios y disponibilidad de los artículos.

Ante esta situación, se desarrolló Stockify, un sistema web responsivo que integra la gestión de inventario, ventas básicas y un catálogo web público. La solución se implementó en tres pequeños comercios de Cuenca con productos y estructuras diferentes. El área administrativa permite gestionar la información operativa, mientras que el catálogo facilita la consulta pública de productos, precios referenciales y disponibilidad.

El desarrollo se organizó de forma incremental mediante una adaptación Scrum. La solución utiliza React, NestJS, TypeScript y PostgreSQL, y su funcionamiento se evaluó mediante pruebas automatizadas, una evaluación de usabilidad con participantes y comprobaciones operativas de despliegue, respaldo y restauración.

El documento se organiza en cinco capítulos que comprenden el marco referencial, la revisión de la literatura, la metodología, el desarrollo e implementación, los resultados y su análisis. Finalmente, se presentan las conclusiones, contribuciones del trabajo, recomendaciones y trabajos futuros.

Capítulo I

Marco referencial

1.1. Planteamiento del problema

La incorporación de tecnologías digitales en los procesos comerciales forma parte de la transformación digital empresarial, debido a que puede incidir en la organización, los procesos productivos, la estrategia competitiva y el modelo de negocio [20]. En España, el Instituto Nacional de Estadística reportó que, en 2024, el 26,6 % de las empresas de 10 o más empleados realizó ventas mediante comercio electrónico; además, en el primer trimestre de 2025, el 44,3 % utilizó servicios de computación en la nube de pago y el 84,5 % de las empresas con conexión a internet disponía de sitio o página web [30].

En América Latina y el Caribe, la digitalización empresarial ha avanzado de manera desigual. La CEPAL define este proceso como una transformación de la organización, los procesos productivos, la estrategia competitiva y el modelo de negocio mediante la adopción de tecnologías digitales; además, señala que el comercio electrónico puede abrir nuevos canales de promoción y distribución, así como reducir costos de transacción. Por su parte, el Banco Interamericano de Desarrollo indica que el porcentaje de empresas que obtiene más de una quinta parte de sus ingresos por canales en línea pasó del 24 % al 45 % durante el impulso generado por la pandemia, aunque con resultados heterogéneos en la región [20,27].

En Ecuador, la evidencia estadística muestra que la incorporación de TIC en las empresas comerciales no siempre se traduce en una digitalización integral de sus procesos. Como antecedente, el INEC reportó que, en 2015, solo el 10,4 % de las empresas del sector comercio realizó ventas por internet. De forma más reciente, la ENESEM 2022 mantiene la medición oficial del uso de TIC en empresas, y los tabulados del sector comercio revisados para esta investigación evidencian alta conectividad, mayor presencia de ventas por internet y una adopción más limitada de software de gestión e inventario [10,28].

Para esta investigación se considero a tres pequeñas empresas comerciales de la ciudad de Cuenca, provincia del Azuay: Ely's Store, dedicada a la venta de ropa para hombres; Light Blue, orientada a accesorios, cosméticos y artículos variados; y Veterinaria Manchas, que comercializa alimentos, juguetes, ropa y otros productos para mascotas. Ely's Store y Light Blue operaban sin sucursales, mientras que Veterinaria Manchas disponía de una sucursal y una bodega interna. Con todo lo observado, estos negocios permitieron levantar los requerimientos conforme a las necesidades relacionadas con productos físicos organizados mediante categorías y variantes.

En los negocios participantes, el registro de productos, entradas, ventas, cantidades dispo-

nibles y algunos precios se apoyaba en cuadernos, anotaciones no sistematizadas y datos recordados por las propietarias. Cuando existían dudas sobre una cantidad, se verificaban físicamente determinados productos; para algunos precios también se consultaba el valor registrado en la compra al proveedor. Estas prácticas dificultaban mantener una consulta organizada de las existencias, los precios y los movimientos de mercadería. La literatura de gestión de inventarios coincide en que la ausencia de registros estructurados limita el conocimiento de los saldos y la trazabilidad de las existencias [4, 11].

Ninguno de los tres negocios disponía de un catálogo web propio. La información sobre artículos, precios, variantes o disponibilidad se comunicaba mediante mensajes directos, redes sociales, fotografías aisladas o consultas presenciales, lo que dificultaba mantener una fuente única y controlada por cada negocio. En términos generales, un catálogo digital estructurado y responsivo permite organizar la oferta y facilitar su consulta desde distintos dispositivos [5, 41].

Frente a esta problemática, se plantea el desarrollo de un sistema web responsivo que integre la gestión interna del inventario con un catálogo web oficial del negocio, permitiendo controlar la disponibilidad de productos y ofrecer a los clientes información actualizada, ordenada y accesible desde distintos dispositivos.

1.2. Justificación

La implementación de un sistema web de inventario y catálogo público resulta relevante desde el punto de vista operativo. En los tres negocios participantes de Cuenca se identificó la necesidad de organizar la información de productos, cantidades, precios y movimientos de mercadería para facilitar su consulta y seguimiento. En términos generales, la gestión de inventarios requiere registros organizados para conocer los saldos y conservar la trazabilidad de las existencias [4, 11]. Un sistema de inventario proporciona mecanismos para registrar, actualizar y consultar estos datos de forma centralizada.

Desde el punto de vista tecnológico, Stockify se plantea como un sistema de información orientado a recopilar, procesar, almacenar y distribuir datos relevantes del negocio para apoyar sus operaciones administrativas y comerciales [9]. Además, el uso de una base de datos relacional permite organizar información de productos, usuarios, ventas e inventario manteniendo consistencia e integridad en los datos [7]. Finalmente, la arquitectura cliente-servidor permite separar la interfaz de usuario, la lógica del negocio y el acceso a datos, favoreciendo el mantenimiento y la escalabilidad de la solución [1].

Desde el punto de vista comercial, el catálogo digital permite que el negocio cuente con un canal propio y oficial para presentar sus productos, variantes, imágenes, precios

referenciales, disponibilidad e información de contacto. Esto fortalece la visibilidad de los artículos ofertados y facilita que los clientes consulten información actualizada sin depender únicamente de mensajes directos, publicaciones aisladas en redes sociales o atención presencial. Además, el diseño responsivo permite que la información pueda consultarse desde computadoras, tabletas y teléfonos móviles, mejorando el acceso a la oferta comercial [5,41].

Desde el punto de vista académico, el proyecto permite aplicar conocimientos de ingeniería de software en el análisis de requerimientos, diseño de sistemas, modelado de base de datos, desarrollo web, pruebas funcionales, pruebas de usabilidad y despliegue de aplicaciones. Por tanto, la investigación posee relevancia académica y tecnológica al proponer una solución informática viable para resolver una problemática real del entorno comercial local.

Finalmente, desde una perspectiva práctica y local, la propuesta se orienta a pequeñas empresas comerciales que requieren herramientas funcionales y adaptadas a sus necesidades de gestión [16,21]. En este sentido, Stockify busca contribuir al control administrativo, a la organización de productos y a la consulta de la oferta comercial mediante una solución web responsiva. El alcance aplicado se evaluó en tres negocios de Cuenca y, por tanto, no permite generalizar sus resultados a todas las pequeñas empresas.

1.3. Objetivos

Objetivo general

Desarrollar un sistema web responsivo para la gestión de inventario, con el fin de mejorar el control administrativo del negocio y facilitar a los clientes el acceso a información actualizada sobre los artículos ofertados.

Objetivos específicos

1. Analizar los requerimientos funcionales y no funcionales relacionados con la gestión de inventario y la visualización de productos en pequeñas empresas comerciales.
2. Diseñar una base de datos relacional SQL que permita organizar, almacenar, actualizar y recuperar la información de productos, usuarios, ventas e inventario.
3. Diseñar la arquitectura del sistema para definir la estructura y el comportamiento que tendrá el software.
4. Desarrollar un sistema web responsivo con funcionalidades administrativas para el registro, edición, control de stock, categorización y visibilidad de productos, y con

funcionalidades públicas para la consulta de productos.

5. Validar el funcionamiento del sistema mediante pruebas funcionales y de usabilidad para comprobar el cumplimiento de los requerimientos establecidos.

1.4. Preguntas de investigación

1. ¿Qué requerimientos funcionales y no funcionales debe cumplir un sistema web responsivo orientado a la gestión de inventario y visualización de productos en pequeñas empresas comerciales?
2. ¿Cómo debe diseñarse una base de datos relacional (SQL) que permita organizar, almacenar, actualizar y recuperar la información de productos, usuarios, ventas e inventario?
3. ¿Qué arquitectura de software permite definir de manera adecuada la estructura y el comportamiento de un sistema web responsivo para la gestión de inventario y catálogo público?
4. ¿Cómo puede desarrollarse un sistema web responsivo con funcionalidades administrativas para el registro, edición, control de stock, categorización y visibilidad de productos, junto con funcionalidades públicas para la consulta de productos?
5. ¿Qué nivel de cumplimiento funcional y usabilidad presenta el sistema desarrollado al ser validado mediante pruebas funcionales y de usabilidad?

Capítulo II

Revisión de la literatura

2.1. Estado del arte

2.1.1. Sistemas web para gestión de inventario

Los sistemas web para gestión de inventario responden a la necesidad de registrar, consultar y actualizar de forma oportuna la información de productos, existencias y movimientos de mercadería; en pequeñas empresas comerciales, el control manual mediante cuadernos, hojas de cálculo aisladas o verificaciones físicas frecuentes dificulta identificar el stock real disponible, retrasa la atención al cliente y limita la trazabilidad de entradas, salidas y ajustes [4, 11]. La incorporación de sistemas de información orientados al inventario permite organizar los datos del negocio y apoyar las operaciones administrativas y comerciales [9].

Entre los antecedentes revisados, el trabajo desarrollado para la empresa JAVSA S.A. integró un sistema web de control de inventarios con un catálogo en línea, por lo que aporta una referencia directa sobre la centralización de productos y oferta comercial con apoyo de Scrum [13]. La investigación de ESPOCH sobre un sistema web con ReactJS para control de inventario aporta un antecedente más reciente sobre la articulación entre interfaz web, administración de productos y metodología ágil [25]. Ambos estudios demuestran la viabilidad de soluciones construidas para organizaciones concretas, pero no resuelven por sí solos la necesidad de configurar una misma solución para negocios con estructuras y tipos de productos diferentes. Por esta razón, Stockify retoma la gestión integrada y el desarrollo incremental, e incorpora categorías, variantes, precios, imágenes, disponibilidad, stock y movimientos como elementos configurables.

2.1.2. Catálogos digitales en pequeñas empresas

El catálogo digital presenta de forma organizada productos o servicios mediante información textual, imágenes y datos comerciales. En pequeñas empresas permite mostrar la oferta de manera accesible y controlada por el propio negocio, sin depender únicamente de publicaciones dispersas en redes sociales o atención presencial [5, 41]. La investigación sobre el catálogo electrónico para la empresa Dolarín aporta criterios para estructurar con claridad la información de productos [5]; además, las prácticas de diseño responsivo permiten presentar esa información adecuadamente en computadoras, tabletas y teléfonos móviles [41].

Para el problema estudiado, la importancia del catálogo no reside únicamente en publicar productos, sino en evitar una segunda fuente de información separada del inventario. Por

ello, se eligió un catálogo propio vinculado con la administración interna: su ventaja es reutilizar productos, variantes, imágenes, precios referenciales y disponibilidad desde una fuente común. Su delimitación es igualmente relevante: el catálogo facilita la consulta y el contacto, pero no procesa pagos, reservas ni pedidos en línea.

2.1.3. Sistemas web comerciales con gestión de ventas e inventario

Los sistemas web comerciales suelen integrar inventario, ventas, administración de productos y consulta de catálogo, de modo que las operaciones internas y la información visible para el público se mantengan relacionadas. El antecedente de TURBOMEKANICS S.A. se orientó a una aplicación web de catálogos privados bajo arquitectura MVC, con énfasis en organización de productos y consulta estructurada por clientes [12]; el trabajo para la ferretería Lascano propuso una tienda online con administración web para ventas, productos e inventario, centrada en un único negocio [22]; y el sistema dirigido al proceso de ventas de Silvagro WSV S.A. incorporó catálogo, control de stock, reportes, facturación y devoluciones en una arquitectura de tres capas [26].

La comparación muestra dos decisiones pertinentes para Stockify. La primera es relacionar las ventas con los movimientos de inventario, porque una venta registrada debe afectar las existencias y conservar trazabilidad. La segunda es limitar la amplitud funcional: aunque los antecedentes incorporan catálogos privados, tiendas o facturación, Stockify se concentra en inventario, ventas básicas y catálogo público. Esta delimitación permite atender el problema identificado sin extender el proyecto a facturación electrónica, contabilidad integral o comercio electrónico transaccional.

2.1.4. Aplicaciones web orientadas a pequeñas empresas comerciales

Las pequeñas empresas comerciales pueden operar con recursos técnicos, financieros y humanos limitados, por lo que la pertinencia de una solución depende tanto de sus funciones como de su alcance operativo. La literatura sobre digitalización de mipymes destaca que la adopción de tecnologías digitales puede transformar procesos, modelos de negocio y canales de comercialización, aunque su avance es desigual y depende de las capacidades internas de cada empresa [20, 21, 27]; la OECD reconoce igualmente que las pymes requieren condiciones y herramientas para mejorar productividad e innovación en mercados digitalizados [16]. En América Latina y el Caribe, la digitalización empresarial ha sido impulsada por la necesidad de ampliar canales de venta y mejorar la relación con clientes [20, 27], y en el ámbito comercial se asocia con procesos de venta y atención apoyados por medios digitales [24].

En los casos analizados, las necesidades se concentraron en ordenar productos, controlar existencias, registrar ventas básicas y presentar la oferta en línea, no en incorporar todos los módulos de una plataforma empresarial. Una aplicación web responsiva fue seleccionada porque permite utilizar las funciones administrativas y públicas desde diferentes dispositivos mediante un navegador. Esta alternativa evita instalaciones específicas por dispositivo, aunque mantiene la dependencia de conectividad y de una infraestructura central disponible.

Las limitaciones de los antecedentes no invalidan sus resultados, sino que delimitan el contexto en el que fueron obtenidos y su correspondencia con el problema de este proyecto. La Tabla 1 reúne los aportes, los mecanismos de construcción o evaluación, las restricciones pertinentes y la respuesta diferenciada de Stockify.

Tabla 1: Síntesis crítica de trabajos académicos previos

Antecedente	Aporte y alcance	Construcción y evaluación	Limitación pertinente	Respuesta de Stockify
Dolarín [5]	Integra clientes, productos, inventario de comercio electrónico, catálogo, pedidos, entrega y cobro a domicilio.	Especificación de requisitos IEEE 830 y modelado web UWE.	Caso empresarial único y alcance transaccional; el pago en línea quedó planteado como mejora.	Integra inventario y ventas básicas con un catálogo público informativo, sin pedidos ni pagos, en configuraciones diferenciadas por negocio.
TURBOMEKANICS S.A. [12]	Centraliza fuentes Sybase y Access en un catálogo privado con acceso y precios diferenciados por cliente.	RUP, MVC, entrevistas, observación y pruebas de caja blanca, caja negra y comprobación funcional.	Caso automotriz único y catálogo restringido; la comprobación final se concentra en cuatro condiciones funcionales, sin métricas operativas ni estudio formal de usabilidad.	Incorpora catálogo abierto, configuración por negocio, inventario y ventas trazables, además de validación técnica, de uso y de operación.
JAVSA S.A. [13]	Integra materias primas, productos terminados, pedidos, salidas, devoluciones, alertas y catálogo adaptable.	Scrum en siete sprints y validación de historias con la responsable del producto.	Modelo propio de una manufacturera; los beneficios no se contrastan con una línea base cuantitativa ni se aplica la solución en varios negocios.	Aplica una base configurable en tres comercios heterogéneos y complementa la integración inventario-catálogo con variantes, ubicaciones y evaluación en varias capas.
Ferretería Lascano [22]	Combina una aplicación Android para catálogo y pedidos con una aplicación web administrativa para productos, proveedores, entradas y reportes.	Estudio exploratorio y diagnóstico, entrevista, observación, desarrollo incremental y diez casos funcionales.	Caso único, dependencia de una aplicación móvil y alcance orientado a pedidos; la optimización declarada no se cuantifica mediante una comparación antes/después.	Ofrece interfaces responsivas desde el navegador y un catálogo no transaccional, con configuración para varios negocios y evaluación con participantes.

Óptica Optimus [25]	Gestiona usuarios, productos, proveedores, clientes, compras, pedidos y ventas; cuantifica el rendimiento del sistema.	Scrum y 384 mediciones de solicitudes, con análisis estadístico y promedio de respuesta de 0,534 s.	Caso único y sin catálogo público; el indicador mide latencia informática, no ahorro de tiempo operativo ni usabilidad.	Vincula el catálogo público con los datos operativos y combina pruebas técnicas con evaluación de usabilidad y validación del despliegue.
Silvagro WSV S.A. [26]	Integra catálogo, carrito, stock, facturas, reportes y devoluciones mediante una arquitectura de tres capas.	Entrevista, definición de requisitos funcionales y no funcionales, y arquitectura con Java y PostgreSQL.	Solución delimitada al proceso de ventas de una organización y con un alcance transaccional más amplio que el definido para este proyecto.	Prioriza inventario, ventas básicas y consulta pública sin carrito, pedidos, pagos ni facturación; aplica configuraciones diferenciadas en tres negocios.

Fuente: Autor.

2.1.5. Plataformas existentes de gestión de inventario y catálogo

En el mercado existen plataformas consolidadas con funciones de inventario, catálogo, ventas y comercio digital que representan enfoques distintos. Odoo es una suite empresarial modular cuyo inventario cubre almacenes, rutas y operaciones logísticas avanzadas, con comercio electrónico y catálogo integrados [42–44]. Shopify se orienta a la creación y administración de tiendas online, con inventario, productos con variantes, ubicaciones y tienda en línea documentados oficialmente [45–48]. Tiendanube, plataforma latinoamericana de comercio electrónico, documenta lista de productos, variantes, stock y carga masiva [36, 49–51]. Amazon actúa como marketplace: sus recursos para vendedores se centran en listados, inventario y herramientas como Seller Central o tiendas de marca; la gestión se realiza dentro de su infraestructura comercial [29, 33–35]. La Tabla 2 sintetiza la comparación y la relación de cada enfoque con Stockify.

Tabla 2: Comparación de plataformas existentes

Plataforma	Enfoque principal	Inventario	Catálogo / tienda	Ventas	Aporte tomado como referencia	Diferencia respecto de Stockify
Odoo	ERP modular con inventario, ventas y comercio electrónico	Almacenes y operaciones logísticas	Sí, mediante catálogo y tienda online	Sí	Organización de productos, ubicaciones y operaciones de inventario.	Stockify adopta un alcance funcional acotado y no incorpora módulos ERP.
Shopify	Plataforma de tienda online y comercio multicanal	Sí, con variantes y ubicaciones	Sí, tienda online propia	Sí	Productos con variantes, ubicaciones y presentación de una tienda propia.	Stockify prioriza el control interno y un catálogo sin compra en línea.
Tiendanube	Plataforma latinoamericana de tienda online	Sí, con stock, variantes y carga masiva	Sí, catálogo y tienda online	Sí	Manejo de variantes, existencias y administración de productos.	Stockify no incorpora pagos, pedidos ni la operación de una tienda transaccional.

Amazon	Marketplace y ecosistema de venta para terceros	Asociado a listados e inventario del vendedor	Sí, dentro del marketplace y tiendas de marca	Sí	Estructuración de listados, atributos, imágenes y variantes.	Stockify publica un catálogo propio y no intermedia transacciones como marketplace.
--------	---	---	---	----	--	---

Fuente: Autor.

2.1.6. Síntesis del estado del arte y delimitación del proyecto

La revisión permite establecer que la integración entre inventario, catálogo digital y gestión comercial ha sido abordada desde distintas perspectivas. Los trabajos académicos muestran la viabilidad de sistemas web para gestionar productos, existencias, ventas y catálogos en organizaciones específicas [12, 13, 22, 26]. Como sintetiza la Tabla 1, estos trabajos aportan referencias útiles sobre la integración de inventario, catálogo y gestión comercial; sin embargo, fueron desarrollados para organizaciones y procesos específicos, lo que limita su transferencia directa a pequeños comercios con productos y estructuras operativas diferentes. Por otra parte, las plataformas comerciales presentan funciones consolidadas de inventario, comercio electrónico y *marketplace* [34, 44, 46, 50], pero responden a alcances más amplios o a modelos de operación diferentes.

El análisis comparativo permitió identificar que los antecedentes resuelven necesidades de inventario, catálogo o ventas en organizaciones específicas. Sin embargo, entre los trabajos revisados no se identificó una solución que reuniera tres características: una base de software configurable para distintos pequeños comercios; la integración de inventario y ventas básicas con un catálogo público; y una evaluación conjunta del funcionamiento técnico, la usabilidad y la operación desplegada. Esta brecha permite precisar el aporte de Stockify dentro del conjunto analizado. En cuanto a las soluciones comerciales, un ERP, una tienda online completa y un *marketplace* incluyen capacidades valiosas, pero también funciones que no formaron parte de las necesidades obtenidas en los tres negocios participantes. La literatura sobre digitalización de mipymes respalda la importancia de seleccionar soluciones pertinentes a las capacidades y necesidades de estas empresas [20, 21, 27].

En este contexto, se eligió desarrollar Stockify como un sistema web responsivo enfocado en inventario interno, ventas básicas y catálogo público para pequeñas empresas comerciales, con o sin local físico. La ventaja esperada de esta selección es mantener productos, variantes, existencias, ventas e información pública en una fuente relacionada, sin depender del modelo comercial de un marketplace. Su propósito no es reemplazar un ERP ni una plataforma de comercio electrónico transaccional. Se excluyen la facturación electrónica, la pasarela de pagos, la contabilidad integral y la logística avanzada para conservar un alcance coherente con las necesidades identificadas y verificable dentro del proyecto.

2.2. Marco teórico referencial

2.2.1. Gestión de inventario y control de existencias

La gestión de inventario comprende el registro, control y seguimiento de las existencias de una empresa para conocer qué productos están disponibles, cuándo deben reponerse y cómo se relacionan con las operaciones de compra y venta; una gestión adecuada reduce faltantes, evita acumulaciones innecesarias y sostiene la continuidad comercial [4, 11]. El stock, por su parte, corresponde al conjunto de bienes disponibles para venta, uso o distribución, y su control permite conocer la cantidad real de productos, prevenir desabastecimientos e identificar excesos [4, 11].

El control de existencias exige registrar entradas, salidas y ajustes asociados a cada producto o variante. Esta trazabilidad diferencia la disponibilidad real, los productos agotados y los de bajo stock. En el levantamiento del proyecto se identificaron registros manuales o dispersos y verificaciones físicas como prácticas de control; en términos generales, la ausencia de registros organizados dificulta conocer los saldos y mantener la trazabilidad de las existencias [4, 11]. Por ello, Stockify centraliza la gestión del inventario por productos, variantes y ubicaciones. Esta organización facilita la consulta de existencias y la trazabilidad de sus movimientos, aunque depende del registro consistente de las operaciones.

2.2.2. Sistema de información

Un sistema de información es un conjunto organizado de recursos humanos, tecnológicos, datos y procedimientos que permite recopilar, procesar, almacenar y distribuir información para apoyar operaciones, coordinación, control y toma de decisiones [9]. Su importancia para este proyecto radica en que el problema no se limita al almacenamiento de productos, sino que relaciona usuarios, ventas, inventario y catálogo. Stockify adopta este enfoque para centralizar dichos datos y ofrecer consultas según el tipo de actor. La ventaja frente a registros separados es la conservación de relaciones entre operaciones; sin embargo, el sistema no sustituye los procedimientos del negocio ni garantiza por sí solo la calidad de los datos ingresados.

2.2.3. Catálogo digital

El catálogo digital es una presentación electrónica organizada de productos o servicios que incorpora descripciones, imágenes y datos comerciales para facilitar su consulta [5]. Se seleccionó un catálogo propio porque los negocios participantes no disponían de una fuente web estructurada y porque sus productos requerían categorías, variantes, fotografías, precios referenciales y disponibilidad. Vincularlo con la administración interna

evita mantener un segundo registro de productos y permite derivar la disponibilidad del inventario. Su alcance es informativo: facilita la consulta y el contacto, pero no equivale a una tienda con pedidos y pagos en línea.

2.2.4. Base de datos relacional

Una base de datos relacional organiza la información en tablas relacionadas mediante claves primarias y foráneas, lo que permite mantener integridad, consistencia y baja redundancia [7, 15]. Este modelo fue elegido porque Stockify maneja relaciones y operaciones transaccionales entre usuarios, roles, productos, variantes, ubicaciones, ventas y movimientos de inventario. Frente a un almacenamiento sin relaciones explícitas, ofrece restricciones y transacciones apropiadas para impedir asociaciones inválidas y coordinar cambios de stock. Como contrapartida, exige diseñar y evolucionar el esquema mediante migraciones controladas.

2.2.5. Arquitectura cliente-servidor

La arquitectura cliente-servidor organiza el software de modo que un cliente solicita servicios o recursos a un servidor, el cual procesa la petición y devuelve una respuesta; esta separación entre interfaz, lógica del negocio y acceso a datos facilita el mantenimiento y la escalabilidad [1]. En una aplicación web, el cliente es el navegador o la interfaz frontend, mientras el servidor concentra la lógica de negocio, la autenticación, la autorización y la comunicación con la base de datos. Este modelo fue seleccionado para que el área administrativa y el catálogo público compartan reglas y datos sin duplicarlos en cada interfaz. Su ventaja es la separación de responsabilidades; su limitación es la dependencia del servidor y de la conectividad para ejecutar las operaciones.

2.2.6. API REST y servicios web

REST es un estilo arquitectónico propuesto por Fielding para sistemas hipermedia distribuidos, orientado a mejorar escalabilidad, simplicidad de interfaces e independencia de componentes [3]; en el desarrollo web se utiliza para diseñar APIs que exponen recursos mediante identificadores y operaciones del protocolo HTTP, cuya semántica de métodos, códigos de estado y representación de recursos define el intercambio de mensajes [23]. Se eligió una API REST para establecer un contrato uniforme entre el frontend y el backend y separar la evolución de ambas aplicaciones. Esta interfaz permite centralizar la validación de usuarios, productos, inventario, ventas y catálogo, aunque requiere controlar explícitamente autenticación, errores y compatibilidad de los contratos.

2.2.7. Aplicaciones web responsivas

Una aplicación web responsiva adapta su interfaz a diferentes tamaños de pantalla y dispositivos, manteniendo la legibilidad y funcionalidad de la experiencia de usuario. MDN Web Docs describe el diseño responsivo como un conjunto de prácticas para adaptar la disposición del contenido a distintos contextos de visualización [41]. Esta alternativa fue seleccionada porque las funciones administrativas y públicas debían poder utilizarse desde computadoras y teléfonos sin desarrollar aplicaciones nativas independientes. La ventaja es ofrecer una misma base funcional mediante el navegador; sin embargo, la adaptación visual debe comprobarse en resoluciones y dispositivos diferentes.

2.2.8. Seguridad, autenticación y control de acceso

La seguridad en aplicaciones web protege datos, funciones y recursos frente a accesos no autorizados: la autenticación verifica la identidad del usuario y la autorización determina qué acciones puede realizar. JSON Web Token (JWT) define un formato compacto para transmitir información verificable mediante objetos JSON firmados [6]; su uso debe seguir las recomendaciones del RFC 8725, que exige validación estricta de algoritmos, emisores y audiencias para evitar configuraciones inseguras o tokens manipulados [17]. Por ello, la autenticación basada en tokens debe complementarse con controles de sesión, expiración y validación del lado del servidor.

El control de acceso requiere además reglas de autorización claras. OWASP recomienda políticas explícitas, validación de permisos en el servidor y enfoques restrictivos por defecto [31], y mantiene el control de acceso roto como categoría crítica de riesgo por fallas que permiten acceder a funciones o datos fuera de los privilegios asignados [32]. Estos principios fueron seleccionados porque Stockify combina recursos públicos con operaciones administrativas que modifican inventario, ventas y configuración. La separación de sesiones, roles y permisos reduce la exposición de funciones sensibles, aunque requiere controles coordinados en cada solicitud del servidor y una gestión adecuada de las credenciales.

2.2.9. Marco Scrum

Scrum es un marco de trabajo ágil orientado a generar valor mediante soluciones adaptativas a problemas complejos. La Guía Scrum 2020 organiza el trabajo en periodos de duración fija denominados *sprints* y define artefactos que favorecen la transparencia, la inspección y la adaptación. El *Product Backlog* mantiene ordenada la lista de funcionalidades, requisitos y mejoras necesarias para el producto; el *Sprint Backlog* reúne el

objetivo del sprint, los elementos seleccionados del *Product Backlog* y el plan de trabajo para desarrollarlos; y el Incremento integra los elementos terminados que cumplen los criterios establecidos [18]. Estos artefactos son importantes porque hacen visible el alcance pendiente, el trabajo previsto para el periodo actual y el resultado obtenido al finalizarlo.

Scrum fue elegido para Stockify porque sus funcionalidades podían priorizarse y desarrollarse progresivamente en módulos relacionados. El *Product Backlog* permitió organizar las funcionalidades generales del sistema, mientras que el *Sprint Backlog* delimitó las tareas correspondientes a cada periodo de trabajo. Como ventajas, esta organización facilitó dividir el alcance, conservar la trazabilidad de las actividades, verificar incrementos funcionales y ajustar la planificación conforme avanzaba el desarrollo.

2.2.10. Despliegue de aplicaciones web en servidor

El despliegue consiste en preparar y publicar sus componentes (backend, frontend, base de datos, variables de entorno, certificados y almacenamiento de archivos) en un entorno accesible por los usuarios; una estrategia consistente reduce diferencias entre desarrollo y producción. Los contenedores empaquetan una aplicación con sus dependencias para ejecutarse de forma aislada y portable: Docker define el contenedor como una unidad ligera y estandarizada que ejecuta software de manera consistente en distintos entornos [37]. Se seleccionaron contenedores para reproducir la composición del backend, el frontend y los servicios asociados, además de separar los recursos de cada negocio desplegado. Esta portabilidad facilita el aprovisionamiento, pero introduce tareas de configuración, supervisión y respaldo que deben documentarse.

2.2.11. Síntesis crítica de los fundamentos seleccionados

Los conceptos revisados no se incorporaron como elementos independientes. La gestión de inventario define el problema operativo; el sistema de información y la base relacional permiten centralizar datos y conservar sus relaciones; la arquitectura cliente–servidor y la API REST separan las interfaces de las reglas del negocio; y el diseño responsivo hace accesibles el área administrativa y el catálogo desde distintos dispositivos. La autenticación y el control de acceso protegen esa separación entre funciones públicas y administrativas, mientras Scrum y los contenedores organizan, respectivamente, la construcción incremental y la entrega reproducible.

La selección conjunta ofrece como ventaja una única fuente operativa para productos, variantes, existencias, ventas y publicación del catálogo. No obstante, también establece límites: el sistema depende de datos ingresados correctamente, conectividad, operación

del servidor y mantenimiento técnico. Estas decisiones se consideran adecuadas para el alcance definido porque atienden las necesidades identificadas sin incorporar facturación electrónica, contabilidad, pagos en línea ni logística avanzada.

Capítulo III

Metodología

3.1. Enfoque metodológico y tipo de investigación

El trabajo se enmarca en una investigación aplicada porque parte de un problema común de las pequeñas empresas —el control de inventario y la publicación de un catálogo actualizado— y orienta el conocimiento de Ingeniería de Software hacia una solución concreta. El Manual de Frascati caracteriza la investigación aplicada por su orientación hacia un objetivo práctico específico [8]; esta definición es coherente con el propósito de analizar, diseñar, construir y validar Stockify, sin pretender formular una teoría general sobre la digitalización empresarial.

La investigación tuvo una orientación tecnológica, puesto que su producto central fue un artefacto de software compuesto por una aplicación web, una base de datos relacional y una infraestructura de despliegue.

El alcance fue descriptivo y evaluativo. Descriptivo porque organizó y describió los actores, requerimientos, arquitectura, módulos, reglas y proceso de despliegue. Evaluativo porque contrastó la solución con criterios de aceptación, pruebas automatizadas y funcionales, requisitos funcionales y no funcionales, evidencia operativa y una evaluación de usabilidad aplicada. Esta delimitación evita atribuir efectos no demostrados y generalizar los resultados al conjunto de las pequeñas empresas [14]. La Tabla 3 sintetiza estas dimensiones.

Tabla 3: Síntesis del enfoque metodológico

Dimensión	Aplicación en Stockify	Producto o evidencia
Aplicada	Resuelve una necesidad práctica mediante un sistema de inventario y catálogo web.	Requerimientos, software desplegable e instrumentos de validación.
Tecnológica	Diseña y construye un artefacto de software a partir de conocimientos de Ingeniería de Software.	Base de datos, backend, frontend e infraestructura.
Descriptiva	Ordena y detalla los actores, reglas, arquitectura, módulos y proceso de desarrollo.	Revisión documental, código, configuración y trazabilidad.
Evaluativa	Compara la solución con criterios definidos y evidencia verificable.	Pruebas, pipelines, verificación operativa y evaluación de usabilidad.

Fuente: Autor.

3.1.1. Negocios participantes

Stockify no fue desarrollado para una empresa específica, sino que se orientó de manera general a pequeñas empresas comerciales, de acuerdo con el alcance del proyecto. Para el levantamiento de información y la evaluación del sistema participaron tres negocios de Cuenca, provincia del Azuay, seleccionados según su accesibilidad, disponibilidad y aceptación voluntaria para colaborar. Los criterios considerados fueron: comercializar productos físicos cuya organización requiriera categorías, variantes o presentaciones; utilizar registros manuales o no sistematizados para productos, existencias o ventas; y no disponer de un catálogo web propio.

Los tres negocios cumplían estas condiciones y aportaron información de referencia para identificar necesidades y configurar los escenarios de evaluación. La Tabla 4 resume sus características.

Tabla 4: Negocios participantes en el proyecto

Negocio	Actividad comercial	Estructura
Ely's Store	Ropa para hombres.	Negocio sin sucursales.
Light Blue	Accesorios, cosméticos y artículos variados.	Negocio sin sucursales.
Veterinaria Manchas	Alimentos, juguetes, ropa y otros productos para mascotas.	Una sucursal y una bodega interna.

Fuente: Autor.

La participación de estos negocios no los convirtió en destinatarios exclusivos de Stockify. La información obtenida permitió orientar el alcance funcional del sistema y considerar actividades comerciales y estructuras operativas diferentes.

3.2. Metodología de desarrollo de software

El desarrollo de Stockify se organizó mediante elementos del marco Scrum, especialmente los sprints, el *Product Backlog*, el *Sprint Backlog* y el Incremento [18]. Estos elementos permitieron distribuir el alcance en periodos de trabajo, ordenar las funcionalidades del sistema y verificar resultados parciales durante el desarrollo.

GitLab apoyó la planificación y la trazabilidad mediante issues, merge requests, commits y pipelines. Estos recursos permitieron registrar tareas, relacionar cambios de código y automatizar controles de construcción y prueba [38–40].

3.2.1. Justificación de la metodología seleccionada

Scrum se consideró adecuado porque Stockify reunía funcionalidades relacionadas que podían priorizarse y construirse como incrementos verificables. La identidad y el acceso debían preceder a las funciones protegidas; los productos y la estructura operativa eran necesarios antes del inventario; y el inventario constituía la base para las ventas y la disponibilidad pública. Estas dependencias permitieron ordenar el desarrollo y definir un resultado esperado para cada sprint.

El *Product Backlog* facilitó organizar y priorizar las funcionalidades generales del sistema. Para cada periodo, el *Sprint Backlog* delimitó los elementos seleccionados y las tareas necesarias para desarrollarlos, mientras el Incremento reunió las funcionalidades terminadas y verificadas. Como ventajas, esta organización permitió dividir el alcance, conservar la trazabilidad entre necesidades y cambios, comprobar resultados parciales e incorporar ajustes sin posponer toda la verificación hasta el final.

3.2.2. Aplicación de Scrum

Retomando los elementos definidos en el marco teórico, la Tabla 5 presenta su aplicación concreta en el proyecto.

