

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

Unidad Académica de Formación Técnica y Tecnológica – PUCE TEC

SISTEMA WEB DESTINADO A FACILITAR LA COMPRA Y ENTREGA DE GEL INSECTICIDA PARA LA EMPRESA EKINATURE

Proyecto de titulación previo a la obtención del título de:

Tecnología Superior en Desarrollo de Software

Autor: Andrés Ricardo Romero Castillo

Tutor: Christian Roberto Tapia Gaibor

Quito, Ecuador

Introducción 5

Capítulo I: 8

Marco teórico y metodológico 8

1.1 Antecedentes 8

1.2 Marco Conceptual y Tecnologías 8

1.3 Metodología de Desarrollo: Scrum 9

1.3.1 Actores Principales 9

1.3.2 Roles del Sistema (Usuarios) 9

1.4 Etapas del Proceso de Desarrollo 9

1.4.1 Historias de Usuario (Backlog) 10

1.5 Infraestructura y Viabilidad 11

1.5.1 Hardware: Requisitos Mínimos 11

1.5.2 Viabilidad Técnica y Económica 12

Capítulo II 13

Construcción del sistema 13

2.1 Alcance del Desarrollo 13

2.2 Análisis de Diseño y Experiencia de Usuario (UX) 13

2.2.1 Mapa de navegación 14

2.3 Estrategia de Diseño de Interfaz (UI) 14

2.4 Arquitectura del Sistema y Lógica de Negocio 15

2.5 Diseño y Normalización de la Base de Datos 16

2.6 Codificación y Módulos de Automatización 17

2.7 Pruebas y Validación 18

Capítulo III: 18

Pruebas y estabilización 19

3.1 Reuniones con el Product Owner 19

3.2 Entorno de Pruebas y Despliegue 20

3.2.1 Proceso de Pruebas en el Ciclo de Desarrollo 20

3.3 Tipos de Pruebas Realizadas **23**

3.3.1 Pruebas de Frontend (Interfaz) 23

3.3.2 Pruebas de Backend (Servidor y Datos) 23

3.4 Herramientas Utilizadas **23**

3.5 Planilla de Pruebas por Sprint **24**

3.6 Estabilización Final **24**

3.6.1 Optimización de la Interfaz y Coherencia Visual 25

3.6.2 Resiliencia del Backend y Manejo de Excepciones 25

3.6.3 Validación de Interoperabilidad Logística 25

Conclusiones **27**

Recomendaciones **28**

Referencias **29**

Lista de Tablas

Tabla 1 Product Backlog.....14
Tabla 2 Planificación del Sprint 1.....15
Tabla 3 Planificación del Sprint 2..... **Error! Bookmark not defined.**
Tabla 4 Planificación del Sprint 3.....16
Tabla 5 Planificación del Sprint 4.....16
Tabla 6 Casos de Uso: Inicio de Sesión Administrativo.....17
Tabla 7 Casos de Uso: Registro de Pedido Cliente.....17
Tabla 8 Casos de Uso: Control de Intentos Logísticos.....18
Tabla 9 Casos de Uso: Gestión del Carrito de Compras..... **Error! Bookmark not defined.**
Tabla 10 Casos de Uso: Administración de Pedidos 19
Tabla 11 Requisitos mínimos de hardware para la implementación del sistema .....**Error! Bookmark not defined.**
Tabla 12 Resumen de resultados de las pruebas funcionales y de integración por Sprint.....38

Lista de Ilustraciones

Ilustración 1 Product Backlog gestionado durante el desarrollo del proyecto. 14
Ilustración 2 Interfaz de la Landing Page de la plataforma EkiNature. 24
Ilustración 3 Interfaz del carrito de compras y proceso de registro del pedido. 25
Ilustración 4 Formulario de registro de pedido (checkout) 26
Ilustración 5 Diagrama de arquitectura cliente-servidor y flujo de datos API 27
Ilustración 6 Figura 6. Diagrama entidad–relación de la base de datos del sistema EkiNature 29
Ilustración 7 Panel de control administrativo con métricas e indicadores de gestión. 31
Ilustración 8 Evidencia de pruebas funcionales del sistema y ejecución de endpoints durante la validación. 32
Ilustración 9 Sesión de revisión de requerimientos 35
Ilustración 10 Figura 10 Validación de avances con el Product Owner. 35
Ilustración 11 Validación de respuesta de la API de productos en Postman 36
Ilustración 12 Evidencia de persistencia de datos y asignación de ID transaccional mediante método POST. 36
Ilustración 13 Ejecución de prueba del módulo de generación de etiquetas logísticas. 37
Ilustración 14 Monitoreo de logs del servidor confirmando el envío de notificaciones SMTP. 37
Ilustración 15 Verificación en mail de administrador 38
Ilustración 16 Implementación de bloques de control de excepciones para asegurar la estabilidad del servidor. 40
Ilustración 17 Prueba de legibilidad del código QR logístico mediante escaneo con dispositivo móvil. 41

Declaración y Autorización

Yo, **Andrés Ricardo Romero Castillo** con C.I. 1722064068 autor(a) del trabajo de integración curricular intitulado: **"SISTEMA WEB DESTINADO A FACILITAR LA COMPRA Y ENTREGA DE GEL INSECTICIDA PARA LA EMPRESA EKINATURE"**, previa a la obtención del título de **Tecnólogo Superior en Desarrollo de Software** en la Unidad Académica de Formación Técnica y Tecnológica PUCE TEC:

1.- Declaro tener pleno conocimiento de la obligación que tiene la Pontificia Universidad Católica del Ecuador, de conformidad con el artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la Pontificia Universidad Católica del Ecuador a difundir a través del sitio web de la Biblioteca de la PUCE el referido trabajo de titulación, respetando las políticas de propiedad intelectual de la Universidad.

Quito, 15 de febrero de 2026

Andrés Ricardo Romero Castillo
C.I. 1722064068

Agradecimientos

A mis padres, mi abuelita y mi hermano, quienes han sido mi motor y mi mayor apoyo en este camino. Cada uno de ustedes aportó algo fundamental para que yo pudiera llegar a esta meta, gracias por los consejos, la paciencia, los recursos y, sobre todo, por ese amor incondicional que me dio las fuerzas para no rendirme nunca. Los amo con la vida y estaré eternamente agradecido por tenerlos a mi lado.

A mi tía, Iveth Castillo, por darme ese impulso inicial y la motivación que necesitaba para empezar esta experiencia tan enriquecedora. Sin tu apoyo, nada de esto habría sido posible, gracias infinitas por creer en mí desde el primer momento.

A Michelle, mi compañera de vida, mi amiga y mi amor. Gracias por tu paciencia infinita, por tu comprensión y por alentarme a ser una mejor versión de mí mismo cada día. Gracias por escucharme y por recorrer este camino a mi lado, ha sido hermoso compartir este logro contigo.

A ese pequeño ser tan especial en mi vida, te dedico este título y todo el esfuerzo que representa. Gracias porque, incluso en la distancia, me enseñaste más de lo que imaginas sobre mí mismo. Me ayudaste a luchar por mis sueños y a ser una mejor persona. Te amo.

Finalmente, me agradezco a mí mismo por la perseverancia y por no rendirme. A pesar de los días emocionalmente agotadores y las cargas laborales pesadas, decidí levantarme cada mañana y dar lo mejor de mí. Andrés, gracias por no rendirte y por seguir siempre adelante.

Introducción

En Quito existe una gran variedad de plagas denominadas plagas urbanas, como roedores, moscas, cucarachas y hormigas, afectando directamente a los negocios de comida, alojamiento, etc. Esto genera incomodidad, pérdida de clientes y contaminación en la comida que ingerimos.

La empresa EkiNature, la cual se dedica a brindar servicios de control de plagas urbanas y esterilización de espacios en hogares, restaurantes, hoteles, hosterías, tiendas, instituciones educativas, etc., cuenta con un gel insecticida que ayuda a la eliminación de plagas como cucarachas y hormigas.

Este producto es de muy baja toxicidad, lo que lo hace seguro para el ser humano, para las mascotas y para la naturaleza, ya que su aplicación es directa y solo elimina la plaga localizada. No huele, no mancha y es cien por ciento efectivo, ofreciendo una alternativa al servicio de control de plagas.

Sin embargo, actualmente no cuenta con un sistema web que permita mostrar información sobre el producto, facilitar su compra y coordinar su entrega, lo que limita su alcance y dificulta la atención a la creciente demanda.

Por esta razón, se propone el desarrollo de un sistema web destinado a facilitar la compra y entrega del gel insecticida, optimizando las ventas y logística de EkiNature.

Antecedentes

Actualmente, la empresa EkiNature comercializa sus productos de manera manual mediante llamadas telefónicas, aplicativos de chat y atención presencial. Este método limita la difusión de información, dificulta organizar los pedidos y reduce la capacidad de respuesta ante la demanda creciente de clientes.

La ausencia de una plataforma digital impide tener un control de las ventas, controlar el estado de los pedidos y llevar un seguimiento correcto en el despacho del producto. Como resultado, varios procesos se realizan de forma manual, lo que incrementa el tiempo de atención, generando posibles errores y no permite el crecimiento de la empresa en términos comerciales.

En este contexto, se identifica la necesidad de implementar una solución tecnológica que permita automatizar la gestión de ventas y optimizar la entrega final del producto, mejorando la eficiencia operativa y la experiencia del cliente.

Planteamiento del problema

Actualmente, la empresa EkiNature no dispone de un sistema web que le permita mostrar información detallada sobre los beneficios, aplicación y valor comercial de sus servicios y del gel insecticida. Las ventas se realizan de forma manual a través de aplicaciones de mensajería como WhatsApp, lo cual genera informalidad en el proceso y provoca la pérdida de datos relevantes al no existir un registro formal.

Esta carencia tecnológica dificulta que los clientes interesados accedan a la información del producto, afectando la posibilidad de compra. Asimismo, la inexistencia de un registro centralizado de pedidos impide una correcta trazabilidad y logística, dificultando la recopilación de estadísticas de ventas, el control de inventario y la fidelización de clientes, factores críticos para el crecimiento y posicionamiento del producto en el mercado.

Justificación

La implementación de un sistema web es indispensable para resolver las limitaciones operativas actuales de EkiNature. Al automatizar el proceso de venta y logística, la empresa podrá escalar su servicio y atender de manera eficiente la creciente demanda en Quito.

El desarrollo de esta plataforma permitirá centralizar la difusión del producto, facilitando el acceso a la información por parte de los clientes y agilizando la toma de decisiones de compra. Además, al contar con un registro digital de pedidos y entregas, la empresa podrá optimizar sus rutas logísticas, reducir errores humanos y construir una base de datos histórica que sirva para futuras estrategias comerciales y de fidelización.

Objetivo general

Desarrollar una aplicación web para EkiNature que centralice la difusión del gel insecticida, su venta en línea y la gestión automatizada de pedidos y entregas, con el propósito de mejorar la experiencia del cliente y optimizar los procesos comerciales.

