UNIVERSIDAD DE EL SALVADOR

FACULTAD MULTIDISCIPLINARIA ORIENTAL

DEPARTAMENTO DE INGENIERÍA Y ARQUITECTURA

SECCIÓN DE INGENIERIA EN SISTEMAS INFORMÁTICOS

INFORME FINAL DE TRABAJO DE GRADO:

MODALIDAD PASANTÍA DE PRÁCTICA PROFESIONAL

TÍTULO DEL PLAN:

DESARROLLO E INTEGRACIÓN DE SOLUCIONES DE SOFTWARE

PARA OPTAR AL GRADO ACADÉMICO DE:

INGENIERO DE SISTEMAS INFORMÁTICOS

PRESENTADO POR:

ROBERTO HERNÁN LAÍNEZ TREJO Nº CARNET LT20007

DOCENTE ASESOR:

ING. JOSÉ ALEXANDER GUANDIQUE FLORES

NOVIEMBRE 2025

SAN MIGUEL, EL SALVADOR, CENTROAMÉRICA

UNIVERSIDAD DE EL SALVADOR

AUTORIDADES

MSC. JUAN ROSA QUINTANILLA

RECTOR

DRA. EVELYN BEATRIZ FARFÁN

VICERRECTORA ACADÉMICA

MSC. ROGER ARMANDO ARIAS ALVARADO

VICERRECTOR ADMINISTRATIVO

LIC. PEDRO ROSALIO CASTANEDA

SECRETARIO GENERAL

LIC. CARLOS AMILCAR SERRANO RIVERA

FISCAL GENERAL

UNIVERSIDAD DE EL SALVADOR

FACULTAD MULTIDISCIPLINARIA ORIENTAL

AUTORIDADES

MCS. CARLOS IVÁN HERNÁNDEZ FRANCO

DECANO

DRA. NORMA AZUCENA FLORES RETANA

VICEDECANA

LIC. CARLOS DE JESÚS SÁNCHEZ

SECRETARIO

ING. JOSÉ LUIS CASTRO CORDERO

JEFE DEL DEPARTAMENTO DE INGENIERÍA Y ARQUITECTURA

ING. MILAGRO ALICIA GONZÁLEZ DE REYES

COORDINADORA DE LOS PROCESOS DE GRADO DE LA CARRERA DE INGENIERÍA DE SISTEMAS INFORMÁTICOS

Índice

Indice de Tablas

Índice de Figuras

Resumen

La presente pasantía profesional tuvo como **objetivo** aplicar y fortalecer los conocimientos adquiridos en la carrera de Ingeniería de Sistemas Informáticos mediante el desarrollo e integración de soluciones de software en un entorno empresarial real. La pasantía se desarrolló de forma individual en la empresa RT Network S.A.S. de C.V., ubicada en San Francisco Gotera, Morazán, durante el período comprendido del 14 de abril al 26 de octubre de 2025.

La **metodología** empleada se basó en un enfoque de desarrollo frontend moderno utilizando React 19 y TypeScript, bajo una arquitectura basada en componentes e integración mediante consumo de APIs REST, desarrolladas en paralelo por el equipo backend de la empresa. Se aplicaron técnicas de diseño UX/UI, validaciones de datos del lado del cliente y control de versiones con Git/GitHub.

Como **resultados**, se logró la implementación de módulos CRUD completamente funcionales para la gestión de productos, usuarios, proveedores, cajas, categorías y subcategorías, así como la integración exitosa con el backend mediante APIs REST. Adicionalmente, se desarrollaron prototipos de diseño UX/UI y se brindó soporte técnico en la instalación y mantenimiento de sistemas de videovigilancia CCTV e IP.

Las **conclusiones** evidencian que los objetivos planteados fueron cumplidos en su totalidad, obteniéndose un frontend moderno, funcional y alineado a las necesidades de la empresa. La pasantía permitió fortalecer competencias técnicas en desarrollo web moderno, integración de sistemas y diseño centrado en el usuario, aportando valor tanto a la formación profesional del estudiante como al proceso de modernización tecnológica de la empresa.