Tabla 5: Aplicación de los elementos de Scrum en Stockify

Componente Scrum	Aplicación en Stockify
Sprint	Ocho periodos de trabajo de 11 días, cada uno con un objetivo y un incremento previsto.
Objetivo del Sprint	Resultado esperado para cada periodo de trabajo, definido según las funcionalidades seleccionadas y presentado en la planificación de los sprints.
Product Backlog	Registro general y ordenado de issues con funcionalidades, requisitos, mejoras, defectos y tareas de mantenimiento.
Sprint Backlog	Issues y tareas seleccionadas para el sprint actual, agrupadas mediante las etiquetas Sprint1–Sprint8.
Incremento	Conjunto de funcionalidades integradas y verificadas mediante criterios de aceptación y controles técnicos aplicables.
Definición de Terminado	Criterio empleado para considerar finalizado un elemento: la funcionalidad debía estar implementada e integrada, cumplir sus criterios de aceptación y superar los controles técnicos aplicables.

Fuente: Autor.

3.2.3. Herramientas de apoyo y trazabilidad

La Tabla 6 presenta los recursos empleados para organizar el backlog, relacionar las actividades con los cambios de código, integrar los componentes y verificar cada incremento.

Tabla 6: Herramientas de apoyo y trazabilidad

Recurso	Función en el proyecto	Implementación
Repositorios Git	Separación de código y documentación.	Repositorios para <i>Backend</i> , <i>frontend</i> e <i>infraestructura</i> .
Issue boards	Visualización y seguimiento del estado de las tareas.	Tablero de desarrollo con 4 columnas: Backlog, En curso, Revisión, Cerrado.
Issues y etiquetas	Organización del backlog, las tareas, los defectos, las mejoras y su asociación con los sprints.	Etiquetas por repositorio: <i>backend</i> , <i>frontend</i> e <i>infra</i> ; agrupación de issues mediante las etiquetas <i>Sprint1–Sprint8</i> .
Merge requests y commits	Integración del código y conservación de trazabilidad.	Merge requests y commits asociados a cada issue.
Pipelines CI/CD	Ejecución automatizada del control de cambios, construcción de imágenes y despliegues.	<i>Backend</i> y <i>frontend</i> ejecutan integración continua y activan el despliegue continuo en <i>infraestructura</i> .

Fuente: Autor.

3.2.4. Planificación y ejecución de sprints

Para la planificación del proyecto se decidió dividir el desarrollo en ocho sprints, cada uno con una duración de 11 días, y para identificarlos se utilizaron las etiquetas correspondientes a cada sprint: *Sprint1–Sprint8*. Los periodos previstos y el incremento planificado para cada sprint se muestran en la Tabla 7.

Tabla 7: Planificación de fechas de los sprints

Sprint	Fechas previstas	Objetivo e incremento
1	14–24 de abril	Inicialización de proyectos <i>frontend</i> y <i>backend</i> , definición de la arquitectura base, convenciones de desarrollo, manejo de errores e integración continua para la verificación, construcción y ejecución de pruebas.

2	25 de abril–5 de mayo	Implementación de la gestión de usuarios, roles y permisos, protección de rutas administrativas y autenticación mediante JWT y Google.
3	6–16 de mayo	Desarrollo de categorías y subcategorías jerárquicas, consulta pública inicial y administración de marcas.
4	17–27 de mayo	Implementación de productos, variantes, imágenes, atributos y valores por categoría.
5	28 de mayo–7 de junio	Implementación de la estructura operativa, ubicaciones, información del negocio y gestión de recursos multimedia.
6	8–18 de junio	Implementación del inventario, integración con la estructura operativa, movimientos, documentos y transferencias.
7	19–29 de junio	Desarrollo de ventas, historial comercial, cambios, devoluciones y ajustes de inventario.
8	30 de junio–10 de julio	Implementación del catálogo público, mejora en la experiencia de usuario y despliegue de la aplicación.

Fuente: Autor.

3.3. Levantamiento y análisis de requerimientos

El levantamiento y análisis de requerimientos permitió identificar necesidades operativas en los tres negocios participantes y convertirlas en requisitos funcionales y no funcionales para Stockify. Los requerimientos fueron organizados y refinados con referencias documentales y criterios de ingeniería para definir el alcance, las reglas y las funcionalidades de la solución.

3.3.1. Técnicas y fuentes

Antes del desarrollo se realizaron entrevistas abiertas con las propietarias de Ely's Store, Light Blue y Veterinaria Manchas. No se utilizó un cuestionario ni una guía fija, sino preguntas formuladas de manera flexible sobre los productos comercializados, su organización en categorías o variantes, las prácticas de registro y las funciones esperadas en el sistema. También se consultó qué información debía mostrarse en el catálogo público. Se tomaron notas de trabajo para trasladar las necesidades al backlog.

En dos negocios se realizaron visitas que permitieron una observación directa. Durante ellas se revisaron cuadernos y anotaciones utilizados para registrar productos, entradas, ventas, cantidades y algunos precios. También se observó que, ante dudas, algunas existencias de productos se comprobaban físicamente y algunos precios se consultaban en

el registro de compra al proveedor. En el tercer negocio no se efectuó una visita de observación; sus prácticas fueron descritas por la propietaria durante la entrevista. Estas actividades se emplearon para comprender el proceso, no como una medición sistemática de tiempos o errores [14].

La información de los negocios se complementó con revisión de literatura, análisis de funciones documentadas de Odoo, Shopify, Tiendanube y Amazon, y decisiones de Ingeniería de Software. La Tabla 8 resume el aporte y el uso de cada fuente.

Tabla 8: Fuentes empleadas para definir y refinar requerimientos

Técnica o fuente	Aporte	Uso e implementación
Entrevistas abiertas	Necesidades comunicadas por las tres propietarias antes del desarrollo.	Definición inicial de funciones administrativas, organización de productos, inventario, ventas y contenido del catálogo.
Observación directa y revisión de registros	Prácticas manuales observadas durante las visitas a dos negocios.	Comprensión de entradas, ventas, cantidades, precios y verificaciones físicas utilizadas en la operación.
Revisión de literatura	Conceptos y antecedentes de inventario, catálogo, ventas y aplicaciones web responsivas.	Fundamentación del problema y delimitación del alcance.
Análisis comparativo	Funciones documentadas de Odoo, Shopify, Tiendanube y Amazon.	Referencias para productos, variantes, categorías, existencias y presentación pública, adaptadas al alcance del proyecto.
Ingeniería de Software	Criterios de arquitectura, datos, seguridad, pruebas, mantenibilidad y trazabilidad.	Definición de requisitos técnicos y decisiones necesarias para construir y verificar la solución.

Fuente: Autor.

La definición de los requisitos integró distintas fuentes. Las necesidades expresadas por las propietarias permitieron identificar el problema y establecer las funcionalidades principales. El análisis comparativo aportó alternativas de organización e interacción acordes con el alcance del sistema, mientras que los criterios de ingeniería orientaron la definición de condiciones de seguridad, integridad, trazabilidad, despliegue y prueba.

3.3.2. Actores y perfiles

La interacción con el sistema se organiza en cuatro contextos técnicos principales: propietario administrador, trabajador interno autorizado, usuario público con cuenta y visitante sin registro. Las denominaciones “vendedor” o “bodeguero” se configuran mediante permisos y se aplican como perfiles operativos. Desde la perspectiva comercial, tanto el *usuario con cuenta* como el *visitante* representan los clientes del negocio. La Tabla 9 resume estos actores y su autenticación.

Tabla 9: Actores e interacción principal en el sistema

Actor	Interacción principal	Autenticación
Propietario administrador	Configura el negocio, usuarios, permisos, catálogo, inventario, ventas y apariencia.	Contexto administrativo.
Trabajador administrativo	Ejecuta funciones autorizadas según los permisos asignados por el administrador.	Contexto administrativo.
Usuario con cuenta	Gestiona su cuenta, sesión pública, perfil y lista de deseos, además de consultar el catálogo.	Contexto público.
Visitante sin registro	Navega por la página de inicio, el catálogo y el detalle de los productos.	No requerida para recursos públicos.

Fuente: Autor.

3.3.3. Requerimientos funcionales

Los requerimientos funcionales se organizaron en tres ámbitos: administrativo, público y transversal. La Tabla 10 presenta su clasificación y descripción.

Tabla 10: Requerimientos funcionales

ID	Actor	Requerimiento	Prioridad
RF-ADM-01	Propietario / Trabajador	Autenticarse en el área administrativa y mantener una sesión renovable y activa.	Alta
RF-ADM-02	Propietario	Gestionar usuarios, invitaciones, roles y permisos.	Media
RF-ADM-03	Propietario	Asignar ubicaciones laborales a trabajadores según sus permisos.	Media
RF-ADM-04	Propietario	Restringir el acceso administrativo según el origen de IP autorizado.	Media
RF-ADM-05	Propietario / Trabajador	Gestionar categorías jerárquicas, atributos, valores heredables y estados públicos.	Alta

RF-ADM-06	Propietario / Trabajador	Gestionar marcas, productos, variantes, precios e imágenes.	Alta
RF-ADM-07	Propietario / Trabajador	Gestionar colecciones de productos y su visibilidad pública.	Baja
RF-ADM-08	Propietario	Configurar sucursales, áreas de almacenamiento y ubicaciones de stock.	Media
RF-ADM-09	Propietario / Trabajador	Consultar existencias, umbrales y kardex por variante de producto y ubicación de stock.	Alta
RF-ADM-10	Propietario / Trabajador	Registrar movimientos de entradas, salidas, ajustes y transferencias de stock entre ubicaciones.	Alta
RF-ADM-11	Propietario / Trabajador	Registrar ventas y consultar el historial de ventas registradas.	Alta
RF-ADM-12	Propietario / Trabajador	Cancelar una venta válida y restablecer el stock asociado sin repetir la operación.	Media
RF-ADM-13	Propietario / Trabajador	Registrar cambios, devoluciones y ajustes comerciales con impacto trazable en stock.	Media
RF-ADM-14	Propietario	Configurar identidad del negocio, datos comerciales, redes sociales, visibilidad del catálogo público y apariencia.	Media
RF-ADM-15	Propietario	Definir si el registro del punto de venta es obligatorio al realizar una venta.	Media
RF-PUB-01	Visitante / Usuario público	Navegar en el catálogo con búsqueda, paginación, filtro de marca y ordenamiento.	Alta
RF-PUB-02	Visitante / Usuario público	Filtrar el catálogo por una categoría o subcategoría específica.	Alta
RF-PUB-03	Visitante / Usuario público	Consultar detalle de la variante de producto, imágenes, precio y disponibilidad sin exponer cantidades internas.	Alta
RF-PUB-04	Visitante / Usuario público	Consultar productos, categorías, marcas y colecciones destacadas.	Media
RF-PUB-05	Usuario público	Registrarse y verificar su correo electrónico.	Alta
RF-PUB-06	Usuario público	Iniciar sesión, recuperar su contraseña y administrar su perfil.	Alta
RF-PUB-07	Usuario público	Agregar y retirar variantes de productos en la lista de deseos.	Baja

RF-PUB-08	Usuario público	Consultar redes y contactar al negocio por los medios habilitados.	Media
RF-TR-01	Todos los actores	Separar sesiones PUBLIC/ADMIN y validar su autorización para cada operación del servidor.	Alta
RF-TR-02	Todos los actores	Validar datos sensibles e inconsistentes antes de procesar operaciones que modifiquen información.	Alta
RF-TR-03	Propietario / Trabajador	Conservar el historial de los registros desactivados sin eliminarlos físicamente.	Alta
RF-TR-04	Propietario / Trabajador	Registrar automáticamente los datos de auditoría asociados a la creación y modificación de registros.	Media

Fuente: Autor.

3.3.4. Requerimientos no funcionales

Los requisitos no funcionales se expresaron mediante condiciones observables. La Tabla 11 presenta los criterios de calidad asociados a las necesidades del sistema.

Tabla 11: Requerimientos no funcionales

ID	Requerimiento	Procedimiento de verificación
RNF-01	Toda ruta que no sea pública debe exigir un token válido.	Inspección de guards globales y casos 401.
RNF-02	Toda operación administrativa debe exigir contexto ADMIN y el rol o permiso correspondiente.	Auditoría de controladores y permisos de autorización.
RNF-03	Un token PUBLIC no debe habilitar rutas ADMIN, ni a la inversa.	Casos cruzados 401/403 por contexto de sesión.
RNF-04	El catálogo no debe exponer costos ni cantidades internas de stock en sus contratos públicos.	Inspección de DTO y pruebas de contrato.
RNF-05	Los campos no permitidos o inválidos deben rechazarse con respuesta controlada.	Validación de payloads en el servidor, esquemas de frontend y casos 400.
RNF-06	La capa HTTP debe aplicar cabeceras de protección, CORS configurado y limitación de solicitudes.	Configuración de validación y pruebas en los casos específicos.

RNF-07	Las contraseñas deben almacenarse con Argon2id y permitir actualización de parámetros de hash.	Inspección y pruebas unitarias de hash/rehash.
RNF-08	Las funciones públicas y administrativas deben poder utilizarse en escritorio y móvil.	Viewports definidos y pruebas de usabilidad responsiva.
RNF-09	La base de datos debe operar con sincronización automática desactivada y manejar cambios explícitos mediante migraciones.	Configuración TypeORM, migraciones y prueba de actualización.
RNF-10	Cada cliente desplegado debe disponer de backend, base de datos y almacenamiento propios, documentando cualquier recurso compartido.	Configuración de aislamiento por cliente sin datos sensibles y con pruebas de aislamiento.
RNF-11	El respaldo debe incluir base de datos, archivos y configuración, checksums y un procedimiento de restauración verificable.	Manifest, checksum y simulacro.

Fuente: Autor.

3.3.5. Historias de usuario y criterios de aceptación

Las siguientes historias de usuario describen comportamientos representativos del sistema y la Tabla 12 reúne sus criterios de aceptación, que orientaron las pruebas unitarias, E2E de API, pruebas automatizadas de interfaz y verificaciones técnicas presentadas posteriormente.

Tabla 12: Historias de usuario representativas y criterios de aceptación

ID	Historia	Criterios de aceptación
HU-01	Como visitante, quiero buscar, filtrar por marca y ordenar el catálogo para localizar productos de interés.	Filtros de búsqueda, marca, orden y paginación; devuelve solo productos activos, visibles y publicables.
HU-02	Como visitante, quiero filtrar el catálogo por una categoría y sus subcategorías.	El slug público incluye el subárbol; un slug inválido u oculto no expone productos; la interfaz permite aplicar la categoría mediante la navegación y limpiar el filtro.
HU-03	Como visitante, quiero consultar un producto y seleccionar una variante para conocer detalles, precio y disponibilidad.	URL canónica única por variante; atributos, imágenes, precio y disponibilidad sin exponer cantidades internas.
HU-04	Como usuario público, quiero registrarme y verificar mi correo para acceder a las funciones de cuenta.	Registro, verificación, reenvío controlado, expiración y posterior sesión PUBLIC.

HU-05	Como usuario público con cuenta, quiero guardar variantes en mi lista de deseos.	Exige sesión PUBLIC; agrega y retira variantes; la lista omite productos que dejaron de ser públicos.
HU-06	Como personal administrativo autorizado, quiero crear un producto con variantes e imágenes para el catálogo público.	Valida permisos, categoría, atributos, combinaciones únicas, precios e imágenes.
HU-07	Como trabajador administrativo autorizado, quiero consultar el stock y el kardex por ubicación.	Valida permisos; muestra cantidades, ubicación, motivo y documento del movimiento; mantiene el historial.
HU-08	Como trabajador administrativo autorizado, quiero transferir stock entre ubicaciones.	Exige permisos, origen y destino distintos, destino activo y saldo suficiente; salida y entrada atómicas con reversión ante fallo.
HU-09	Como personal administrativo autorizado, quiero registrar una venta y actualizar las existencias asociadas.	Exige permisos, ítems y stock válidos, sucursal según configuración; totales en servidor y descuento en la misma transacción.
HU-10	Como personal administrativo autorizado, quiero cancelar una venta o registrar un ajuste sin perder trazabilidad.	Exige motivo; evita dobles cancelaciones o ajustes inválidos; registra atómicamente estado y movimientos.
HU-11	Como propietario administrador, quiero configurar roles y permisos por responsabilidades.	ADMIN conserva bypass; los demás roles reciben solo permisos asignados y restricciones de origen IP; un cambio de rol revoca las sesiones administrativas activas.

Fuente: Autor.

3.4. Diseño de la solución

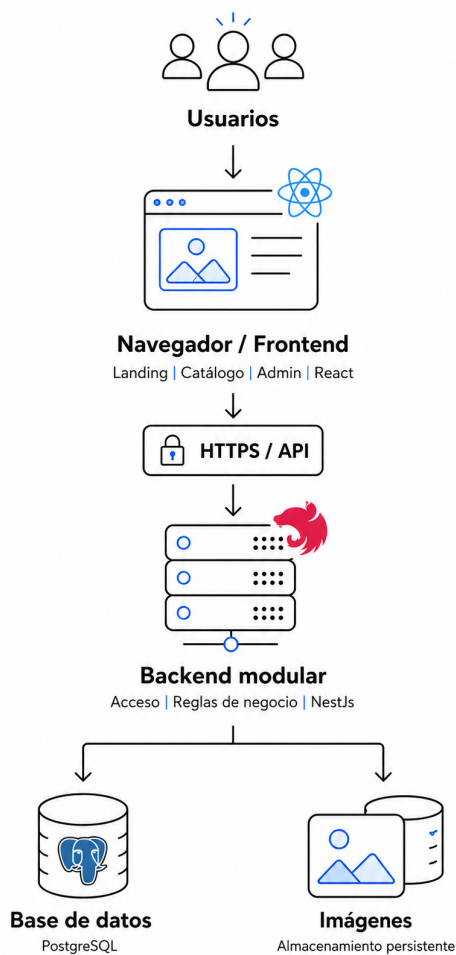
Stockify se diseñó a partir de sus requerimientos funcionales y no funcionales como una aplicación web responsiva con arquitectura cliente-servidor. El navegador gestiona la interacción pública y administrativa, la API concentra las reglas de negocio y PostgreSQL almacena los datos transaccionales. Aunque sus funciones se organizan por módulos, estos forman parte de una única aplicación backend monolítica.

3.4.1. Arquitectura lógica general

En la interfaz, según el subdominio se identifica un negocio en específico. React organiza las rutas públicas, de autenticación, cuenta y administración, todas conectadas a la API NestJS mediante una capa HTTP común. Los archivos de imagen se conservan en un

volumen persistente, mientras que PostgreSQL registra sus metadatos y asociaciones. La Figura 1 resume estas responsabilidades.

Figura 1: Arquitectura lógica general de Stockify



Fuente: Autor.

3.4.2. Diseño del modelo de datos

El modelo relacional que sustenta la solución está compuesto por 50 entidades con relaciones y restricciones explícitas que contribuyen a la integridad de los datos y a la consistencia de las consultas [7, 15]. La Tabla 13 las agrupa por subdominio.

Tabla 13: Dominios y entidades principales del modelo de datos

Dominio	Entidades representativas	Relación esencial
Identidad y acceso	Persona, usuario, identidad, sesión, rol, permiso y orígenes administrativos.	Usuario–rol y rol–permiso; las sesiones se separan por contexto.
Catálogo	Categoría, marca, producto, variante, atributos y colecciones.	Producto–variantes; categoría jerárquica; variante–valores de atributo.
Estructura operativa	Sucursal, área de almacenamiento y ubicación.	Sucursal–áreas–ubicaciones, con estados lógicos.
Inventario	Existencia, movimiento, operación, transferencia y documentos asociados.	Combinación única ubicación–variante y movimientos vinculados con su origen.
Ventas	Venta, líneas, ajustes comerciales y líneas de ajuste.	El ajuste referencia la venta original y ambos producen efectos de inventario.
Medios y presentación	Activo, uso del activo, negocio, redes, configuraciones y apariencia.	El uso relaciona un archivo con una entidad propietaria y un propósito visual.
Funciones públicas	Registro/verificación pública y lista de deseos.	Usuario público–variante, filtrada por las reglas de publicación vigentes.

Fuente: Autor.

El **producto** representa la agrupación comercial y la **variante** identifica una combinación vendible con precio, atributos y visibilidad propios. La existencia se modela mediante la combinación única de **variante** y **ubicación**; el **movimiento** registra el tipo, motivo, cantidad y documento que lo originó. La **venta** conserva sus líneas, mientras que los **cambios**, **devoluciones** y **ajustes** se relacionan con la venta original. El catálogo público no duplica la información, sino que expone una proyección restringida del mismo modelo y aplica criterios de estado, visibilidad, imagen y disponibilidad antes de responder. Además, para controlar la evolución del esquema, TypeORM mantiene desactivada la sincronización automática y aplica los cambios mediante migraciones y seeds.

3.4.3. Diseño del backend

El backend se estructuró por funcionalidades de negocio: cada módulo puede contener presentación, servicios de aplicación, entidades o adaptadores de persistencia. Los controladores traducen solicitudes HTTP y DTO hacia operaciones; los servicios coordinan reglas, repositorios y transacciones; TypeORM conecta las entidades con PostgreSQL. La separación es pragmática y modular, no una aplicación estricta de Clean Architecture. La

configuración global del backend define el prefijo de la API, CORS, Helmet, validación de entradas, manejo de excepciones, rate limit y guards globales.

Figura 2: Flujo lógico de una operación en el backend



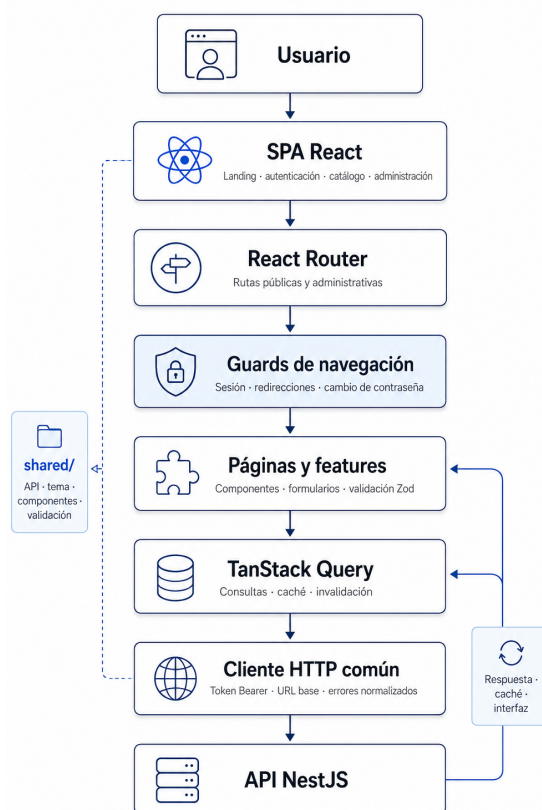
Fuente: Autor.

3.4.4. Diseño del frontend

El frontend se diseñó como una SPA React con cuatro áreas lógicas: landing institucional, autenticación, experiencia pública y administración. React Router define rutas y límites de navegación; los guards del navegador resuelven carga de sesión, redirecciones y cambio obligatorio de contraseña, pero la autorización definitiva permanece en el backend. TanStack Query administra estado remoto, caché e invalidación, y una capa HTTP común normaliza la URL base, adjunta el token de acceso y coordina un único refresh por contexto ante una respuesta no autorizada. Los formularios administrativos combinan React Hook Form y Zod para la validación de interfaz, que mejora la experiencia pero no sustituye los DTO y reglas del servidor.

Las rutas públicas y administrativas de mayor tamaño se cargan de forma diferida. La responsividad se incorporó mediante Tailwind CSS, navegación móvil, drawer administrativo, filtros adaptativos, vistas CRUD en tarjetas y acciones fijas; su presencia en el código constituye una decisión de diseño y la utilizabilidad real se comprueba con viewports y tareas definidos.

Figura 3: Flujo lógico general del frontend

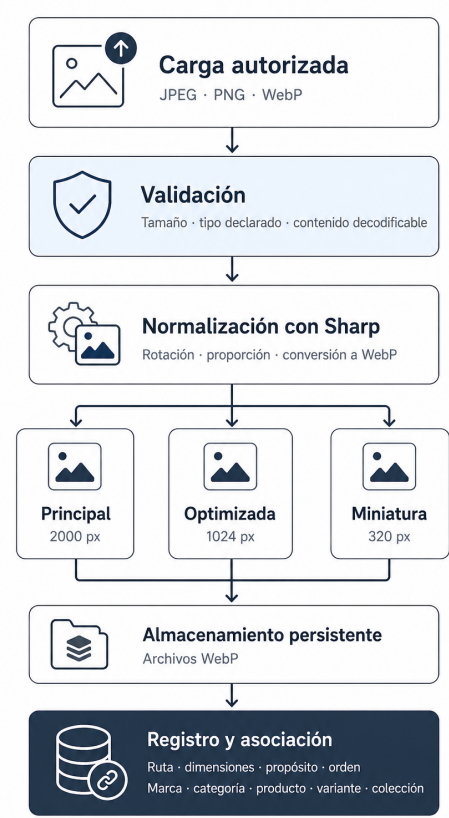


Fuente: Autor.

3.4.5. Diseño del almacenamiento de imágenes

El subsistema multimedia separa el archivo físico de su descripción relacional. Una carga autorizada se valida por tamaño, tipo de archivo declarado y contenido decodificable; después se normaliza a tres archivos WebP: versión principal, optimizada y miniatura, y la base de datos registra la ruta de almacenamiento. El uso de cada archivo se registra mediante una entidad que lo relaciona con una marca, categoría, producto, variante o colección, indicando su propósito visual y su orden de presentación.

Figura 4: Flujo lógico del almacenamiento de imágenes



Fuente: Autor.

3.4.6. Seguridad y control de acceso

La seguridad sigue una política cerrada por defecto. Salvo las rutas marcadas explícitamente como públicas, la solicitud atraviesa autenticación JWT, rol, contexto de sesión y permisos. Los contextos ADMIN y PUBLIC se codifican en la sesión y evitan utilizar un token de cliente como credencial administrativa. La Tabla 14 resume los controles previstos y la Figura 5 el ciclo de autenticación.

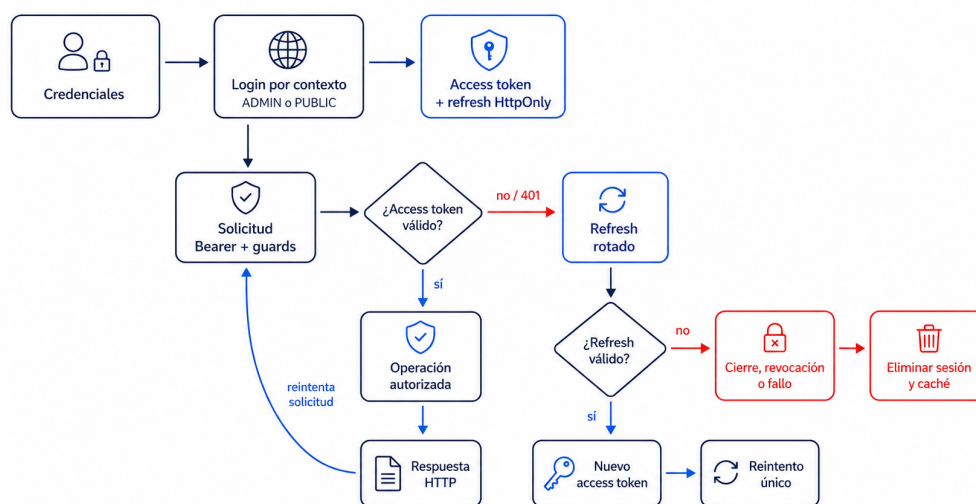
Tabla 14: Controles de seguridad previstos en el diseño

Control	Finalidad	Alcance y cautela
JWT Bearer	Autenticar y verificar expiración, emisor y audiencia.	Comprueba además la sesión activa en contexto administrativo.
Contextos de sesión	Separar credenciales ADMIN y PUBLIC.	Requiere casos negativos cruzados.
RBAC y permisos	Restringir módulos y operaciones.	ADMIN posee bypass explícito.
Refresh rotado	Renovar y revocar sesiones sin exponer el refresh.	Cookie HttpOnly, SameSite=Lax.

Argon2id	Proteger contraseñas y permitir rehash.	Parámetros explícitos y prueba de rehash.
Validación global	Transformar DTO y rechazar campos no permitidos.	Complementa las reglas del dominio.
Helmet, CORS y rate limit	Reducir superficie HTTP y abuso básico.	No equivalen a pentest ni WAF.
Origen administrativo	Restringir perfiles por IP o sucursal.	No universal; owner conserva bypass.
Guards de interfaz (frontend)	Ocultar acciones no permitidas ni autorizadas.	Solo presentación visual; el servidor reautoriza.

Fuente: Autor.

Figura 5: Flujo de autenticación, renovación y autorización



Fuente: Autor.

3.5. Estrategia de desarrollo e integración

La estrategia relaciona el incremento funcional con evidencia rastreable. Una necesidad o hallazgo se registra como issue; el cambio se desarrolla en una rama, se asocia con commits y una merge request, atraviesa los controles automatizados aplicables y, después de integrarse, actualiza el estado del backlog. Las herramientas de GitLab soportan esa trazabilidad, aunque una merge request no prueba por sí sola revisión independiente [38–40].

3.5.1. Construcción incremental

Los ocho sprints planificados agruparon incrementos de arquitectura, identidad, catálogo, inventario, ventas y experiencia pública; posteriormente, se desarrollaron las actividades de infraestructura. Esta secuencia explica las dependencias: la identidad precede a los paneles protegidos, productos y estructura preceden al inventario, e inventario precede a ventas y disponibilidad pública. El flujo documental definido para cada incremento es:

Figura 6: Flujo documental de cada incremento



Fuente: Autor.

La relación no siempre es uno a uno: una issue puede requerir cambios coordinados en backend y frontend, y una refactorización puede afectar varias issues. Por ello, la matriz final se apoya también en rutas de código, pruebas y commit, no únicamente en el estado del backlog.

3.5.2. Organización de repositorios

El trabajo se distribuye en tres repositorios: `stockify-back` concentra la API, reglas, entidades, migraciones, seeds y pruebas del servidor, y produce la imagen backend; `stockify-front` contiene landing, catálogo, autenticación, cuenta y área administrativa, y produce la imagen frontend; `stockify-infra` versiona Compose, Nginx, scripts, pipelines de despliegue, provisión y backup. Esta separación permite que cada repositorio mantenga su propio ciclo de versiones.

3.5.3. Ramas y merge requests

El flujo promueve cambios desde una rama `feature` hacia `develop`, de `develop` hacia `staging` y de `staging` hacia `main`: `develop` concentra integración, `staging` alimenta QA y `main` habilita la entrega productiva. Las tres ramas se mantienen protegidas y sin `force-push` permitido. Las merge requests conservan diff, commits, conversación y relación con pipelines.

3.5.4. Integración continua y puertas de calidad

Los pipelines de aplicación automatizan comprobaciones antes de producir una imagen. El backend ejecuta análisis estático, construcción, pruebas unitarias y E2E con PostgreSQL; el

frontend ejecuta análisis estático, construcción, pruebas unitarias y pruebas automatizadas de interfaz con Playwright y API interceptada. La Tabla 15 resume las puertas definidas.

Tabla 15: Puertas de calidad definidas en integración continua

Puerta	Backend	Frontend	Criterio
Comprobación estática	TypeScript, ESLint y formato.	TypeScript, ESLint y formato.	Comandos sin error.
Construcción	Compilación NestJS.	Build Vite y contrato de landing.	Artefacto construible.
Unitarias y componentes	Jest.	Vitest y Testing Library.	Suite sin fallos.
Pruebas automatizadas	Jest, Supertest y PostgreSQL.	Playwright con API interceptada.	Escenarios sin fallos.
Publicación y entrega	Imagen por SHA y trigger a infraestructura.	Imagen por SHA y trigger a infraestructura.	Push al Registry y referencia exacta en downstream.

Fuente: Autor.

3.6. Estrategia de pruebas y validación

La validación combinó comprobaciones técnicas y evaluación de usabilidad. Las primeras contrastaron el software y la infraestructura con los criterios definidos; la segunda examinó la finalización de tareas y la percepción de uso en las interfaces administrativa y pública. Esta combinación permitió obtener evidencia complementaria sin atribuir a una técnica un alcance que no le correspondía.

3.6.1. Pruebas técnicas

Las pruebas técnicas incluyeron pruebas unitarias, E2E de API, pruebas automatizadas de interfaz y verificaciones de infraestructura. Estas cubrieron, respectivamente, reglas aisladas, la integración entre API y persistencia, navegación e interacción con respuestas controladas de la API, y contratos y scripts operativos.

Tabla 16: Niveles de prueba

Nivel	Herramienta o medio	Ámbito verificado
Pruebas unitarias backend	Jest.	Reglas, servicios, guards y utilidades.
E2E de API	Jest, Supertest y PostgreSQL aislado.	API NestJS, autenticación, persistencia y flujo comercial.
Pruebas unitarias y de componentes frontend	Vitest, Testing Library y jsdom.	Renderizado, formularios, utilidades e interacción.
Pruebas automatizadas de interfaz	Playwright sobre Desktop Chrome con API interceptada.	Navegación pública y administrativa, autenticación y estados visibles.
Verificaciones técnicas de infraestructura	Scripts Bash y Node.js.	Rutas, contratos de repositorio, transporte de secretos y proxy simulado.

Fuente: Autor.

3.6.2. Evaluación de usabilidad

La evaluación de usabilidad se realizó al finalizar el desarrollo mediante dos recorridos diferenciados. El área administrativa fue evaluada por las tres propietarias que participaron en el levantamiento, una por negocio. El catálogo público fue evaluado por doce personas seleccionadas según su disponibilidad y aceptación voluntaria para realizar las pruebas. Estas personas no debían ser clientes ni mantener una relación previa con los negocios, porque el propósito era comprobar si un usuario podía consultar el catálogo y completar tareas representativas.

Cada propietaria ejecutó diez tareas administrativas con datos de prueba vinculados a su negocio. Los doce participantes públicos ejecutaron nueve tareas de consulta desde teléfonos móviles y se distribuyeron entre los catálogos de Ely's Store, Light Blue y Veterinaria Manchas. Cada participante realizó personalmente las actividades y completó el cuestionario posterior; el resultado, el tiempo aproximado y las observaciones de cada tarea se consignaron en la misma ficha.

Los instrumentos incluyeron perfil de uso, actividades, la *System Usability Scale* (SUS), tres valoraciones complementarias y preguntas abiertas. Los resultados se analizaron por separado para cada interfaz: las tareas se clasificaron como completadas sin ayuda, completadas con ayuda, no completadas o no aplicables, y se calcularon sus frecuencias, medianas y rangos de tiempo. Para obtener la puntuación SUS se aplicó el procedimiento establecido por Brooke: se restó 1 a la respuesta de cada ítem impar y se restó de 5 la respuesta de cada ítem par; las contribuciones resultantes se sumaron y se multiplicaron por

2,5 [2]. Las tres valoraciones complementarias se analizaron por separado y no formaron parte de este cálculo. Las plantillas se presentan en el anexo del instrumento administrativo y en el anexo del instrumento del catálogo público.

3.7. Estrategia de despliegue y entrega continua

El despliegue se diseñó con imágenes separadas de frontend y backend, dos ambientes Compose y una instancia backend por cliente. Las aplicaciones publican imágenes por SHA y activan una pipeline de infraestructura; QA se actualiza de forma automática según las reglas del downstream, mientras que producción conserva un gate manual. La Tabla 17 resume la separación por ambiente y la Tabla 18 los recursos aislados y compartidos por cliente.

3.7.1. Contenerización, construcción y Registry

El backend utiliza una construcción multietapa, instala dependencias productivas, compila NestJS y ejecuta como usuario no privilegiado; el frontend se compila con Vite y se sirve mediante Nginx sin privilegios. Docker Compose coordina frontend, reverse proxy, PostgreSQL y los backends generados por cliente. Los datos de PostgreSQL se conservan en un volumen del ambiente y cada cliente recibe un volumen propio para `/app/storage`; los healthchecks comprueban procesos y rutas específicas, sin probar por sí solos la disponibilidad sostenida.

Cada imagen recibe una etiqueta derivada del SHA corto y un alias móvil de ambiente `staging` o `prod`. La referencia enviada a infraestructura utiliza la etiqueta SHA; el alias solo identifica el artefacto más reciente y no debe emplearse como versión inmutable.