Objetivos específicos

- Analizar los requerimientos funcionales y no funcionales para entender mejor el control de plagas, la necesidad del cliente y brindar un sistema robusto.

- Implementar una solución tecnológica utilizando Node.js, Express y una base de datos relacional (MySQL), para garantizar una estructura eficiente y escalable que permita gestionar los procesos de venta, pedidos e información del producto.
- Automatizar la gestión de pedidos de forma que, una vez completada la compra, se genere una notificación al administrador y se planifique la entrega teniendo en cuenta otros envíos programados, para optimizar la logística y mejorar la eficiencia en la atención al cliente.

Metodología de desarrollo de software scrum

Para el desarrollo del sistema web se ha seleccionado el marco de trabajo ágil Scrum. Esta elección ayuda a trabajar en etapas más manejables, facilitando revisar constantemente el avance y realizar ajustes de manera eficiente.

Scrum es un marco de trabajo que divide el proceso de desarrollo en ciclos cortos llamados *sprints*. En este proyecto, cada sprint tiene una duración establecida de una semana e incluye las fases de planificación, ejecución, revisión y retroalimentación.

La metodología se ajustó al proyecto y se organizaron las siguientes actividades:

- **Planificación:** Establecimiento de tareas y objetivos basados en un backlog priorizado.
- **Desarrollo:** Ejecución de las tareas asignadas, asegurando pruebas y que todo funcione antes de avanzar.
- **Revisión y Retrospectiva:** Evaluación de resultados al final de cada sprint para validar el cumplimiento de objetivos e identificar mejoras para el siguiente ciclo.

Capítulo I:

Marco teórico y metodológico

1.1 Antecedentes

La empresa **EkiNature**, dedicada a la comercialización de soluciones efectivas y de baja toxicidad para el control de plagas, se enfrenta a retos significativos en la gestión de sus ventas y logísticas debido a la dependencia de procesos manuales. Actualmente, el registro de pedidos y el seguimiento de entregas se realizan mediante aplicativos de chat, lo que genera retrasos en entregas y pérdidas en dinero.

Para modernizar estos procesos, surge la necesidad de desarrollar una plataforma web. Esta solución utiliza un *frontend* en React.js y un *backend* en Node.js y MySQL, automatizando las ventas, desde la captación del cliente hasta la generación de etiquetas para su despacho.

1.2 Marco Conceptual y Tecnologías

El sistema se basa en una arquitectura cliente-servidor, diseñada para ser escalable y segura.

Tecnologías del Backend

- **Node.js:** Es un entorno de ejecución de JavaScript en el servidor. Su diseño basado en eventos facilita la atención de múltiples solicitudes de manera simultánea, lo que lo convierte en una herramienta eficiente para aplicaciones web.
- **Express.js:** Framework que facilita la creación de la API RESTful. Maneja las rutas de productos, pedidos y contacto.
- **MySQL:** Sistema de gestión de bases de datos relacional elegido por su robustez en transacciones financieras. Se integra mediante el driver mysql2 para ejecutar consultas optimizadas.
- **Dotenv:** Herramienta para la gestión segura de variables de entorno (credenciales de DB, puertos).

Tecnologías del Frontend

- **React.js:** Biblioteca para la construcción de interfaces dinámicas. Permite el uso de componentes reutilizables y la gestión del estado global mediante *Hooks*.

- **Bootstrap:** Framework de diseño utilizado para garantizar que la plataforma sea responsiva y profesional, facilitando el acceso desde celulares y computadoras.

1.3 Metodología de Desarrollo: Scrum

Se optó por implementar Scrum como marco de trabajo debido a su flexibilidad y a la posibilidad de organizar el desarrollo en etapas breves. El sistema puede ajustarse progresivamente a las necesidades de EkiNature, garantizando que la solución sea efectiva a la realidad operativa de la empresa.

1.3.1 Actores Principales

De acuerdo con el marco de trabajo, los roles se distribuyeron de la siguiente manera:

- **Product Owner / Tutor:** Fredy Jovanny Romero Paredes.
- **Scrum Master:** Ing. Christian Roberto Tapia Gaibor.
- **Equipo de Desarrollo:** Andrés Ricardo Romero Castillo.

1.3.2 Roles del Sistema (Usuarios)

- **Administrador:** Gestiona el catálogo, monitorea ventas y revisa los mensajes de contacto.
- **Cliente:** Usuario final que navega por el catálogo, gestiona su carrito y registra su pedido.
- **Operador Logístico:** Responsable de la gestión de estados y generación de etiquetas QR para el despacho.

1.4 Etapas del Proceso de Desarrollo

1. **Levantamiento de Requisitos:** Reuniones con el *Product Owner* para definir las funcionalidades de la interfaz. La más importante, pasarela de pagos con QR y trasferencias bancarias.
2. **Diseño:** Creación de mockups de la interfaz de usuario, diagramas de entidad-relación y definición del *Backlog*.
3. **Construcción:** Desarrollo progresivo dividido en *Sprints*, priorizando la persistencia de datos y el flujo de compra.
4. **Pruebas:** Validación de *endpoints* con Postman y pruebas de usabilidad en el navegador.

5. **Implementación:** Finalmente, el sistema fue desplegado en un entorno productivo gratuito (Vercel/Render) y se llevó a cabo la capacitación técnica correspondiente.1.4.1 Historias de usuario

A continuación, se detallan las historias de usuario identificadas durante el levantamiento de requisitos para la plataforma EkiNature, así como el flujo operativo dentro del sistema:

- **HU01:** Como cliente, deseo visualizar el catálogo de productos con sus beneficios técnicos, para conocer las ventajas del gel insecticida.
- **HU02:** Como cliente, se busca administrar un carrito de compras que incluya diversas opciones de pago, con el propósito de reunir y confirmar los productos seleccionados antes de completar la transacción.
- **HU03:** Como cliente deseo registrar mis datos para formalizar el pedido de forma digital.
- **HU04:** Como administrador, quiero acceder mediante un inicio de sesión, para gestionar la información de ventas y logística de envíos.
- **HU05:** Como administrador, quiero visualizar un panel o página de pedidos pendientes, para organizar las actividades de envío.
- **HU06:** Como administrador, quiero programar fechas de entrega y registrar intentos de envío, para optimizar las entregas a los clientes.
- **HU07:** Como administrador, quiero generar etiquetas de envío automáticas con códigos QR, para agilizar la preparación de paquetes.
- **HU08:** Como sistema, automatizar el cambio de estado a "Retiro en oficina" tras dos intentos fallidos de entrega, para cumplir con las políticas de la empresa.

1.4.2 Product Backlog

A partir de las reuniones realizadas con el Product Owner, se obtuvieron los requerimientos funcionales del sistema. Estos requerimientos fueron estructuradas en historias de usuario y después organizadas en un Product Backlog priorizado, el cual guía la planificación de los Sprints.

Tabla 1 Product Backlog

ID	Historia	Prioridad	Tipo
HU01	Visualizar catálogo	Alta	Funcional
HU02	Carrito de compras	Alta	Funcional
HU03	Checkout pedido	Alta	Funcional
HU04	Login admin	Alta	Seguridad
HU05	Dashboard pedidos	Media	Gestión
HU06	Programar entregas	Media	Logística
HU07	QR etiquetas	Baja	Automatización
HU08	Cambio automático de estado	Baja	Regla de negocio

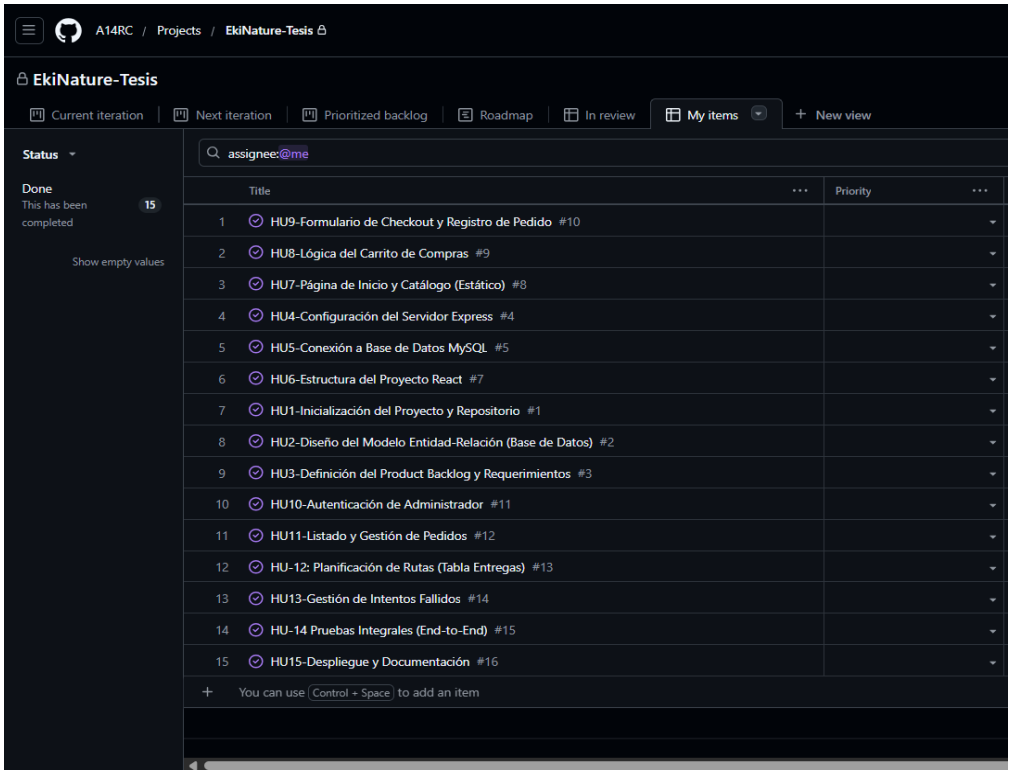


Ilustración 1Product Backlog gestionado durante el desarrollo del proyecto.

1.4.3 Planificación de Sprints

Con base en el Product Backlog priorizado, el desarrollo del sistema se organizó en ciclos cortos de trabajo denominados Sprints. Cada Sprint tiene un conjunto de funcionalidades, lo que permitió entregar partes funcionales del sistema de manera continua y revisar el avance con el Product Owner. Esta forma de trabajo facilitó un progreso gradual, ayudó a detectar errores a tiempo y mantuvo la calidad del software en cada etapa.