Palabras clave: desarrollo frontend; React; TypeScript; API REST; UX/UI; módulos CRUD; videovigilancia.

Abstract

The objective of this professional internship was to apply and strengthen the knowledge acquired during the Computer Systems Engineering program through the development and integration of software solutions in a real business environment. The internship was carried out individually at RT Network S.A.S. de C.V., located in San Francisco Gotera, Morazán, during the period from April 14 to October 26, 2025.

The methodology was based on a modern frontend development approach using React 19 and TypeScript, following a component-based architecture and integration through REST APIs developed in parallel by the company's backend team. UX/UI design techniques, client-side data validation, and version control using Git/GitHub were applied.

The results include the successful implementation of fully functional CRUD modules for the management of products, users, suppliers, cash registers, categories, and subcategories, as well as proper integration with the backend through REST APIs. Additionally, UX/UI design prototypes were developed, and technical support was provided for the installation and maintenance of CCTV and IP video surveillance systems.

The conclusions indicate that the established objectives were fully achieved, resulting in a modern, functional frontend aligned with the company's needs. This internship strengthened technical skills in modern web development, system integration, and user-centered design, contributing to both professional training and the company's technological modernization process.

Keywords: frontend development; React; TypeScript; REST API; UX/UI; CRUD modules; video surveillance.

Introducción

La presente pasantía profesional se realizó de forma individual en RT Network S.A.S. de C.V., una empresa dedicada al desarrollo de software, instalación de sistemas de videovigilancia y servicios de seguridad electrónica. El propósito principal de la pasantía fue aplicar los conocimientos adquiridos durante mi formación en la carrera de Ingeniería de Sistemas Informáticos, contribuyendo simultáneamente al fortalecimiento tecnológico de la empresa mediante el desarrollo, rediseño e integración de soluciones informáticas modernas.

La justificación de este proceso formativo se estructura en tres dimensiones.
A nivel profesional, la pasantía representó una oportunidad para adquirir experiencia real en el desarrollo de interfaces web modernas, diseño UX/UI, consumo de APIs REST, gestión de infraestructura tecnológica y soporte técnico especializado, habilidades altamente demandadas en el entorno laboral actual.
A nivel institucional, esta modalidad de graduación permitió a la Universidad de El Salvador cumplir con su misión de formar profesionales competentes y capaces de integrarse de manera efectiva en el sector productivo del país.
A nivel empresarial, mi incorporación contribuyó al fortalecimiento de las dos áreas estratégicas de RT Network: el desarrollo de software y los servicios de videovigilancia, apoyando su proceso de modernización digital y su competitividad en el mercado salvadoreño.

La importancia de esta pasantía radica en que permitió intervenir en un entorno donde convergen dos líneas de trabajo clave para la empresa: la migración parcial del sistema interno hacia tecnologías modernas y la implementación de infraestructura de seguridad electrónica. En el área de software, participé en la construcción de un nuevo frontend en React 19 y TypeScript, mientras el backend se desarrollaba en paralelo por el equipo interno. Esto permitió abordar problemáticas previamente identificadas en el sistema existente, tales como la obsolescencia tecnológica, deficiencias en la experiencia de usuario y un deficiente flujo de datos entre módulos. En el área técnica, la pasantía facilitó reforzar los conocimientos prácticos mediante la instalación, mantenimiento y configuración de sistemas de videovigilancia IP y CCTV.

Objetivos

Objetivo General:

Aplicar y fortalecer los conocimientos teóricos adquiridos a lo largo de la carrera en un entorno empresarial real, desarrollando e integrando soluciones de software a través de una experiencia de aprendizaje práctico.