3.7.2. Separación de QA y producción

Tabla 17: Estrategia de despliegue por ambiente

Dimensión	QA	Producción
Rama y raíz operativa	staging; /opt/stockify/qa. main; /opt/stockify/prod.	
Proyecto Compose	stockify-clients.	stockify-prod.
Origen local	Puerto de proxy ligado a loopback.	Puerto diferente ligado a loopback.
Red y persistencia	Red, PostgreSQL y volúmenes propios de QA.	Red, PostgreSQL y volúmenes propios de producción.

Actualización y uso	Downstream automático; integración y validación previa.	Job de despliegue siempre manual; servicio de clientes autorizados.
---------------------	---	---

Fuente: Autor.

La separación evita que QA comparta base, red o secretos con producción. Dentro de cada ambiente sí existen recursos compartidos entre clientes, como se detalla a continuación.

3.7.3. Dominios, enrutamiento y multiinstancia

La convención de host distingue landing y clientes: en QA, `staging.webstockify.com` y `<slug>-staging.webstockify.com`; en producción, `webstockify.com` (con `www`) y `<slug>.webstockify.com`. El host determina qué configuración de negocio utiliza el frontend. Cloudflare Tunnel dirige el tráfico hacia el puerto local del ambiente y Nginx selecciona el bloque de servidor: para un dominio de cliente, `/` se envía al frontend compartido, `/api/` al backend de ese cliente y `/img/` al almacenamiento servido por el mismo backend.

Tabla 18: Elementos aislados y compartidos en la multiinstancia

Recurso	Por cliente	Compartido dentro del ambiente
Backend	Servicio back- <code><slug></code> y configuración propia.	Imagen versionada y host Docker.
Base de datos	Nombre de base único.	Contenedor PostgreSQL y usuario/credencial configurada.
Archivos	Volumen persistente propio.	Motor y almacenamiento físico del host.
JWT y configuración	Archivo de cliente y material lógico propio.	Variables comunes del ambiente.
Dominio/Nginx	Host y upstream generados.	Reverse proxy y red Docker común.
Frontend	No se crea una SPA por cliente.	Una instancia de frontend por ambiente.

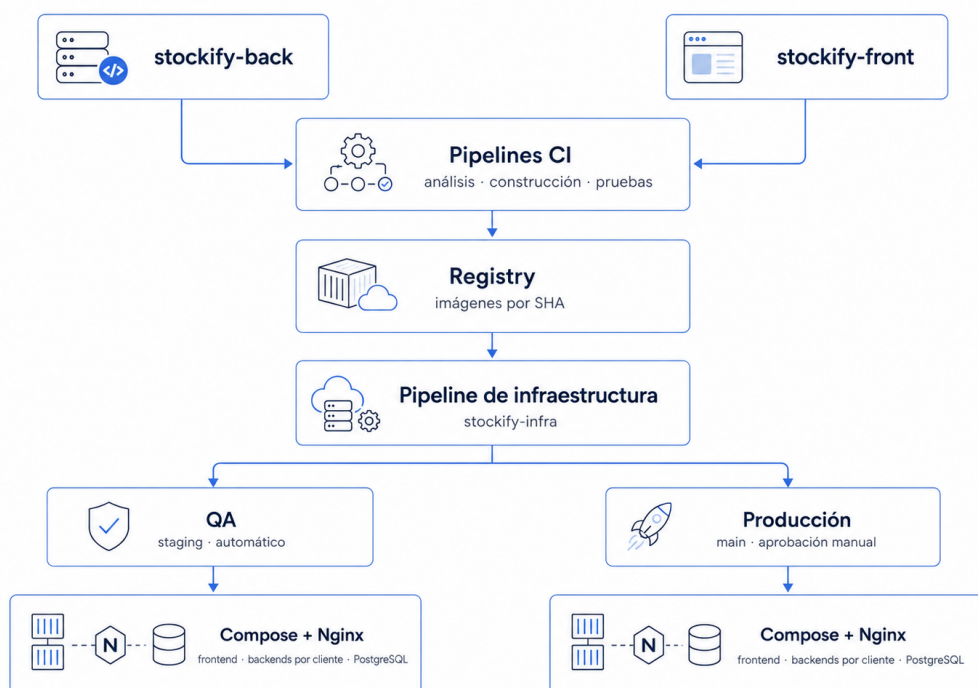
Fuente: Autor.

La separación de nombres y volúmenes limita cruces accidentales, pero el usuario PostgreSQL común y la red compartida reducen el aislamiento ante la compromisión de un backend. La prueba RNF-10 debe verificar privilegios efectivos y acceso cruzado.

3.7.4. Automatización, aprobación productiva y recuperación

Cuando los cambios se integran en `staging`, se construye la imagen y se activa la pipeline downstream que despliega QA. Al integrarlos en `main`, se envía la referencia de la imagen a infraestructura, pero `deploy_prod` permanece manual. Los *resource groups* serializan operaciones por ambiente y la provisión de un nuevo cliente se mantiene separada del despliegue global. La aprobación manual reduce el riesgo de publicar por el solo hecho de integrar en `main`. Para una entrega final se registran commit, pipeline, tag SHA, digest, ambiente, aprobación y smoke test. La Figura 7 representa el flujo de despliegue y entrega continua desde los repositorios de aplicación hasta los ambientes de QA y producción.

Figura 7: Flujo de despliegue y entrega continua de Stockify



Fuente: Autor.

La estrategia de despliegue productivo contempla generar un respaldo previo y conservar referencias que permitan retornar a una imagen anterior si falla el *healthcheck*. El procedimiento de respaldo se define para identificar las bases de datos, generar *dumps*, copiar el almacenamiento y la configuración, calcular *checksums* y restaurar cada *dump* en una base temporal antes de cerrar el paquete. La validación prevista consideró la integridad del paquete, la comprobación mediante *checksum*, la restauración en un entorno aislado, el arranque de la aplicación, la verificación de imágenes y el registro de la duración.

Capítulo IV

Desarrollo e implementación

4.1. Introducción

Este capítulo describe cómo se materializó Stockify en componentes de software y de infraestructura. A diferencia del Capítulo III, que expuso el diseño y la estrategia de validación, aquí se documentan tecnologías, estructuras y mecanismos observables en el código fuente; los resultados de pruebas y la comprobación funcional se reservan para el Capítulo V, de modo que la existencia de una configuración o un caso de prueba no se interprete anticipadamente como un resultado aprobado.

La revisión utiliza los tres repositorios técnicos del proyecto: el backend concentra contratos HTTP, reglas de negocio y persistencia; el frontend implementa las experiencias administrativa y pública; y la infraestructura define composición, enrutamiento, aprovisionamiento y operación. Los identificadores exactos de la versión evaluada se reservan para el capítulo de resultados y los anexos técnicos, donde cumplen una función de trazabilidad.

La exposición combina tres planos complementarios: primero el entorno tecnológico y la arquitectura del sistema desplegado —topología cliente–servidor, despliegue multiinstancia, organización modular, modelo relacional, contratos API, frontend e integración HTTP—; después el comportamiento funcional observado en el área administrativa y en el catálogo público; y finalmente los mecanismos transversales de seguridad y auditoría, junto con la integración, entrega continua y operación del ambiente productivo.

4.2. Entorno técnico y tecnologías

4.2.1. Base de datos y persistencia

La capa de datos emplea PostgreSQL como motor relacional y TypeORM como ORM. La aplicación registra la conexión con parámetros validados por ConfigService, y el archivo `data-source.ts` crea un DataSource equivalente para los comandos de migración; ambos comparten el registro explícito de 50 entidades (`ormEntities`), sin depender de patrones de descubrimiento. El esquema Joi comprueba tipos, rangos y variables obligatorias antes de inicializar la aplicación, y `DB_SYNCHRONIZE` permanece en `false`; la evolución del esquema es una actividad explícita mediante migraciones, no una modificación automática al iniciar la API. La Tabla 19 resume estos componentes.

Tabla 19: Componentes de la capa de persistencia implementada

Componente	Implementación	Responsabilidad
Motor relacional	PostgreSQL 16 y controlador pg	Datos estructurados, relaciones, restricciones y transacciones.
Mapeo persistente	TypeORM 0.3.30	Relaciona entidades con tablas, índices y claves foráneas.
Integración NestJS	@nestjs/typeorm 11.0.3	Registra conexión y repositorios mediante inyección de dependencias.
Configuración	@nestjs/config y Joi	Valida el entorno y suministra opciones tipadas.
Evolución del esquema	Migraciones TypeORM	Cambios explícitos con sincronización automática desactivada.
Datos iniciales	Seeds TypeScript	Inicializa permisos, usuario administrativo, catálogo base, negocio, apariencia y estructura operativa.

Fuente: Autor.

4.2.2. Backend

El backend se construyó con NestJS y TypeScript como una aplicación modular ejecutada en un solo proceso. El arranque establece el prefijo global de la API, habilita CORS con credenciales, incorpora Helmet, registra una tubería global de validación y aplica un filtro uniforme de excepciones. Swagger publica documentación interactiva OpenAPI del contrato HTTP e incorpora el esquema Bearer. La Tabla 20 resume las tecnologías y sus versiones utilizadas.

Tabla 20: Tecnologías principales del backend implementado

Tecnología	Versión resuelta	Responsabilidad
NestJS	11.1.24	Estructura modular, controladores y ejecución HTTP de la API sobre Express.
TypeScript	5.9.3	Tipado estático de la aplicación.
Autenticación	@nestjs/jwt 11.0.2; @nestjs/passport 11.0.5; passport-jwt 4.0.1	Emisión y validación de tokens, junto con estrategias de autenticación.
Validación	class-validator 0.15.1 y Joi 18.2.3	Validación de solicitudes y de la configuración de entorno.

Seguridad HTTP	Helmet 8.1.0; @nestjs/throttler 6.5.0	Cabeceras de seguridad y control de solicitudes repetidas.
Swagger/OpenAPI	@nestjs/swagger 11.4.4	Documentación interactiva del contrato HTTP y del esquema Bearer.

Fuente: Autor.

Sobre este entorno técnico, **AppModule** reúne los módulos de autenticación, usuarios, roles, permisos, estructura operativa, catálogo, inventario, ventas, configuración, apariencia, medios, catálogo público y lista de deseos. La capa de presentación contiene controladores y DTO; los servicios de aplicación coordinan reglas, repositorios y transacciones; y la infraestructura contiene entidades y adaptadores técnicos. La autorización se aplica mediante guards globales en el orden de limitación de tasa, JWT, roles, contexto de sesión y permisos; las rutas públicas se declaran con `@Public()` y los tokens distinguen los contextos **PUBLIC** y **ADMIN**. La Tabla 21 resume la distribución funcional resultante.

Tabla 21: Responsabilidades principales del backend

Componente	Elementos representativos	Responsabilidad
Presentación HTTP	Controladores y DTO	Recibir solicitudes, validar contratos y producir respuestas.
Documentación de API	Swagger/OpenAPI	Publicar documentación interactiva del contrato HTTP y del esquema Bearer.
Identidad y acceso	Auth, usuarios, roles, permisos y sesiones	Autenticar, renovar sesiones y restringir operaciones.
Catálogo	Categorías, marcas, productos, variantes y atributos	Mantener la estructura comercial y sus reglas de publicación.
Operación interna	Sucursales, áreas, ubicaciones e inventario	Representar la estructura física y registrar existencias y movimientos.
Documentos comerciales	Ventas, ajustes y fulfillment	Calcular importes y coordinar efectos transaccionales sobre inventario.
Servicios públicos	Catálogo, negocio y lista de deseos	Exponer proyecciones limitadas del modelo para visitantes y clientes.
Servicios transversales	Configuración, errores, auditoría y medios	Proveer comportamiento común sin duplicarlo por controlador.

Fuente: Autor.

4.2.3. Frontend

El frontend se implementó como una aplicación de página única (SPA) con React y TypeScript. Vite actúa como servidor de desarrollo y constructor del artefacto estático; Tailwind CSS se emplea para los estilos y las adaptaciones responsivas. La Tabla 22 resume las tecnologías y sus versiones utilizadas.

Tabla 22: Tecnologías principales del frontend implementado

Tecnología	Versión resuelta	Responsabilidad en Stockify
React y React DOM	19.2.6	Composición de la interfaz y providers de alcance global.
React Router DOM	7.15.1	Enrutamiento de landing, catálogo, cuenta pública y administración.
TanStack Query	5.101.0	Consulta, caché e invalidación del estado obtenido desde la API.
React Hook Form y Zod	7.76.1 y 4.4.3	Formularios y validación tipada de los datos de la interfaz.
TypeScript	6.0.3	Tipado estático en modo estricto.
Vite	8.1.0	Desarrollo y generación del paquete estático de producción.
Tailwind CSS	4.3.0	Utilidades de presentación y adaptación responsiva.
Vitest y Playwright	4.1.8 y 1.61.1	Pruebas unitarias, de componentes y de navegador.

Fuente: Autor.

Sobre esta base técnica, el frontend separa las pantallas navegables (pages), las capacidades de negocio reutilizables (features) y los recursos comunes (shared). La interfaz consume la API mediante un cliente fetch centralizado y utiliza TanStack Query para el estado remoto cacheable. La Tabla 23 resume la distribución funcional de estas áreas.

Tabla 23: Responsabilidades principales del frontend

Componente	Elementos representativos	Responsabilidad
Aplicación y navegación	src/app, App.tsx, enrutador, rutas diferidas y cliente de consultas.	Ensamblar proveedores, resolver las rutas y coordinar la composición inicial de la interfaz.
Autenticación y cuenta	features/auth, guards de ruta, formularios y esquemas.	Gestionar sesiones públicas y administrativas, acceso, perfil y seguridad de cuenta.

Administración	features/admin, pages/admin y componentes por dominio.	Implementar el panel para la gestión de catálogo, estructura, inventario, ventas, usuarios y configuración.
Catálogo público	features/public-catalog, pages/public y lista de deseos.	Presentar la información comercial, la navegación pública y las acciones de cuenta asociadas.
Landing institucional	pages/landing.	Presentar la página de aterrizaje independiente de la interfaz de cada negocio.
Recursos compartidos	shared/api, componentes, tema, consultas y validaciones.	Reutilizar el cliente HTTP, controles de interfaz, apariencia y utilidades transversales.

Fuente: Autor.

4.2.4. Infraestructura y DevOps

La infraestructura se mantiene en un repositorio independiente que combina Docker Compose, Nginx, GitLab CI/CD, el Container Registry y scripts de operación, lo que permite versionar la topología y los procedimientos de despliegue por separado del código de aplicación; la Tabla 24 resume estos componentes y su función.

Tabla 24: Componentes de infraestructura y su función

Componente	Aplicación en Stockify	Artefacto versionado
Docker y Compose	Frontend, proxy, PostgreSQL y backends generados por cliente.	Dockerfiles y Compose base, QA y producción.
Node.js 22	Construcción de las imágenes y ejecución del backend.	Dockerfiles de backend y frontend.
Nginx 1.29	Proxy inverso por dominio; publica frontend, API e imágenes.	Configuración global y bloques generados.
PostgreSQL 16	Motor compartido por ambiente con base diferenciada por cliente.	Servicios y volúmenes de Compose.
GitLab CI/CD y Registry	Imágenes identificadas por commit y flujos de infraestructura.	Canalizaciones de aplicación e infraestructura.
Scripts de provisión	Contrato validado, servicio, base, volumen y bloque Nginx por cliente.	Scripts de provisión y manifiesto de clientes.

Cloudflare Tunnel	Asocia el dominio del cliente con el punto de entrada local.	Sincronización de DNS e ingress con reversión ante error.
Despliegue y respaldo	Migraciones, healthchecks, respaldo previo y restauración de imagen.	Scripts de despliegue, operación y backup.

Fuente: Autor.

La composición base declara frontend y proxy con comprobaciones de salud; las extensiones de QA y producción añaden PostgreSQL, redes y volúmenes propios. El proxy se publica solo en la interfaz local y la exposición pública se delega al túnel. Nginx posee un servidor por defecto que rechaza dominios no reconocidos y bloques por cliente que dirigen / al frontend compartido, /api a su backend y /img al mismo servicio.

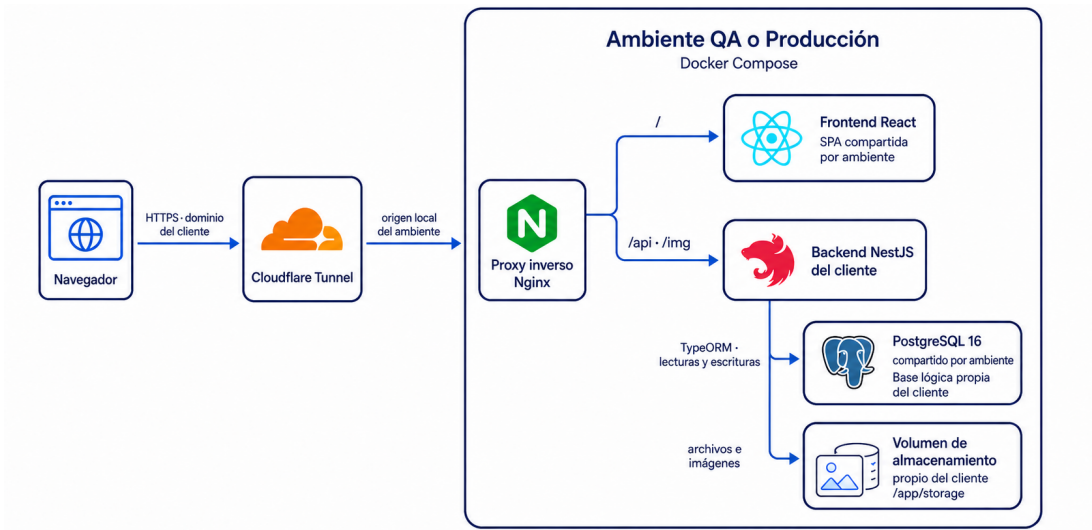
4.3. Arquitectura y fundamentos técnicos implementados

La implementación distribuye la solución entre una aplicación web ejecutada en el navegador, un proxy inverso, un servicio backend por cliente, PostgreSQL y volúmenes persistentes. Esta estructura corresponde a una arquitectura cliente–servidor desplegada mediante contenedores; la separación modular del código no se interpreta como microservicios, porque los módulos funcionales del servidor se entregan como una única aplicación backend para cada cliente.

4.3.1. Arquitectura cliente–servidor y despliegue multiinstancia

El navegador recibe la SPA React desde el servicio frontend. La interfaz no accede directamente a la base de datos ni al volumen de archivos: las operaciones se envían por HTTP al backend, que concentra el acceso a persistencia. El proxy Nginx constituye el punto de entrada del ambiente: para cada dominio registrado dirige la raíz al frontend compartido, traduce el prefijo público /api hacia el backend del cliente y envía /img al mecanismo de publicación de imágenes del mismo servicio, como resume la Figura 8.

Figura 8: Arquitectura cliente-servidor implementada



Fuente: Autor.

La infraestructura implementa dos ambientes Compose independientes, QA y producción, cada uno con su proyecto, red, volumen de PostgreSQL, proxy y frontend propios. Dentro de un ambiente, cada cliente registrado obtiene backend, base de datos y volumen de almacenamiento propios, con el bloque Nginx correspondiente incorporado automáticamente. Por tanto, la solución se clasifica como multiinstancia con separación lógica por cliente, no como *multi-tenant* de esquema compartido. La Tabla 25 resume la distribución de recursos resultante.

Tabla 25: Distribución de recursos en la arquitectura multiinstancia

Nivel	Recursos separados	Recursos compartidos
Entre QA y producción	Proyecto Compose, red, PostgreSQL, volúmenes, proxy, frontend, configuración y estado de despliegue.	La definición versionada y los scripts reutilizables del repositorio.
Entre clientes Stockify	Backend, dominio, nombre de base de datos y volumen de almacenamiento.	Frontend, proxy Nginx, instancia PostgreSQL, red y credenciales comunes de conexión.

Fuente: Autor.

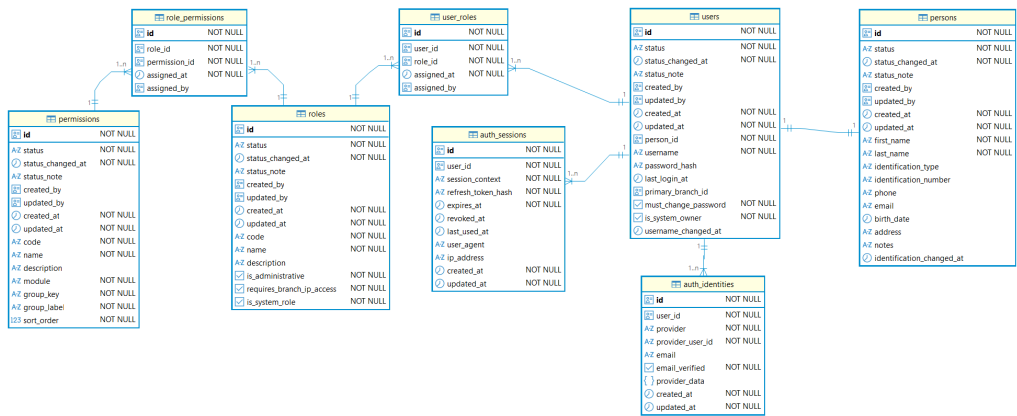
4.3.2. Modelo relacional y dominios de datos

El registro de entidades de TypeORM contiene 50 entidades. La Tabla 13 las organiza por responsabilidad funcional. Para mantener legible el modelo completo, las figuras de esta subsección presentan vistas relacionales separadas por dominio y muestran únicamente las tablas y claves foráneas necesarias para comprender cada recorrido.

Identidad y acceso

Este dominio separa la información personal de las credenciales de acceso. Cada usuario se vincula con una persona, puede disponer de una o más identidades de autenticación y conserva sesiones por contexto. La autorización se resuelve mediante las relaciones usuario-rol y rol-permiso, mientras los orígenes administrativos permiten asociar restricciones de acceso con una sucursal cuando corresponde.

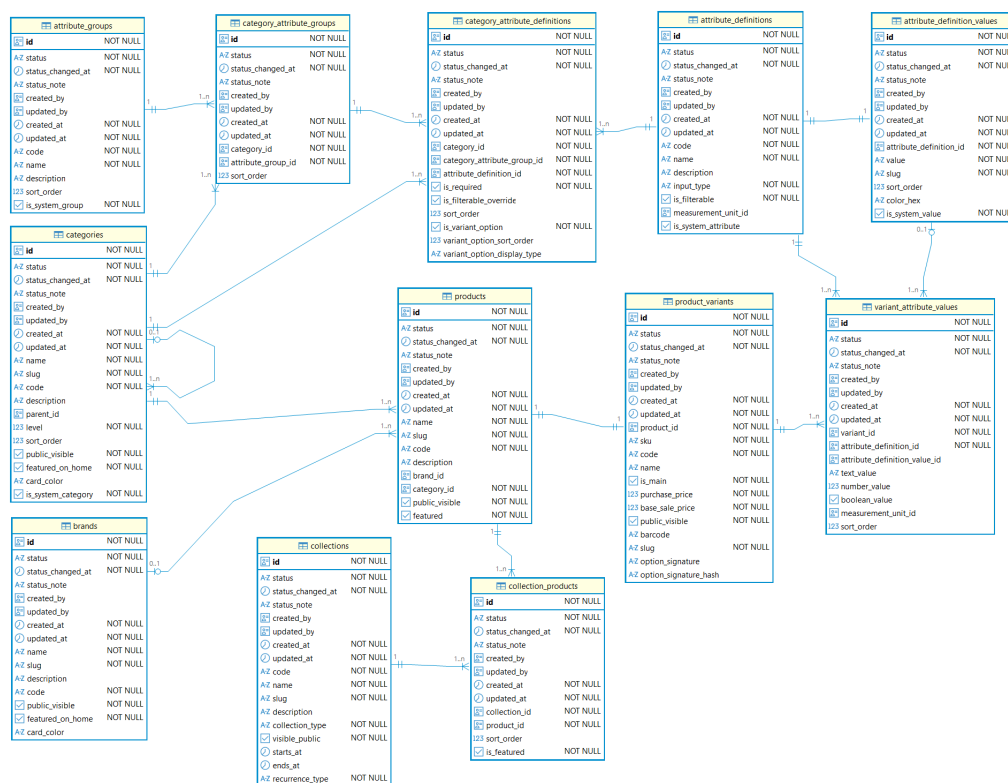
Figura 9: Relaciones del dominio de identidad y acceso



Fuente: Autor.

Catálogo

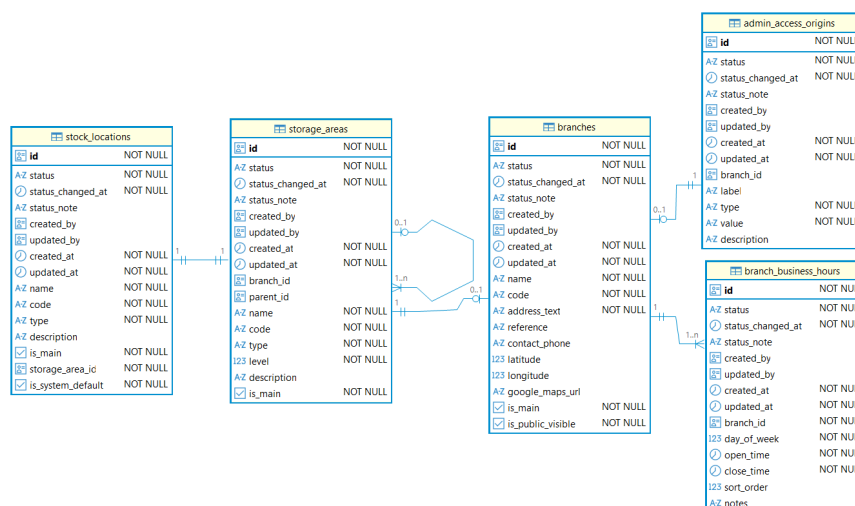
El catálogo organiza la clasificación y la información comercial en capas. Las categorías admiten jerarquía y determinan los atributos aplicables; las marcas identifican el origen comercial; el producto reúne los datos comunes y cada variante representa una combinación vendible con valores de atributo propios. Las colecciones relacionan productos visibles mediante un orden definido. La disponibilidad pública se deriva de las existencias elegibles y no se conserva como una propiedad manual de la variante.



Estructura operativa

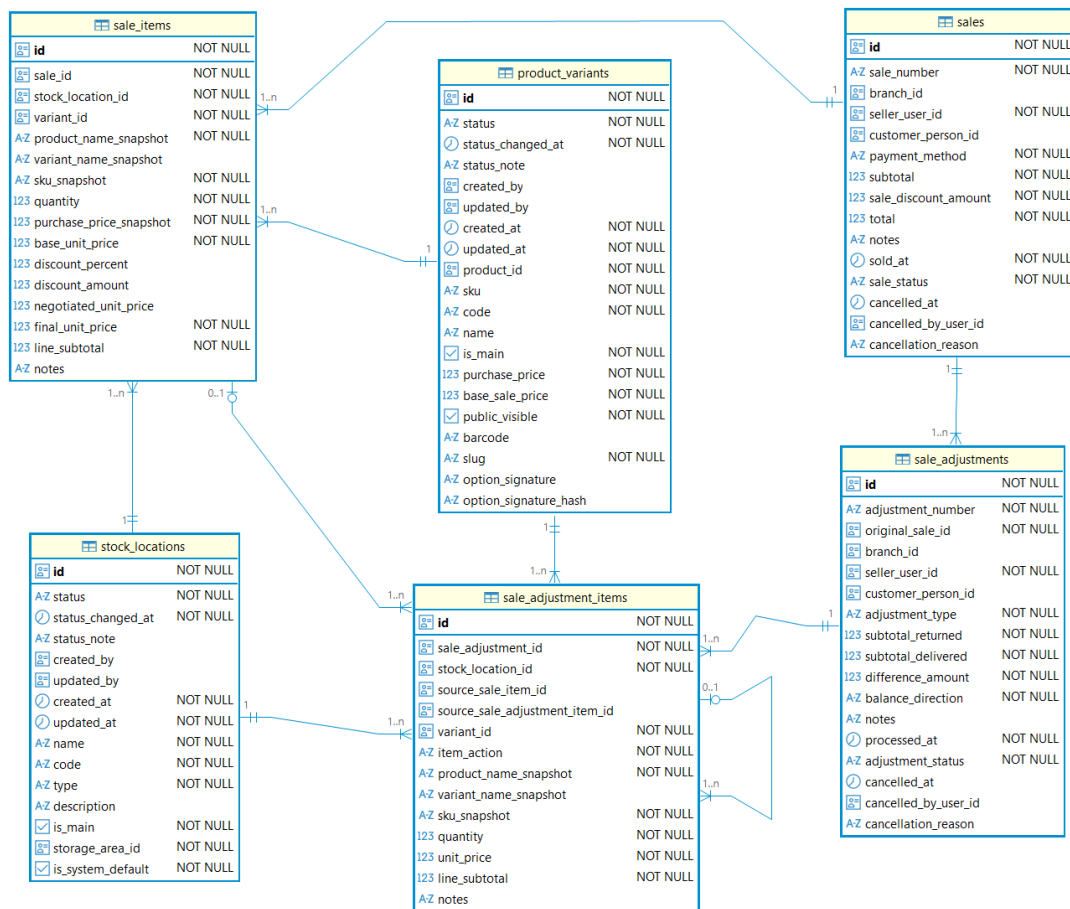
La estructura operativa modela los lugares donde puede existir stock. Una sucursal agrupa áreas de almacenamiento y estas contienen ubicaciones físicas; también pueden existir áreas independientes cuando la configuración del negocio lo permite. Los estados lógicos conservan el historial administrativo y evitan usar nodos inactivos o eliminados como si estuvieran disponibles para la operación.

Figura 11: Relaciones del dominio de estructura operativa



Fuente: Autor.

Figura 13: Relaciones del dominio de ventas

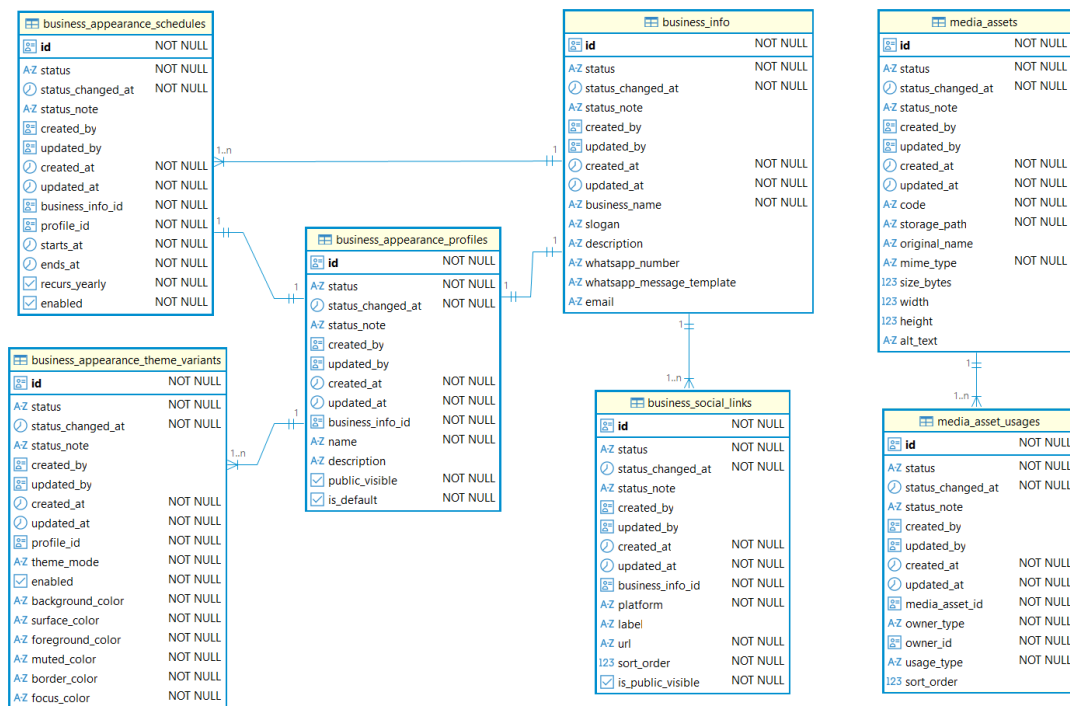


Fuente: Autor.

Medios y presentación

Los archivos multimedia se registran como activos con metadatos técnicos y se relacionan con sus usos funcionales mediante propietario y propósito visual. La información del negocio, los enlaces sociales, las configuraciones y los perfiles de apariencia complementan la presentación administrativa y pública sin mezclarse con las existencias ni los documentos de venta.

Figura 14: Relaciones del dominio de medios y presentación

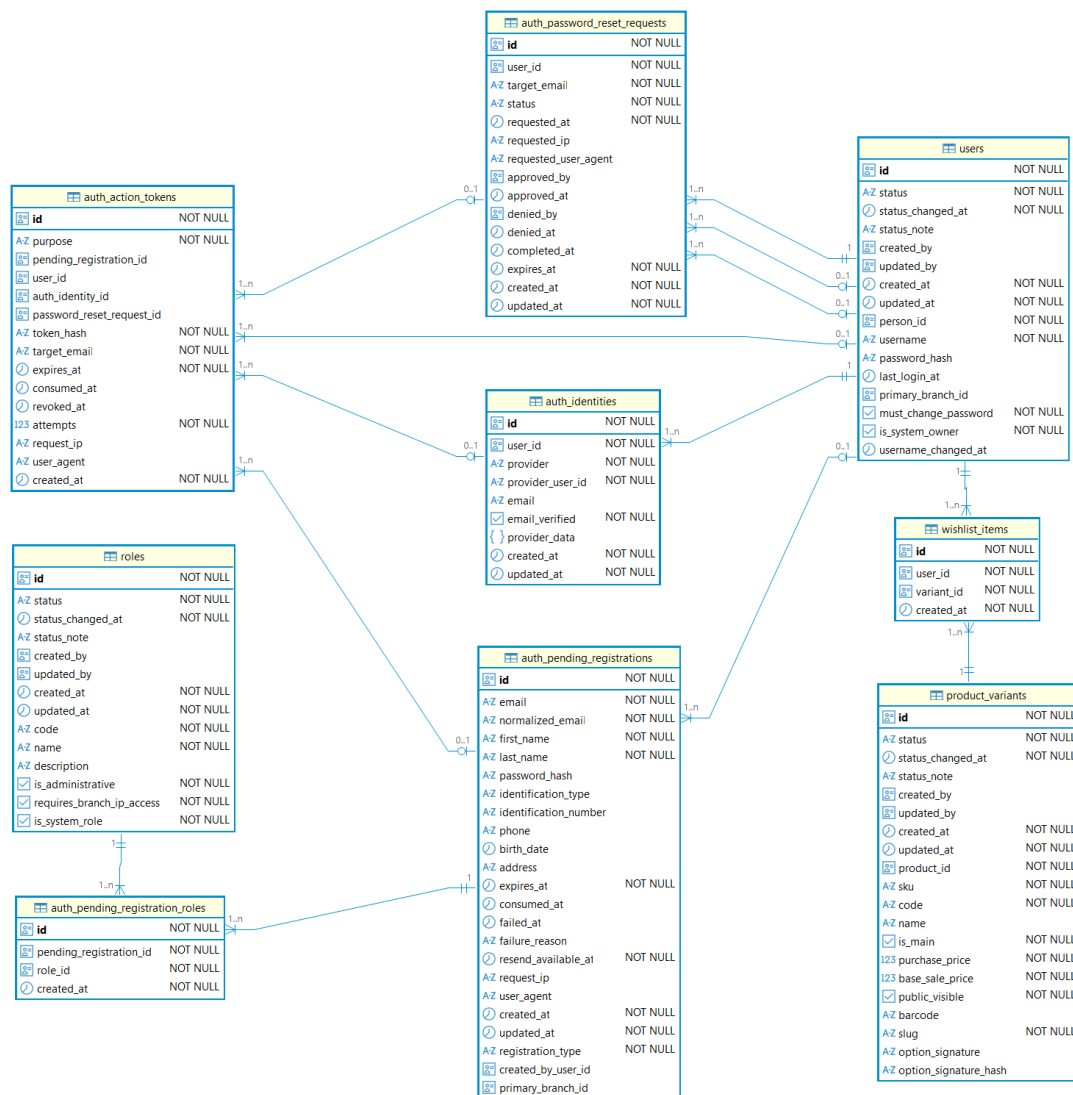


Fuente: Autor.