Sprint 1 – Configuración y base del sistema

Tabla 2 Planificación del Sprint 1

Actividad	Descripción
Configuración del entorno	Instalación de Node.js, React y dependencias del proyecto
Diseño de base de datos	Creación de tablas principales en MySQL
Estructura backend	Implementación del servidor Express
API de productos	Endpoints CRUD para gestión del catálogo
Pruebas iniciales	Verificación de endpoints con Postman

Sprint 2 – Módulo de compra del cliente

Tabla 3 Planificación del Sprint 2

Actividad	Descripción
Interfaz del catálogo	Desarrollo de vista de productos en React
Carrito de compras	Gestión de productos mediante almacenamiento local

Formulario checkout	Registro de datos del cliente
Validaciones	Control de campos obligatorios
Integración frontend–backend	Conexión de API con la interfaz

Sprint 3 – Gestión de pedidos

Tabla 4 Planificación del Sprint 3

Actividad	Descripción
Registro de pedidos	Persistencia de compras en base de datos
Generación de identificadores	Creación de IDs únicos por transacción
Notificaciones	Envío de confirmación por correo electrónico
Panel administrativo	Visualización de pedidos pendientes
Pruebas funcionales	Validación del flujo completo de compra

Sprint 4 – Logística y control de entregas

Tabla 5 Planificación del Sprint 4

Actividad	Descripción
Gestión de estados	Cambio de estado de pedidos (pendiente, en proceso, entregado, cancelado)
Registro de intentos	Control de número máximo de entregas
Generación de etiquetas QR	Creación automática de códigos para despacho
Reportes básicos	Visualización de ventas en dinero y cantidad.
Ajustes finales	Corrección de errores y optimización del sistema

1.4.4 Casos de uso expandidos

Caso de Uso CU1. Inicio de Sesión Administrativo

Tabla 6 Casos de Uso: Inicio de Sesión Administrativo

Campo	Detalle
Nombre del caso de uso	CU1 – Acceso administrativo
Actor principal	Administrador del sistema
Descripción	El sistema valida las credenciales del administrador y permite el acceso al panel de administrador o dashboard.
Precondiciones	El administrador debe tener una cuenta activa registrada en la base de datos.
Postcondiciones	Se permite el acceso a las funcionalidades protegidas del sistema.
Requisitos no funcionales	RNF01: Protección de credenciales / RNF02: Tiempo de respuesta menor a 2 segundos

Caso de Uso CU2. Registro de Pedido y Notificación

Tabla 7 Casos de Uso: Registro de Pedido Cliente

Campo	Detalle
Nombre del caso de uso	CU2 – Registro de pedido

Actor principal	Cliente
Descripción	El sistema recibe los datos del pedido, registra la transacción y envía una confirmación al cliente.
Precondiciones	El cliente debe tener productos agregados al carrito de compras.
Postcondiciones	Se genera un pedido con identificador único y estado inicial pendiente.
Requisitos no funcionales	RNF01: Persistencia de datos / RNF02: Validación de información obligatoria

Caso de Uso CU3. Gestión de Intentos de Entrega

Tabla 8 Casos de Uso: Control de Intentos Logísticos

Campo	Detalle
Nombre del caso de uso	CU3 – Gestión de intentos fallidos
Actor principal	Administrador / Sistema
Descripción	El sistema registra los intentos de entrega y aplica reglas de negocio cuando se supera el número máximo permitido.
Precondiciones	El pedido debe encontrarse en estado en proceso de entrega.

Postcondiciones	El estado del pedido se actualiza automáticamente al darle presionar la el ícono de etiqueta.
Requisitos no funcionales	RNF01: Integridad de estados / RNF02: Trazabilidad de registros

Caso de Uso CU4. Gestión del Carrito de Compras

Tabla 9 Casos de Uso: Gestión del Carrito de Compras

Campo	Detalle
Nombre del caso de uso	CU4 – Gestión del carrito de compras
Actor principal	Cliente
Descripción	El usuario puede agregar, modificar o eliminar productos antes de confirmar la compra.
Precondiciones	El cliente debe haber ingresado al catálogo de productos.
Postcondiciones	El sistema actualiza el contenido del carrito y recalcula el total de la compra.
Requisitos no funcionales	RNF01: Respuesta rápida del sistema / RNF02: Conservación temporal de la información del carrito

Caso de Uso CU5. Administración y Seguimiento de Pedidos

Tabla 10 Casos de Uso: Administración de Pedidos

Campo	Detalle
Nombre del caso de uso	CU5 – Seguimiento y actualización de pedidos
Actor principal	Administrador

Descripción	El administrador consulta la lista de pedidos y actualiza su estado dentro del flujo logístico.
Precondiciones	El administrador debe haber iniciado sesión correctamente.
Postcondiciones	Los cambios de estado quedan registrados en el historial del sistema.
Requisitos no funcionales	RNF01: Control de acceso / RNF02: Registro histórico de modificaciones

1.5 Infraestructura y viabilidad

1.5.1 Requerimientos de hardware

Tabla 11 Requisitos mínimos de hardware para la implementación del sistema

Componente	Cliente Web	Servidor Backend
Procesador	Intel i3 / Ryzen 3 o superior	Intel i5 / Ryzen 5 o superior
Memoria RAM	4 GB mínimo	8 GB recomendado
Almacenamiento	500 MB libres	SSD con al menos 10 GB
Navegador	Chrome o Firefox actualizado	N/A

1.5.2 Viabilidad técnica y económica

Desde el punto de vista técnico, el proyecto es viable debido al uso de tecnologías consolidadas y ampliamente documentadas como React, Node.js y MySQL, las cuales permiten desarrollar aplicaciones web escalables y mantenibles.

En el aspecto económico, la solución resulta viable al usar herramientas de código abierto y servicios de despliegue gratuitos en la nube, lo que elimina la necesidad de adquirir licencias propietarias o infraestructura física. Esto reduce significativamente los costos iniciales y facilita la adopción del sistema por parte de la empresa.

En resumen, en este capítulo se concretaron las bases teóricas, metodológicas y tecnológicas que mantienen el desarrollo de la plataforma EkiNature. Primero se analizó el problema y sus antecedentes, luego se justificaron las tecnologías elegidas y la metodología ágil que se va a seguir. También se organizaron las historias de usuario, el backlog, la planificación de los Sprints y los casos de uso, que básicamente sirven como guía para saber qué se va a construir y cómo hacerlo.

Todo esto funciona como el punto de partida y la estructura sobre la que se desarrolló el sistema. Con este contexto ya claro, en el siguiente capítulo se explica la parte más técnica, es decir, cómo se construyó la plataforma, su arquitectura, el diseño y la integración de cada uno de sus componentes.

Capítulo II

Construcción del sistema

En este capítulo se describe cómo se realizó la implementación técnica de la plataforma web EkiNature. Se explican las decisiones de diseño tomadas, la arquitectura de software utilizada y las tecnologías que forman parte de la solución. También, se detalla la estructura del sistema tanto en el frontend como en el backend y la base de datos, con el objetivo de asegurar que la aplicación pueda escalar, sea fácil de mantener y esté acorde a las necesidades operativas de la empresa.

Además, se muestran las actividades de desarrollo realizadas, la integración de los distintos módulos, la automatización de procesos y las pruebas de validación aplicadas durante la construcción del sistema, garantizando así un funcionamiento correcto y estable.

2.1 Alcance del Desarrollo

El desarrollo de la plataforma EkiNature se basó en la creación de un sistema web orientado a digitalizar el proceso de compra, venta, registro de pedidos y su posterior despacho.

El alcance incluyó tanto la experiencia del cliente final como las herramientas que necesita la persona encargada para el control interno de las operaciones.

Desde la perspectiva del usuario, se implementaron funcionalidades que permiten la visualización del catálogo de productos, la gestión del carrito de compras y el registro de pedidos a través de formularios. Estas características facilitan la automatización del proceso de compra y reducen la dependencia de los chats y procesos manuales.

A nivel administrativo, el sistema incorpora un panel de control que ayuda la consulta de pedidos, actualización de estados de los productos a despachar, seguimiento de intentos de entrega y generación automática de etiquetas con códigos QR para el despacho. Estas herramientas ayudan a controlar las transacciones de forma más ordenada y a coordinar mejor el trabajo diario.

A nivel técnico, el sistema se desarrolló con una arquitectura cliente–servidor, separando el frontend en React.js, el backend en Node.js con Express y una base de datos MySQL para guardar la información. Además, el código se organizó de forma modular, esto facilita el mantenimiento, permite que el sistema crezca sin problemas y hace más sencillo agregar nuevas funcionalidades en el futuro.

En esta etapa no se incluyó la integración con pasarelas de pago externas ni el desarrollo de aplicaciones móviles nativas. El proyecto se centró únicamente en una solución web responsiva, diseñada para acceder desde navegadores modernos.

2.2 Análisis de Diseño y Experiencia de Usuario (UX)

El diseño de la interfaz de usuario fue creado con el objetivo de garantizar claridad, facilidad de uso y accesibilidad durante la interacción con la plataforma. Para lograr esto, se usó un enfoque minimalista, reduciendo los elementos innecesarios y organizando la información en bloques claros, con el fin de no cansar o aburrir al usuario y facilitar la navegación por la plataforma.

La estructura gráfica se basó en una distribución jerárquica de componentes, usando tarjetas informativas, bloques modulares y secciones claras. Con esto se logra presentar el contenido de manera progresiva, guiando al usuario desde la página principal hacia el catálogo de productos, el carrito de compras y el proceso de registro del pedido, evitando confusiones o pasos redundantes.

Además, se aplicaron principios de usabilidad para que la plataforma sea más clara y fácil de utilizar, manteniendo una apariencia visual permanente, respuestas inmediatas a las acciones del usuario y un diseño responsivo que funcione correctamente tanto en celulares como en computadoras. Estas decisiones permiten una interacción más fluida y ayudan a reducir errores durante el proceso de compra.

Desde el punto de vista estético, se seleccionó una identidad visual alineada con la temática ecológica de la marca, utilizando colores, elementos gráficos minimalistas que ayudan a la percepción de orden y profesionalismo del sistema.

La Figura 2 muestra la interfaz de la página principal (Home), donde se puede observar cómo los componentes están organizados por secciones y cómo la información se distribuye de forma

ordenada, guiando al usuario en su navegación inicial.