Objetivos Específicos:

- **Desarrollo frontend moderno:** desarrollar interfaces web funcionales, responsivas y accesibles utilizando React 19 y TypeScript, garantizando una experiencia de usuario fluida en dispositivos de escritorio.
- **Implementación de módulos CRUD:** implementar módulos completos de creación, lectura, actualización y eliminación (CRUD) para la gestión integral de inventario, ventas y usuarios, integrados correctamente con el backend.
- **Infraestructura de seguridad:** instalar y mantener sistemas de videovigilancia CCTV/IP, garantizando la operatividad continua de la infraestructura de seguridad de los clientes.
- Soporte técnico: Optimizar el funcionamiento de equipos informáticos mediante mantenimiento preventivo y correctivo.

Información de la empresa RT Network S.A.S. DE C.V.

Datos generales

Localización

RT Network S.A.S. de C.V. se encuentra ubicada en Plaza Gloria, 1.ª Calle Poniente, local #1, San Francisco Gotera, departamento de Morazán, El Salvador.

Antecedentes

RT Network S.A.S. de C.V. es una empresa salvadoreña fundada en el año 2022 como respuesta a la creciente demanda de soluciones tecnológicas integrales en la zona oriental del país. Su creación surge con la visión de ofrecer servicios especializados que integraran el desarrollo de software con la implementación de sistemas de seguridad electrónica, dos áreas esenciales para el fortalecimiento tecnológico y la protección empresarial en la región.

Desde sus inicios, la empresa identificó una oportunidad significativa en la provisión de servicios tecnológicos profesionales y de calidad, orientados especialmente a pequeñas y medianas empresas. Con este enfoque, RT Network estableció dos líneas de negocio principales:

- **Sistemas de videovigilancia CCTV e IP, incluyendo diseño, instalación, configuración y mantenimiento.**
- **Desarrollo de software empresarial**, con énfasis en sistemas de inventario, facturación electrónica y soluciones administrativas personalizadas.

Durante sus primeros años de operación, RT Network ha mostrado un crecimiento sostenido, consolidándose como un proveedor confiable en el departamento de Morazán y extendiendo gradualmente sus servicios a otras zonas del país. Su estrategia se ha fundamentado en el uso de tecnología moderna, el respaldo profesional y la calidad en la prestación de servicios.

La misión de la empresa es ofrecer soluciones tecnológicas con altos

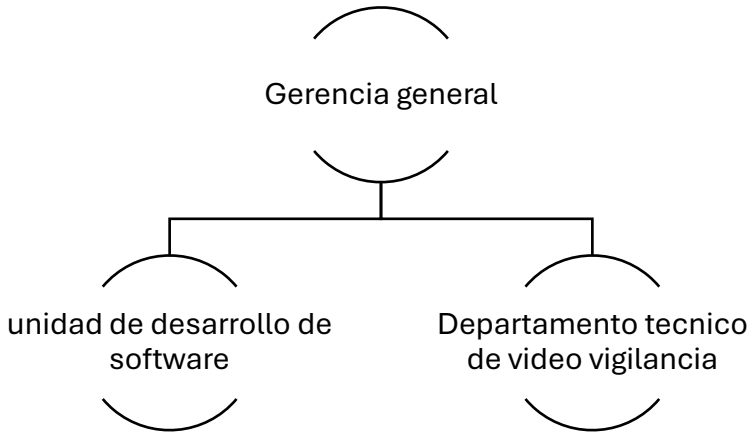
estándares de calidad, formadas por capital humano calificado y respaldo técnico continuo. Su visión es posicionarse como la empresa líder en servicios integrales de facturación electrónica, redes informáticas y videovigilancia CCTV/IP. Los valores institucionales que guían sus operaciones son: profesionalismo, integridad, compromiso, calidad y satisfacción del cliente.

Estructura Organizacional de RT Network

RT Network S.A.S. de C.V. cuenta con una estructura organizacional compacta, diseñada para asegurar eficiencia operativa y una comunicación directa entre sus áreas funcionales. La empresa se organiza de la siguiente manera:

Organigrama General de la Empresa

Figura 1. Organigrama de RT Network S.A.S. DE C.V.



Ubicación del Área de Programación dentro de la Empresa

La Unidad de Desarrollo de Software forma parte directa de la Gerencia General y constituye una de las áreas estratégicas de la empresa. Sus principales funciones son:

- Análisis y diseño de sistemas.
- Desarrollo de aplicaciones web.
- Integración con APIs y servicios internos.
- Pruebas de funcionalidad.