Funciones públicas

Las funciones públicas conservan los datos necesarios para registro, verificación, recuperación de credenciales y lista de deseos. Las solicitudes pendientes se mantienen separadas de la cuenta definitiva; una vez verificada la identidad, el usuario puede asociar variantes visibles a su lista de deseos.

Figura 15: Relaciones de las funciones públicas



Fuente: Autor.

4.3.3. Persistencia, migraciones y seeds

TypeORM utiliza un DataSource para los comandos de migración y una configuración de runtime para NestJS que comparten el registro de entidades. La opción migrationsRun permanece en false; por ello, los cambios del esquema no se sincronizan automáticamente y su aplicación se controla mediante comandos explícitos. El comando npm run typeorm - migration:show identifica las migraciones pendientes en el orden en que deben ejecutarse.

Figura 16: Migraciones pendientes identificadas mediante TypeORM

```
$ npm run typeorm -- migration:show

> stockify-back@0.0.1 typeorm
> typeorm-ts-node-commonjs -d src/database/data-source.ts migration:show

[ ] AutoMigration1777433535251
[ ] AutoMigration1777691727611
[ ] AutoMigration1777699661357
[ ] CreateAuthSessions1777900000000
[ ] ExpandRoleCodeLength1778000000000
[ ] CreateCategories1778100000000
[ ] ConsolidateCategoryIndexes1778200000000
[ ] AddNormalizedRoleUniqueness1778300000000
```

Fuente: Autor.

Cuando se modifica el modelo de entidades, `npm run migration:generate` compara el modelo con el esquema de la base de datos y genera una migración inicial. El archivo generado como `1778426341227-AutoMigration` se revisó y se renombró como `1778426341227-CreateBrands` para identificar el cambio incorporado. A continuación, `npm run migration:run` aplica las migraciones pendientes y actualiza la estructura de persistencia.

Figura 17: Inicio de la ejecución de migraciones pendientes

```
$ npm run migration:run

> stockify-back@0.0.1 migration:run
> npm run typeorm -- migration:run

> stockify-back@0.0.1 typeorm
> typeorm-ts-node-commonjs -d src/database/data-source.ts migration:run

query: SELECT version()
query: SELECT * FROM current_schema()
query: CREATE EXTENSION IF NOT EXISTS "uuid-oss"
query: SELECT * FROM "information_schema"."tables" WHERE "table_schema" = 'public' AND "table_name" = 'migrations'
query: SELECT * FROM "migrations" "migrations" ORDER BY "id" DESC
0 migrations are already loaded in the database.
9 migrations were found in the source code.
9 migrations are new migrations must be executed.
query: START TRANSACTION
query: CREATE EXTENSION IF NOT EXISTS "uuid-oss"
query: CREATE TYPE "public"."entity_status" AS ENUM('ACTIVE', 'INACTIVE', 'DELETED')
```

Fuente: Autor.

Figura 18: Migraciones aplicadas después de la actualización del esquema

```
$ npm run typeorm -- migration:show
> stockify-back@0.0.1 typeorm
> typeorm-ts-node-commonjs -d src/database/data-source.ts migration:show

[X] 1 AutoMigration1777433535251
[X] 2 AutoMigration1777691727611
[X] 3 AutoMigration1777699661357
[X] 4 CreateAuthSessions1777900000000
[X] 5 ExpandRoleCodeLength1778000000000
[X] 6 CreateCategories1778100000000
[X] 7 ConsolidateCategoryIndexes1778200000000
[X] 8 AddNormalizedRoleUniqueness1778300000000
[ ] CreateBrands1778426341227
```

Fuente: Autor.

Tras aplicar las migraciones, `npm run seed` inicializa los datos en un orden definido: roles y permisos, usuario administrativo, catálogo base, información y configuraciones del negocio, apariencia y estructura operativa. Las configuraciones se procesan por clave y conservan el valor editado por el administrador; el bootstrap operacional crea sucursal, área y ubicación dentro de una transacción, rechazando configuraciones que dejarían al sistema sin estructura inicial.

Figura 19: Ejecución del seed de datos iniciales

```
$ npm run seed
> stockify-back@0.0.1 seed
> ts-node -r tsconfig-paths/register src/database/seeds/seed.ts
```

Fuente: Autor.

4.3.4. Backend y contratos API

La API REST utiliza un prefijo configurable y agrupa las rutas por dominio y contexto. Los controladores administrativos se sitúan bajo `admin`; las operaciones abiertas usan `public` o el decorador `@Public()`; y la autenticación separa los flujos público y administrativo.

Los contratos de entrada se expresan mediante DTO y `class-validator`. Un mecanismo global de validación elimina campos ajenos, rechaza propiedades no permitidas y transforma tipos; un manejador global de errores normaliza las respuestas de error.

Figura 20: Configuración global de validación y normalización de errores

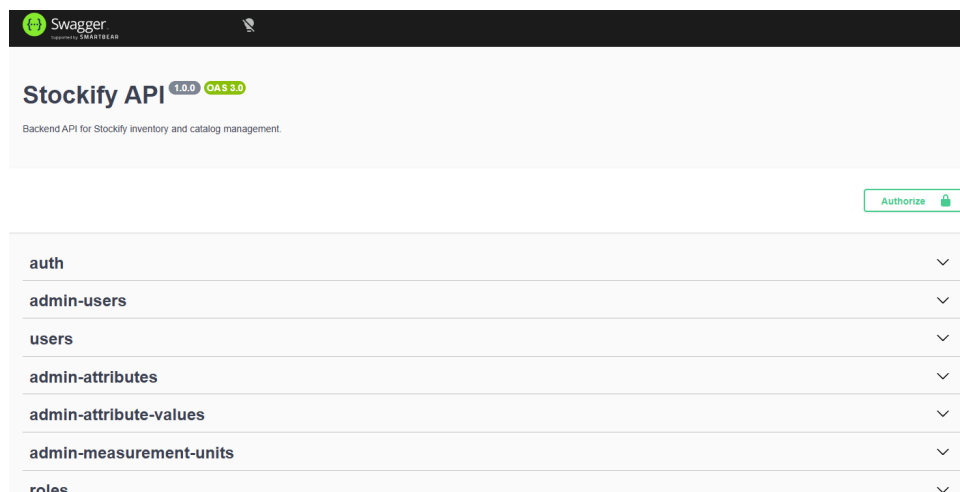
```
app.useGlobalPipes(
  new ValidationPipe({
    whitelist: true,
    forbidNonWhitelisted: true,
    transform: true,
    transformOptions: {
      enableImplicitConversion: true,
    },
    validationError: {
      target: false,
      value: false,
    },
  }),
);
app.useGlobalFilters(globalHttpExceptionHandler);
```

Fuente: Autor.

La autorización no se delega al cliente; JwtAuthGuard protege las rutas por defecto y RolesGuard, SessionContextGuard y PermissionsGuard aplican las restricciones declaradas como metadatos, con el bypass explícito del rol administrativo.

La documentación se genera a partir de los controladores y sus metadatos, y presenta el contrato HTTP agrupado por contexto funcional. Esta organización distingue las rutas administrativas, públicas, de autenticación y de consulta técnica sin exponer la implementación interna de cada módulo. La Figura 21 muestra esta agrupación.

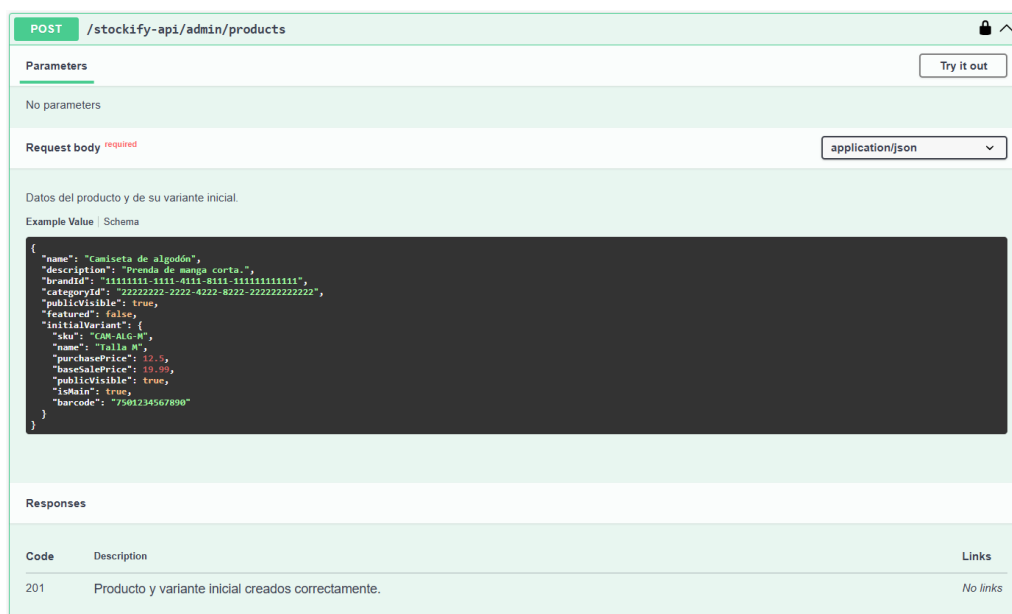
Figura 21: Documentación agrupada por contexto funcional



Fuente: Autor.

Cada operación expuesta por la API documenta el método, la ruta, el esquema de autenticación, los datos de entrada y las respuestas esperadas. Por ejemplo, el registro administrativo de productos de la Figura 22 ilustra esta estructura mediante una operación protegida que incorpora un cuerpo JSON y una respuesta de creación.

Figura 22: Contrato HTTP documentado para el registro administrativo de productos



Fuente: Autor.

4.3.5. Frontend: rutas y gestión de estado

La raíz de la SPA reúne el cliente de consultas, la identidad visual, la apariencia, la autenticación, las notificaciones y el enrutador. Estos servicios se inicializan una sola vez y quedan disponibles para las páginas de la aplicación.

Figura 23: Proveedores globales que componen la aplicación de página única

```
export function App() {
  const hostname = globalThis.location?.hostname ?? ''

  if (isLandingHost(hostname)) return <LandingApp />

  const app = (
    <QueryClientProvider client={queryClient}>
      <DocumentBrandingSync />
      <AppearanceRuntimeProvider>
        <AuthProvider>
          <ToastProvider>
            <RouterProvider router={appRouter} />
          </ToastProvider>
        </AuthProvider>
      </AppearanceRuntimeProvider>
      {ReactQueryDevtools ? (
        <Suspense fallback={null}>
          <ReactQueryDevtools initialIsOpen={false} />
        </Suspense>
      ) : null}
    </QueryClientProvider>
  )

  if (!env.GOOGLE_CLIENT_ID) return app

  return (
    <GoogleOAuthProvider clientId={env.GOOGLE_CLIENT_ID}>
      {app}
    </GoogleOAuthProvider>
  )
}
```

Fuente: Autor.

El área administrativa organiza sus módulos mediante un registro central de pestañas y paneles diferidos para usuarios, catálogo, inventario, ventas, estructura operativa, negocio y apariencia. La visibilidad de cada opción depende de los permisos, mientras el servidor conserva la decisión final de autorización.

El enrutador separa las rutas públicas de las administrativas. La sección pública incluye inicio, catálogo, colecciones, autenticación, perfil y lista de deseos; la administrativa mantiene su inicio de sesión y un layout protegido. Los *guards* consultan y restauran la sesión, aplican el cambio obligatorio de contraseña y las rutas no existentes muestran una vista común. Las páginas de mayor tamaño se cargan de forma diferida.

Figura 24: Enrutamiento de las áreas pública y administrativa de la aplicación

```
export const appRouter = createBrowserRouter([
  {
    path: '/',
    element: (
      <PublicLayout>
        <HomePage />
      </PublicLayout>
    ),
  },
  // Resto de rutas públicas y protegidas
  {
    path: '/admin',
    element: (
      <AdminProtectedRoute>
        <AdminShellRoute />
      </AdminProtectedRoute>
    ),
    children: [
      { index: true, element: <AdminPageRoute /> },
      { path: 'account', element: <AdminAccountPageRoute /> },
      { path: 'account/security', element: <AdminAccountSecurityPageRoute /> },
      {
        path: 'password-reset-requests/:id',
        element: <AdminPasswordResetRequestDetailRoute />,
      },
    ],
  },
  {
    path: '*',
    element: <NotFoundPage />,
  },
])
```

Fuente: Autor.

Los mecanismos de estado utilizados se resumen en la Tabla 26.

Tabla 26: Mecanismos de estado utilizados en el frontend

Tipo de estado	Mecanismo	Aplicación observada
Estado remoto	TanStack Query	Negocio, catálogo, detalle, colecciones, wishlist, sesión y datos administrativos.
Estado de navegación	URL y React Router	Búsqueda, categoría, marca, orden y página del catálogo.

Estado de sesión	AuthProvider y almacenamiento del navegador	Usuario y access token por contexto PUBLIC/ADMIN.
Estado de formularios	React Hook Form y Zod	Valores, errores y validación de formularios.
Estado de interfaz	Hooks locales de React	Menús, diálogos, filtros móviles y selección de variantes.

Fuente: Autor.

El catálogo conserva búsqueda, filtros, orden y página en la URL, por lo que la consulta puede reconstruirse al recargar o compartir la dirección. Como política predeterminada, TanStack Query considera vigente una consulta durante treinta segundos y conserva los datos inactivos durante cinco minutos; algunas consultas establecen valores específicos. No reintenta errores 4xx ni mutaciones. El proveedor de autenticación separa las consultas y credenciales de los contextos público y administrativo y las elimina al cerrar sesión. El *access token* se almacena en `localStorage` para restaurar la interfaz tras una recarga, con la exposición ante XSS reconocida como límite de seguridad del cliente.

4.3.6. Integración entre frontend y backend

La integración se concentra en un cliente HTTP común y módulos API por funcionalidad. La URL base proviene de la configuración del frontend; en desarrollo Vite dirige `/api` y `/img`, y en los ambientes contenedorizados Nginx conserva esos prefijos. El cliente serializa JSON, conserva `FormData` sin fijar el tipo multipart, convierte respuestas no satisfactorias en un error de API con estado y detalle, y aplica `no-store` como política HTTP predeterminada, concentrando la caché funcional en TanStack Query. Las claves de consulta representan el recurso y sus filtros y no incluyen tokens.

Para solicitudes protegidas, los módulos API indican el contexto y el cliente añade el token Bearer. Ante un 401 (excepto en login, logout y refresh) se comparte una única promesa de renovación por contexto: la renovación envía la cookie mediante `credentials: include`, persiste el nuevo access token y repite una sola vez la solicitud original; un error definitivo se convierte en el estado de error de la consulta o mutación correspondiente. Los tipos TypeScript, esquemas de formulario y módulos API constituyen el contrato consumido por la interfaz, pero no una frontera de confianza: el backend vuelve a validar cuerpo, identidad y permisos en cada operación.

4.3.7. Almacenamiento y exposición de imágenes

La gestión de imágenes separa el archivo físico de su utilización funcional: `media_assets` conserva código, ruta, tipo, tamaño, dimensiones y texto alternativo, y `media_asset_usages` relaciona el activo con un propietario polimórfico (marca, categoría, producto, variante, colección o negocio) y un propósito visual. Esta separación permite reutilizar un activo y conocer sus asociaciones antes de eliminarlo.

El servicio admite cargas JPEG, PNG, WebP y GIF, valida el tipo de archivo declarado (MIME) y tamaño, comprueba con Sharp que el contenido sea decodificable, rota la imagen según sus metadatos y la transforma en tres variantes WebP de la Tabla 27. Los archivos se almacenan bajo una raíz configurable, y Express monta la ruta pública con `Cross-Origin-Resource-Policy: cross-origin` para que la SPA servida desde otro origen pueda mostrarlas.

Tabla 27: Variantes de imagen generadas por el backend

Variante	Archivo	Dimensión máxima	Uso previsto
Original procesada	<code>original.webp</code>	2000 píxeles	Presentación de mayor tamaño.
Optimizada	<code>optimized.webp</code>	1024 píxeles	Fichas y componentes de contenido.
Miniatura	<code>thumbnail.webp</code>	320 píxeles	Listados, selectores y vistas compactas.

Fuente: Autor.

4.4. Área administrativa

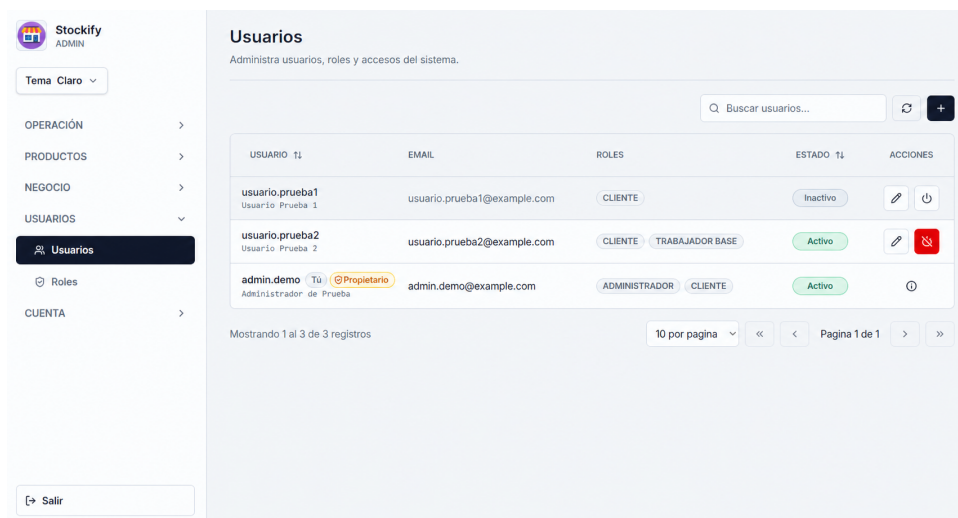
El área administrativa reúne funciones de identidad, configuración, catálogo, inventario y ventas bajo un contexto de sesión independiente al catálogo público. La interfaz organiza los recorridos y condiciona acciones según permisos, mientras el backend vuelve a validar autorización, estado, relaciones y transacciones. Las subsecciones describen las capacidades implementadas y mantienen visibles los recorridos parciales o riesgos pendientes.

4.4.1. Usuarios, roles y permisos

La administración de identidad separa la consulta del perfil propio de las operaciones del contexto administrativo. La interfaz ofrece búsqueda, detalle, invitación de usuarios, edición, cambio de estado y asignación de roles; una invitación es un registro pendiente y no crea una cuenta con credenciales hasta validar su identidad. Los datos ordinarios y

sensibles se tratan de forma distinta: modificar identificadores como el nombre de usuario exige el permiso de actualización sensible o el rol ADMIN, y los cambios de estado y de roles poseen permisos separados, de modo que un operador reciba solo la capacidad necesaria para su función.

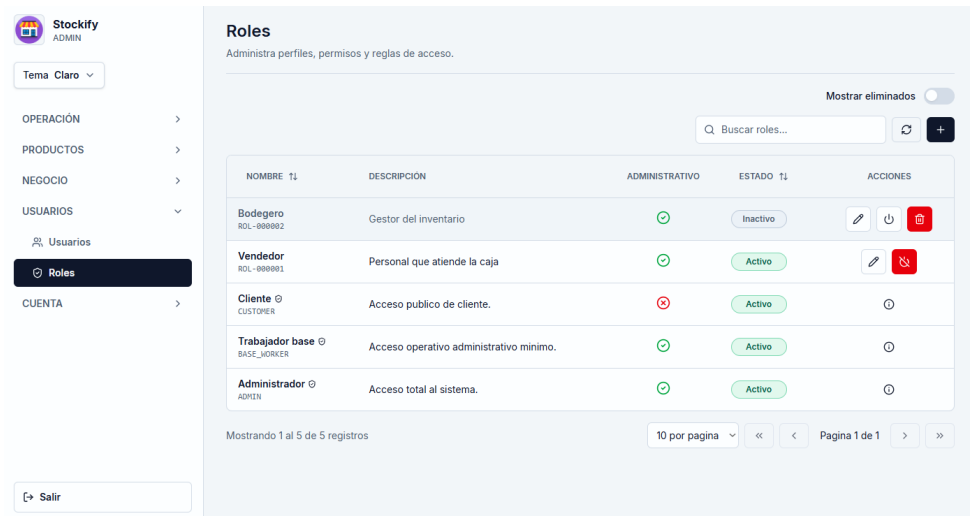
Figura 25: Panel administrativo de gestión de usuarios



Fuente: Autor.

Los roles de sistema son de solo lectura y no se eliminan; el catálogo de permisos se consume desde la edición del rol. La revocación de sesiones administrativas tras inactivar un usuario o modificar sus roles evita que un access token siga representando concesiones anteriores; la misma medida se aplica a los usuarios afectados cuando cambia la configuración de un rol. Estas reglas se complementan con las protecciones que impiden aplicar ciertas operaciones al propio operador o al propietario de sistema. La interfaz replica la convención de permisos para ocultar o deshabilitar acciones, con bypass visual de ADMIN, pero la decisión final permanece en el backend.

Figura 26: Panel administrativo de roles y permisos



Fuente: Autor.

4.4.2. Biblioteca y selección de imágenes

La gestión de imágenes se centraliza en una biblioteca reutilizable desde los formularios que requieren asociar archivos, como marcas, categorías, productos, variantes, colecciones y el logo del negocio. Estos formularios incorporan un componente que permite abrir la galería, localizar todas las imágenes disponibles y asociarlas al registro que se está editando, sin duplicar archivos entre los distintos módulos administrativos.

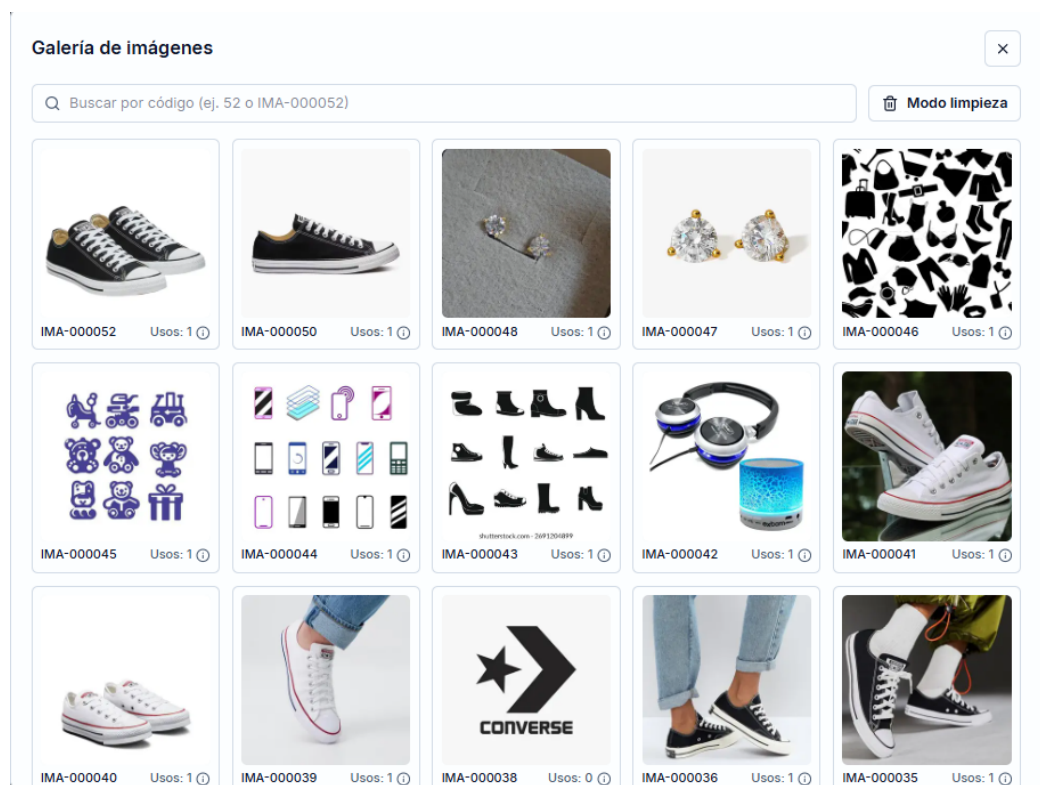
Figura 27: Formulario con acceso reutilizable a la galería de imágenes



Fuente: Autor.

La biblioteca presenta las imágenes cargadas y su código; además, permite consultar las entidades que las utilizan. Quitar una asignación solo desvincula la imagen del registro y conserva el archivo en la biblioteca. En cambio, el modo de eliminación solo está disponible con imágenes sin usos asociados y las elimina de manera permanente de su ubicación física de almacenamiento.

Figura 28: Biblioteca administrativa de imágenes y consulta de usos



Fuente: Autor.

4.4.3. Negocio y estructura operativa

La configuración del negocio reúne información institucional, enlaces de redes sociales y configuraciones que alteran el comportamiento comercial: publicación de productos agotados, habilitación de áreas independientes y exigencia de sucursal al vender. La interfaz permite consultarlos y modificarlos con los permisos correspondientes.

Figura 29: Panel de información del negocio

Negocio
Configura la identidad, contacto, catálogo público y sucursal principal del negocio.

Información general
Identidad del negocio, logo y datos de contacto.

Logo  Ver galería

Nombre del negocio 8/75

Eslogan 20/255

Descripción

WhatsApp Correo de contacto

CREADO POR Sistema - jueves, 9 de julio de 2026, 10:17 p. m. ACTUALIZADO POR Kevin Andres Tarqui Tapia - jueves, 23 de julio de 2026, 1:00 a. m.

Guardar

Redes sociales
Enlaces a tus redes sociales para el catálogo público.
Aun no agregaste redes sociales. + Agregar

Sacar

Fuente: Autor.

Figura 30: Panel de configuraciones del negocio

Negocio
Configura la identidad, contacto, catálogo público y sucursal principal del negocio.

Catálogo público
Controla si los clientes ven el catálogo público.

Catálogo público activo ☒ El catálogo público se muestra normalmente a los clientes.

Disponibilidad de áreas independientes
Define si el stock de áreas independientes (sin sucursal) cuenta para la disponibilidad del catálogo público. Útil para negocios sin sucursales.

Contar áreas independientes en disponibilidad pública ☒ El stock de áreas independientes activas se cuenta como disponible en el catálogo.

Productos agotados en el catálogo
Controla si el catálogo público muestra productos sin stock disponible.

Mostrar productos agotados ☐ Los productos agotados se ocultan de listados, búsqueda y detalle (los "Próximamente" siguen visibles).

Sucursal en ventas
Controla si registrar una venta exige seleccionar una sucursal.

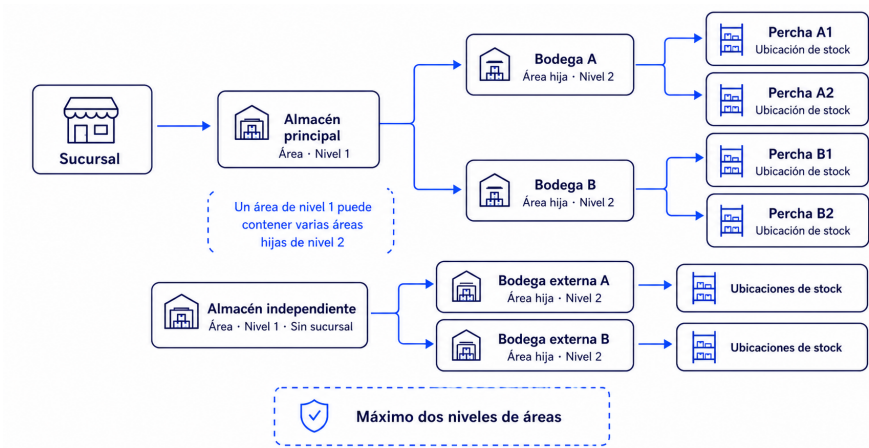
Exigir sucursal al registrar ventas ☒ Cada venta debe registrarse en una sucursal.

Sacar

Fuente: Autor.

La estructura física admite dos tipos de raíz: una sucursal con áreas de almacenamiento y ubicaciones de stock, o un área independiente con ubicaciones de stock, de modo que no es necesario inventar una sucursal para representar una bodega autónoma. Cada rama admite hasta dos niveles de áreas. El área de nivel 1 sería de tipo almacén y puede agrupar varias bodegas de nivel 2. La Figura 31 representa de mejor manera esta jerarquía.

Figura 31: Jerarquías admitidas por la estructura operativa



Fuente: Autor.

El servicio verifica que el negocio conserve al menos una raíz operativa activa y bloquea inactivaciones o eliminaciones que dejarían la organización sin estructura utilizable o que afectan nodos con stock. La Tabla 28 resume las reglas verificadas.

Tabla 28: Reglas verificadas sobre la estructura operativa

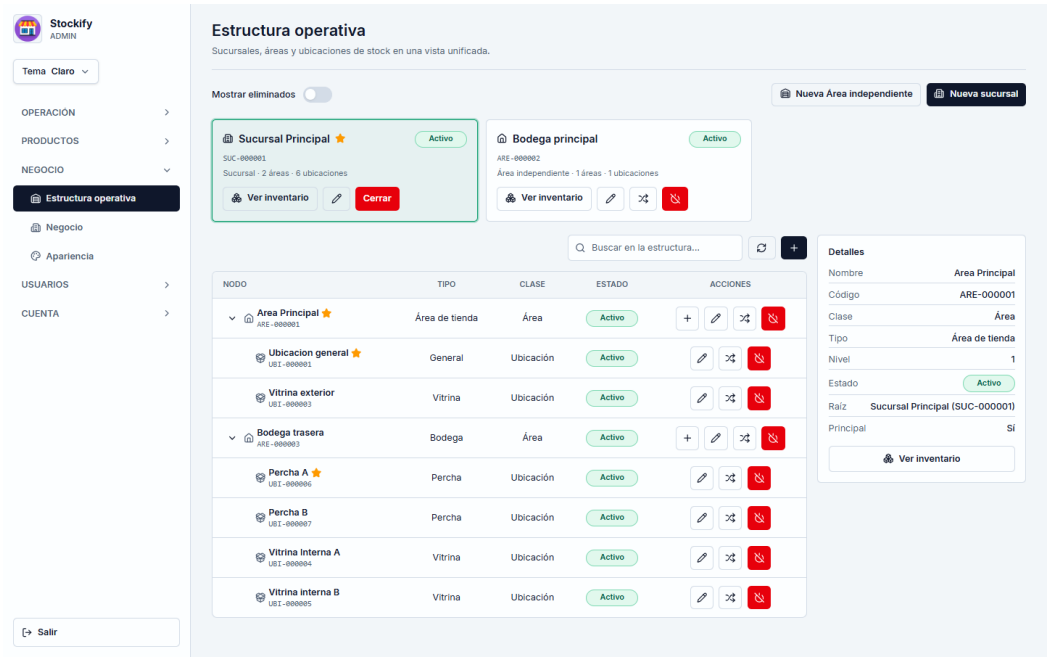
Operación	Precondición principal	Efecto implementado
Cierre de sucursal	Sucursal activa, estructura restante operativa y sin stock positivo afectado.	Cambia a inactivo, conserva el motivo y registra auditoría operativa.
Eliminación controlada	Nodo inactivo y ubicaciones sin stock positivo.	Modifica transaccionalmente los estados del nodo y sus dependencias.

Reasignación de área o ubicación	Destino y relación jerárquica válidos; no pretende trasladar existencias.	Corrección administrativa con valores anterior y nuevo registrados.
----------------------------------	---	---

Fuente: Autor.

El panel de estructura presenta el árbol de raíces y nodos, condiciona las acciones a los permisos de la sesión y enlaza con la consulta de inventario del elemento seleccionado. La distinción entre reasignación y traslado es esencial: cambiar la relación de una ubicación con un área corrige la configuración sin crear entradas ni salidas de stock; una transferencia física registra dos extremos y sus movimientos en el kardex.

Figura 32: Panel administrativo de estructura operativa



Fuente: Autor.

4.4.4. Marcas, categorías, atributos, productos y variantes

La gestión administrativa del catálogo separa las referencias que preparan la clasificación de los productos y de sus variantes. Marcas, categorías y atributos definen la información reutilizable; los productos agrupan los datos comunes y las variantes representan la unidad comercial concreta.

Marcas

Las marcas se administran como referencias reutilizables para identificar el origen comercial de los productos. El panel permite las acciones básicas CRUD; al asociarlas a un producto, el backend exige que permanezcan activas y controla el impacto de los cambios sobre los registros relacionados. Si se oculta una marca, el catálogo público oculta sus productos asociados. Además, una marca puede marcarse como destacada para aparecer en la pantalla de inicio del catálogo público.

Figura 33: Panel administrativo de marcas

Stockify

ADMIN

Tema Claro

OPERACIÓN

PRODUCTOS

PRODUCTOS

- Productos
- Categorías
- Marcas**
- Colecciones
- Atributos

NEGOCIO

USUARIOS

CUENTA

Salir

Marcas

Administra marcas comerciales y logos del catálogo.

Mostrar eliminadas

Q. Buscar marcas...

MARCA	LOGO	DESCRIPCIÓN	VISIBILIDAD	ACTUALIZADO	ESTADO	ACCIONES
Converse			Visible	23/7/26, 3:25 a. m.	Activo	  
JBL			Visible	23/7/26, 2:46 a. m.	Activo	  
Skechers			Visible	23/7/26, 2:21 a. m.	Activo	  
Axe			Oculto	23/7/26, 3:27 a. m.	Activo	  
Resona			Visible	23/7/26, 3:27 a. m.	Inactivo	  
Lego			Visible	23/7/26, 2:17 a. m.	Activo	  
Adidas			Visible	23/7/26, 2:17 a. m.	Activo	  
Samsung			Visible	23/7/26, 2:15 a. m.	Activo	  

Mostrando 1 al 8 de 8 registros

10 por página

Página 1 de 1

Fuente: Autor.

Categorías y atributos

Las categorías organizan el catálogo en una jerarquía flexible de hasta tres niveles. Una categoría puede actuar como raíz o depender de otra, sin relaciones circulares; esto permite representar familias, subfamilias y agrupaciones más específicas sin fijar una única estructura comercial. Antes de asociar un producto, el sistema comprueba que la categoría esté activa y sea asignable; si se oculta, el catálogo público también oculta sus productos asociados. Al igual que las marcas, las categorías pueden marcarse como destacadas para mostrarse en la pantalla de inicio del catálogo público.

Figura 34: Panel administrativo de categorías

Stockify

ADMIN

Tema Claro

OPERACIÓN

PRODUCTOS

PRODUCTOS

- Productos
- Categorías**
- Marcas
- Colecciones
- Atributos

NEGOCIO

USUARIOS

CUENTA

Salir

Categorías

Organiza categorías y subcategorías del catálogo.

Mostrar eliminadas

Q. Buscar categorías...

CATEGORÍA	IMAGEN	DESCRIPCIÓN	NIVEL	VISIBILIDAD	ESTADO	ACCIONES
Aretes	Sin imagen		Nivel 1	Visible	Activo	  
Audio			Nivel 1	Visible	Activo	  
Audifonos	Sin imagen		Nivel 2	Visible	Activo	  
Parlantes	Sin imagen		Nivel 2	Visible	Activo	  
Calzado			Nivel 1	Visible	Activo	  
Casual	Sin imagen		Nivel 2	Visible	Activo	  
Deportivo	Sin imagen		Nivel 2	Visible	Activo	  
Formal	Sin imagen		Nivel 2	Visible	Activo	  
Celulares			Nivel 1	Visible	Inactivo	  
Desodorantes	Sin imagen		Nivel 1	Oculto	Inactivo	  

Fuente: Autor.

Los atributos describen características que diferencian o complementan las variantes de una categoría, como color, tamaño, material, capacidad o dimensiones. Cada definición establece su tipo de entrada —selección, selección múltiple, texto, número o valor booleano— y puede disponer de valores predefinidos o de una unidad de medida. La categoría selecciona cuáles de esos atributos resultan aplicables, si son obligatorios, si intervienen en filtros y el orden en el que se presentan sus valores (en caso de ser tipo selección). Estas asociaciones se heredan hacia las categorías hijas, que pueden añadir atributos propios; así se evita repetir una misma configuración en cada rama. Por último, el seed incorpora el atributo de sistema *Color*, con valores predefinidos y representación hexadecimal.