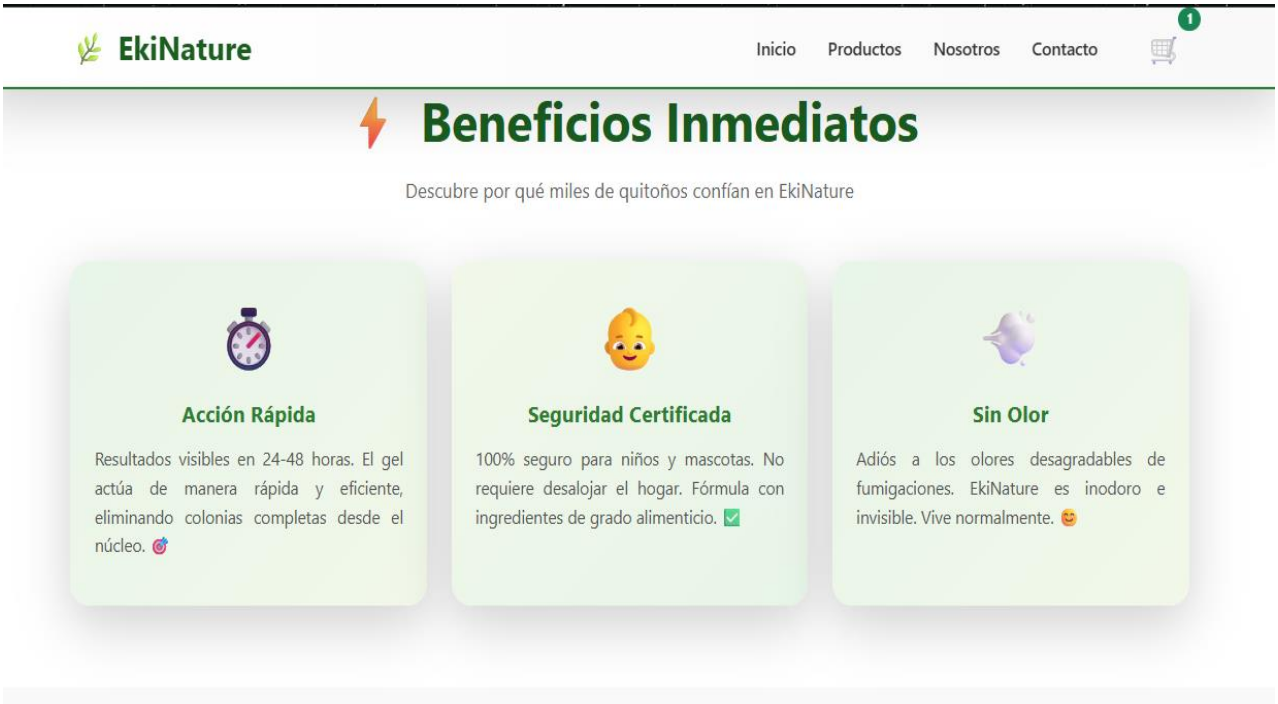


Ilustración 2 Interfaz de la Landing Page de la plataforma EkiNature.

2.3 Estrategia de Diseño de Interfaz (UI)

La interfaz gráfica del sistema fue diseñada para que todas las pantallas mantengan una uniformidad en el uso de colores, mientras que los iconos y botones tienen un mismo estilo, lo que ayuda a una experiencia de usuario clara y consistente.

Se usaron tonalidades verdes y cremas, manteniendo el concepto con sostenibilidad, seguridad y control focalizado en plagas a tratar, valores que representan la identidad de la marca EkiNature. Esta selección ayuda a mantener uniformidad estética en todas las páginas del sistema y refuerza la percepción de profesionalismo de la plataforma.

Los elementos interactivos, tales como botones, formularios, campos de entrada y tarjetas de productos, fueron diseñados bajo un enfoque modular y reutilizable. Esto permitió mantener patrones visuales homogéneos en cada sección del sistema, reduciendo la curva de aprendizaje del usuario y mejorando la eficiencia durante la navegación.

El proceso de registro de pedidos (checkout) se estructuró mediante formularios y validaciones de los campos que rellena el usuario en tiempo real, con el objetivo de prevenir errores de digitación y garantizar que la información ingresada sea la correcta. Para facilitar aún más la

navegación por la página, se aplicó un diseño responsivo, permitiendo una correcta visualización tanto en celulares como en computadoras.

La Figura 3 muestra la interfaz correspondiente al carrito de compras y al flujo de confirmación del pedido, donde se ve de manera clara la disposición de los productos seleccionados, el cálculo automático del total a pagar, incluyendo envío e IVA y la organización del formulario de datos del cliente.

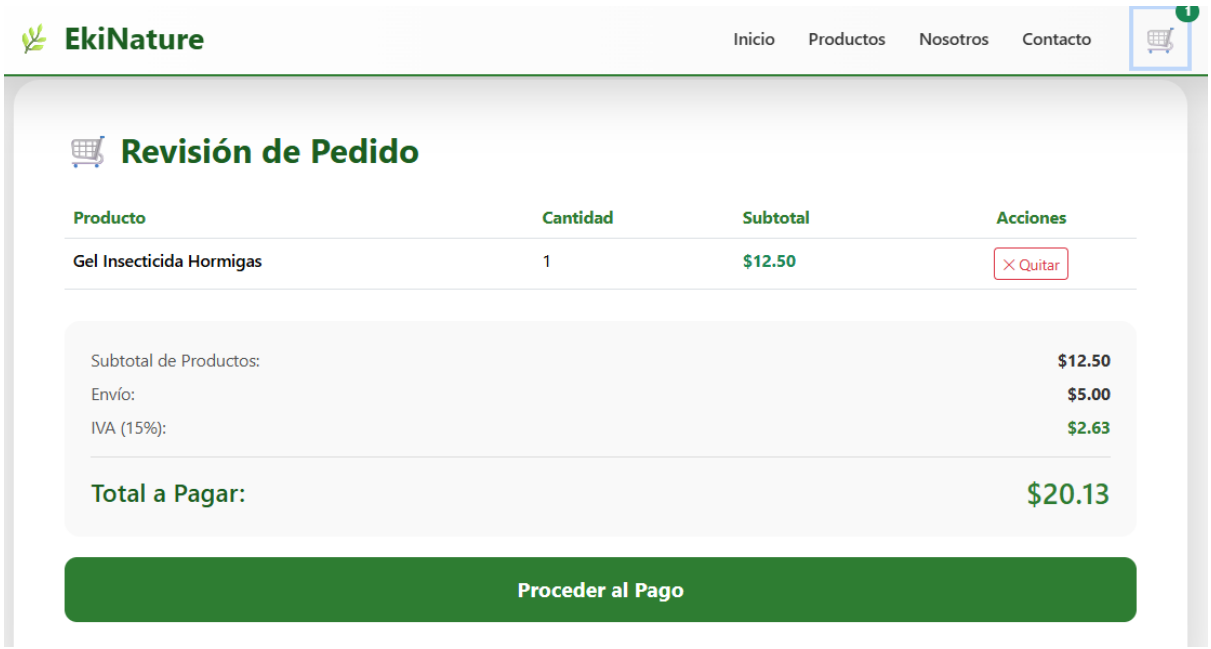


Ilustración 3 Interfaz del carrito de compras y proceso de registro del pedido.

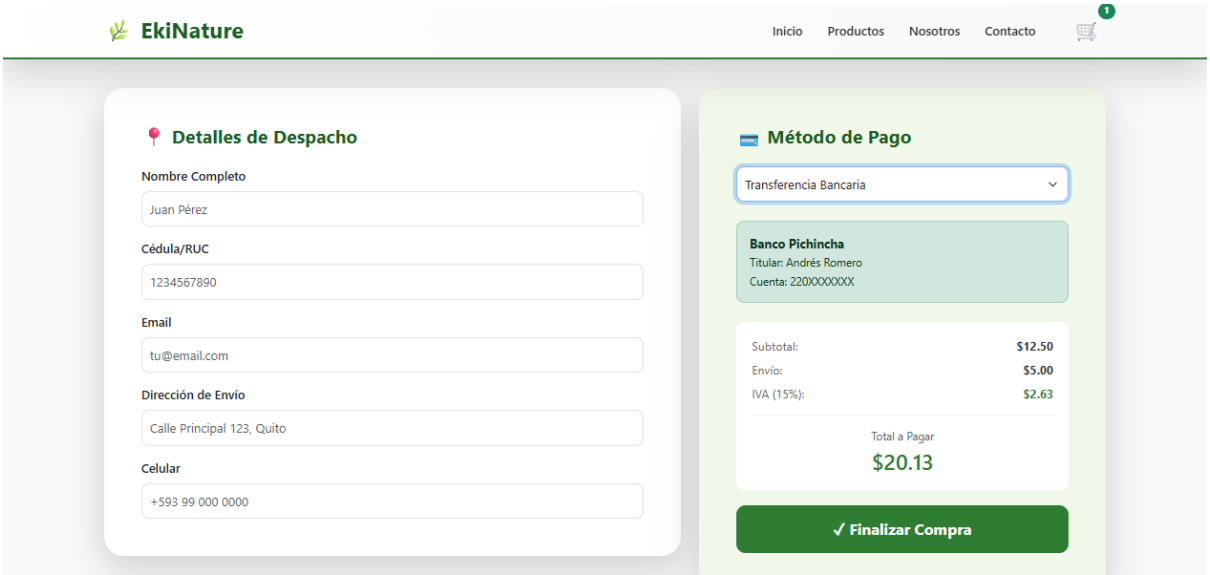


Ilustración 4 Formulario de registro de pedido (checkout)

2.4 Arquitectura del Sistema y Lógica de Negocio

La plataforma EkiNature fue creada con una arquitectura cliente–servidor, separando la capa de presentación, la capa de servicios y la capa de persistencia de datos. Esta arquitectura permitió mantener una estructura modular, facilitando el mantenimiento del código, la escalabilidad del sistema y la incorporación de nuevas funcionalidades sin afectar los componentes que ya existen y funcionan de manera correcta.

En la capa cliente se desarrolló una aplicación web utilizando React.js, responsable de la interacción directa con el usuario. Desde esta interfaz se gestionan las acciones de navegación, selección de productos, administración del carrito de compras y registro de pedidos. El estado temporal del carrito se conserva mediante almacenamiento local del navegador (localStorage), garantizando que la información no se pierda por si el usuario recarga la página o la cierra de manera accidental.

La capa de servicios fue construida con Node.js y el framework Express, implementando una API RESTful para el intercambio de información en formato JSON. Esta API centraliza la lógica del negocio, validando los datos recibidos, procesando las transacciones y controlando las reglas operativas del sistema. Cada módulo fue organizado en controladores independientes para productos, pedidos y gestión logística, aplicando operaciones CRUD para la manipulación de registros.

La persistencia de datos se gestionó mediante MySQL como sistema de base de datos relacional. Las transacciones correspondientes a compra se almacenan de forma estructurada, asegurando integridad referencial entre pedidos, detalles de pedido y usuarios. Esta normalización reduce la redundancia de información y mejora la consistencia de los datos.

En la lógica de negocio se configuraron reglas automáticas que facilitan la gestión del proceso logístico. Una de las reglas destacadas es el control de intentos de entrega, donde el sistema registra envío o despacho y, tras dos intentos fallidos, actualiza el estado del pedido a “Retiro en oficina”, evitando salidas adicionales innecesarias. También, se incorporó la generación de etiquetas con códigos QR e información relevante para la empresa encargada y el responable, permitiendo identificar rápidamente los pedidos durante el despacho.

Esta arquitectura permitió desacoplar responsabilidades entre interfaz, procesamiento y almacenamiento, garantizando un sistema robusto, eficiente y adaptable a futuros requerimientos de escalabilidad.