Figura 35: Panel administrativo de atributos

ATRIBUTO	TIPO	UNIDAD	FILTRO	ESTADO	ACCIONES
Color COLOR	Selección	-	Si	Activo	[Edit] [Delete]
Version Bluetooth ATR-0000007	Selección	-	No	Activo	[Edit] [Delete]
Bluetooth ATR-0000008	Booleano	-	No	Activo	[Edit] [Delete]
Peso g ATR-0000007	Número	Gramo (g)	No	Inactivo	[Edit] [Toggle] [Delete]
Peso Kg ATR-0000005	Número	Kilogramo (kg)	No	Activo	[Edit] [Delete]
Almacenamiento ATR-0000005	Selección	-	No	Activo	[Edit] [Delete]
Intensidad ATR-0000004	Selección	-	No	Activo	[Edit] [Delete]
Aroma ATR-0000003	Texto	-	No	Activo	[Edit] [Delete]
Material ATR-0000002	Selección	-	No	Activo	[Edit] [Delete]
Talla ATR-0000001	Selección	-	Si	Activo	[Edit] [Delete]

Fuente: Autor.

Cuando un atributo se marca como opción de variante, la configuración define su orden y la forma en que el catálogo público presenta el selector, por ejemplo mediante botones, lista desplegable, muestra de color o muestra de imagen. Esta opción facilita que la persona usuaria cambie entre variantes por una característica relevante, mientras los valores asociados conservan el detalle propio de cada variante.

Figura 36: Configuración de atributos aplicables a una categoría

Atributos de Audio

Atributo *

Color (Selección)

Grupo *

General

Filtro categoría

Usar valor global

Requerido

Opción de variante

Eje comercial para seleccionar variantes públicas (independiente del filtro). Solo aplica a "Selección única", "Número", "Si/No".

Visualización pública *

Muestra de color

Cancelar edición

Actualizar

General

Color

Selección

Propio de esta categoría

Requerido

Filtrable

Material

Selección

Propio de esta categoría

Requerido

Conectividad

Bluetooth

Si / No

Propio de esta categoría

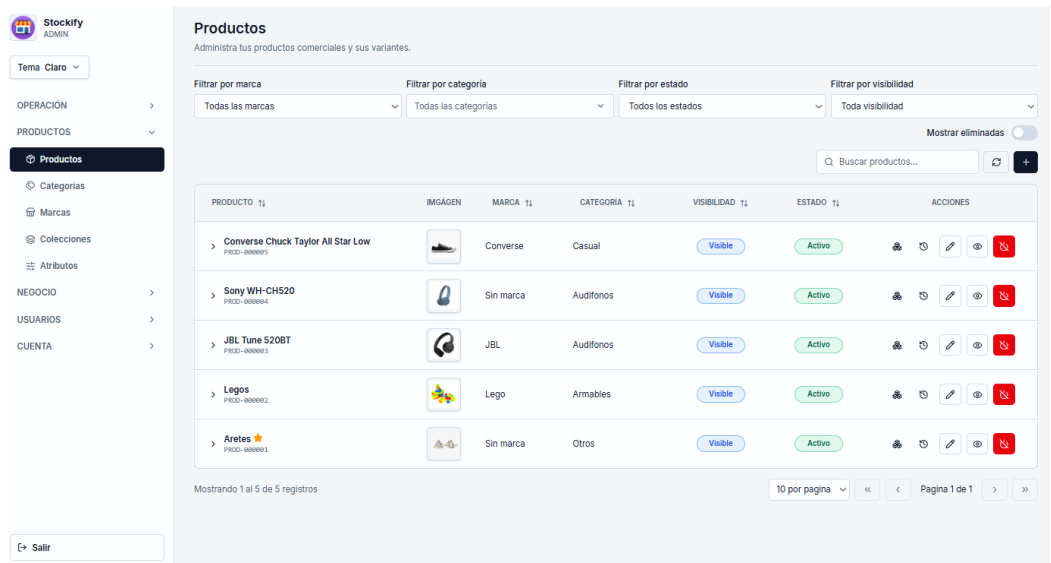
Requerido

Fuente: Autor.

Productos y variantes

El producto no representa por sí solo la unidad que se vende o se descuenta del inventario; reúne la información compartida por una familia comercial, como nombre, descripción, marca, categoría, visibilidad general y destacado. La variante representa la combinación comercial concreta: posee código, SKU, precios, código de barras cuando corresponde, atributos, imágenes, estado y visibilidad propios. Por ello, las existencias, ventas y consultas públicas se resuelven sobre las variantes y no directamente sobre el producto.

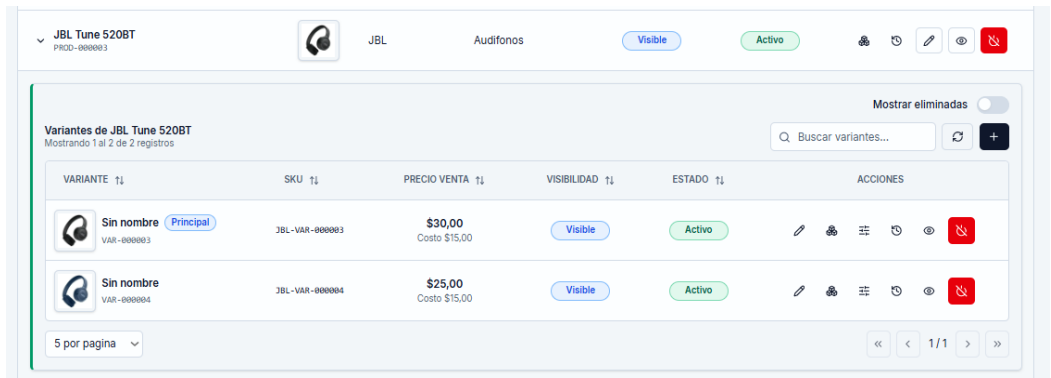
Figura 37: Panel administrativo de productos y variantes



Fuente: Autor.

La interfaz concentra en productos un asistente de creación, filtros, consulta de detalle y filas expandibles para sus variantes. Al crear o modificar un producto, el backend verifica que la marca esté activa y que la categoría pueda recibir productos. Los valores de atributos de cada variante se normalizan para formar una firma estable; si otra variante del mismo producto presenta la misma combinación, la operación se rechaza, por lo que el frontend no puede forzar combinaciones inválidas.

Figura 38: Listado de variantes asociado a un producto



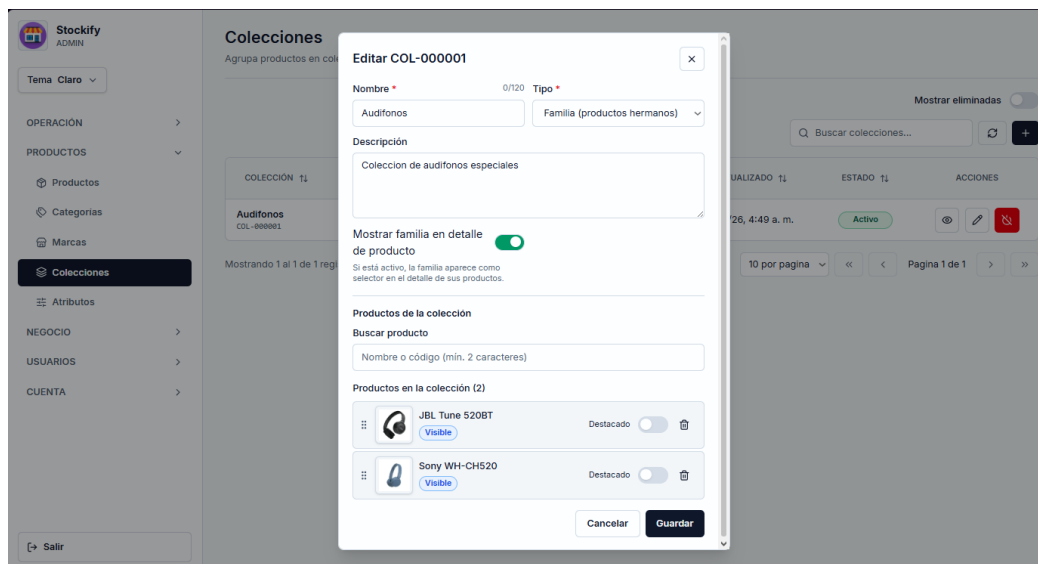
Fuente: Autor.

Colecciones

Las colecciones agrupan productos en un orden definido y se administran en dos tipos. Las colecciones de familia reúnen artículos relacionados, como los de una misma familia o serie, y se muestran como opciones seleccionables en el detalle del catálogo público. Las colecciones de marketing se presentan en la pantalla inicial y exigen al menos cuatro productos 100 % visibles; también pueden configurarse con recurrencia anual para eventos

como Navidad o Viernes Negro, por ejemplo. El backend valida los productos asociados y actualiza transaccionalmente el conjunto ordenado, su visibilidad, portada y programación según el tipo de colección.

Figura 39: Modal y panel administrativo de colecciones



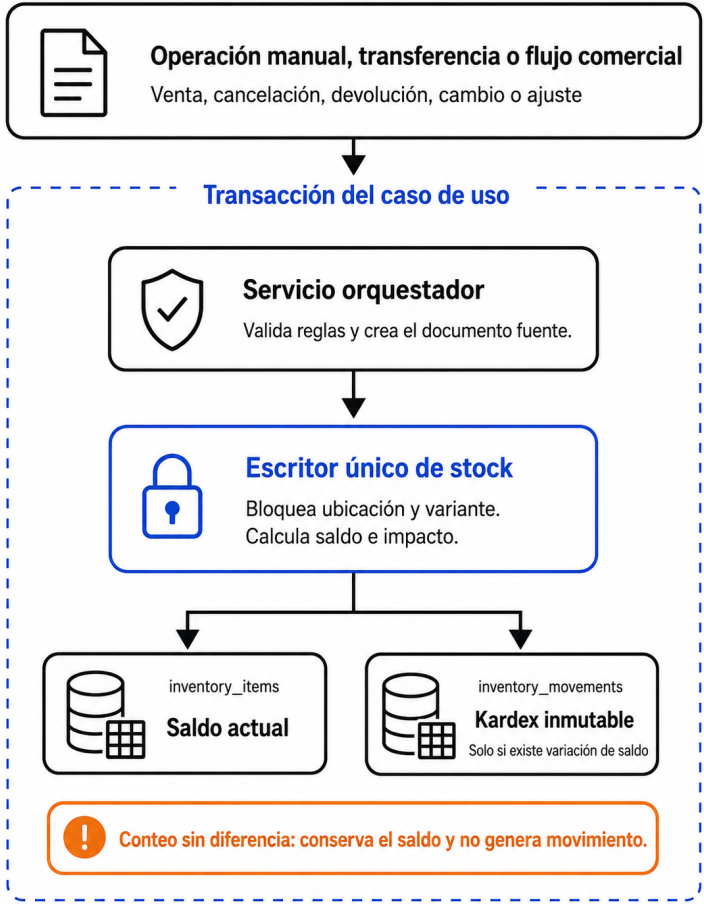
Fuente: Autor.

4.4.5. Inventario, movimientos, operaciones y transferencias

El inventario separa el saldo vigente de su historial: la entidad de ítems conserva una única cantidad por combinación de ubicación y variante, mientras que la entidad de movimientos registra cada modificación con dirección, cantidad, saldos anterior y nuevo, motivo, impacto, documento de origen, actor y fecha. El primer recurso responde cuánto stock existe; el segundo constituye el kardex que explica cómo se alcanzó ese saldo.

Todos los flujos que modifican existencias utilizan el escritor compartido, que obtiene un bloqueo pesimista sobre el ítem existente, calcula el saldo y persiste el saldo y su movimiento dentro de la transacción del servicio orquestador (Figura 40); cuando un conteo coincide con el saldo registrado, no se genera un movimiento. Las restricciones de unicidad y no negatividad complementan estas comprobaciones en la base.

Figura 40: Persistencia coordinada del saldo y su movimiento



Fuente: Autor.

Las operaciones manuales permiten registrar entradas, salidas y ajustes por conteo bajo un mismo documento, con uno o varios ítems. El backend valida la correspondencia entre el tipo de operación y su motivo, deriva el impacto y exige una nota en los casos sensibles o ambiguos. La Tabla 29 presenta estos flujos junto con las transferencias manuales.

Tabla 29: Operaciones manuales y transferencias de inventario

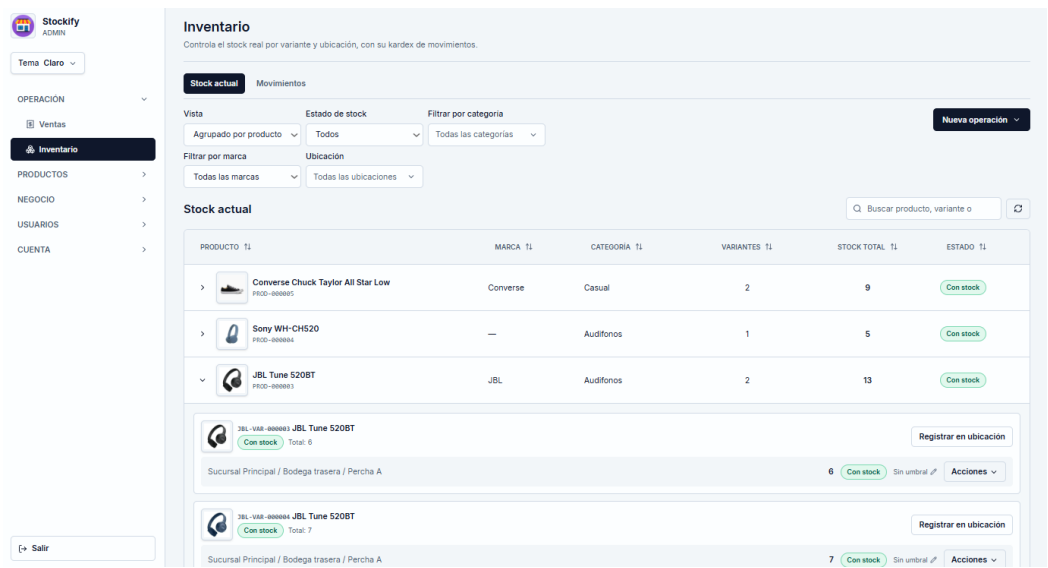
Tipo de operación	Motivos o evento registrado	Origen y efecto registrado
Entrada manual	Compra o reposición de mercadería; reingreso interno; ajuste positivo; migración desde un registro anterior u otro.	Operación creada desde Inventario; incrementa el saldo de las variantes seleccionadas.
Salida manual	Producto dañado; pérdida o extravío; uso interno; muestra o exhibición; retiro manual u otro.	Operación creada desde Inventario; disminuye el saldo disponible.

Conteo físico	Conteo físico; diferencia de inventario; error de registro anterior; corrección por auditoría; regularización u otro.	Operación creada desde Inventario; el sistema calcula la diferencia entre el conteo y el saldo registrado.
Transferencia manual	Reposición de punto de venta; consolidación de stock; movimiento a almacenamiento o bodega; reubicación física del local; reorganización interna u otro.	Documento de transferencia; genera una salida en el origen y una entrada en el destino.

Fuente: Autor.

La pestaña de existencias permite consultar el inventario agrupado por producto o por variante. Además de filtros por categoría, marca, estado y búsqueda, la consulta puede concentrarse en una sucursal, un área (incluidas sus áreas descendientes) o una ubicación específica. Al desplegar un producto se visualizan las variantes y el saldo asociado a cada ubicación física tal como lo presenta la Figura 41.

Figura 41: Consulta de existencias por producto, variante y ubicación



Fuente: Autor.

El registro de una operación manual solicita el motivo, el destino y las variantes involucradas; una misma operación puede aplicar una ubicación común o un destino distinto para cada ítem. La interfaz muestra el impacto estimado, mientras que la validación definitiva de cantidades, estados y relaciones se mantiene en el backend.

Figura 42: Registro de una operación manual de entrada de stock

Nueva operación — Entrada de stock

Motivo *
Compra / reposición de mercadería Impacto estimado: Entrada neutral

Usar diferente destino para cada ítem ☐

Destino *
Bodega principal / General

Detalle del motivo (opcional)
Compra de mercadería para la bodega

Ítems de la operación

Variante *
SON-VAR-000005 - Sony WH-CH520

SON-VAR-000005 - Sony WH-CH520
Nueva ubicación: sin stock previo registrado.

Cantidad *
12

Quitar ítem

Variante *
JER-VAR-000001 - Aretes

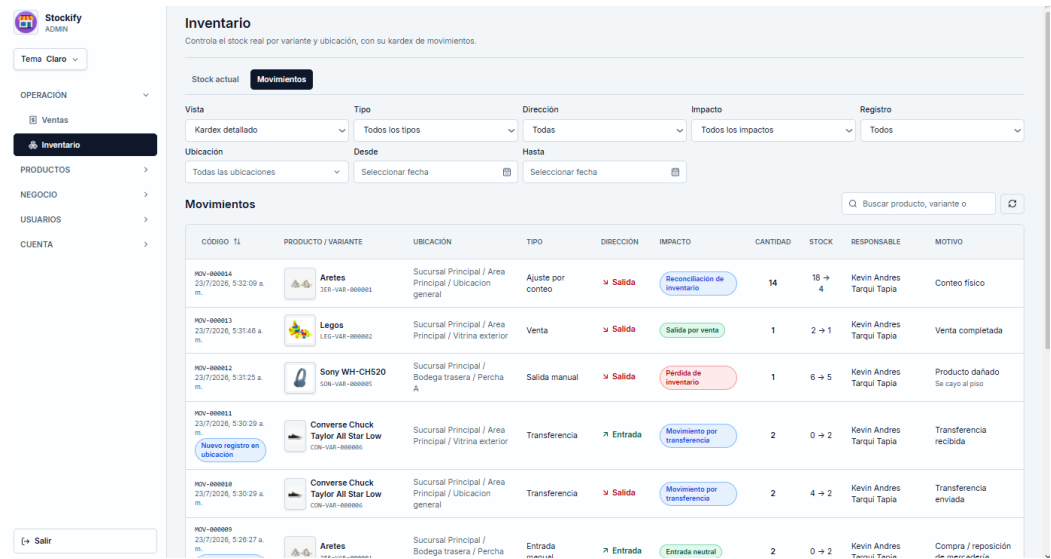
JER-VAR-000001 - Aretes
Nueva ubicación: sin stock previo registrado.

Fuente: Autor.

Las transferencias se registran mediante un documento independiente que exige origen y destino distintos. Por cada variante transferida se aplican una salida en la ubicación de origen y una entrada en la de destino dentro de la misma transacción, por lo que se distinguen de la reasignación administrativa de una ubicación, que no modifica existencias.

El historial de movimientos admite dos vistas; el kardex detallado muestra cada movimiento individual, mientras que la vista agrupada por documento reúne los movimientos originados por una operación de inventario, transferencia, venta o ajuste comercial. Sus filtros permiten delimitar el resultado por sucursal, área, ubicación, tipo, dirección, impacto, creación de ítem y rango de fechas. Tras una mutación, la interfaz actualiza las consultas de existencias, movimientos y documentos relacionados.

Figura 43: Kardex detallado de movimientos de inventario



Fuente: Autor.

4.4.6. Ventas, cambios, devoluciones y ajustes

El módulo reúne el registro de nuevas ventas, el historial de ventas realizadas y el historial de cambios y devoluciones. Constituye un registro comercial interno: la forma de pago se conserva como dato informativo, sin sustituir la facturación legal ni procesar cobros.

En la pestaña de nueva venta, el buscador rápido localiza variantes por SKU, producto, variante o código de barras y muestra las ubicaciones con existencias disponibles. Se seleccionan los artículos, la ubicación desde la que se entregan y las cantidades; además, puede ajustarse el precio negociado de cada artículo respecto de su precio base y aplicarse un descuento sobre el total de la venta. El registro puede asociarse a un cliente existente o crear su ficha con los datos personales requeridos. La selección de sucursal se exige cuando la configuración `sales_require_branch` se encuentra activa.

Figura 44: Registro administrativo de una nueva venta

Stockify
ADMIN

Tema Claro

OPERACIÓN

Ventas

Inventario

PRODUCTOS

NEGOCIO

USUARIOS

CUENTA

Salir

Ventas

Registra ventas, consulta ventas realizadas y gestiona cambios, devoluciones y ajustes.

Nueva venta

Ventas realizadas

Cambios y devoluciones

Sucursal *

Sucursal Principal (principal)

Forma de pago *

Efectivo

Items de la venta

Legos

LEGO-VAR-000002 - Sucursal Principal / Area Principal / Vitrina exterior

Stock disponible: 1

Cantidad

1

Precio (base \$50.00)

\$50.00

Subtotal \$50.00

Cliente

Maricela Tapia

Nuevo cliente

Consumidor final

Buscar cliente por nombre, identificación, teléfono o correo

Agregar productos

Buscar por SKU, producto, variante o código de barras

Aretes

JER-VAR-000001

Precio base \$50.00

Sucursal Principal / Bodega trasera / Percha B - stock 2

Agregar

Sucursal Principal / Area Principal / Ubicacion general - stock 4

Agregar

Bodega principal / General - stock 1

Agregar

Converse Chuck Taylor All Star Low

CON-VAR-000006

Precio base \$65.99

Sucursal Principal / Area Principal / Ubicacion general - stock 2

Agregar

Sucursal Principal / Area Principal / Vitrina exterior - stock 2

Agregar

Descuento de la venta

Notas

Subtotal

\$50.00

Descuento

-\$0.00

Total

\$50.00

Vista previa. El total definitivo lo calcula el servidor al confirmar.

Registrar venta

Limpiar venta

Fuente: Autor.

La interfaz presenta una previsualización de subtotal, descuento y total antes de confirmar. Al registrar la operación, el backend recalcula los precios y totales, conserva el documento comercial y descuenta las existencias correspondientes dentro de una misma transacción. La Tabla 30 resume el tratamiento de la venta y de sus operaciones posteriores.

Tabla 30: Tratamiento de ventas y ajustes posteriores

Operación	Efecto comercial	Relación con inventario
Venta	Registra las variantes, cantidades, precio negociado por artículo, descuento global, cliente y forma de pago.	Descuenta las variantes desde las ubicaciones que atienden la operación.
Cancelación de venta	Marca la venta como cancelada con una razón explícita; solo procede cuando no existen ajustes posteriores completados.	Revierte las salidas asociadas a la venta.
Devolución	Registra los artículos recibidos de la venta original o de un ajuste anterior y el saldo comercial resultante.	Genera entradas en la ubicación de retorno indicada.
Cambio	Vincula los artículos devueltos con las nuevas variantes entregadas y calcula la diferencia monetaria.	Registra entradas por los artículos recibidos y salidas por los entregados dentro de una misma transacción.

Ajuste comercial	Registra una corrección vinculada a la venta, con artículos entregados o con un saldo monetario declarado según el caso.	Solo genera movimientos cuando la corrección incorpora artículos.
Anulación de ajuste	Marca el ajuste como anulado con razón explícita y recompone el estado de la venta original.	Genera movimientos inversos; se bloquea si otro ajuste posterior depende de aquel.

Fuente: Autor.

El historial de ventas realizadas muestra el número del documento, cliente, forma de pago, total, estado y fecha. Cada registro permite abrir un detalle con los artículos e importes de la operación; desde ese detalle se puede iniciar un cambio o devolución y, cuando las reglas comerciales lo permiten, cancelar la venta.

Figura 45: Historial de ventas realizadas

 Stockify

ADMIN

Tema Claro

OPERACIÓN

Ventas

Inventario

PRODUCTOS

NEGOCIO

USUARIOS

CUENTA

Salir

Ventas

Registra ventas, consulta ventas realizadas y gestiona cambios, devoluciones y ajustes.

Nueva venta

Ventas realizadas

Cambios y devoluciones

Ventas realizadas

Q Buscar...

NÚMERO	CLIENTE	PAGO	TOTAL	ESTADO	ÍTEM	FECHA	ACCIONES
VTA-000006	Kevin Tarqui	Efectivo	\$62.99	Ajustada	1	23/7/2026, 6:19:13 a. m.	
VTA-000005	Consumidor final	Efectivo	\$48.00	Ajustada parcialmente	2	23/7/2026, 6:12:23 a. m.	
VTA-000004	Consumidor final	Efectivo	\$65.99	Completada	1	23/7/2026, 6:12:11 a. m.	
VTA-000003	Maricela Tapia	Efectivo	\$50.00	Ajustada	1	23/7/2026, 6:11:54 a. m.	
VTA-000002	Consumidor final	Efectivo	\$50.00	Cancelada	1	23/7/2026, 5:31:46 a. m.	
VTA-000001	Gabriela Patiño	Efectivo	\$95.00	Completada	1	13/7/2026, 10:53:10 p. m.	

Mostrando 1 al 6 de 6 registros

10 por página

Página 1 de 1

Fuente: Autor.

Un cambio registra los artículos devueltos y los que se entregan en sustitución. El sistema compara los importes de ambos grupos y determina si el cliente debe pagar una diferencia, el negocio debe devolverla o no existe saldo cuando sus valores son equivalentes. La devolución registra únicamente los artículos recibidos; ambos casos generan un ajuste vinculado con la venta original y conservan su trazabilidad comercial e inventariable.

Figura 46: Confirmación de un cambio con cálculo del saldo comercial

Cambio / devolución / ajuste

Tipo de ajuste *

Cambio

Items a devolver

Ubicación de retorno *

Sucursal Principal / Área Pri

Legos

LEG-VAR-000002

Items a entregar

Agregar productos

Sony WH-CH520

Sony WH-CH520

SON-VAR-000005

Precio base \$35.00

Sucursal Principal / Bodega tras

Bodega principal / General - sto

Sony WH-CH520

Confirmar ajuste

Cambio

ITEMS A DEVOLVER

Legos

LEG-VAR-000002

1 x \$50.00

ITEMS A ENTREGAR

Sony WH-CH520

SON-VAR-000005

2 x \$35.00

Subtotal devuelto

\$50.00

Subtotal entregado

\$70.00

Saldo

El cliente paga - \$20.00

MOTIVO

Cliente quiere cambio y ajuste de legos por 2 audifonos sony

Cancelar

Confirmar ajuste

Cantidad *

1

Agregar

Agregar

Fuente: Autor.

El historial de cambios y devoluciones reúne los ajustes generados a partir de las ventas y permite consultar su detalle. Desde ese detalle puede anularse un ajuste mediante una razón explícita; la anulación revierte sus movimientos de inventario y actualiza el estado de la venta. La acción se restringe cuando un ajuste posterior depende de artículos entregados por el ajuste que se pretende anular.

Figura 47: Historial de cambios y devoluciones

Stockify

ADMIN

Tema Claro

OPERACION

Ventas

Inventario

PRODUCTOS

NEGOCIO

USUARIOS

CUENTA

Salir

Ventas

Registra ventas, consulta ventas realizadas y gestiona cambios, devoluciones y ajustes.

Nueva venta

Ventas realizadas

Cambios y devoluciones

Cambios y devoluciones

Buscar...

NÚMERO	VENTA	TIPO	CLIENTE	DIFERENCIA	ESTADO	FECHA	ACCIONES
AJU-000003	VTA-000006	Devolucion	Kevin Tarqui	\$62.99 · El negocio devuelve	Aplicado	23/7/2026, 6:19:40 a. m.	
AJU-000002	VTA-000005	Devolucion	Consumidor final	\$35.00 · El negocio devuelve	Aplicado	23/7/2026, 6:18:36 a. m.	
AJU-000001	VTA-000003	Cambio	Maricela Tapia	\$20.00 · El cliente paga	Aplicado	23/7/2026, 6:17:08 a. m.	

Mostrando 1 al 3 de 3 registros

10 por pagina

Página 1 de 1

Fuente: Autor.

4.5. Landing institucional y catálogo público

El catálogo público expone la información comercial publicada sin mostrar costos de adquisición ni cantidades internas. Reúne navegación abierta, consulta de productos y rutas protegidas de cuenta y lista de deseos.

4.5.1. Landing institucional

La landing institucional constituye la página base de Stockify y presenta la aplicación sin asociarla al catálogo de un cliente específico. Su contenido comunica el propósito y las funciones generales de la plataforma. Los dominios generales muestran esta página, mientras que los subdominios de cada negocio cargan su catálogo público.

Figura 48: Landing institucional de Stockify



Fuente: Autor.

4.5.2. Estructura, rutas y navegación

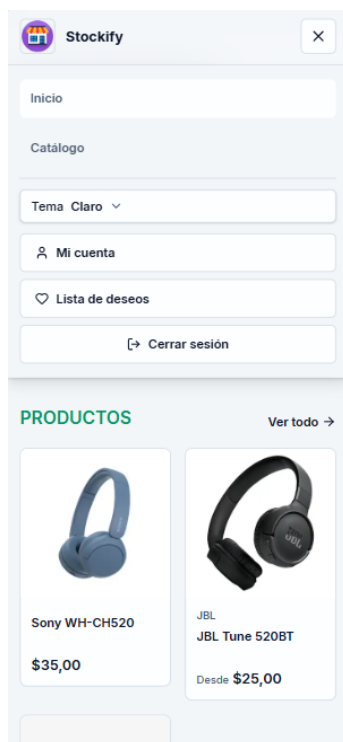
La aplicación identifica el dominio solicitado y presenta la experiencia institucional o el catálogo del negocio. El catálogo integra identidad visual, apariencia, sesión pública, notificaciones y rutas de inicio, catálogo, detalle de producto, colecciones, autenticación, cuenta, seguridad y lista de deseos. Las vistas de mayor tamaño se cargan de manera diferida y las rutas de cuenta y lista de deseos requieren una sesión pública válida.

La barra superior conduce al inicio y al catálogo; en pantallas menores compacta los enlaces y el menú de cuenta. Con sesión activa, este menú incorpora perfil, lista de deseos y cierre de sesión; sin sesión, muestra los accesos de autenticación. El pie de página conserva los destinos principales y los enlaces sociales publicados.

4.5.3. Inicio, catálogo, búsqueda y filtros

Cuando el catálogo está habilitado, la página de inicio carga la configuración pública del negocio, productos, categorías y marcas destacadas, colecciones de marketing e información de contacto; las secciones sin datos se omiten. El buscador dirige al catálogo mediante el parámetro `search`, mientras que las categorías, marcas y colecciones publicadas conducen a sus respectivos recorridos.

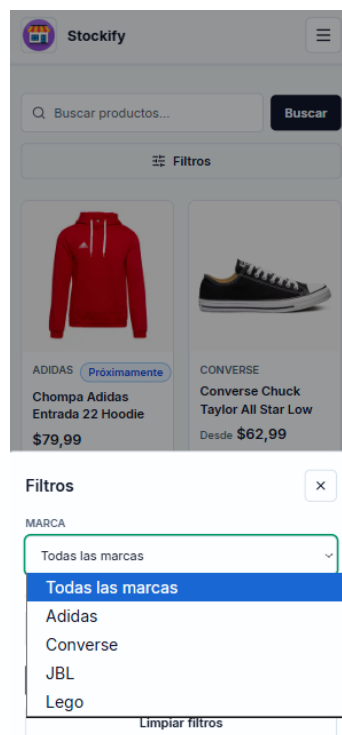
Figura 49: Página de inicio del catálogo público en vista móvil



Fuente: Autor.

El catálogo admite búsqueda, categoría, marca, orden y página mediante parámetros de la URL. La consulta es paginada y vuelve a la primera página cuando cambia la búsqueda o el filtro. En pantallas amplias, los filtros de marca y orden se presentan en una columna lateral; en pantallas menores se muestran en un modal.

Figura 50: Catálogo público en vista móvil con búsqueda y filtros



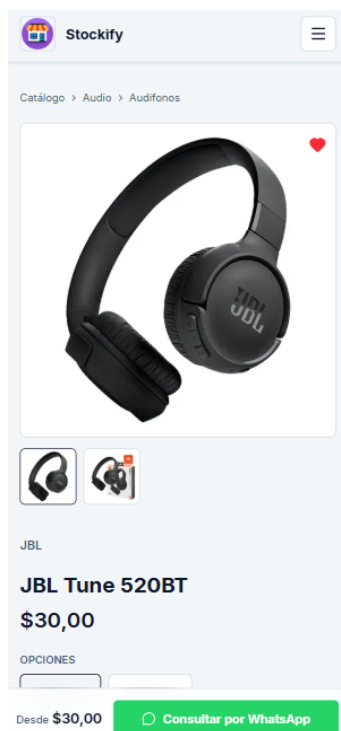
Fuente: Autor.

4.5.4. Ficha, variantes e imágenes

La ficha resuelve el producto y la variante indicada en la dirección, actualiza la URL canónica de la selección y presenta galería, marca, nombre, precio, disponibilidad, descripción, atributos, sucursales públicas, productos relacionados, acción de compartir y consulta por WhatsApp. Cuando el producto pertenece a una colección de familia, el visitante puede desplazarse a artículos relacionados conservando la combinación de opciones más cercana.

Las opciones de una variante pueden mostrarse mediante botones, listas, muestras de color o imágenes, según el atributo configurado. La galería prioriza las imágenes de la variante activa y utiliza las del producto cuando aquella no posee imágenes. La ficha es un recorrido de consulta comercial: no reserva existencias ni crea una venta.

Figura 51: Ficha pública de producto en vista móvil



Fuente: Autor.

4.5.5. Disponibilidad

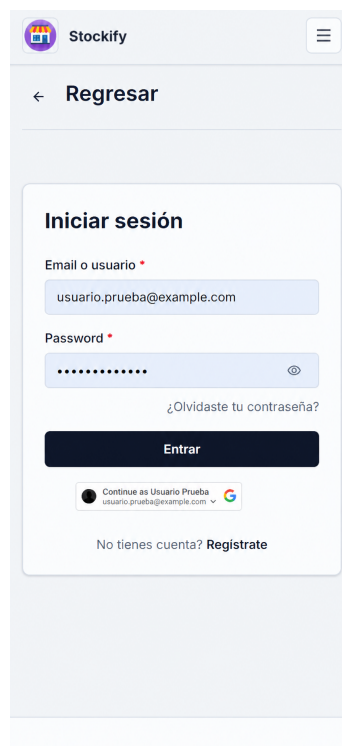
El contrato público define los estados disponible, agotado y próximamente. El backend los determina a partir del inventario público elegible: una cantidad positiva produce AVAILABLE; una cantidad igual a cero con registros elegibles produce OUT_OF_STOCK; y la ausencia de registros elegibles produce COMING_SOON. La elegibilidad exige ubicación y área activas, además de una sucursal activa o un área independiente permitida por configuración.

La interfaz recibe el estado resumido, no cantidades exactas; por ello, disponible no representa una reserva ni una garantía de existencia. Las opciones agotadas se distinguen visualmente, pero pueden mantenerse seleccionables para permitir la comparación entre variantes.

4.5.6. Cuenta pública y lista de deseos

El acceso público permite iniciar sesión mediante credenciales o Google. El registro se realiza mediante un formulario y requiere la verificación posterior del correo electrónico antes de habilitar la cuenta; también incluye recuperación de contraseña. Cuando no existe una sesión válida, las rutas protegidas intentan renovarla y, si no es posible, conducen al inicio de sesión.

Figura 52: Acceso de usuarios al catálogo público en vista móvil

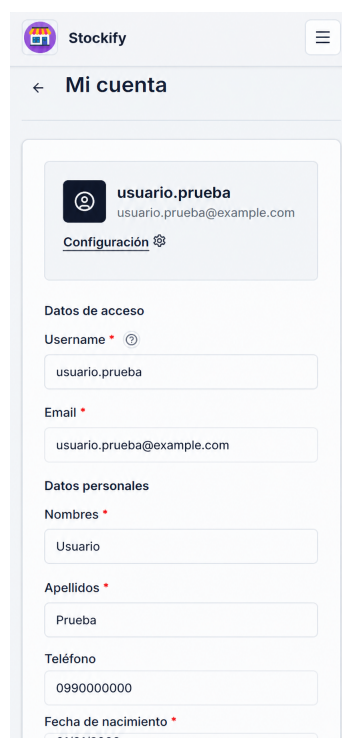


The screenshot shows the 'Iniciar sesión' (Login) screen of the Stockify mobile app. At the top, there is a header with the Stockify logo and a menu icon. Below the header is a back arrow and the text 'Regresar'. The main content area is titled 'Iniciar sesión'. It contains two input fields: 'Email o usuario' with the value 'usuario.prueba@example.com' and 'Password' with masked characters. Below the password field is a link '¿Olvidaste tu contraseña?'. A large black button labeled 'Entrar' is positioned below the inputs. At the bottom, there is a section for 'Continue as Usuario Prueba' with a Google Assistant icon and a 'Registrate' link for users who do not have an account.