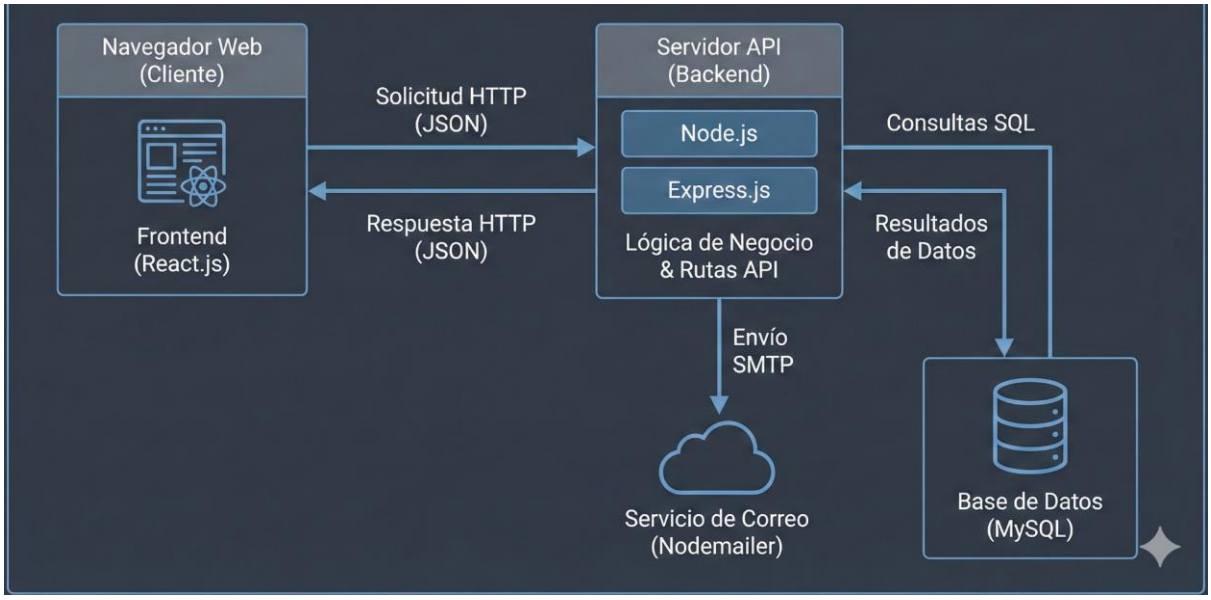


Ilustración 5 Diagrama de arquitectura cliente-servidor y flujo de datos API

2.5 Diseño y Normalización de la Base de Datos

La persistencia de la información de la página EkiNature se implementó mediante un modelo de base de datos relacional utilizando MySQL, seleccionado por su estabilidad, soporte transaccional y compatibilidad con aplicaciones web de arquitectura cliente–servidor. El diseño se orientó a garantizar integridad referencial, consistencia de datos y eficiencia en las consultas.

Durante la etapa de modelado se identificaron las entidades principales del dominio del negocio, considerando el flujo completo de venta, registro de pedidos y gestión logística. A partir de este análisis se definieron tablas independientes y relacionadas, evitando redundancias mediante técnicas de normalización hasta tercera forma normal (3FN).

La entidad **Productos** gurada la información del catálogo comercial, incluyendo nombre, descripción, precio y recursos visuales. Esta tabla permite mantener una gestión centralizada de los artículos disponibles para la venta.

La entidad **Pedidos** registra la información general de cada transacción, como datos del cliente, dirección de entrega, estado logístico y fecha de creación. Cada pedido se asocia a un identificador único que facilita su seguimiento dentro del sistema.

Para representar la relación entre pedidos y productos se implementó la tabla **Detalle_Pedidos**, ésta establece una relación de uno a muchos, permitiendo que un pedido contenga múltiples productos. Esta estructura posibilita almacenar cantidades, precios unitarios y subtotales sin duplicar información del catálogo.

Adicionalmente, se incorporaron tablas complementarias como **Usuarios/Administradores**, destinadas a la autenticación y control de acceso, y **Mensajes_Contacto**, orientada al registro de consultas realizadas por los clientes. Estas entidades contribuyen a mantener separadas las responsabilidades funcionales del sistema.

El uso de claves primarias y foráneas permitió reforzar la integridad referencial entre tablas, asegurando que no existan registros huérfanos ni inconsistencias durante las operaciones de inserción o eliminación. Este enfoque estructurado facilita la escalabilidad del sistema y mejora el rendimiento de las consultas asociadas al procesamiento de pedidos.

En conjunto, el modelo relacional diseñado proporciona una base sólida para soportar el crecimiento de la plataforma, manteniendo la confiabilidad de la información y la correcta gestión de las transacciones comerciales.

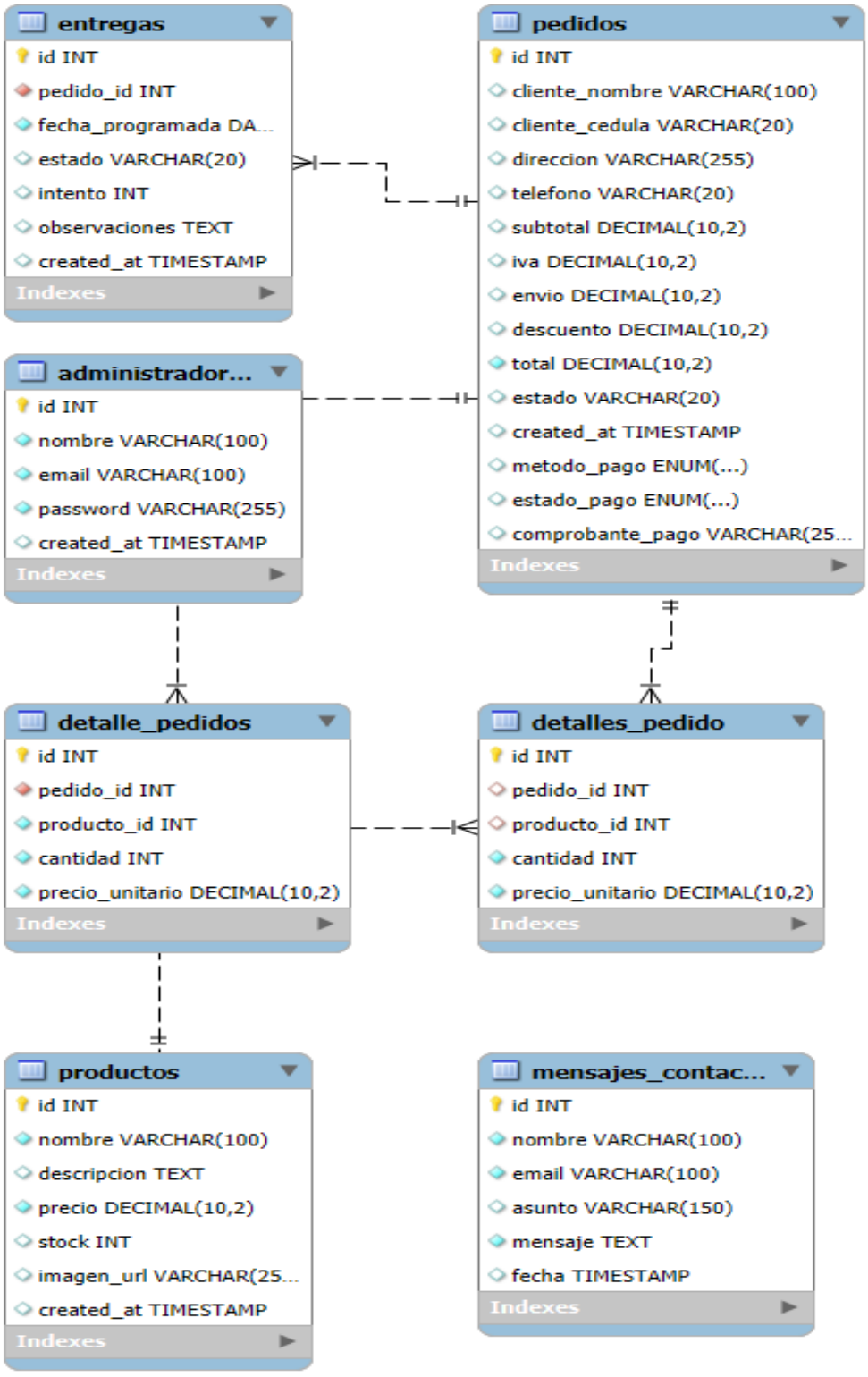


Ilustración 6 Figura 6. Diagrama entidad–relación de la base de datos del sistema EkiNature

2.6 Codificación y Módulos de Automatización

La fase de codificación se realizó siguiendo principios de organización modular y buenas prácticas de programación, asegurando mantenibilidad, escalabilidad y facilidad para ampliaciones del sistema en un futuro. Para lograrlo, se usó una separación clara entre la capa de presentación (frontend), la lógica de negocio (backend) y la persistencia de datos (base de datos).

En el **frontend**, la aplicación fue construida con React.js empleando componentes funcionales y el uso de Hooks, permitiendo gestionar el estado y el comportamiento de los componentes en React, ayudando a que la interfaz se actualice de forma dinámica sin recargar la página. También, se integró almacenamiento local (localStorage) para conservar temporalmente la información del carrito de compras, evitando la pérdida de datos durante la navegación.

En el **backend**, se utilizó Node.js junto con el framework Express para estructurar una API RESTful. Las funcionalidades fueron organizadas en rutas, controladores y middlewares, aplicando operaciones CRUD para la gestión de productos, pedidos y usuarios administrativos. Se incorporaron bloques de control de errores mediante estructuras try/catch, asegurando la estabilidad del sistema ante fallos inesperados.

Adicionalmente, se implementaron **módulos de automatización** orientados a optimizar los procesos de venta, compra y despacho de los productos, entre los que destacan:

- **Dashboard administrativo:** visualización de pedidos, estados y métricas de ventas en tiempo real.

- **Gestión dinámica de estados:** transición automática entre estados como pendiente, en proceso, entregado o cancelado.

- **Generación de etiquetas QR:** creación automática de códigos QR con la información del destinatario para agilizar la preparación de envíos.

- **Validaciones de integridad:** comprobación de datos obligatorios en formularios antes de registrar transacciones.

Estas automatizaciones reducen la intervención manual, minimizan errores operativos y mejoran la eficiencia del flujo logístico, aportando mayor confiabilidad al sistema.

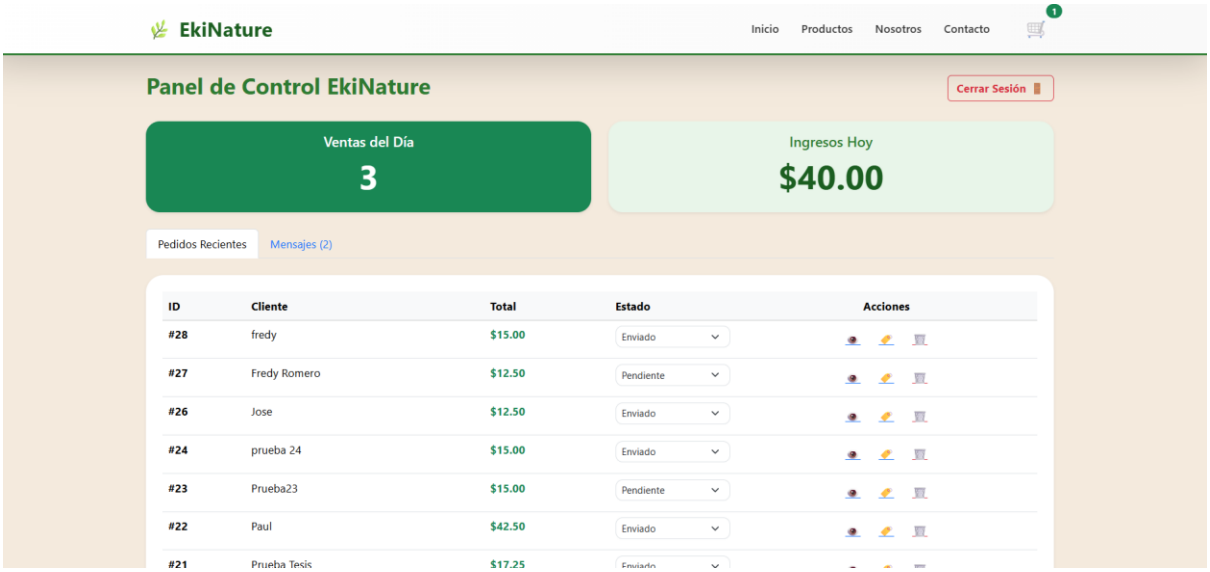


Ilustración 7 Panel de control administrativo con métricas e indicadores de gestión.

2.7 Pruebas y Validación

Con el fin de garantizar el correcto funcionamiento de la plataforma EkiNature, se ejecutó un proceso de verificación integral con el fin de validar la estabilidad del sistema, la consistencia de los datos y la experiencia del usuario final. Las pruebas se realizaron de manera progresiva durante cada Sprint, esto permite detectar y corregir errores en etapas tempranas del desarrollo.

Se aplicaron tres niveles principales de evaluación:

Pruebas unitarias.

Se comprobaron de forma individual los controladores del backend y las funciones críticas del sistema, como el registro de pedidos, el cálculo automático de totales, la validación de campos obligatorios y la actualización del estado de los pedidos (pendiente, enviado, entregado o cancelado) Estas pruebas permitieron asegurar que cada módulo funcione correctamente de manera independiente.

Pruebas de integración.

Se verificó la comunicación entre el frontend, la API REST y la base de datos MySQL. En esta etapa se validó el flujo completo de la compra, empezando desde la selección de productos hasta la persistencia final del pedido. También, se realizaron pruebas de endpoints mediante

herramientas como Postman para confirmar respuestas correctas del servidor y tiempos de respuesta adecuados.

Pruebas de usuario.

Se evaluó la interacción real con la plataforma simulando escenarios operativos diarios. Se comprobó la facilidad de navegación en el catálogo, la correcta gestión del carrito de compras, el registro exitoso de pedidos y la actualización de estados desde el panel administrativo. También se validó la generación automática de etiquetas con códigos QR y su lectura por parte de la persona encargada.

Los resultados obtenidos evidenciaron la estabilidad y funcionalidad, tiempos de respuesta aceptables y la reducción significativa de errores manuales en los procesos de registro y despacho. Confirmando que el sistema es apto para su despliegue en un entorno productivo.

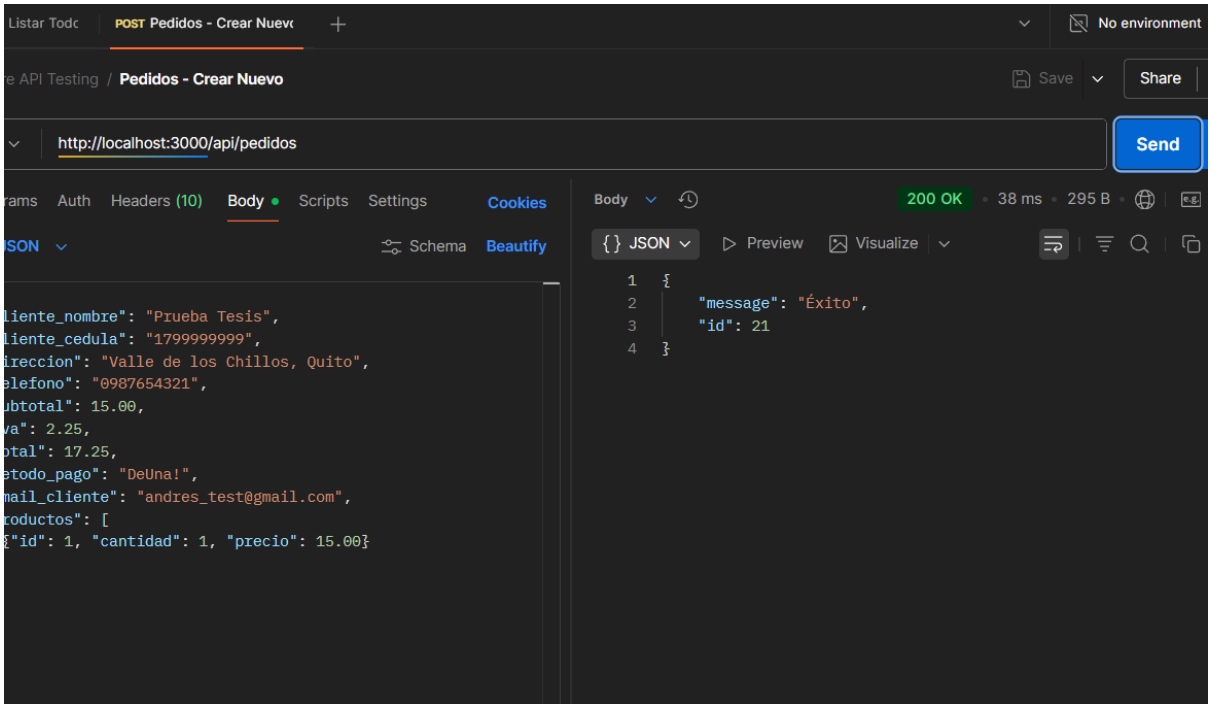


Ilustración 8 Evidencia de pruebas funcionales del sistema y ejecución de endpoints durante la validación.

En resumen, la construcción del sistema permitió dar vida a los requerimientos levantados durante la fase de análisis, integrando una arquitectura web moderna, una base de datos relacional normalizada y módulos de automatización orientados a optimizar los procesos de información, venta y despacho de los productos y servicios ofrecidos por la empresa. La validación funcional demostró que la plataforma opera de forma estable y eficiente,

constituyendo una solución tecnológica viable para la digitalización de las operaciones de EkiNature.

Capítulo III:

Pruebas y estabilización

El presente capítulo describe el proceso de validación funcional, pruebas técnicas y actividades de estabilización realizadas sobre la plataforma EkiNature una vez concluida su construcción. Se detallan las revisiones con el Product Owner, el entorno de despliegue, las pruebas ejecutadas en frontend y backend, así como las acciones de optimización implementadas para garantizar la confiabilidad, integridad y correcto desempeño del sistema en un entorno de producción real.

3.1 Reuniones con el Product Owner

En el desarrollo de la plataforma EkiNature, la figura del *Product Owner* (PO) desempeñó un rol fundamental como puente entre las necesidades comerciales de la marca y las especificaciones técnicas. El PO se aseguró de que las funcionalidades, especialmente el módulo de logística y la pasarela de pagos, cumplieran con los objetivos de control y eficiencia de la empresa.

La distribución del equipo de trabajo se definió de la siguiente manera:

- **Product Owner:** Fredy Romero Paredes
- **Equipo de Desarrollo (Fullstack):** Andrés Romero

Se mantuvieron reuniones por videollamada para solventar observaciones sobre la disposición visual de las tarjetas de productos, la legibilidad de las etiquetas de envío y el flujo del carrito de compras.



Ilustración 9 Sesión de revisión de requerimientos

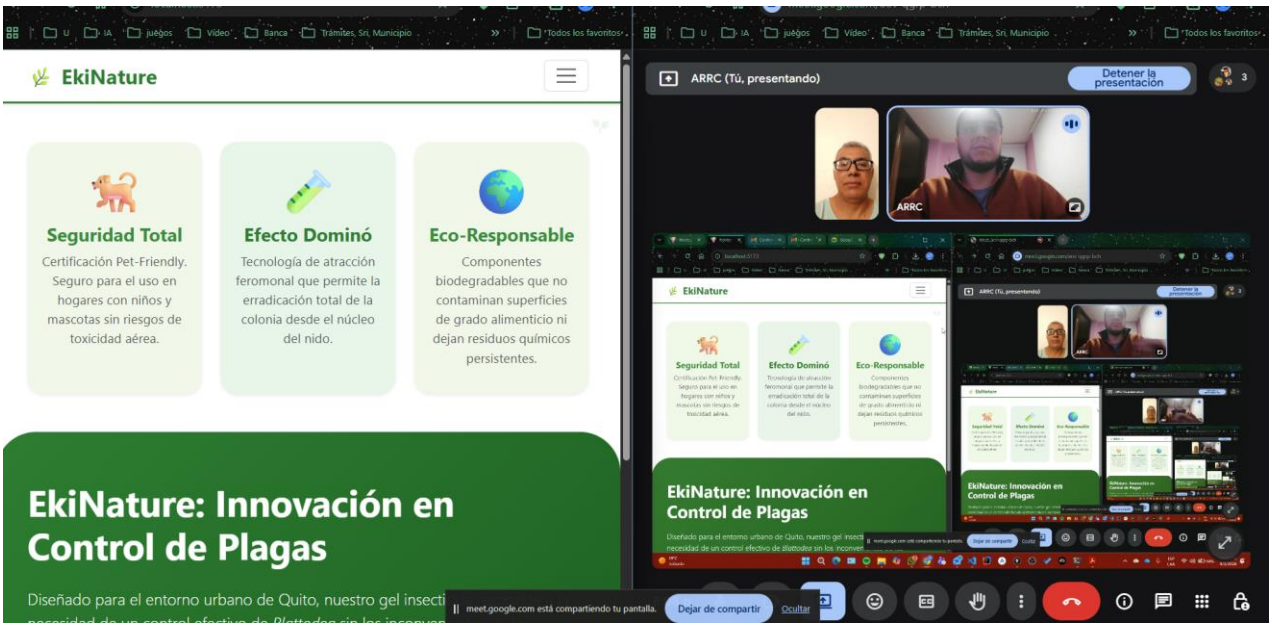


Ilustración 10 Figura 10 Validación de avances con el Product Owner.