Fuente: Autor.

La cuenta pública permite consultar y actualizar el perfil, así como acceder a las opciones de seguridad. Estas vistas se habilitan únicamente después de validar la sesión pública.

Figura 53: Cuenta pública en vista móvil

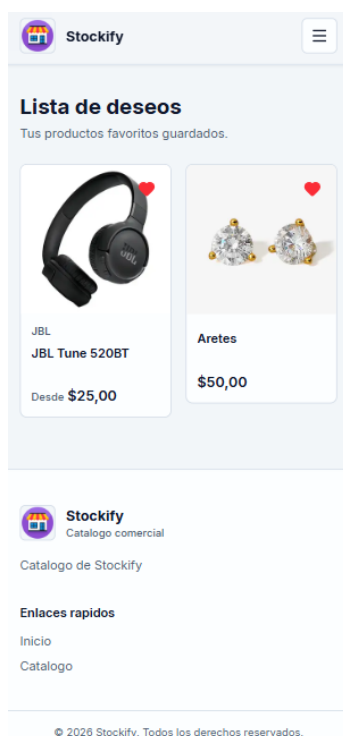


The screenshot shows the 'Mi cuenta' (My account) screen of the Stockify mobile app. At the top, there is a header with the Stockify logo and a menu icon. Below the header is a back arrow and the text 'Mi cuenta'. The main content area displays the user's profile information, including a profile picture placeholder, the username 'usuario.prueba', and the email 'usuario.prueba@example.com'. Below this is a 'Configuración' (Settings) link. The screen is divided into sections: 'Datos de acceso' (Access data) with fields for 'Username' (usuario.prueba) and 'Email' (usuario.prueba@example.com); 'Datos personales' (Personal data) with fields for 'Nombres' (Usuario), 'Apellidos' (Prueba), 'Teléfono' (0990000000), and 'Fecha de nacimiento' (01/01/2000).

Fuente: Autor.

La lista de deseos se organiza por variante. Si el visitante no ha iniciado sesión, la acción de guardar un favorito conduce al acceso; con sesión activa, la lista conserva el enlace hacia la variante seleccionada y presenta los estados de carga, error, vacío y cuadrícula de productos.

Figura 54: Lista de deseos en vista móvil



Fuente: Autor.

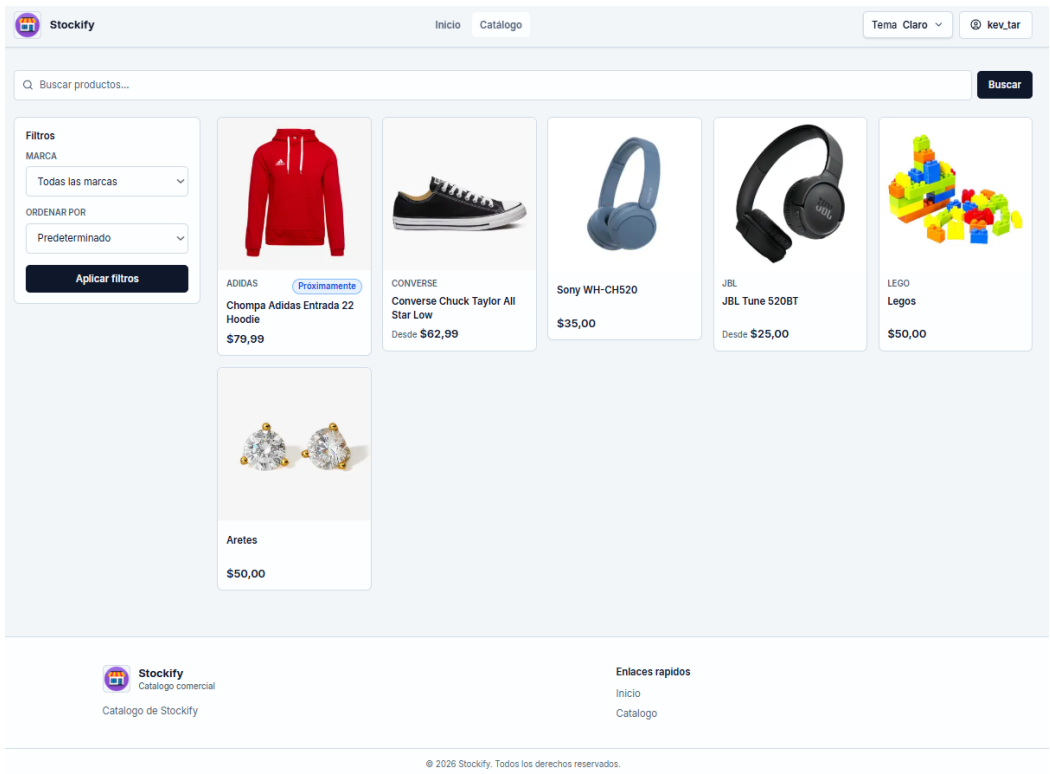
4.5.7. Información comercial y contacto

La configuración pública proporciona el nombre del negocio, la identidad visual, el número de WhatsApp y los enlaces sociales activos. El inicio y el pie de página muestran esta información comercial; en la ficha, la consulta por WhatsApp prioriza los teléfonos públicos de las sucursales asociadas a la variante y, si no existen, utiliza el número general del negocio. La acción de compartir emplea la capacidad nativa del navegador o copia la dirección como alternativa.

4.5.8. Diseño responsivo y personalización

La responsividad adapta la navegación, el catálogo, la ficha y la lista de deseos al ancho de pantalla, como resume la Tabla 31. Las capturas anteriores muestran primero estos recorridos en vista móvil; a continuación se presenta el catálogo en escritorio como contraste. En móvil, el precio y el acceso a WhatsApp permanecen disponibles en una barra inferior de la ficha.

Figura 55: Catálogo público en vista de escritorio



Fuente: Autor.

Tabla 31: Adaptaciones responsivas observadas en el catálogo

Componente	Pantallas menores	Pantallas amplias
Navegación	Botón de menú y enlaces apilados.	Enlaces, tema y cuenta visibles en la barra.
Catálogo	Filtros en modal y cuadrícula de dos columnas.	Filtros laterales fijos y más columnas.
Ficha	Una columna y CTA inferior fijo.	Galería y panel comercial en dos columnas.
Wishlist	Dos columnas iniciales.	Tres a cinco columnas según el ancho.

Fuente: Autor.

La apariencia pública utiliza el perfil activo del negocio para aplicar variantes clara u oscura y sus colores de presentación. El nombre del negocio actualiza el título de la pestaña y el logotipo puede utilizarse como favicon. Estas decisiones permiten adaptar la experiencia a distintas marcas y tamaños de pantalla.

4.6. Seguridad y trazabilidad transversal

La implementación incorpora mecanismos compartidos por los distintos módulos para controlar el acceso y conservar trazabilidad sobre las operaciones relevantes. Las sesiones persistentes resuelven la continuidad de la autenticación, mientras que los registros de auditoría, el kardex y los logs técnicos responden a propósitos diferentes.

4.6.1. Sesiones y control de acceso

La autenticación utiliza *access tokens* JWT y sesiones persistentes asociadas a *refresh tokens*. El *access token* viaja como credencial Bearer e incorpora la identidad, los roles, los permisos efectivos, el contexto de sesión y, cuando corresponde, su identificador. Los contextos PUBLIC y ADMIN se mantienen separados y disponen de cookies de renovación independientes, configuradas con `HttpOnly`, `SameSite=Lax` y el indicador `Secure` según el ambiente. La Tabla 32 resume los elementos involucrados.

Tabla 32: Elementos de autenticación y sesión

Elemento	Transporte o persistencia	Finalidad y control
Access token	Cabecera Bearer enviada a la API.	Identidad, contexto y concesiones efectivas; vigencia configurada y validación por ruta protegida.
Refresh token	Cookie distinta para PUBLIC y ADMIN.	Renovar sin entregar el valor a JavaScript; se rota tras una renovación válida.
Sesión persistente	Registro en <code>auth_sessions</code> .	Usuario, contexto, hash del refresh, expiración, revocación y metadatos de uso.
Credencial almacenada	Solo el hash del refresh permanece en la sesión.	La comparación con el hash evita persistir directamente la credencial de renovación.

Fuente: Autor.

Durante la renovación se selecciona la cookie del contexto, se localiza la sesión y se comprueban el hash, la expiración, la revocación y el estado del usuario. Si estas verificaciones son válidas, se rota el *refresh token* y se emite un nuevo *access token* con los roles y permisos recalculados. Las contraseñas se verifican mediante `Argon2id` y el servicio puede actualizar el hash cuando detecta parámetros anteriores.

En el contexto administrativo, la validación del *access token* también exige una sesión persistente vigente; por ello, la inactivación de un usuario o un cambio de rol o permiso

interrumpe el acceso antes de la expiración del JWT. En el contexto público, la sesión se renueva mediante su cookie correspondiente y el token de acceso emitido conserva su vigencia. Sobre la identidad autenticada, los guards descritos en la Sección 4.3 aplican las restricciones de rol, contexto y permisos; estos permisos se recalculan al iniciar o renovar la sesión y pueden considerar restricciones por origen administrativo autorizado.

4.6.2. Auditoría y trazabilidad

La implementación distingue cuatro mecanismos con propósitos complementarios, resumidos en la Tabla 33: los campos heredados de la clase auditoría describen el estado administrativo actual; los eventos operativos registran cambios sensibles de configuración; los movimientos de inventario constituyen el kardex; y los logs registran el comportamiento técnico de la aplicación.

Tabla 33: Capas de auditoría y trazabilidad

Mecanismo	Pregunta que responde	Contenido y alcance
AuditableEntity	¿Cuál es el estado del registro y quién lo modificó?	Estado, nota, creador, actualizador y fechas.
Eventos operativos	¿Qué configuración sensible cambió y por qué?	Entidad, tipo, valores anterior y nuevo, razón, actor y fecha.
Movimientos de inventario	¿Qué cambio real de stock llevó al saldo nuevo?	Ubicación, variante, dirección, cantidad, saldos, motivo, fuente y actor.
Logs de aplicación	¿Qué ocurrió durante la ejecución técnica?	Evento, severidad, contexto y diagnóstico saneado; no es libro de negocio.

Fuente: Autor.

La tabla `operational_audit_events` se destina a cambios administrativos sensibles, como el cierre de una sucursal, la reasignación de áreas o ubicaciones, el cambio de categoría de un producto y operaciones relevantes. Cuando uno de estos cambios se ejecuta dentro de una transacción, el evento correspondiente se registra en esa misma transacción, de modo que ambos se confirman o se revierten juntos. En cambio, `inventory_movements` conserva el historial de entradas y salidas de existencias, cada registro enlaza la variante, la ubicación, los saldos, el motivo, el documento de origen y el usuario ejecutor. De este modo, una reasignación administrativa no se interpreta como un movimiento de inventario y una transferencia conserva su efecto en el kardex, tal como se ilustra en la Figura 56.

Figura 56: Mecanismos de auditoría y trazabilidad



Fuente: Autor.

4.7. Integración, entrega continua y operación

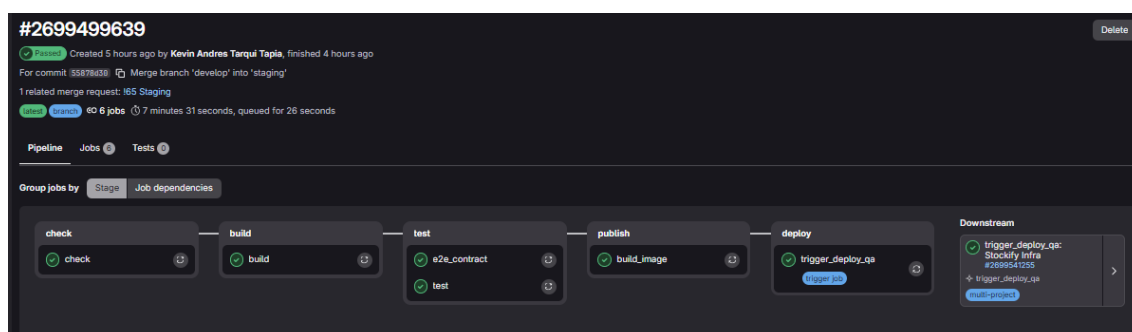
La integración operativa vincula los repositorios de aplicación e infraestructura para construir imágenes, desplegarlas por ambiente, provisionar clientes y resguardar su información.

4.7.1. Canalizaciones y artefactos versionados

Las canalizaciones de backend y frontend ejecutan comprobaciones, construcción, pruebas, publicación de imágenes y activación del *downstream* de infraestructura. En el backend, las pruebas de extremo a extremo inician PostgreSQL aislado, aplican migraciones y ejecutan la suite de la API. En el frontend, *e2e_contract* ejecuta pruebas automatizadas de contratos de interfaz con Playwright y respuestas de API controladas.

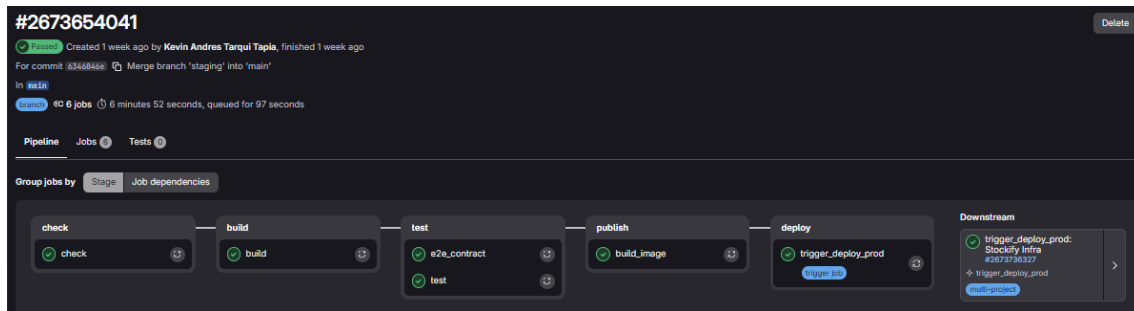
Las integraciones en *staging* publican una imagen candidata y activan la ruta hacia QA mediante *trigger_deploy_qa*; las integraciones en *main* publican la imagen destinada a producción y activan *trigger_deploy_prod*. Las Figuras 57 y 58 muestran ambas trayectorias. Las etapas de control se mantienen porque cambia el ambiente de destino, no el proceso de construcción.

Figura 57: Canalización del frontend hacia QA



Fuente: Autor.

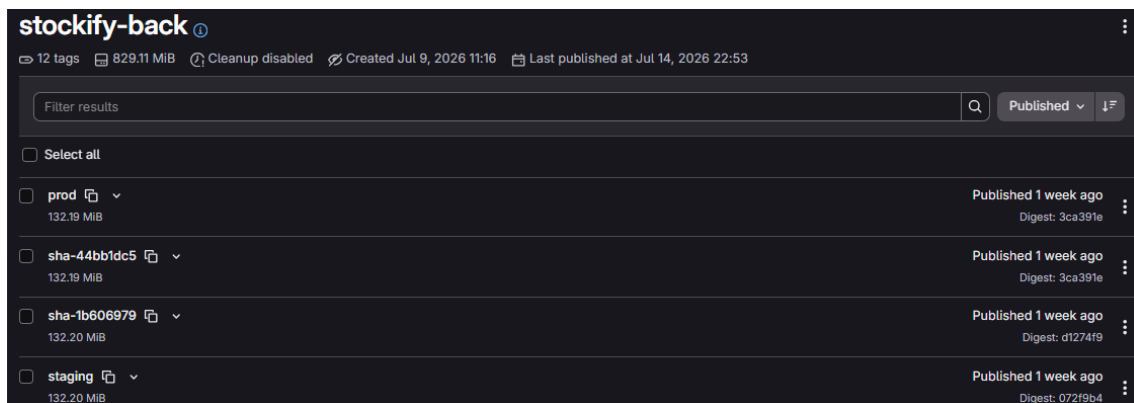
Figura 58: Canalización del frontend hacia producción



Fuente: Autor.

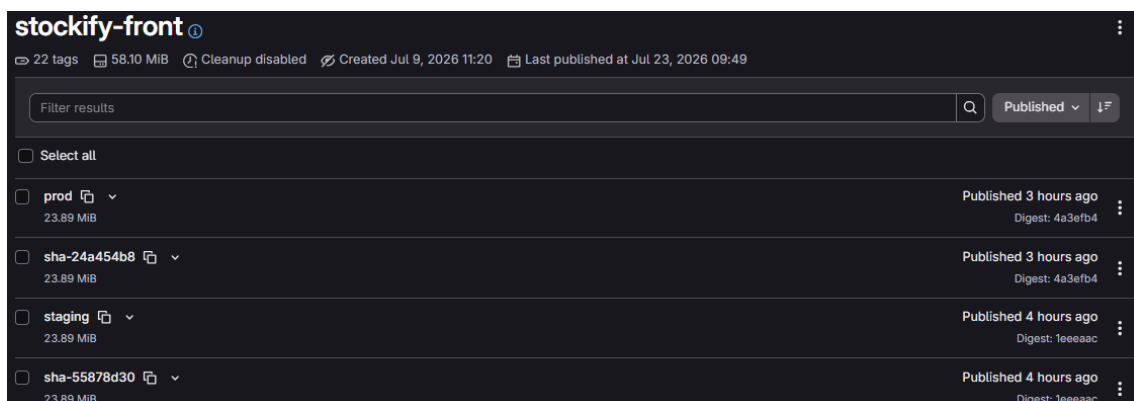
Cada publicación de staging o main genera una etiqueta sha- asociada al commit y conserva un alias móvil de ambiente. El *downstream* recibe la referencia SHA, no el alias, junto con el componente y el ambiente; antes de desplegar, valida que la imagen corresponda al componente declarado. El estado de despliegue conserva la referencia seleccionada y la anterior para permitir la reversión de imagen cuando el proceso falla.

Figura 59: Imágenes versionadas del backend en GitLab Container Registry



Fuente: Autor.

Figura 60: Imágenes versionadas del frontend en GitLab Container Registry



Fuente: Autor.

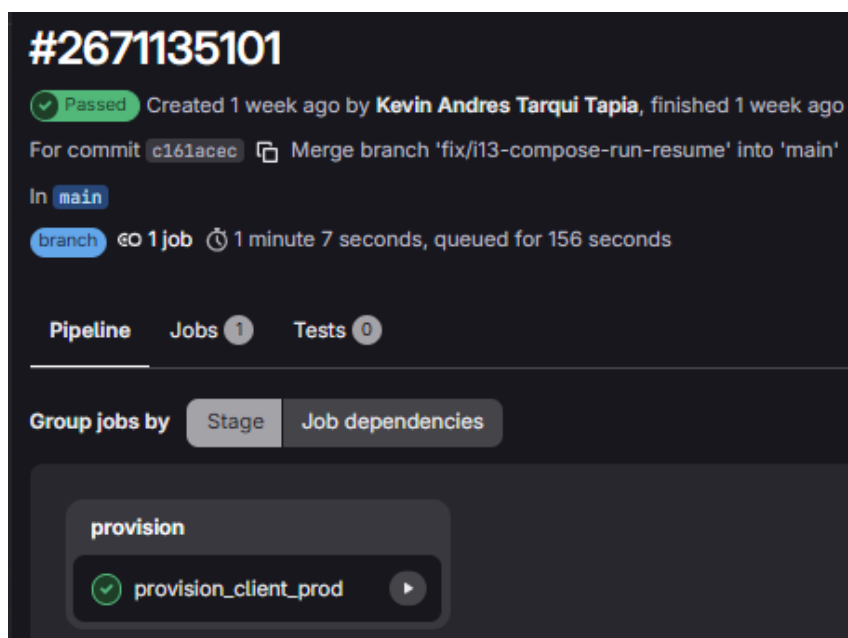
4.7.2. Despliegue y provisionamiento controlados

El *downstream* de QA sincroniza la infraestructura con *staging*, valida el componente y la imagen recibida, obtiene el artefacto desde el Registry e invoca el script común de despliegue. Al actualizar el backend, el proceso ejecuta las migraciones de cada cliente registrado antes de reiniciar sus servicios; al actualizar el frontend, reemplaza la SPA compartida y recrea el proxy cuando corresponde. Al finalizar, comprueba la ruta principal y las rutas de salud de los dominios configurados.

El *downstream* de producción sincroniza exclusivamente la infraestructura con *main*. Su trabajo *deploy_prod* requiere activación manual y se serializa mediante un grupo de recursos. Antes de actualizar un backend con clientes registrados, el script exige un respaldo previo válido; si el cambio falla después de iniciarse, recupera la imagen anterior y conserva el respaldo para recuperación.

El provisionamiento de clientes se ejecuta como un trabajo manual separado del despliegue global. El procedimiento valida el identificador y el dominio, prepara los archivos generados de Compose y Nginx, crea la base de datos y el volumen de almacenamiento, aplica migraciones y datos iniciales, levanta el backend y comprueba las rutas pública, administrativa y de salud antes de incorporar la ruta del cliente al túnel.

Figura 61: Ejecución manual del trabajo de provisionamiento productivo



Fuente: Autor.

Figura 62: Servicios de producción asociados a clientes provisionados

```
stockify@stockify-main:~$ docker ps --format "table {{.Names}}\t{{.Status}}\t{{.Image}}"
```

NAMES	STATUS	IMAGE
stockify-prod-frontend	Up 4 hours (healthy)	registry.gitlab.com/tesis-kevin-tarqui/stockify-front:sha-24a454b8
stockify-clients-proxy	Up 5 hours (healthy)	nginx:1.29-alpine
stockify-clients-frontend	Up 5 hours (healthy)	registry.gitlab.com/tesis-kevin-tarqui/stockify-front:sha-55878d30
stockify-prod-proxy	Up 7 days (healthy)	nginx:1.29-alpine
stockify-prod-back-light-blue	Up 7 days (healthy)	3ca391e35115
stockify-prod-back-veterinaria-manchas	Up 8 days (healthy)	3ca391e35115
stockify-prod-back-elys-store	Up 9 days (healthy)	0da0f8c0b59b
stockify-prod-postgres	Up 9 days (healthy)	postgres:16-alpine
stockify-clients-back-demo	Up 9 days (healthy)	registry.gitlab.com/tesis-kevin-tarqui/stockify-back:sha-1ec75c11
stockify-clients-postgres	Up 9 days (healthy)	postgres:16-alpine

Fuente: Autor.

4.7.3. Exposición mediante Nginx y Cloudflare Tunnel

Cada ambiente mantiene su propio proxy Nginx, publicado únicamente sobre su interfaz local. Cloudflare Tunnel dirige las solicitudes públicas hacia el puerto local del ambiente correspondiente mediante servicios de túnel separados. El servidor Nginx por defecto rechaza hosts no registrados; los bloques generados para cada cliente dirigen la raíz al frontend compartido, /api al backend asociado y /img al mismo servicio que publica sus archivos.

La sincronización de Cloudflare valida la convención de dominio, rechaza registros incompatibles o rutas duplicadas y actualiza de forma coordinada el *ingress* del túnel y el registro DNS. Si la actualización de DNS falla, el procedimiento restaura la configuración previa del *ingress*. La Figura 63 muestra los túneles configurados para los ambientes de QA y producción.

Figura 63: Túneles configurados para los ambientes de QA y producción

Name	Status	Replicas	Routes
stockify-prod	Healthy	1	@ 5 apps
stockify-qa	Healthy	1	@ 2 apps

Fuente: Autor.

4.7.4. Backups, restauración y verificación

El mecanismo de respaldo descubre las bases de datos aplicables, genera un *dump* por cada una, conserva conteos de referencia y restaura temporalmente los *dumps* de la base de datos para comprobar la estructura y los registros críticos. También archiva los volúmenes

de almacenamiento de cada cliente y las rutas de configuración, preserva los permisos de la configuración y genera *checksums* para los artefactos del paquete. La Tabla 34 resume estas comprobaciones.

Tabla 34: Artefactos incluidos y comprobaciones del respaldo

Artefacto	Creación	Comprobación implementada
Bases de datos	<i>Dump</i> individual por base y conteos de referencia.	Restauración temporal y comparación de tablas y registros críticos.
Almacenamiento	Archivo comprimido por volumen de cliente.	Hash del contenido antes y después del archivado.
Configuración	Archivo comprimido con permisos preservados.	<i>Checksum</i> del artefacto generado.
Paquete cifrado	Copia cifrada en la ruta local de salida cuando existe una <i>passphrase</i> .	<i>Checksum</i> del paquete cifrado.

Fuente: Autor.

Antes de una actualización productiva de backend, el script de despliegue exige un respaldo previo marcado como válido y lo mantiene protegido mientras se completa el cambio. La rotación conserva cuatro respaldos semanales en QA y, en producción, catorce diarios, ocho semanales y doce mensuales. La verificación técnica comprueba los *checksums* de los paquetes de respaldo y la restauración temporal de las bases de datos respaldadas.

Figura 64: Verificación de restauración de respaldos mediante bases temporales

```

=== Verificación de integridad del respaldo QA ===
Checksums válidos

=== Restauración temporal y validación de bases ===
14:48:51 Verificación de restore (individual + conjunto completo)...
14:48:51 restore stockify_demo -> rv_1784818131_stockify_demo
NOTICE: database "rv_1784818131_stockify_demo" does not exist, skipping
14:48:59 ok: stockify_demo restaurada y verificada (53 tablas, conteos críticos exactos)
14:48:59 Restore del conjunto completo verificado (todas las bases del paquete).
14:48:59 Verificación standalone OK: /opt/stockify/qa/backups/final-cutover/weekly-20260714T032621Z
stockify@stockify-main:~$ |

```

Fuente: Autor.

Capítulo V

Resultados y análisis

5.1. Introducción

Los resultados técnicos se sustentan en ejecuciones reproducibles, pipelines de GitLab, evaluación de usabilidad e inspección operativa. La evidencia de pruebas corresponde a los commits f8bbe0a6 del backend, ad5dbb7e del frontend y 55a5e10 de infraestructura, con comandos, conteos y resultados trazables. Los resultados se organizan en cumplimiento de requerimientos, pruebas unitarias, E2E de API, pruebas automatizadas de interfaz, verificaciones técnicas de infraestructura, usabilidad y evidencia de operación.

5.2. Cumplimiento de los requerimientos funcionales

El cumplimiento funcional se determinó mediante la trazabilidad entre los requerimientos, la implementación versionada, las pruebas automatizadas y las tareas de usabilidad. La Tabla 35 reúne los requerimientos que contaron con evidencia automatizada directa.

Tabla 35: Requerimientos con evidencia automatizada ejecutada

ID	Evidencia ejecutada	Alcance verificado
RF-ADM-01	Login administrativo válido e inválido y consulta de perfil en E2E backend; redirecciones, login, logout y expiración en pruebas automatizadas de interfaz.	Autenticación de la API y comportamiento visible de la interfaz con respuestas controladas.
RF-ADM-11	Creación de catálogo, incorporación de existencias, registro de venta, disminución y reposición de stock en un caso E2E backend.	Flujo comercial delimitado en API y PostgreSQL aislado.
RF-ADM-12	Cancelación de venta y reposición de existencias en el caso E2E backend.	Cambio de estado comercial y actualización persistente de stock.
RF-PUB-01	Ocho pruebas automatizadas de interfaz para navegación inicio-catálogo, búsqueda, vacío, error/reintento, fallback de imagen, detalle, producto no encontrado y catálogo deshabilitado.	Navegación, búsqueda y estados seleccionados del catálogo con API interceptada.
RF-PUB-03	Consulta de producto y variante en E2E backend; apertura de detalle en pruebas automatizadas de interfaz.	Consulta del detalle y sus variantes mediante API; presentación con datos controlados en navegador.
RF-TR-01	Rutas administrativas muestreadas respondieron 401 sin sesión.	Protección por defecto de endpoints seleccionados.
RF-TR-02	Credenciales inválidas respondieron 401; payloads y consultas inválidos respondieron 400 en E2E backend.	Validación de autenticación y contratos HTTP seleccionados.

Fuente: Autor.

Las ejecuciones anteriores se complementaron con los tests unitarios, la implementación funcional presentada en el Capítulo IV y las tareas realizadas durante la evaluación de usabilidad. Esta combinación permitió comprobar los flujos administrativos, públicos y transversales definidos para el sistema. La Tabla 36 presenta el cierre consolidado.

Tabla 36: Cumplimiento consolidado de los requerimientos funcionales

Ámbito	Cantidad	Evidencia consolidada	Resultado
Administrativo	15	Autenticación; usuarios, roles, sucursal principal y orígenes autorizados; catálogo y colecciones; estructura, umbrales, kardex y transferencias; ventas, cancelaciones, ajustes y obligatoriedad de sucursal.	15 de 15 cumplidos
Público	8	Búsqueda, paginación, filtros, categorías y destacados; detalle y disponibilidad; registro, verificación, acceso, recuperación, perfil, lista de deseos y contacto.	8 de 8 cumplidos
Transversal	4	Separación de sesiones, validación de datos, conservación lógica de registros y auditoría de operaciones.	4 de 4 cumplidos
Total	27	Implementación, pruebas automatizadas y evaluación de usabilidad.	27 de 27 cumplidos

Fuente: Autor.

En consecuencia, los 27 requerimientos funcionales definidos para Stockify se cumplieron dentro del alcance establecido. La Sección 5.4 detalla los recorridos automatizados y su distribución por área. La trazabilidad individual de cada requerimiento, desde su implementación hasta la evidencia que respalda su comprobación, se documenta en el anexo *Inventario técnico y trazabilidad funcional de la solución*.

5.3. Pruebas automatizadas

Los comandos se ejecutaron sobre worktrees limpios. Cada registro conserva comando, duración, código de salida y conteo informado por el runner; las duraciones identifican la ejecución registrada. La configuración vigente de CI utiliza `node:22-alpine` como imagen base para los jobs de comprobación y construcción, mientras que los contratos de interfaz utilizan la imagen de Playwright.

5.3.1. Backend

La Tabla 37 presenta los resultados del backend. El conjunto unitario aprobó 99 suites y 882 pruebas. La base E2E dedicada tenía una migración aplicada y sobre ella aprobaron tres suites con ocho casos.

Tabla 37: Resultados locales de pruebas y controles del backend

Comando	Duración	Resultado	Alcance
<code>npm run check</code>	60,51 s	Código 0	Typecheck, ESLint y formato.
<code>npm run build</code>	18,40 s	Código 0	Compilación NestJS.
<code>npm run test:ci</code>	101,25 s	99 suites; 882 pruebas; 0 fallos	Pruebas unitarias ejecutadas en serie.
<code>npm run migration:show:e2e</code>	6,60 s	1 migración aplicada	Base PostgreSQL E2E dedicada.
<code>npm run test:e2e</code>	11,12 s	3 suites; 8 casos; 0 fallos	API NestJS y PostgreSQL aislado.

Fuente: Autor.

El pipeline asociado al mismo commit creó una instancia PostgreSQL, aplicó la migración y ejecutó la suite E2E.

5.3.2. Frontend

El frontend aprobó check, build, 96 archivos unitarios con 523 pruebas y 15 pruebas automatizadas de interfaz con Playwright, como se resume en la Tabla 38. Las pruebas de Playwright utilizaron Desktop Chrome y respuestas HTTP controladas de la API.

Tabla 38: Resultados locales de pruebas y controles del frontend

Comando	Duración	Resultado	Interpretación
<code>npm run check</code>	103,84 s	Código 0	Typecheck, ESLint y formato.
<code>npm run build</code>	27,73 s	Código 0	Typecheck y build Vite.
<code>npm run test:ci</code>	60,25 s	96 archivos; 523 pruebas; 0 fallos	Pruebas unitarias y de componentes en jsdom.
<code>npm run test:e2e:contract</code>	38,46 s	15/15 aprobados	Pruebas automatizadas de interfaz con Playwright, Desktop Chrome y API interceptada.

Fuente: Autor.

5.3.3. Pipelines CI

Los pipelines de GitLab asociados a los commits de aplicación registraron las ejecuciones correspondientes: el backend 2677153017 (f8bbe0a6) ejecutó check, build, test y test:e2e; el frontend 2677245031 (ad5dbb7e) ejecutó check, build, test y e2e_contract. Sus trazas consignan los mismos conteos: 99 suites y 882 pruebas

unitarias, además de 3 suites y 8 casos E2E en backend; 96 archivos, 523 pruebas unitarias y 15 pruebas automatizadas de interfaz en frontend.

5.3.4. Verificaciones técnicas de infraestructura

La Tabla 39 reúne las verificaciones técnicas ejecutadas localmente sobre el repositorio de infraestructura.

Tabla 39: Resultados de verificaciones técnicas de infraestructura

Verificación	Resultado
<code>stockify-env.test.sh</code>	26 casos aprobados para rutas, selectores y dominios por ambiente.
<code>repository-simplification.test.sh</code>	Código 0 para endpoints, documentación y plantillas de entorno.
<code>provision-client-env.test.sh</code>	Renderizado de QA y producción, incluido <code>-validate-only</code> , con Docker simulado.
<code>ci-secret-transport.test.sh</code>	3 pares de codificación y decodificación aprobados.
<code>validate-prod-client-login.test.mjs</code>	Host y rutas proxy aprobados con servidor simulado.

Fuente: Autor.

5.4. Pruebas funcionales automatizadas

Las pruebas funcionales automatizadas se distribuyen en E2E de API y pruebas de interfaz. Los ocho casos E2E del backend recorren la API NestJS y PostgreSQL aislado; las quince pruebas Playwright recorren la interfaz en navegador con respuestas HTTP controladas de la API.

5.4.1. Área administrativa

La evidencia del área administrativa combina recorridos de interfaz con API controlada y casos E2E ejecutados sobre el backend. La Tabla 40 resume la autenticación, la navegación administrativa y el flujo de catálogo, inventario, venta y cancelación verificados durante las ejecuciones automatizadas.

Tabla 40: Recorridos automatizados ejecutados del área administrativa

Recorrido	Resultado	Ámbito verificado
Login válido, login inválido y perfil actual	3 casos E2E aprobados	API NestJS, sesión y PostgreSQL aislado.
Redirecciones, login, logout y expiración	5 pruebas de interfaz aprobadas	Navegador con respuestas HTTP controladas.
Navegación entre productos y marcas y acceso restringido	2 pruebas de interfaz aprobadas	Router y estados visibles de la interfaz.
Catálogo, stock, venta y cancelación	1 caso E2E aprobado	Disminución y reposición de existencias en PostgreSQL aislado.

Fuente: Autor.

5.4.2. Catálogo público

La evidencia del catálogo público integra pruebas de interfaz con API interceptada y casos E2E ejecutados sobre los endpoints públicos del backend. La Tabla 41 resume la navegación, los estados visibles del catálogo y las respuestas de la API verificadas durante las ejecuciones automatizadas.

Tabla 41: Recorridos automatizados ejecutados del catálogo público

Recorrido	Resultado	Ámbito verificado
Health y endpoints públicos principales	2 casos E2E aprobados	API NestJS y PostgreSQL aislado.
Búsqueda, vacío, error y reintento, y catálogo deshabilitado	4 pruebas de interfaz aprobadas	Navegador con API interceptada.
Fallback de imagen, detalle y producto no encontrado	3 pruebas de interfaz aprobadas	Estados visibles del catálogo con API interceptada.
Navegación inicio-catálogo	Prueba de interfaz aprobada	Router público en navegador.

Fuente: Autor.

5.4.3. Controles transversales

Los controles transversales completan los ocho casos E2E ejecutados sobre la API y PostgreSQL aislado.

Tabla 42: Controles transversales verificados mediante E2E

Recorrido	Resultado	Ámbito verificado
Rutas administrativas protegidas	1 caso E2E aprobado	Respuesta 401 en endpoints administrativos seleccionados sin sesión.
Payloads y consultas inválidos	1 caso E2E aprobado	Respuesta 400 para contratos HTTP seleccionados.

Fuente: Autor.

En conjunto, los ocho casos E2E y las quince pruebas automatizadas de interfaz confirman el cumplimiento de los criterios funcionales automatizados seleccionados.