3.2 Entorno de Pruebas y Despliegue

El sistema fue migrado desde un entorno de desarrollo local hacia la nube, con el fin de validar su funcionamiento en condiciones reales de red. Para ello, se utilizó Vercel para el alojamiento del frontend y Render para desplegar el backend, mientras que la base de datos MySQL se gestionó mediante un servicio administrado en Clever Cloud. Esta configuración permitió contar con una URL pública accesible, facilitando las pruebas con usuarios finales.

3.2.1 Proceso de Pruebas en el Ciclo de Desarrollo

Para dar por finalizada una funcionalidad, debía cumplir con los siguientes criterios.

- **Validación de Endpoints (Backend):** Se utilizó **Postman** para verificar que las rutas de la API (como /api/pedidos) respondieran con los códigos HTTP correctos (200 OK, 201 Created).

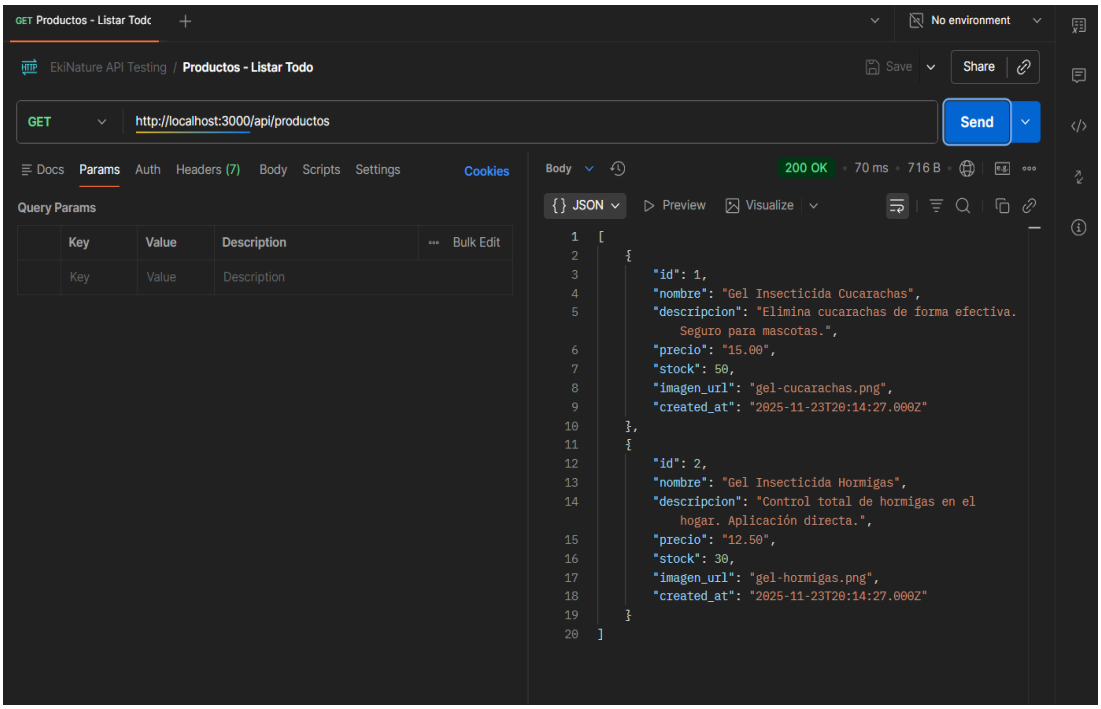


Ilustración 11 Validación de respuesta de la API de productos en Postman

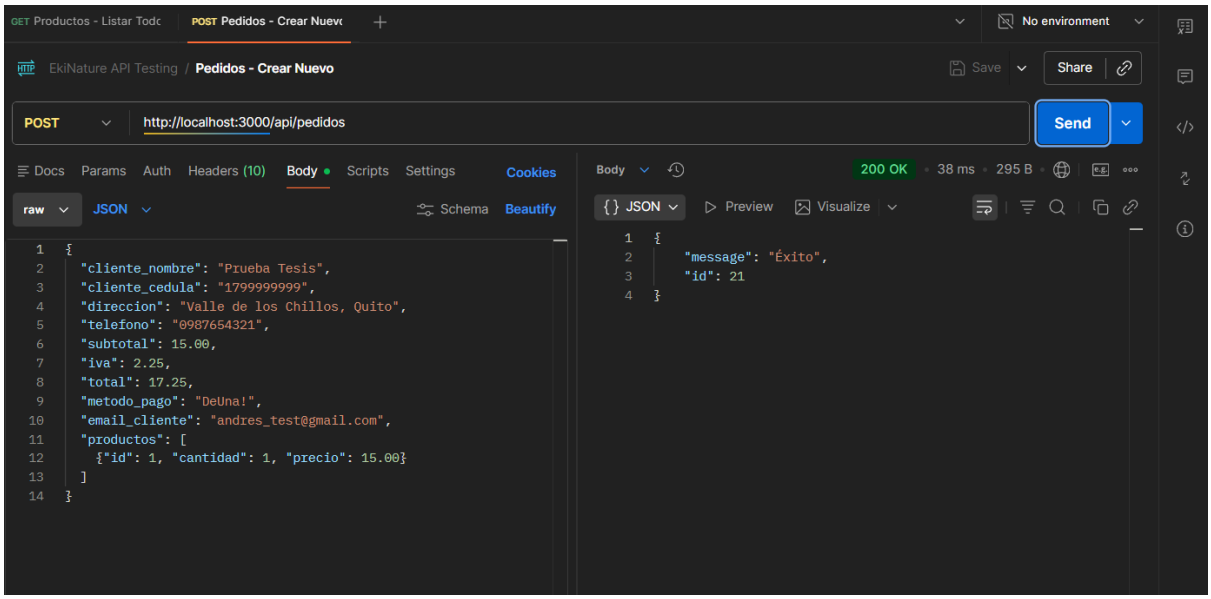


Ilustración 12 Evidencia de persistencia de datos y asignación de ID transaccional mediante método POST.

- **Validación Funcional (Frontend):** Se comprobó manualmente que al presionar o accionar el botón de "Imprimir Etiqueta", se generara la etiqueta con el código QR y los datos del cliente.

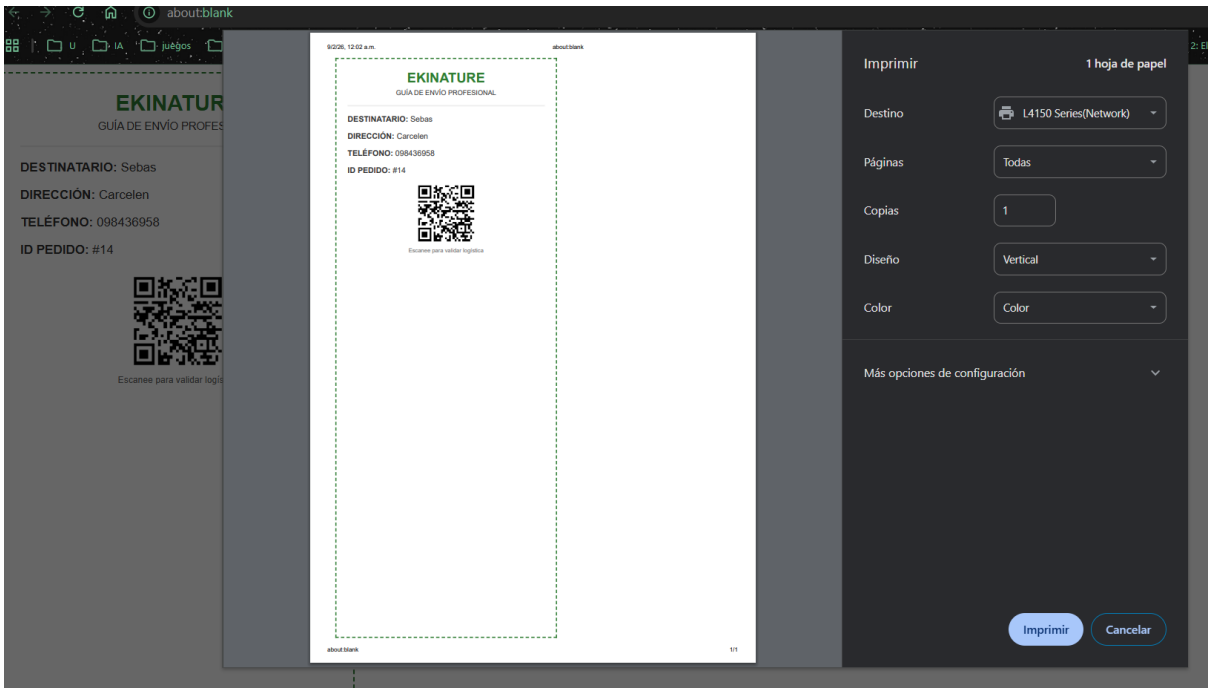


Ilustración 13 Ejecución de prueba del módulo de generación de etiquetas logísticas.

- **Integración y Persistencia:** Se verificó que la información ingresada en el formulario de contacto se guarde de manera correctamente en MySQL y envíe la alerta por correo electrónico.

```
----------------------------------------
🚀 Servidor EkiNature corriendo en puerto 3000
----------------------------------------
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
[dotenv@17.2.3] injecting env (6) from .env -- tip: 🔐 prevent committing .env to code: https://dotenvx.com/precommit
[dotenv@17.2.3] injecting env (0) from .env -- tip: ⚙️ write to custom object with { processEnv: myObject }
----------------------------------------
🚀 Servidor EkiNature corriendo en puerto 3000
----------------------------------------
Correo enviado con éxito: 250 2.0.0 OK  1770611661 a1e0cc1a2514c-949014a724csm2919321241.2 - gsmtp
```

Ilustración 14 Monitoreo de logs del servidor confirmando el envío de notificaciones SMTP.