5.5. Cumplimiento de los requerimientos no funcionales

El cumplimiento de los requerimientos no funcionales se comprobó mediante pruebas automatizadas, revisión de la configuración, evaluación en distintos dispositivos y validación operativa de los ambientes desplegados. La Tabla 43 presenta la evidencia consolidada para los once criterios definidos.

Tabla 43: Cumplimiento de los requerimientos no funcionales

ID	Evidencia	Resultado comprobado	Estado
RNF-01	Guard JWT global y E2E de rutas administrativas protegidas.	Las rutas protegidas exigieron una sesión válida y respondieron 401 sin autenticación.	Cumplido
RNF-02	Guards de contexto, roles y permisos; controladores administrativos y tareas de gestión de usuarios.	Las operaciones administrativas aplicaron el contexto ADMIN y las restricciones declaradas.	Cumplido
RNF-03	Contexto incorporado al token, cookies separadas y validación mediante SessionContextGuard.	Las solicitudes públicas y administrativas se procesaron según el contexto de sesión correspondiente.	Cumplido
RNF-04	DTO públicos, mapeadores y pruebas de contrato del catálogo.	La representación pública presentó disponibilidad sin exponer costos ni cantidades internas.	Cumplido
RNF-05	Validación global y E2E de payloads y consultas inválidos.	Los datos no permitidos o inválidos produjeron respuestas 400 controladas.	Cumplido
RNF-06	Configuración global de Helmet, CORS y limitación de solicitudes.	La capa HTTP incorporó las protecciones y restricciones configuradas.	Cumplido
RNF-07	Servicio Argon2id, actualización de parámetros y pruebas unitarias de hash.	El almacenamiento y la renovación de hashes aprobaron sus comprobaciones.	Cumplido
RNF-08	Evaluación administrativa en computadora y teléfono, además de doce sesiones móviles del catálogo público.	Los flujos responsivos evaluados se completaron en escritorio y dispositivos móviles.	Cumplido
RNF-09	Sincronización automática desactivada, migraciones versionadas y PostgreSQL preparado por el pipeline.	El esquema se actualizó mediante migraciones antes de ejecutar las pruebas E2E.	Cumplido

RNF-10	Proyectos Compose separados y provisionamiento multiinstancia verificado.	Cada cliente dispuso de backend, base lógica, almacenamiento y dominio propios, con recursos compartidos documentados.	Cumplido
RNF-11	Copias de las bases, almacenamiento y configuración incluidos en el respaldo; checksums y restauración temporal.	Se verificaron la integridad de los artefactos y la restauración temporal de las bases incluidas.	Cumplido
Total	Evidencia técnica, de usabilidad y operación.	Cumplimiento consolidado de los once criterios definidos.	11 de 11

Fuente: Autor.

5.5.1. Seguridad y contratos HTTP

Los E2E backend aprobaron login válido e inválido, perfil autenticado, rutas protegidas por defecto y validación HTTP de payloads y consultas. Las credenciales inválidas respondieron 401, mientras que los payloads y consultas inválidos respondieron 400. Las pruebas unitarias y la configuración global complementaron estos resultados mediante verificaciones de guards, contextos de sesión, DTO, cabeceras de protección y hash de contraseñas.

5.5.2. Responsividad, persistencia y operación

La evaluación de usabilidad comprobó los flujos responsivos desde computadoras y teléfonos móviles. El pipeline backend creó PostgreSQL, aplicó las migraciones y ejecutó la suite E2E; a su vez, la validación operativa comprobó la separación por cliente y la restauración temporal de los respaldos. Los controles de tipado, estilo, formato, construcción y pruebas automatizadas se presentan en la Sección 5.3, mientras que la evidencia de despliegue y respaldos se desarrolla en la Sección 5.7.

5.6. Usabilidad

La evaluación de usabilidad reunió 15 participantes y se aplicó mediante los dos instrumentos descritos en la Sección 3.6: uno para el área administrativa y otro para el catálogo público. Las plantillas utilizadas se presentan en blanco en el anexo del instrumento administrativo y en el anexo del instrumento del catálogo público. El análisis mantiene separadas ambas interfaces porque sus tareas, perfiles y condiciones de uso son diferentes.

5.6.1. Participantes y condiciones

La ruta administrativa contó con tres propietarias, una por negocio, mientras que la ruta pública reunió 12 participantes seleccionados según su disponibilidad y aceptación voluntaria para realizar las pruebas, distribuidos entre los tres catálogos. Estos últimos no

necesariamente eran clientes de los negocios. Todas las fichas registraron la aceptación para participar; los resultados se analizaron mediante códigos y los nombres no se incorporaron al informe. La Tabla 44 presenta la distribución del grupo evaluado.

Tabla 44: Distribución de participantes por negocio

Negocio	Área administrativa	Catálogo público	Total
Ely's Store	1	5	6
Light Blue	1	3	4
Veterinaria Manchas	1	4	5
Total	3	12	15

Fuente: Autor.

En la ruta administrativa se utilizaron un teléfono Android, una computadora de escritorio y una computadora portátil, con Chrome, Edge y Firefox, respectivamente. Los tres perfiles presentaron distintos niveles de experiencia previa con sistemas de inventario o ventas: uno no los había utilizado, uno los utilizaba ocasionalmente y uno con frecuencia. Dos participantes gestionaban productos, existencias o ventas una o varias veces por semana y uno lo hacía diariamente.

La ruta pública se ejecutó desde teléfonos móviles: nueve dispositivos Android y tres iPhone. Se utilizaron Chrome móvil en cinco sesiones, Safari en tres, Firefox móvil en dos y Edge móvil en dos. Siete participantes consultaban catálogos o tiendas en línea con frecuencia y cinco de manera ocasional; nueve navegaban diariamente desde el teléfono y tres varias veces por semana. Las tareas se realizaron con datos de prueba asociados al contexto de cada negocio.

Para el análisis, la categoría *completada* del formulario se interpretó como finalización sin ayuda, debido a que el mismo instrumento registró por separado la categoría *completada con ayuda*. No se registraron tareas no completadas ni no aplicables.

5.6.2. Resultados administrativos

Cada participante realizó diez tareas: ingreso al área administrativa, gestión de usuarios, configuración del negocio, preparación del catálogo, registro de productos, actualización y consulta de inventario, registro de ventas y una actividad final adaptada al negocio. Las 30 ejecuciones se completaron; 27 se realizaron sin ayuda y tres requirieron una indicación breve. Esto representa una tasa de finalización del 100 %, con 90 % sin ayuda y 10 % con ayuda.

La Tabla 45 resume los resultados por tarea. Los tiempos se presentan mediante mediana y rango porque cada tarea fue ejecutada por tres participantes.

Tabla 45: Resultados de las tareas administrativas

Tarea	Actividad principal	Sin ayuda	Con ayuda	Mediana	Rango
T01	Ingreso y confirmación del negocio	3	0	1 min 18 s	1 min 02 s–1 min 26 s
T02	Creación de usuario y asignación de permisos	2	1	2 min 46 s	2 min 21 s–3 min 06 s
T03	Información y configuración del negocio	3	0	1 min 52 s	1 min 39 s–2 min 03 s
T04	Categoría, marca y atributos	3	0	2 min 58 s	2 min 47 s–3 min 19 s
T05	Producto, variantes, precio e imagen	3	0	4 min 12 s	3 min 54 s–4 min 31 s
T06	Registro y consulta de existencias	2	1	1 min 54 s	1 min 36 s–2 min 38 s
T07	Localización del movimiento de inventario	3	0	1 min 31 s	1 min 24 s–2 min 04 s
T08	Registro de la venta de prueba	3	0	2 min 49 s	2 min 33 s–3 min 07 s
T09	Consulta de venta y efecto en existencias	3	0	1 min 46 s	1 min 28 s–1 min 57 s
T10	Actividad final según el contexto del negocio	2	1	3 min 26 s	3 min 08 s–3 min 42 s
Total		27	3		

Fuente: Autor.

Las ayudas se concentraron en tres acciones diferentes: reconocer el alcance de un permiso al crear un usuario, confirmar la ubicación activa antes de registrar existencias y localizar la vista previa de apariencia. Una vez recibida la indicación, las actividades continuaron hasta alcanzar el resultado esperado. La duración total por sesión presentó una mediana de 24 min 23 s, con un rango de 22 min 37 s a 27 min 17 s.

5.6.3. Resultados del catálogo público

Los 12 participantes realizaron nueve tareas desde la vista móvil: identificación del negocio, búsqueda, filtrado, consulta de la ficha, selección de variante, interpretación de disponibilidad, uso de la lista de deseos, localización del contacto y navegación de regreso al producto. Las 108 ejecuciones se completaron; 104 se realizaron sin ayuda y cuatro con ayuda. La tasa de finalización fue del 100 %, con 96,3 % sin ayuda y 3,7 % con ayuda.

Tabla 46: Resultados de las tareas del catálogo público

Tarea	Actividad principal	Sin ayuda	Con ayuda	Mediana	Rango
T01	Identificación del negocio	12	0	21,5 s	17–30 s
T02	Búsqueda del producto	12	0	32,5 s	27–46 s
T03	Aplicación de filtros	10	2	42 s	34–62 s
T04	Consulta de la ficha del producto	12	0	30,5 s	26–39 s
T05	Selección de variante	11	1	27,5 s	23–45 s
T06	Interpretación de disponibilidad	12	0	31 s	27–39 s
T07	Uso de la lista de deseos	11	1	43,5 s	35–54 s
T08	Localización del medio de contacto	12	0	29 s	25–42 s
T09	Navegación móvil de regreso al producto	12	0	42,5 s	33–58 s
Total		104	4		

Fuente: Autor.

Las cuatro ayudas correspondieron a la aplicación de filtros en dos sesiones, la selección de una variante en una sesión y el acceso a la lista de deseos desde la navegación móvil en otra. Las tareas de filtrado, lista de deseos y navegación final presentaron las mayores medianas de tiempo, mientras que la identificación del negocio fue la actividad más breve.

5.6.4. SUS y valoraciones complementarias

Los 15 cuestionarios incluyeron respuestas para los diez ítems de la *System Usability Scale* (SUS). Las puntuaciones se obtuvieron mediante el procedimiento estándar de SUS descrito en la Sección 3.6. Estas se expresan en una escala de 0 a 100, pero no representan un porcentaje de tareas correctas. Los resultados administrativos y públicos se analizaron por separado.

Tabla 47: Resultados de la escala SUS

Grupo evaluado	n	Mediana SUS	Rango
Área administrativa	3	67,5	65–70
Catálogo público, conjunto total	12	67,5	60–70
Catálogo público de Ely's Store	5	67,5	62,5–70
Catálogo público de Light Blue	3	70	60–70
Catálogo público de Veterinaria Manchas	4	65	60–70

Fuente: Autor.

Las tres valoraciones complementarias de cada instrumento se analizaron por separado de SUS. En el área administrativa, la claridad de la información y la satisfacción obtuvieron una mediana de 4, mientras que la facilidad para completar las tareas obtuvo 3. En el catálogo público, la claridad de la información presentó una mediana de 4; la facilidad de las tareas y la navegación desde el dispositivo utilizado obtuvieron 3. En todos los casos, las respuestas se ubicaron entre 3 y 5.

5.6.5. Dificultades y mejoras

Las observaciones administrativas muestran que el menú, la organización por secciones, las imágenes de producto y los filtros de inventario facilitaron el recorrido. Las dificultades se relacionaron con el significado de los permisos, la visibilidad de la ubicación activa y la localización de la vista previa de apariencia. Las mejoras sugeridas se orientaron a presentar un resumen de permisos más directo, destacar la ubicación seleccionada y hacer más visible la vista previa durante la personalización.

En el catálogo público, la búsqueda, los filtros, las imágenes y la organización de la ficha ayudaron a reconocer los productos y sus variantes. Las indicaciones registradas se concentraron en la aplicación de filtros, el selector de variantes y el acceso móvil a la lista de deseos. Los comentarios también propusieron acercar la disponibilidad al selector correspondiente, facilitar el reinicio de filtros y dar mayor visibilidad a la lista de deseos en la navegación móvil.

5.6.6. Síntesis

El grupo evaluado completó todas las tareas planteadas en ambas interfaces. En el área administrativa, nueve de cada diez ejecuciones se realizaron sin ayuda; en el catálogo público, la proporción fue de 96,3 %. Ambos grupos obtuvieron una mediana SUS de 67,5, mientras que las valoraciones complementarias se mantuvieron entre posiciones

neutrales y favorables.

Los resultados aportaron evidencia de uso para los requerimientos RF-ADM-01, RF-ADM-02, RF-ADM-05, RF-ADM-06, RF-ADM-09, RF-ADM-10, RF-ADM-11 y RF-ADM-14 del área administrativa, así como para RF-PUB-01, RF-PUB-03, RF-PUB-07 y RF-PUB-08 del catálogo público. La ejecución desde computadoras y dispositivos móviles también respaldó el requerimiento RNF-08. El cumplimiento consolidado se presenta en las Secciones 5.2 y 5.5.

Los resultados respaldan la utilización de los flujos evaluados y, al mismo tiempo, identifican oportunidades concretas para mejorar permisos, selección de ubicaciones, filtros, variantes y navegación móvil. Debido al tamaño del grupo y a que su participación dependió de la disponibilidad y aceptación voluntaria, estas conclusiones describen únicamente las sesiones realizadas y no se generalizan a otras poblaciones.

5.7. Resultados del despliegue y la validación operativa

La validación operativa reunió evidencia del flujo de entrega, el provisionamiento multi-instancia, el estado de los servicios, el enrutamiento y los respaldos. Los identificadores de pipeline, referencias de imagen, resúmenes SHA-256, manifiestos y datos sanitizados del VPS se conservan en el anexo de evidencia operativa.

5.7.1. Promoción y provisionamiento

Las promociones desde `main` activaron satisfactoriamente los procesos de infraestructura destinados a producción, mientras que la promoción del frontend desde `staging` activó el despliegue automático de QA. El pipeline de infraestructura 2677251952 completó el trabajo manual de provisionamiento productivo y dejó operativas dos instancias de negocio en producción.

5.7.2. Servicios y rutas

El VPS Contabo con Ubuntu 24.04.4 LTS ejecutó nueve contenedores en estado `healthy` y con el contador de reinicios en cero: `proxy`, `frontend`, `PostgreSQL` y un backend de referencia en QA; `proxy`, `frontend`, `PostgreSQL` y dos backends de negocio en producción. Las rutas locales de inicio y salud respondieron 200 en los tres contextos, al igual que las rutas públicas de producción. El acceso público de QA se mantuvo restringido con respuesta 403, mientras que su ruta local respondió 200. El servicio `cloudflared` permaneció cargado, habilitado y activo.

5.7.3. Separación de recursos y respaldos

QA y producción utilizaron proyectos Compose, redes, instancias PostgreSQL, puertos y secretos diferenciados. Dentro de cada ambiente, los clientes compartieron frontend, proxy, motor PostgreSQL y red, mientras que cada uno dispuso de backend, base lógica, volumen de almacenamiento y dominio propios.

La verificación de respaldos utilizó un paquete semanal de QA y uno diario de producción generados por la versión 2.1.1 del script. Ambos registraron ejecución satisfactoria, checksums válidos y el indicador `restore_verified=true`. La comprobación restauró temporalmente las bases incluidas y validó sus tablas y conteos críticos. La Tabla 48 sintetiza los resultados.

Tabla 48: Síntesis de los resultados de la validación operativa

Dimensión	Evidencia comprobada	Resultado
Entrega	Promociones hacia QA y producción con procesos de infraestructura finalizados satisfactoriamente.	Exitosa
Provisionamiento	Pipeline 2677251952, una instancia QA y dos instancias productivas.	Operativo
Servicios	Nueve contenedores healthy y contador de reinicios en cero.	Saludables
Rutas	Respuesta local 200 en tres contextos, producción pública 200 y acceso público de QA restringido con 403.	Enrutamiento verificado
Separación	Ambientes diferenciados y backend, base lógica, volumen y dominio propios por cliente.	RNF-10 cumplido
Respaldos	Checksums válidos y restauración temporal satisfactoria de las bases incluidas.	RNF-11 cumplido

Fuente: Autor.

La convergencia de los resultados de entrega, operación, separación y respaldo confirmó el funcionamiento de los ambientes desplegados y el cumplimiento de los requisitos no funcionales asociados.

5.8. Cumplimiento de los objetivos

La comprobación de los requerimientos, la implementación, las pruebas automatizadas, la evaluación de usabilidad y la validación operativa permitieron determinar el cumplimiento de los objetivos del proyecto. La Tabla 49 presenta la evidencia consolidada.

Tabla 49: Cumplimiento de los objetivos del proyecto

Objetivo	Evidencia obtenida	Resultado
General	Sistema web responsivo implementado para la gestión administrativa del inventario y la consulta pública de información comercial; 27 requerimientos funcionales y 11 no funcionales comprobados.	Cumplido dentro del alcance
Específico 1	Actores, requerimientos funcionales y no funcionales, historias de usuario, criterios de aceptación y trazabilidad definidos y comprobados.	Cumplido
Específico 2	Modelo relacional SQL, entidades, relaciones, restricciones, migraciones y datos iniciales implementados y versionados.	Cumplido
Específico 3	Arquitectura cliente–servidor, organización modular, comunicación entre capas y despliegue multiinstancia implementados.	Cumplido
Específico 4	Backend, área administrativa, catálogo público responsivo, inventario, ventas y mecanismos operativos implementados.	Cumplido
Específico 5	882 pruebas unitarias backend, 523 frontend, ocho E2E de API, quince pruebas automatizadas de interfaz, validación operativa y evaluación de usabilidad con 15 participantes.	Cumplido

Fuente: Autor.

En consecuencia, el objetivo general y los cinco objetivos específicos se cumplieron dentro del alcance definido para Stockify, con base en la evidencia de implementación, validación técnica, usabilidad y operación presentada en este capítulo.

5.9. Discusión

Los resultados muestran correspondencia entre los requerimientos definidos, la implementación y los mecanismos utilizados para comprobarla. El cierre de los 27 requerimientos funcionales y los 11 no funcionales se sustentó en evidencia complementaria, de modo que el cumplimiento comprendió tanto el comportamiento del software como su uso y operación.

Las 882 pruebas unitarias backend y 523 frontend comprobaron reglas y componentes aislados. Los ocho casos E2E ampliaron esta evidencia mediante recorridos sobre la API NestJS y PostgreSQL, mientras que las quince pruebas Playwright verificaron navegación, interacción y estados visibles con respuestas controladas de la API. Esta separación permite

atribuir a cada conjunto el alcance que realmente ejecutó y evita confundir las pruebas de interfaz con recorridos completos entre frontend y backend. La coincidencia entre los resultados locales y las canalizaciones de CI fortaleció su trazabilidad.

La validación operativa complementó las pruebas de aplicación al comprobar promoción, provisionamiento, rutas, separación de ambientes y restauración temporal de respaldos. La presencia de un backend, una base lógica, un volumen y un dominio por cliente confirmó que el modelo multiinstancia descrito en la arquitectura se reflejó en los ambientes desplegados.

La evaluación de usabilidad incorporó la perspectiva de tres propietarias y doce participantes del catálogo público. Las 30 ejecuciones administrativas y las 108 del catálogo público alcanzaron el resultado esperado; el 90 % de las primeras y el 96,3 % de las segundas se completaron sin ayuda. La mediana SUS de 67,5 en ambos grupos muestra una experiencia funcional y, al mismo tiempo, orienta mejoras relacionadas con permisos, ubicación activa, filtros, variantes y navegación móvil.

En conjunto, los resultados respaldan el cumplimiento de los objetivos porque relacionan las capacidades implementadas con pruebas reproducibles, uso por participantes y operación en ambientes desplegados. Esta combinación aporta una valoración más completa que la revisión aislada del código o de una única clase de prueba.

5.10. Limitaciones de la evaluación

Los resultados se interpretan dentro del alcance de los instrumentos aplicados. La validación automatizada se concentró en criterios funcionales representativos: los casos E2E recorrieron la API y PostgreSQL, mientras que las pruebas Playwright comprobaron navegación, interacción y estados visibles mediante respuestas controladas de la API. Cada conjunto sustenta conclusiones acordes con su nivel de ejecución y conserva sus comandos, commits y resultados en el anexo Evidencia operativa y de pruebas de la versión evaluada.

La validación operativa abarcó el estado de los servicios, las rutas y la separación de recursos por ambiente y cliente. También comprobó los *checksums* de los paquetes de respaldo y la restauración temporal de las bases de datos incluidas. Sus referencias sanitizadas se presentan en ese mismo anexo. Las métricas de disponibilidad sostenida, capacidad y recuperación integral corresponden a evaluaciones longitudinales especializadas.

La evaluación de usabilidad correspondió a un grupo de 15 personas: tres propietarias y doce participantes del catálogo público. Sus conclusiones se limitan a los perfiles, dispositivos y tareas evaluados y no se generalizan a otras poblaciones. El estudio no

incorporó una línea base ni una comparación antes y después de la implementación; por tanto, la contribución al control administrativo se valoró mediante la finalización de tareas, la trazabilidad operativa y la percepción de uso, sin cuantificar causalmente el grado de mejora producido.

Conclusiones y recomendaciones

Conclusiones

El desarrollo de Stockify permitió cumplir el objetivo general al integrar, en una aplicación web responsiva, la gestión administrativa de productos, variantes, existencias y ventas con un catálogo público alimentado por la misma información operativa. La solución se orientó a las necesidades identificadas en Ely's Store, Light Blue y Veterinaria Manchas, tres pequeñas empresas comerciales de Cuenca que utilizaban registros manuales y no disponían de catálogo web propio. La evidencia de implementación, pruebas, usabilidad y operación respaldó el cumplimiento de los 27 requerimientos funcionales y los 11 no funcionales definidos dentro del alcance del proyecto.

1. El análisis de requerimientos combinó entrevistas abiertas con las tres propietarias, observación directa y revisión de registros manuales en dos negocios, literatura, comparación de plataformas y criterios de Ingeniería de Software. Esta combinación permitió convertir las necesidades operativas y las decisiones técnicas en requerimientos, historias de usuario y criterios verificables. Los 15 requerimientos administrativos, ocho públicos y cuatro transversales fueron cumplidos mediante funciones correspondientes al alcance establecido para inventario, ventas básicas y catálogo público.
2. El modelo relacional organizó 50 entidades distribuidas entre identidad y acceso, catálogo, estructura operativa, inventario, ventas, medios y funciones públicas. La separación entre producto y variante, la unicidad del saldo por ubicación y variante, el kardex y la relación entre ventas, ajustes y movimientos permitieron mantener consistencia y trazabilidad. Las restricciones, transacciones, migraciones y datos iniciales complementaron este diseño al controlar la evolución del esquema y las operaciones que modifican existencias.
3. La arquitectura cliente–servidor, compuesta por una SPA, una API REST organizada como monolito modular y PostgreSQL, centralizó las reglas del negocio y el acceso a persistencia. Su despliegue multiinstancia permitió configurar distintos negocios dentro de cada ambiente, con backend, base lógica, almacenamiento y dominio propios por cliente, además de componentes compartidos por ambiente. Esta organización separó las interfaces, la lógica y los datos, y permitió entregar una configuración diferenciada para cada uno de los negocios evaluados.
4. El área administrativa integró la gestión de usuarios, estructura operativa, imágenes, catálogo, inventario, transferencias, ventas y ajustes. A partir de estos datos, el catálogo público presentó productos, variantes, colecciones, disponibilidad, información

comercial y funciones de cuenta desde interfaces adaptadas a dispositivos móviles y de escritorio. De esta manera, la información pública se derivó de la operación interna sin duplicar existencias ni exponer costos o cantidades reservadas para la administración.

5. La validación aportó evidencia complementaria sobre el funcionamiento del sistema. En el backend se aprobaron 882 pruebas unitarias y ocho casos E2E contra la API y PostgreSQL; en el frontend se aprobaron 523 pruebas unitarias y de componentes, además de 15 pruebas automatizadas de interfaz con respuestas controladas de la API. Las tres propietarias y los doce participantes del catálogo público completaron las tareas planteadas; el 90 % de las ejecuciones administrativas y el 96,3 % de las públicas se realizaron sin ayuda, y ambos grupos obtuvieron una mediana SUS de 67,5. La validación operativa también comprobó los ambientes desplegados, la separación lógica por cliente, los *checksums* de los paquetes de respaldo y la restauración temporal de las bases de datos incluidas.

En conjunto, los resultados respaldan el cumplimiento del objetivo general y de los cinco objetivos específicos dentro del contexto evaluado. Stockify constituye una solución aplicada que centraliza información, conserva trazabilidad y facilita la consulta pública. La evaluación se concentró en comprobar el funcionamiento, la usabilidad y la operación del sistema en los negocios participantes; la medición de efectos a largo plazo sobre la productividad, las ventas o la precisión del inventario quedó fuera del alcance del estudio. Por ello, las conclusiones deben interpretarse dentro del contexto evaluado. Las oportunidades identificadas en usabilidad y operación proporcionan una base concreta para su evolución y para evaluaciones posteriores de mayor alcance.

Contribuciones del trabajo

Respecto de los trabajos previos analizados, la contribución no radica en integrar por primera vez inventario y catálogo, pues antecedentes como JAVSA S.A. ya relacionaron ambos componentes [13]; tampoco en reproducir una tienda electrónica, un catálogo privado o un sistema de ventas construido para una sola organización [5, 12, 22, 26]. La contribución diferenciada consiste en construir una base de software configurable, aplicarla en tres pequeños comercios con productos y estructuras distintas y documentar conjuntamente su funcionamiento técnico, su usabilidad y su operación desplegada.

El aporte funcional consiste en relacionar la gestión interna de pequeñas empresas comerciales con un catálogo público propio. La información de productos, variantes, imágenes y disponibilidad se administra desde una misma solución, de modo que el negocio conserva el control de sus registros y el cliente consulta una presentación derivada de los datos ope-

rativos vigentes. La configuración de categorías, atributos, estructura física, apariencia y reglas comerciales permite utilizar esta base en distintas configuraciones de negocio, con o sin local físico.

En el ámbito técnico, el trabajo aporta un modelo que mantiene separados el saldo vigente, el kardex y los documentos comerciales, pero coordina sus modificaciones mediante reglas y transacciones. Esta decisión permite reconstruir entradas, salidas, transferencias, ventas, cambios, devoluciones y anulaciones sin confundir los movimientos reales de existencias con los cambios administrativos. También se incorporaron autorización por contexto, roles y permisos, una biblioteca reutilizable de imágenes y mecanismos diferenciados de auditoría.

El aporte metodológico y operativo se encuentra en la trazabilidad entre requerimientos, issues, cambios versionados, pruebas y canalizaciones de integración continua. La separación de repositorios, las imágenes identificadas por commit, la promoción controlada entre ambientes y el provisionamiento multiinstancia vinculan la construcción del software con su operación. A esto se suma un mecanismo de respaldo que reúne bases de datos, almacenamiento y configuración, y que comprueba su integridad mediante sumas de verificación y restauraciones temporales.

La contribución académica se expresa en la evidencia obtenida al aplicar y evaluar la propuesta en tres contextos comerciales, en lugar de limitar su comprobación a una única configuración. Las pruebas automatizadas delimitaron el alcance comprobado en backend, frontend e infraestructura, mientras que las sesiones con participantes permitieron contrastar los recorridos administrativos y públicos. La validación operativa agregó la revisión de ambientes desplegados, respaldos y restauraciones. Esta combinación evita fundamentar el resultado únicamente en la revisión del código y proporciona observaciones concretas para continuar mejorando la experiencia. Debido a que no se estableció una línea base antes de la implementación, esta contribución no se presenta como evidencia de ahorro de tiempo o reducción de errores, sino como una solución tecnológica construida, aplicada y evaluada de forma documentada.

En el plano formativo, la amplitud funcional y operativa del proyecto constituyó un reto de integración de conocimientos adquiridos sobre análisis, modelado de datos, arquitectura, desarrollo web y calidad de software. Los aspectos que exigieron conocimientos adicionales motivaron la investigación técnica y el aprendizaje autónomo sobre seguridad, pruebas, integración continua, despliegue y recuperación, fortaleciendo las competencias de ingeniería de software aplicadas durante el desarrollo.

Recomendaciones

Las recomendaciones siguientes se derivan de los resultados de usabilidad, las comprobaciones operativas y las limitaciones de la evaluación:

1. Priorizar los ajustes identificados durante la evaluación de usabilidad: explicar con mayor claridad el alcance de los permisos, hacer más visible la vista previa de apariencia, facilitar el reinicio de filtros, agregar la disponibilidad al selector de variantes y destacar la lista de deseos en la navegación móvil. Estas mejoras deben contrastarse nuevamente con los perfiles administrativo y público.
2. Mantener sincronizados el modelo de datos, las migraciones, los datos iniciales, la documentación de la API y las pruebas automatizadas. Cada modificación de una regla de inventario, venta, autorización o disponibilidad pública debe conservar evidencia en los niveles de prueba correspondientes y aprobar las modificaciones antes de su promoción.
3. Ejecutar periódicamente la verificación de respaldos mediante restauraciones aisladas y revisar tanto las bases de datos como los archivos y la configuración incluidos. También se recomienda incorporar seguimiento continuo del estado de los servicios, capacidad, almacenamiento, tiempos de respuesta y alertas operativas para complementar las comprobaciones puntuales.
4. Ampliar gradualmente la evaluación de usabilidad a más negocios, perfiles, dispositivos y navegadores. El registro longitudinal de tiempos, solicitudes de ayuda, errores y puntuaciones permitiría comparar las mejoras incorporadas y conocer su efecto con mayor alcance que la muestra inicial.

Trabajos futuros

Las líneas de trabajo siguientes amplían el alcance funcional y operativo de Stockify a partir de la base implementada:

1. Incorporar un módulo configurable de promociones y descuentos por producto o variante. Este módulo podría aplicar condiciones por cantidad, periodo, categoría, colección o segmento, además de registrar la promoción aplicada en cada operación para identificar el origen del descuento y mantener la trazabilidad de los importes.
2. Definir planes de servicio, por ejemplo básico, intermedio y completo, que agrupen módulos, capacidades y límites de uso. La disponibilidad de cada función deberá

controlarse en el backend mediante las condiciones asociadas al plan y no únicamente mediante la visibilidad de opciones en la interfaz.

3. Integrar los planes con un flujo de suscripción y pago que automatice el registro de nuevos clientes. La confirmación de una suscripción podría iniciar la asignación del plan, la preparación de los recursos y del dominio, la ejecución de migraciones, la carga de datos iniciales y las comprobaciones de funcionamiento. Este proceso debería evitar la duplicación de recursos ante intentos repetidos, además de mantener un registro de las operaciones y permitir la reversión de cambios en caso de fallos.
4. Evaluar una evolución gradual del backend multiinstancia hacia una arquitectura *multitenant*. Un servicio compartido podría reducir la cantidad de procesos por negocio, centralizar despliegues y migraciones, aprovechar mejor los recursos y agilizar el provisionamiento. Su conveniencia deberá determinarse mediante pruebas de carga y seguridad, considerando el aislamiento del cliente en consultas, cachés, tareas, archivos y respaldos, así como el alcance de una falla sobre los demás negocios.
5. Ampliar el módulo comercial con facturación electrónica conforme a los requisitos técnicos y normativos vigentes del Servicio de Rentas Internas (SRI). Esta evolución deberá integrar los comprobantes con las ventas, las formas de pago y los ajustes de inventario, además de conservar la trazabilidad de su emisión y autorización.
6. Extender la personalización del catálogo mediante plantillas, distribuciones de página, estilos, tarjetas y componentes configurables por negocio. La evolución deberá conservar la responsividad, incluir una vista previa clara y mantener una representación consistente de productos, variantes, colecciones y disponibilidad.

Referencias

- [1] A. Berson, *Client/Server Architecture*. New York, NY, USA: McGraw-Hill, 1992.
- [2] J. Brooke, “SUS: A ‘quick and dirty’ usability scale,” in *Usability Evaluation in Industry*, P. W. Jordan, B. Thomas, B. A. Weerdmeester, and I. L. McClelland, Eds. London, U.K.: Taylor & Francis, 1996, pp. 189–194, doi: 10.1201/9781498710411-35.
- [3] R. T. Fielding, *Architectural Styles and the Design of Network-based Software Architectures*, Ph.D. dissertation, Univ. California, Irvine, CA, USA, 2000.
[Online]. Available:
<https://ics.uci.edu/~fielding/pubs/dissertation/top.htm>
- [4] L. A. Mora García, *Indicadores de la gestión logística*, 2nd ed. Bogotá, Colombia: Ecoe Ediciones, 2008, ISBN: 978-958-648-563-0. [Online]. Available:
<https://ecoeediciones.com/product/indicadores-de-la-gestion-logistica-2da-edicion/>
- [5] C. E. Gómez Herrera, *Análisis, diseño y construcción del sistema de comercio electrónico para la empresa Dolarin*. Trabajo de titulación, Dpto. de Ciencias de la Computación, Escuela Politécnica del Ejército, Sangolquí, Ecuador, Sep. 2011.
[Online]. Available:
https://observatorioecuadordigital.mintel.gob.ec/wp-content/uploads/2022/11/T-ESPE-032795-1-DOLARIN_compressed.pdf
- [6] M. B. Jones, J. Bradley, and N. Sakimura, “JSON Web Token (JWT),” RFC 7519, May 2015, doi: 10.17487/RFC7519.
- [7] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, 7th ed. Boston, MA, USA: Pearson, Jun. 8, 2015, ISBN: 978-0-13-397077-7.
- [8] OECD, *Frascati Manual 2015: Guidelines for Collecting and Reporting Data on Research and Experimental Development*, The Measurement of Scientific, Technological and Innovation Activities. Paris, France: OECD Publishing, Oct. 2015, doi: 10.1787/9789264239012-en.
- [9] K. C. Laudon and J. P. Laudon, *Sistemas de información gerencial*, 14th ed. Ciudad de México, México: Pearson Educación de México, 2016, ISBN: 978-607-32-3696-6.

- [10] Instituto Nacional de Estadística y Censos (INEC), *Tecnologías de la Información y Comunicación – Empresas: Módulo de Tecnologías de la Información y la Comunicación (TIC) de las Encuestas de Manufactura y Minería, Comercio Interno y Servicios 2015*, versión 1.3, Ecuador, 2016, IDD-ECU-INEC-DECON-TIC-EMPRESAS-2015-v1.3. [Online]. Available: <https://anda.inec.gob.ec/anda5/index.php/catalog/733/study-description>
- [11] H. Guerrero Salas, *Inventarios: manejo y control*, 2nd ed. Bogotá, Colombia: Ecoe Ediciones, Jul. 2017, ISBN: 978-958-771-491-3. [Online]. Available: <https://www.ecoeediciones.com/product/inventarios-manejo-y-control-2da-edicion-impreso/>
- [12] E. R. Jiménez Ruiz, *Aplicación web de catálogos privados utilizando la arquitectura MVC para la empresa TURBOMEKANICS S.A. de la ciudad de Ambato*, Trabajo de titulación, Facultad de Ingeniería en Sistemas, Electrónica e Industrial, Universidad Técnica de Ambato, Ambato, Ecuador, Aug. 2017. [Online]. Available: <https://repositorio.uta.edu.ec/handle/123456789/26218>
- [13] J. E. Buri Naula and D. F. Pillajo Pacheco, *Desarrollo de un sistema web para control de inventarios que cuente con catálogo en línea de productos de la empresa JAVSA S.A.*, Trabajo de titulación, Escuela Politécnica Nacional, Quito, Ecuador, May 15, 2018. [Online]. Available: <https://bibdigital.epn.edu.ec/handle/15000/19417>
- [14] R. Hernández-Sampieri and C. P. Mendoza Torres, *Metodología de la investigación: las rutas cuantitativa, cualitativa y mixta*, 1st ed. Ciudad de México, México: McGraw-Hill Education, Jun. 15, 2018, ISBN: 978-1-4562-6096-5.
- [15] A. Silberschatz, H. F. Korth, and S. Sudarshan, *Database System Concepts*, 7th ed. New York, NY, USA: McGraw-Hill Education, Mar. 2019, ISBN: 978-0-07-802215-9.
- [16] Organisation for Economic Co-operation and Development, *OECD SME and Entrepreneurship Outlook 2019*. Paris, France: OECD Publishing, May 2019, doi: 10.1787/34907e9c-en.
- [17] Y. Sheffer, D. Hardt, and M. B. Jones, “JSON Web Token Best Current Practices,” RFC 8725, BCP 225, Feb. 2020, doi: 10.17487/RFC8725.
- [18] K. Schwaber and J. Sutherland, *La Guía de Scrum: La Guía Definitiva de Scrum: Las Reglas del Juego*, Nov. 2020. [Online]. Available:

<https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Spanish-Latin-South-American.pdf>

- [19] M. D. R. Sevilla-Gonzalez, L. Moreno Loaeza, L. S. Lazaro-Carrera, B. Bourguet Ramirez, A. Vázquez Rodríguez, M. L. Peralta-Pedrero, and P. Almeda-Valdes, “Spanish Version of the System Usability Scale for the Assessment of Electronic Tools: Development and Validation,” *JMIR Human Factors*, vol. 7, no. 4, Art. no. e21161, Dec. 16, 2020, doi: 10.2196/21161.
- [20] M. Dini, N. Gligo, and A. Patiño, *Transformación digital de las mipymes: elementos para el diseño de políticas*, Documentos de Proyectos, LC/TS.2021/99. Santiago, Chile: Comisión Económica para América Latina y el Caribe (CEPAL), Aug. 2021.
- [21] C. Suaznabar, D. Herrera, and A. Cathles, *Convivir con el coronavirus: ¿Cómo aprovechar la inercia para digitalizar a las pymes de la región?*, M. Alfonso, Ed. Washington, DC, USA: Inter-American Development Bank, Feb. 2022, doi: 10.18235/0004007.
- [22] A. A. Bacilio Tumbaco, *Desarrollo de una tienda online con administración de una aplicación web para ventas de productos de la ferretería Lascano*, Examen complejo, Facultad de Sistemas y Telecomunicaciones, Universidad Estatal Península de Santa Elena, La Libertad, Ecuador, Jun. 2022. [Online]. Available: <https://repositorio.upse.edu.ec/handle/46000/7702>
- [23] R. T. Fielding, M. Nottingham, and J. Reschke, Eds., “HTTP Semantics,” RFC 9110, STD 97, Jun. 2022, doi: 10.17487/RFC9110.
- [24] J. A. Castro-Pacheco and D. A. Zaldumbide-Peralvo, “Proceso de digitalización comercial y su relación con las ventas de los negocios gastronómicos. Estudio de caso en el contexto local en el periodo 2020–2021,” *593 Digital Publisher CEIT*, vol. 7, no. 4-1, pp. 499–509, Aug. 2022, doi: 10.33386/593dp.2022.4-1.1253.
- [25] D. F. Sánchez Espín, *Desarrollo e implementación de un sistema web utilizando la librería Reactjs para el control de inventario de la empresa óptica Optimus*. Trabajo de titulación, Escuela Superior Politécnica de Chimborazo, Riobamba, Ecuador, May 17, 2023. [Online]. Available: <https://dspace.esPOCH.edu.ec/handle/123456789/20833>
- [26] A. A. Castro Pérez, *Desarrollo de un sistema web dirigido al proceso de ventas de la empresa Silvagro WSV S.A.*, Trabajo de titulación, Facultad de Ingeniería Industrial, Carrera de Licenciatura en Sistemas de Información, Universidad de

- Guayaquil, Guayaquil, Ecuador, Sep. 2023. [Online]. Available: <https://repositorio.ug.edu.ec/handle/redug/69975>
- [27] J. Hirs and F. Vargas, *Prioridades para la digitalización empresarial en América Latina y el Caribe*. Washington, DC, USA: Inter-American Development Bank, Sep. 2023, doi: 10.18235/0005166.
- [28] Instituto Nacional de Estadística y Censos (INEC), *Encuesta Estructural Empresarial 2022: Tomo I, Tomo II, Establecimientos, TIC*, versión 1.3, 2024. [Online]. Available: <https://anda.inec.gob.ec/anda5/index.php/catalog/1046/study-description>
- [29] M. Toogood, “How to create Amazon product listings,” *Sell on Amazon*, Oct. 3, 2025. [Online]. Available: <https://sell.amazon.com/blog/amazon-product-listings>
- [30] Instituto Nacional de Estadística, *Encuesta sobre el uso de TIC y del comercio electrónico en las empresas. Año 2024 - Primer trimestre 2025. Datos definitivos*. Madrid, España: INE, 2025.
- [31] OWASP Foundation, “Authorization Cheat Sheet,” *OWASP Cheat Sheet Series*. [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html
- [32] OWASP Foundation, “A01:2025 Broken Access Control,” *OWASP Top 10:2025*, 2025. [Online]. Available: https://owasp.org/Top10/2025/A01_2025-Broken_Access_Control/
- [33] Amazon, “Inventory management techniques and best practices,” *Sell on Amazon*. [Online]. Available: <https://sell.amazon.com/learn/inventory-management>.
- [34] Amazon, “Amazon Seller Central: Your digital portal for selling with Amazon,” *Sell on Amazon*. [Online]. Available: <https://sell.amazon.com/tools/seller-central>.
- [35] D. Quaker, “5 steps for building an Amazon Brand Store,” *Sell on Amazon*, Mar. 6, 2026. [Online]. Available: <https://sell.amazon.com/blog/amazon-storefront>.
- [36] Tiendanube, “¿Qué son las variantes y cómo configurarlas en mis productos?”, *Centro de Ayuda*, Mar. 8, 2026. [Online]. Available: <https://>

`//ayuda.tiendanube.com/es_MX/160437-variantes-del-producto/
que-son-las-variantes-y-como-configurarlas-en-mis-productos.`

- [37] Docker Inc., “What is a container?,” *Docker Docs*. [Online]. Available:
[https://docs.docker.com/get-started/docker-concepts/the-basics/
what-is-a-container/](https://docs.docker.com/get-started/docker-concepts/the-basics/what-is-a-container/).
- [38] GitLab B.V., “CI/CD pipelines,” *GitLab Docs*. [Online]. Available:
<https://docs.gitlab.com/ci/pipelines/>.
- [39] GitLab B.V., “Issues,” *GitLab Docs*. [Online]. Available:
<https://docs.gitlab.com/user/project/issues/>.
- [40] GitLab B.V., “Merge requests,” *GitLab Docs*. [Online]. Available:
https://docs.gitlab.com/user/project/merge_requests/.
- [41] MDN contributors, “Diseño receptivo,” *MDN Web Docs*, Apr. 29, 2026. [Online].
Available: [https://developer.mozilla.org/es/docs/Learn_web_
development/Core/CSS_layout/Responsive_Design](https://developer.mozilla.org/es/docs/Learn_web_development/Core/CSS_layout/Responsive_Design).
- [42] Odoo S.A., “Product catalog,” *Odoo 19.0 Documentation*. [Online]. Available:
[https://www.odoo.com/documentation/19.0/applications/websites/
ecommerce/ecommerce_design/catalog.html](https://www.odoo.com/documentation/19.0/applications/websites/ecommerce/ecommerce_design/catalog.html).
- [43] Odoo S.A., “eCommerce,” *Odoo 19.0 Documentation*. [Online]. Available:
[https://www.odoo.com/documentation/19.0/applications/websites/
ecommerce.html](https://www.odoo.com/documentation/19.0/applications/websites/ecommerce.html).
- [44] Odoo S.A., “Inventory,” *Odoo 19.0 Documentation*. [Online]. Available:
[https://www.odoo.com/documentation/19.0/applications/inventory_
and_mrp/inventory.html](https://www.odoo.com/documentation/19.0/applications/inventory_and_mrp/inventory.html).
- [45] Shopify, “Locations,” *Shopify Help Center*. [Online]. Available:
<https://help.shopify.com/en/manual/fulfillment/setup/locations>.
- [46] Shopify, “Managing inventory,” *Shopify Help Center*. [Online]. Available:
<https://help.shopify.com/en/manual/products/inventory>.
- [47] Shopify, “Online Store,” *Shopify Help Center*. [Online]. Available:
<https://help.shopify.com/en/manual/online-store>.
- [48] Shopify, “Variants,” *Shopify Help Center*. [Online]. Available:
<https://help.shopify.com/en/manual/products/variants>.

- [49] Tiendanube, “¿Cómo completar el Excel de carga masiva de productos?”, *Centro de Ayuda*, Jun. 21, 2026. [Online]. Available: https://ayuda.tiendanube.com/es_CO/122710-importar-y-exportar-productos/como-completar-el-excel-de-carga-masiva-de-productos.
- [50] Tiendanube, “Lista de productos,” *Centro de Ayuda*. [Online]. Available: https://ayuda.tiendanube.com/es_ES/122708-lista-de-productos.
- [51] Tiendanube, “¿Cómo editar el stock de mis productos?”, *Centro de Ayuda*, Jul. 13, 2026. [Online]. Available: https://ayuda.tiendanube.com/es_MX/gestion-de-inventario/como-editar-el-stock-de-mis-productos.

Anexos

Formulario de evaluación de usabilidad del área administrativa

Consentimiento

Esta actividad evalúa la experiencia de uso del área administrativa de Stockify. Se utilizarán datos de prueba y las respuestas se registrarán con un código para el análisis académico del trabajo de titulación. La participación es voluntaria; la persona puede solicitar una pausa, dejar una actividad o retirarse durante la sesión. Las respuestas se presentan de forma codificada y agregada.

☐ He leído la información y acepto participar.

Código de participante: _____

Nombre y firma: _____

Fecha: _____

Perfil breve

1. ¿Ha utilizado sistemas de inventario o ventas?

☐ Nunca ☐ Ocasionalmente ☐ Con frecuencia

2. ¿Con qué frecuencia gestiona productos, existencias o ventas?

☐ Menos de una vez por semana

☐ Una o varias veces por semana

☐ Diariamente

Dispositivo utilizado: _____

Navegador: _____

Actividades

Utilice únicamente los datos de prueba proporcionados y realice cada actividad como lo haría normalmente.

Tabla 50: Actividades del formulario administrativo

N.	Actividad
1	Ingresa al área administrativa con la cuenta proporcionada y confirme el negocio asignado.
2	Revise el rol indicado y cree un usuario de prueba con los permisos solicitados.
3	Consulte o actualice la información del negocio según la indicación recibida.
4	Prepare la categoría, marca y atributos necesarios para el producto de prueba.
5	Registre el producto con sus variantes, precio e imagen.
6	Registre la cantidad indicada y consulte la disponibilidad resultante.
7	Localice el movimiento de inventario relacionado con el producto.
8	Registre la venta de prueba con el producto, variante y cantidad proporcionados.
9	Consulte la venta registrada e identifique su efecto en las existencias.
10	Complete la actividad final indicada para su escenario, relacionada con colecciones, apariencia o ubicación de inventario.

Fuente: Autor.

Cuestionario posterior

Marque una opción por enunciado: 1 significa *totalmente en desacuerdo*; 2, *en desacuerdo*; 3, *neutral*; 4, *de acuerdo*; y 5, *totalmente de acuerdo*.

Tabla 51: Escala SUS del formulario administrativo

N.	Enunciado	1	2	3	4	5
1	Me gustaría usar esta herramienta frecuentemente.	☐	☐	☐	☐	☐
2	Considero que esta herramienta es innecesariamente compleja.	☐	☐	☐	☐	☐
3	Considero que la herramienta es fácil de usar.	☐	☐	☐	☐	☐
4	Considero necesario el apoyo de personal experto para poder utilizar esta herramienta.	☐	☐	☐	☐	☐
5	Considero que las funciones de la herramienta están bien integradas.	☐	☐	☐	☐	☐
6	Considero que la herramienta presenta muchas contradicciones.	☐	☐	☐	☐	☐
7	Imagino que la mayoría de las personas aprenderían a usar esta herramienta rápidamente.	☐	☐	☐	☐	☐
8	Considero que el uso de esta herramienta es tedioso.	☐	☐	☐	☐	☐

9	Me sentí muy confiado al usar la herramienta.	☐ ☐ ☐ ☐ ☐
10	Necesité saber bastantes cosas antes de poder empezar a usar esta herramienta.	☐ ☐ ☐ ☐ ☐

Fuente: Adaptado de la versión española de la escala SUS de Sevilla-González et al. [19], bajo licencia CC BY 4.0.

Tabla 52: Valoración general del área administrativa

Enunciado	1	2	3	4	5
La información presentada fue clara.	☐	☐	☐	☐	☐
Completar las tareas resultó fácil.	☐	☐	☐	☐	☐
La experiencia de uso fue satisfactoria.	☐	☐	☐	☐	☐

Fuente: Autor.

Comentarios opcionales

- ¿Qué le ayudó a realizar las tareas?

- ¿Qué fue difícil o confuso?

- ¿Hubo algún momento en que el sistema se comportó de forma distinta a lo que esperaba? Describa qué ocurrió.

- ¿Cómo se relaciona este flujo con la forma en que realiza actualmente estas actividades?

- ¿Qué cambio mejoraría primero?

- ¿Desea agregar algo sobre la experiencia de uso?

Registro del responsable

Resultado: C = completada; CA = completada con ayuda; NC = no completada; NA = no aplicable.

Tabla 53: Registro de las tareas administrativas

Tarea	Resultado	Tiempo aprox.	Observación breve
ADM-T01	_____	_____	_____
ADM-T02	_____	_____	_____
ADM-T03	_____	_____	_____
ADM-T04	_____	_____	_____
ADM-T05	_____	_____	_____
ADM-T06	_____	_____	_____
ADM-T07	_____	_____	_____
ADM-T08	_____	_____	_____
ADM-T09	_____	_____	_____
ADM-T10	_____	_____	_____

Fuente: Autor.

Comentario final: _____

Formulario de evaluación de usabilidad del catálogo público

Consentimiento

Esta actividad evalúa la experiencia de uso del catálogo público de Stockify. Se utilizarán datos de prueba y las respuestas se registrarán con un código para el análisis académico del trabajo de titulación. La participación es voluntaria; la persona puede solicitar una pausa, dejar una actividad o retirarse durante la sesión. Las respuestas se presentan de forma codificada y agregada.

☐ He leído la información y acepto participar.

Código de participante: _____

Nombre y firma: _____

Fecha: _____

Perfil breve

1. ¿Ha consultado antes catálogos o tiendas en línea?

☐ Nunca ☐ Ocasionalmente ☐ Con frecuencia

2. ¿Ha utilizado antes este catálogo?

☐ Sí ☐ No ☐ No lo recuerda

3. ¿Con qué frecuencia navega en la web desde un teléfono móvil?

☐ Ocasionalmente

☐ Varias veces por semana

☐ Diariamente

Dispositivo utilizado: _____

Navegador: _____

Actividades

Utilice el enlace proporcionado y realice las actividades como lo haría normalmente desde su dispositivo.

Tabla 54: Actividades del formulario del catálogo público

N.	Actividad
1	Abra el catálogo e identifique el negocio al que pertenece.
2	Encuentre el producto indicado mediante la búsqueda.
3	Use los filtros para reducir los resultados según la condición proporcionada.
4	Abra el producto e identifique su descripción, precio e imagen.
5	Seleccione la variante indicada.
6	Determine si la variante está disponible y explique qué información utilizó.
7	Agregue el producto a la lista de deseos y encuéntralo nuevamente desde esa lista.
8	Localice el medio de contacto del negocio y explique qué acción realizaría.
9	Desde la vista móvil, vuelva al listado, abra la navegación y regrese al producto guardado.

Fuente: Autor.

Cuestionario posterior

Marque una opción por enunciado: 1 significa *totalmente en desacuerdo*; 2, *en desacuerdo*; 3, *neutral*; 4, *de acuerdo*; y 5, *totalmente de acuerdo*.

Tabla 55: Escala SUS del formulario de catálogo público

N.	Enunciado	1	2	3	4	5
1	Me gustaría usar esta herramienta frecuentemente.	☐	☐	☐	☐	☐
2	Considero que esta herramienta es innecesariamente compleja.	☐	☐	☐	☐	☐
3	Considero que la herramienta es fácil de usar.	☐	☐	☐	☐	☐
4	Considero necesario el apoyo de personal experto para poder utilizar esta herramienta.	☐	☐	☐	☐	☐
5	Considero que las funciones de la herramienta están bien integradas.	☐	☐	☐	☐	☐
6	Considero que la herramienta presenta muchas contradicciones.	☐	☐	☐	☐	☐
7	Imagino que la mayoría de las personas aprenderían a usar esta herramienta rápidamente.	☐	☐	☐	☐	☐
8	Considero que el uso de esta herramienta es tedioso.	☐	☐	☐	☐	☐
9	Me sentí muy confiado al usar la herramienta.	☐	☐	☐	☐	☐

- 10 Necesité saber bastantes cosas antes de poder empezar a usar esta herramienta. ☐ ☐ ☐ ☐ ☐

Fuente: Adaptado de la versión española de la escala SUS de Sevilla-González et al. [19], bajo licencia CC BY 4.0.

Tabla 56: Valoración general del catálogo público

Enunciado	1	2	3	4	5
La información de los productos, variantes y disponibilidad fue clara.	☐	☐	☐	☐	☐
Completar las tareas del catálogo resultó fácil.	☐	☐	☐	☐	☐
Navegar desde el dispositivo utilizado resultó fácil.	☐	☐	☐	☐	☐

Fuente: Autor.

Comentarios opcionales

1. ¿Qué le ayudó a encontrar o comprender un producto?

2. ¿Qué fue difícil o confuso?

3. ¿Cómo interpretó la información del producto y su disponibilidad?

4. ¿Encontró el medio de contacto que esperaba? Describa qué esperaba encontrar.

5. ¿Qué cambio mejoraría primero en el catálogo?

6. ¿Desea agregar algo sobre su experiencia de uso?

Registro del responsable

Resultado: C = completada; CA = completada con ayuda; NC = no completada; NA = no aplicable.

Tabla 57: Registro de las tareas del catálogo público

Tarea	Resultado	Tiempo aprox.	Observación breve
PUB-T01	_____	_____	_____
PUB-T02	_____	_____	_____
PUB-T03	_____	_____	_____
PUB-T04	_____	_____	_____
PUB-T05	_____	_____	_____
PUB-T06	_____	_____	_____
PUB-T07	_____	_____	_____
PUB-T08	_____	_____	_____
PUB-T09	_____	_____	_____

Fuente: Autor.

Comentario final: _____

Inventario técnico y trazabilidad funcional de la solución

Manifiesto de repositorios y versiones

Tabla 58: Repositorios y commits de la versión evaluada

Repositorio	Rama local	Commit	Responsabilidad observada
stockify-back	develop	f8bbe0a60386999adbc8dfdc521679f4536199a6	API NestJS, reglas, persistencia, migraciones y almacenamiento de medios.
stockify-front	develop	ad5dbb7e9e98098519eb52f54c0d3547d522384d	SPA React, rutas públicas, cuenta y administración.
stockify-infra	develop	55a5e104292d3bc6fa8d08bc601a3685455cf622	Composición, Nginx, provisión y operación documentada.

Fuente: Autor.

Tecnologías y magnitudes verificadas

Tabla 59: Tecnologías, versiones y magnitudes de la versión evaluada

Aspecto	Valor declarado en los artefactos
Construcción y runtime	Node.js 22 (node:22-alpine) en los Dockerfiles de backend y frontend.
Backend	NestJS ^11.1.24; TypeORM ^0.3.30; controlador PostgreSQL ^8.21.0.
Frontend	React ^19.2.6; React Router ^7.15.1; TanStack Query ^5.101.0; Zod ^4.4.3; Vite ^8.0.14.
Servido y persistencia	Nginx 1.29 alpine (SPA sin privilegios y proxy de ambiente); postgres:16-alpine en los overlays QA y producción.
Persistencia estructurada	50 entidades ORM registradas en ormEntities, distribuidas en dominios de autenticación, usuarios, RBAC, negocio, estructura operativa, catálogo, atributos, medios, inventario, ventas, apariencia, colecciones, auditoría operativa y lista de deseos.
Contrato HTTP	24 módulos funcionales; 52 rutas base @Controller; 216 operaciones HTTP (95 GET, 52 POST, 44 PATCH, 20 DELETE, 5 PUT).

Fuente: Autor.

Matriz de trazabilidad de requerimientos funcionales

Tabla 60: Trazabilidad individual de los requerimientos funcionales

ID	Criterio o comportamiento esperado	Evidencia de implementación	Evidencia de validación	Tipo de comprobación	Resultado	Estado
RF-ADM-01	Autenticación administrativa con sesión renovable.	Secciones 4.3 y 4.4.	Tabla 35.	Prueba E2E de API; prueba de interfaz con API controlada.	Aprobado	Cumplido
RF-ADM-02	Gestión de usuarios, invitaciones, roles y permisos.	Sección 4.4 y Figura 25.	Tabla 45, T02.	Tarea de usabilidad.	Completado	Cumplido
RF-ADM-03	Asignación de ubicaciones laborales según permisos.	Sección 4.4.	Implementación descrita en la Sección 4.4.	Inspección de implementación.	Verificado	Cumplido
RF-ADM-04	Restricción administrativa por origen IP autorizado.	Secciones 4.3 y 4.6.	Controles de acceso descritos en la Sección 4.6.	Inspección de implementación.	Verificado	Cumplido
RF-ADM-05	Gestión de categorías, atributos, valores heredables y estados públicos.	Sección 4.4.	Tabla 45, T04.	Tarea de usabilidad.	Completado	Cumplido
RF-ADM-06	Gestión de marcas, productos, variantes, precios e imágenes.	Sección 4.4.	Tabla 45, T05.	Tarea de usabilidad.	Completado	Cumplido
RF-ADM-07	Gestión de colecciones y su visibilidad pública.	Sección 4.4.	Implementación descrita en la Sección 4.4.	Inspección de implementación.	Verificado	Cumplido
RF-ADM-08	Configuración de sucursales, áreas y ubicaciones de stock.	Sección 4.4.	Tabla 45, T03.	Tarea de usabilidad.	Completado	Cumplido
RF-ADM-09	Consulta de existencias, umbrales y kardex por variante y ubicación.	Sección 4.4.	Tabla 45, T06.	Tarea de usabilidad.	Completado	Cumplido
RF-ADM-10	Registro de entradas, salidas, ajustes y transferencias de stock.	Sección 4.4.	Tabla 45, T07.	Tarea de usabilidad.	Completado	Cumplido
RF-ADM-11	Registro y consulta de ventas.	Sección 4.4.	Tabla 35 y Tabla 45, T08.	Prueba E2E de API; tarea de usabilidad.	Aprobado	Cumplido
RF-ADM-12	Cancelación de venta con reposición idempotente de stock.	Sección 4.4.	Tabla 35.	Prueba E2E de API.	Aprobado	Cumplido
RF-ADM-13	Cambios, devoluciones y ajustes comerciales trazables.	Sección 4.4.	Implementación descrita en la Sección 4.4.	Inspección de implementación.	Verificado	Cumplido
RF-ADM-14	Configuración de identidad, datos, redes, visibilidad y apariencia del negocio.	Sección 4.4.	Tabla 45, T03.	Tarea de usabilidad.	Completado	Cumplido

RF-ADM-15	Configuración de obligatoriedad de sucursal en ventas.	Sección 4.4.	Implementación descrita en la Sección 4.4.	Inspección de implementación.	Verificado	Cumplido
RF-PUB-01	Navegación, búsqueda, paginación, filtro de marca y ordenamiento.	Sección 4.5.	Tabla 35 y Tabla 46, T02–T03.	Prueba de interfaz con API controlada; tarea de usabilidad.	Aprobado	Cumplido
RF-PUB-02	Filtrado por categoría o subcategoría.	Sección 4.5.	Tabla 46, T03.	Tarea de usabilidad.	Completado	Cumplido
RF-PUB-03	Consulta de variante, imágenes, precio y disponibilidad sin datos internos.	Sección 4.5.	Tabla 35 y Tabla 46, T04–T06.	Prueba E2E de API; prueba de interfaz con API controlada; tarea de usabilidad.	Aprobado	Cumplido
RF-PUB-04	Consulta de productos, categorías, marcas y colecciones destacadas.	Sección 4.5.	Implementación descrita en la Sección 4.5.	Inspección de implementación.	Verificado	Cumplido
RF-PUB-05	Registro y verificación del correo electrónico.	Sección 4.5 y Figura 52.	Implementación descrita en la Sección 4.5.	Inspección de implementación.	Verificado	Cumplido
RF-PUB-06	Inicio de sesión, recuperación de contraseña y administración del perfil.	Sección 4.5 y Figuras 52 y 53.	Implementación descrita en la Sección 4.5.	Inspección de implementación.	Verificado	Cumplido
RF-PUB-07	Gestión de variantes en la lista de deseos.	Sección 4.5.	Tabla 46, T07.	Tarea de usabilidad.	Completado	Cumplido
RF-PUB-08	Consulta de redes y contacto del negocio.	Sección 4.5.	Tabla 46, T08.	Tarea de usabilidad.	Completado	Cumplido
RF-TR-01	Separación de sesiones PUBLIC/ADMIN y autorización por operación.	Sección 4.6.	Tabla 35.	Prueba E2E de API.	Aprobado	Cumplido
RF-TR-02	Validación de datos sensibles e inconsistentes antes de modificar información.	Secciones 4.3 y 4.6.	Tabla 35.	Prueba E2E de API.	Aprobado	Cumplido
RF-TR-03	Conservación lógica de registros desactivados.	Sección 4.6.	Implementación descrita en la Sección 4.6.	Inspección de implementación.	Verificado	Cumplido
RF-TR-04	Registro automático de datos de auditoría en altas y modificaciones.	Sección 4.6 y Figura 56.	Implementación descrita en la Sección 4.6.	Inspección de implementación.	Verificado	Cumplido

Fuente: Autor.

Trazabilidad funcional y mecanismos transversales

Matriz de capacidades administrativas

Tabla 61: Capacidades administrativas y control efectivo del backend

Dominio	Capacidad solicitada desde administración	Control o invariante del backend
Usuarios	Consultar, invitar, editar, cambiar estado y asignar roles o sucursal principal.	Separa permisos por operación, protege al operador y al propietario de sistema y revoca sesiones administrativas tras cambios relevantes.
Roles y permisos	Mantener roles y seleccionar permisos desde el catálogo.	Conserva roles de sistema como solo lectura, aplica bypass de ADMIN y revoca sesiones de usuarios afectados por cambios de rol.
Negocio y settings	Editar información, enlaces sociales y configuración comercial.	Solo admite claves conocidas creadas por seed, valida el tipo del valor y aplica los settings en los casos de uso que regulan catálogo y ventas.
Estructura operativa	Administrar sucursales, áreas y ubicaciones; cerrar o reasignar nodos.	Exige una raíz operativa, bloquea inactivación o eliminación con stock positivo y trata la reasignación como corrección administrativa.
Clasificación de catálogo	Mantener marcas, categorías, atributos y unidades.	Valida estados y relaciones, limita la categoría a tres niveles, impide ciclos y controla el impacto sobre productos visibles.
Productos y variantes	Crear y editar productos, precios, variantes, atributos y visibilidad.	Requiere marca y categoría utilizables, rechaza variantes equivalentes mediante una firma estable y audita transaccionalmente el cambio de categoría.
Colecciones y medios	Agrupar productos, seleccionar activos y ordenar imágenes.	Sustituye asociaciones dentro de transacciones, conserva usos del activo y limita a diez las imágenes activas de una variante.
Inventario	Consultar saldos y movimientos; registrar operaciones y transferencias.	Un escritor común bloquea el ítem, impide saldos negativos y persiste saldo y kardex en la transacción del documento.
Ventas y ajustes	Registrar, cancelar, devolver, cambiar o ajustar.	Recalcula precios, aplica <code>sales_require_branch</code> , valida cantidades y coordina documento, estado y movimientos en una transacción.
Apariencia	Crear perfiles, variantes visuales y programación.	Valida estados y programación, conserva un perfil efectivo y registra eventos operativos para cambios sensibles.
Proyección pública	Activar el catálogo y publicar recursos comerciales.	Exige estado y visibilidad de producto, categoría, marca, variante e imagen; calcula la disponibilidad desde inventario y aplica el setting de agotados en todas las vistas.

Fuente: Autor.

Disponibilidad pública derivada

Tabla 62: Disponibilidad pública derivada de una variante

Ítem elegible	Suma pública	Estado derivado	Efecto con <code>show_out_of_stock_products</code> desactivado
Sí	Mayor que cero	AVAILABLE	El producto permanece visible.
Sí	Igual a cero	OUT_OF_STOCK	El producto se excluye solo si todas sus variantes públicas están en este estado.
No	Igual a cero	COMING_SOON	El producto permanece; este estado no se considera agotado.

Fuente: Autor.

Riesgos representativos y condición de resolución

Tabla 63: Riesgos funcionales y condiciones verificables para su resolución

ID	Tipo	Condición observada	Condición verificable de resolución
F6-R01	Revocación pública	La estrategia consulta la sesión persistente por cada access token ADMIN, pero no para PUBLIC.	Implementar revocación efectiva del contexto público o aceptar formalmente la ventana hasta expiración y verificarla.
F6-R02	Almacenamiento cliente	El frontend conserva access tokens en localStorage; la cookie HttpOnly protege solo el refresh.	Migrar la credencial accesible desde JavaScript o documentar el modelo de amenaza con controles compensatorios verificables.
F6-R03	Catálogo RBAC	Las rutas de gestión de permisos requieren permissions:manage, código ausente del seed base.	Añadir el permiso al catálogo inicial o declarar esas operaciones como exclusivas de ADMIN.
F6-R04	Cobertura de auditoría	Se declaran eventos de eliminación para sucursal, área y ubicación, pero los servicios de eliminación no los insertan.	Agregar los eventos faltantes en la transacción o acotar explícitamente el alcance documentado de la auditoría operativa.
F6-R05	Restauración de stock	Cancelaciones y devoluciones pueden escribir sobre la ubicación conservada sin revalidar el estado de la ruta operativa.	Centralizar la validación o definir un destino de recuperación válido que impida reintroducir stock en nodos inactivos.
F6-R06	Semántica pública	La disponibilidad suma stock de sucursales activas aunque is_public_visible sea falso.	Confirmar la regla con el negocio y documentarla, o alinear el cálculo con la visibilidad de sucursal.
F6-R07	Evolución del esquema	La migración consolidada refleja el esquema actual; las incrementales previas están en migrations_archive.	Incorporar una migración que reconcilie bases existentes y documentar instalación nueva frente a actualización.

Fuente: Autor.

Evidencia operativa y de pruebas de la versión evaluada

Pipelines relevantes verificados

Tabla 64: Pipelines y downstream relevantes verificados

Flujo	Pipeline(s)	Referencia registrada	Ámbito	Resultado observado
Validación backend	2677153017	develop@f8bbe0a6	CI	check, build, test y test:e2e exitosos.
Validación frontend	2677245031	develop@ad5dbb7e	CI	check, build, test y e2e_contract exitosos.
Promoción backend	2677009278 → 2677014688	main@44bb1dc5	PROD	Publicación y downstream de infraestructura exitosos.
Promoción frontend	2677215880 → 2677221138	main@e8becbab	PROD	Publicación y downstream de infraestructura exitosos.
Promoción frontend	2677207438 → 2677214461	staging@9d913af2	QA	Publicación y downstream automático de QA exitosos.
Provisión controlada	2677251952	Ejecución web sobre main	PROD	Trabajo provision_client_prod exitoso; la segunda instancia productiva se observó posteriormente en el VPS.

Fuente: Autor.

Estado observado del VPS

Tabla 65: Detalle sanitizado del VPS inspeccionado

Aspecto	Observación puntual	Límite
Plataforma	VPS Contabo con Ubuntu 24.04.4 LTS.	Se omiten hostname, IP y datos de acceso.
Separación ambiental	Proyectos Compose, redes, PostgreSQL, secretos y puertos diferenciados para QA (checkout staging@51a76f0f) y producción (checkout main@78361c61).	Los recursos siguen compartiéndose entre clientes dentro de un mismo ambiente.
Contenedores	Nueva en ejecución, todos healthy y con cero reinicios: proxy, frontend, PostgreSQL y un backend en QA; proxy, frontend, PostgreSQL y dos backends de negocio en PROD.	Estado instantáneo; no equivale a uptime ni alta disponibilidad.
Persistencia	Una base lógica y un volumen de archivos por cliente; PostgreSQL aceptó conexiones.	PostgreSQL, red y credenciales comunes se comparten dentro de cada ambiente.
Rutas y túnel	Inicio y healthcheck respondieron 200 en los tres contextos locales y en los dos contextos productivos públicos; QA público respondió 403 con ruta local 200, compatible con una restricción en Cloudflare. Cloudflare Tunnel cargado, habilitado y activo.	Comprobación puntual; no sustituye monitoreo continuo ni verifica la política exacta del túnel.

Fuente: Autor.

Paquetes de backup inspeccionados

Tabla 66: Backups QA y producción inspeccionados

Evidencia	QA	PROD	Límite comprobado
Paquete y script	weekly-20260714T032621Z; versión 2.1.1	daily-20260715T015419Z; versión 2.1.1	Solo se inspeccionaron estos paquetes.
Contenido	Una base de cliente; cuatro checksums válidos.	Dos bases de cliente; seis checksums válidos; paquete cifrado saliente presente localmente.	Se omiten nombres internos; no se verificó transferencia externa automática.
Estado y verificación	success; 0 errores; restore_verified: true.	success; 0 errores; restore_verified: true.	La rutina restaura bases temporales; no restaura integralmente storage y configuración.
Programación	No se observó cron o timer instalado.	No se observó cron o timer instalado.	No se afirma ejecución periódica automática.

Fuente: Autor.

Evidencia reproducible de pruebas técnicas

Backend

Tabla 67: Detalle reproducible de las pruebas backend

Comando	Código	Salida resumida	Precisión
npm run check	0	Typecheck, lint y formato aprobados	Sin correcciones automáticas.
npm run build	0	Build NestJS aprobado	Artefacto local.
npm run test:ci	0	99 suites; 882 pruebas; 0 fallos	Pruebas unitarias.
npm run migration:show:e2e	0	1 migración aplicada	Base PostgreSQL E2E dedicada.
npm run test:e2e	0	3 suites; 8 casos; 0 fallos	API NestJS y PostgreSQL aislado.

Fuente: Autor.

Frontend

Tabla 68: Detalle reproducible de las pruebas frontend

Comando	Código	Salida resumida	Precisión
npm run check	0	Typecheck, lint y formato aprobados	ESLint configurado con 0 warnings.
npm run build	0	Build Vite aprobado	Incluye typecheck.
npm run test:ci	0	96 archivos; 523 pruebas	Pruebas unitarias y de componentes.
npm run test:e2e:contract	0	15 casos aprobados	Pruebas automatizadas de interfaz, HTTP controlado y Desktop Chrome.

Fuente: Autor.

Verificaciones técnicas de infraestructura

Tabla 69: Verificaciones técnicas de infraestructura

Verificación	Comprobación realizada
Configuración por ambiente	Validación de la configuración diferenciada para QA y producción.
Simplificación del repositorio	Comprobación de la estructura y los contratos técnicos del repositorio.
Provisionamiento de cliente	Renderizado de QA y producción mediante <code>-validate-only</code> y Docker simulado.
Transporte de secretos	Verificación del manejo previsto para los secretos de despliegue.
Acceso del cliente	Comprobación del host y de las rutas proxy mediante un servidor simulado.

Fuente: Autor.

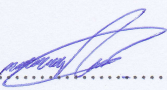


Universidad
Católica
de Cuenca

**AUTORIZACIÓN DE PUBLICACIÓN EN EL
REPOSITORIO INSTITUCIONAL**

Kevin Andres Tarqui Tapia portador(a) de la cédula de ciudadanía N° **0150943934**. En calidad de autor/a y titular de los derechos patrimoniales del trabajo de titulación **“Implementación de un sistema de inventario y catálogo web responsivo para pequeñas empresas comerciales”** de conformidad a lo establecido en el artículo 114 Código Orgánico de la Economía Social de los Conocimientos, Creatividad e Innovación, reconozco a favor de la Universidad Católica de Cuenca una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos y no comerciales. Autorizo además a la Universidad Católica de Cuenca, para que realice la publicación de éste trabajo de titulación en el Repositorio Institucional de conformidad a lo dispuesto en el artículo 144 de la Ley Orgánica de Educación Superior.

Cuenca, **04 de agosto de 2026**

F: 

Kevin Andres Tarqui Tapia

C.I. 0150943934