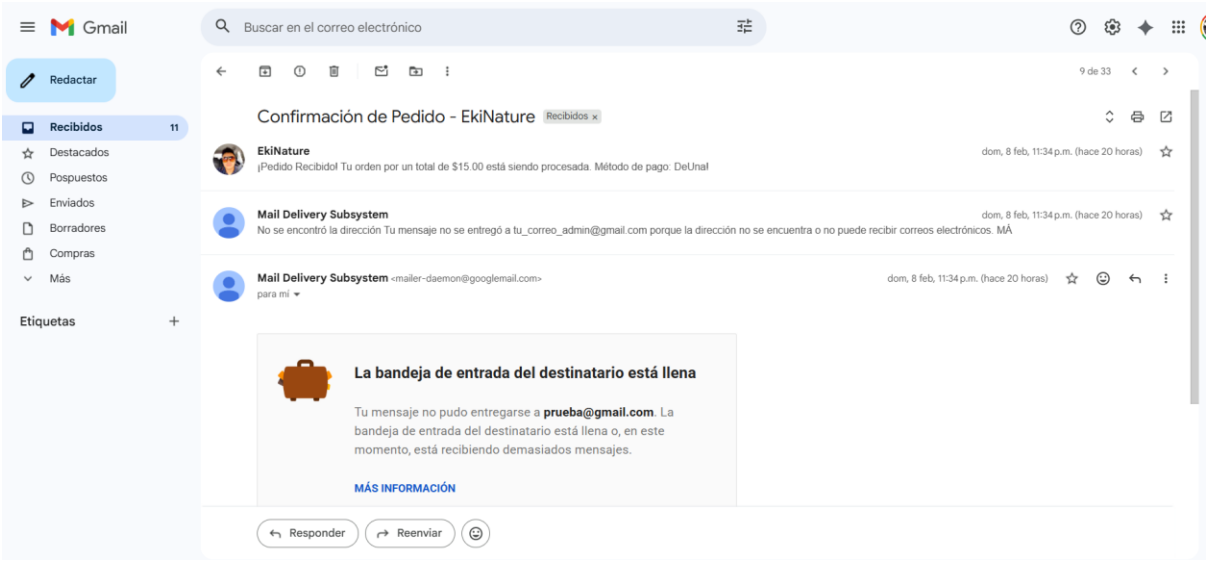


Ilustración 15 Verificación en mail de administrador

3.3 Tipos de Pruebas Realizadas

3.3.1 Pruebas de Frontend (Interfaz)

- **Funcionales:** Verificación de botones, validación de formularios de pago y navegación del menú.
- **Usabilidad:** Comprobación de la estética "naturaleza" (tonos verdes) y buena visibilidad en los mensajes de confirmación de pedido.

3.3.2 Pruebas de Backend (Servidor y Datos)

- **Base de Datos:** Se verificó la integridad referencial en las tablas *pedidos* y *detalle_pedidos.*
- **Integración:** Se comprobó la correcta entre controladores y el servicio de mensajería **Nodemailer**.

3.4 Herramientas Utilizadas

- **Postman:** Para probar la lógica RESTful y respuestas JSON.
- **MySQL Workbench:** Para el monitoreo de la persistencia de datos financieros.
- **Consola de Node.js:** Para la corrección de errores en tiempo real y seguimiento de peticiones.

3.5 Planilla de Pruebas por Sprint

Tabla 12 Resumen de resultados de las pruebas funcionales y de integración por Sprint

Sprint	Tarea	Descripción	Tipo de Prueba	Resultado Esperado	Estado
S1	Configuración de la DB	Migración de tablas pedidos y contacto.	Base de Datos	Estructura SQL sin errores en Clever Cloud.	Exitoso
S2	Autenticación	Acceso administrativo.	Funcional	Bloqueo de rutas no autorizadas.	Exitoso
S3	Checkout	Integración de pagos y selección QR.	Integración	Persistencia del metodo_pago en la DB.	Exitoso
S4	Logística	Generación de etiqueta con QR.	Automatización	Renderizado de PDF con datos dinámicos.	Exitoso

3.6 Estabilización Final

Después de ejecutar las pruebas, se realizó una fase de ajustes y mejoras técnicas con el objetivo de asegurar la estabilidad del sistema y la consistencia de la interfaz. Durante esta etapa se corrigieron pequeños errores y se optimizó el flujo de datos, permitiendo que la aplicación funcione correctamente en condiciones reales de red (entornos de producción).

3.6.1 Optimización de la Interfaz y Coherencia Visual

Se configuraron estilos globales con el fin de unificar el uso de los tonos verdes (#2e7d32) en toda la plataforma. Esto permitió mantener una identidad visual igual en todos los módulos, desde la página de inicio hasta el panel del administrador. Además, se comprobó que los componentes de Bootstrap conservaran su diseño responsivo en distintas resoluciones de pantalla, evitando desplazamientos horizontales innecesarios en celulares o tablets.

3.6.2 Estabilidad del Backend y Manejo de Errores

Uno de los puntos principales fue la reorganización y mejora de la lógica de los controladores en Node.js. Se implementaron bloques try/catch para manejar los errores de forma controlada, especialmente en la comunicación con la base de datos MySQL y en el envío de correos mediante Nodemailer.
Así, cuando ocurre un fallo inesperado, el servidor puede responder con mensajes claros y controlados, evitando que la API se detenga y facilitando la identificación del problema a través de los logs del sistema.

```
catch (err) {
    console.error("Error detallado en la creación del pedido:", err);
    res.status(500).json({
        error: "No se pudo procesar la orden de compra",
        detalle: err.message
    });
}
```

Ilustración 16 Implementación de bloques de control de excepciones para asegurar la estabilidad del servidor.

3.6.3 Pruebas del Sistema de Envíos

Finalmente, se realizaron pruebas en condiciones reales para verificar que la automatización del sistema de envíos funcione de manera correcta. Se generaron etiquetas de envío con códigos QR y se comprobó que su tamaño fuera el ideal para su lectura con escáneres y cámaras de celular o tablets. Además, mediante un celular, se validó que la dirección y los datos de contacto

del cliente se obtuvieran correctamente desde el código, reduciendo errores humanos en la planificación de las entregas.

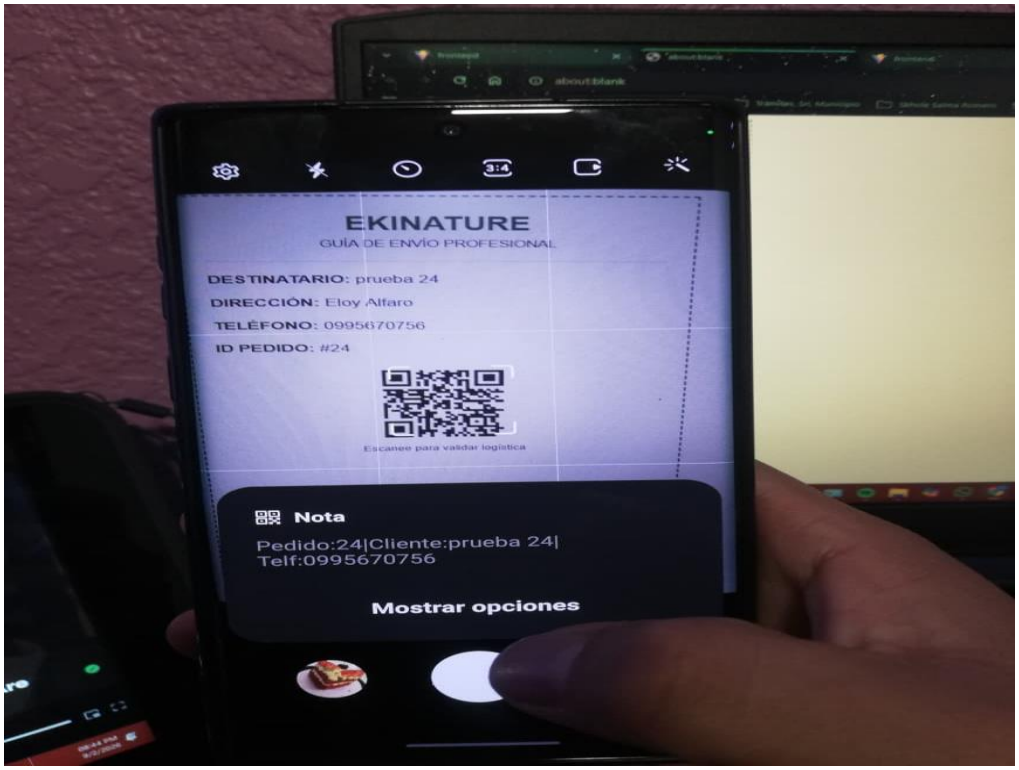


Ilustración 17 Prueba de legibilidad del código QR logístico mediante escaneo con dispositivo móvil.

En resumen, las pruebas realizadas confirmaron que la plataforma EkiNature funciona de manera estable y confiable, tanto en la interfaz de usuario como en el manejo interno de datos. La revisión de los servicios en la nube y el seguimiento de la comunicación entre las distintas partes del sistema mostraron un buen desempeño en escenarios reales. Con base en estos resultados, se concluye que la solución está preparada para ser implementada en producción.

Conclusiones

Análisis de requerimientos y robustez del sistema.

El levantamiento de información permitió identificar la necesidad de una plataforma que centralice y ayude a la automatización de los procesos de venta, compra y despacho de los productos y servicios de la empresa EkiNature. La solución desarrollada creó un sistema web confiable, seguro y fácil de usar, garantizando la integridad de las transacciones y mejorando el control operativo del negocio.

Optimización logística mediante automatización.

La incorporación de funciones automatizadas, como las notificaciones por mail y la generación de etiquetas con códigos QR, disminuyó significativamente la gestión manual en el despacho de pedidos. Estas mejoras ayudaron a la eficiencia del proceso de entrega del producto, reduciendo errores humanos y tiempos de etiquetado manual.

Viabilidad técnica y despliegue en la nube

Se puso a prueba el sistema para asegurar que todos los componentes funcionen correctamente. Después se configuró y publicó la aplicación en la nube, permitiendo su ejecución en producción con conexión estable entre frontend, backend y la base de datos.

Recomendaciones

Integrar pasarela de pagos electrónica, permitiendo el pago mediante tarjeta de crédito para automatizar el proceso de cobro y completar el ciclo de venta en línea.

Como mejora futura, se recomienda implementar un dominio propio y habilitar conexión segura HTTPS mediante certificados SSL/TLS. Para proteger la información de los usuarios, brindar mayor confianza y mostrar una imagen más profesional de la plataforma o sistema.

Como medida preventiva, se pueden configurar respaldos automáticos y revisiones constantes de la base de datos para proteger la información, garantizando así que el sistema continúe funcionando correctamente ante cualquier posible fallo.

Referencias

Banks, A. (2018). *Learning React: Functional Web Development with React and Redux* (2nd ed.). O'Reilly Media.

Denso Wave Incorporated. (2024). *QR Code features: Standards and capabilities*. https://www.qrcode.com/en/about/features.html

Fielding, R. T., & Taylor, R. N. (2002). *Principled design of the modern web architecture*. University of California, Irvine.

Laudon, K. C., & Traver, C. G. (2020). *E-commerce 2020: Business, technology and society* (16th ed.). Pearson.

Myers, G. J., Sandler, C., & Badgett, T. (2011). *The art of software testing* (3rd ed.). John Wiley & Sons.

MySQL AB. (2020). *MySQL 8.0 reference manual*. https://dev.mysql.com/doc/refman/8.0/en/

Node.js Foundation. (2024). *Introduction to Node.js*. https://nodejs.org/en/learn/getting-started/introduction-to-nodejs

Postman. (2024). *Postman learning center: Introduction to Postman and API testing*. https://learning.postman.com/docs/getting-started/introduction/

Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide*. Scrum.org.

https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Spanish.pdf

Sommerville, I. (2016). *Software engineering* (10th ed.). Pearson.