Dentro de esta unidad se desempeñó el estudiante, específicamente como desarrollador frontend, trabajando de manera coordinada con el desarrollador backend y reportando al responsable de la unidad.

Funciones Principales del Área de Desarrollo de Software

- Desarrollar y mantener sistemas de inventario, ventas y facturación.
- Implementar mejoras tecnológicas y rediseño de módulos.
- Crear interfaces modernas basadas en patrones UX/UI.

Recursos

Instalaciones y Equipo

Mobiliario

Tabla .1 Mobiliario asignado en la unidad tecnológica

Mobiliario	Cantidad	Descripción
Escritorio	1	Proporcionado por la empresa
Silla	1	Proporcionado por la empresa
Herramientas varias	1	Incluye herramientas básicas utilizadas en instalaciones y mantenimiento técnico.
Equipo de instalación	1	Materiales y equipo utilizados para el montaje y configuración de sistemas de videovigilancia.

Nota: El espacio de trabajo y el equipo mobiliario que se detallan fueron proporcionados por la empresa receptora durante el periodo de la pasantía.

Tabla .2 Equipo de computo utilizado en el desarrollo

Equipo	Cantidad	Descripción
Modelo: HP 15-f387wm Procesador: AMD Athlon Silver 3050u 2.30GHz Gráficos: AMD Radeon R5HD Memoria: 8GB Ram Capacidad del disco: 500 m.2	1	Equipo personal

Nota: Las especificaciones del computador personal empleado para el desarrollo del proyecto se detallan en la tabla anterior.

Humanos

Tabla .3 Recursos Humanos y Equipo de asesoría

Personal	Cantidad	Descripción
Desarrollador Backend	1	Colaboro en el desarrollo del backend del sistema, proporcionando los endpoints y API necesarios para la integración del frontend
Responsable de la unidad de tecnología	1	Brindo la inducción a los procedimientos y tareas a realizar durante la pasantía

Nota: El proceso de pasantía fue guiado tanto por un asesor académico y un asesor externo responsable de la unidad de tecnología de la empresa.

Actividades Actuales

Producción Principal y otras

El quehacer fundamental de RT Network S.A.S. de C.V. se estructura en dos áreas estratégicas:

- el desarrollo de soluciones de software, y
- la implementación de sistemas de seguridad electrónica mediante videovigilancia.

En el ámbito del desarrollo de software, la empresa se encuentra ejecutando un proceso de modernización tecnológica. Antes de este proceso, RT Network operaba un sistema interno de inventario y ventas desarrollado en PHP, el cual presentaba limitaciones notables en cuanto a experiencia de usuario, escalabilidad, mantenimiento y flujo de datos. Debido a estas condiciones, la empresa inició un proyecto de renovación tecnológica que contempló la migración gradual del sistema hacia React 19 con TypeScript, con el objetivo de construir interfaces modernas, accesibles e intuitivas.

El proceso de actualización se desarrolló bajo un enfoque parcial y progresivo, donde el frontend fue reconstruido completamente en React 19, mientras que el backend continuó implementándose y mejorándose en paralelo. Esto permitió reutilizar lógica existente, desarrollar nuevos endpoints cuando fue necesario y garantizar la continuidad operativa del sistema. Los módulos contemplados incluyen: gestión de inventario, ventas, facturación electrónica, usuarios, productos, proveedores, cajas, categorías y subcategorías.

En el área de seguridad electrónica, RT Network diseña e implementa soluciones completas de videovigilancia mediante cámaras IP y sistemas CCTV. Estas actividades abarcan el tendido de cableado estructurado, configuración de dispositivos, implementación de NVR/DVR y habilitación de monitoreo local y remoto. Como actividades complementarias, la empresa también ofrece soporte técnico informático, diseño UX/UI para proyectos externos y consultoría en infraestructura tecnológica.

Situación técnica

situación técnica de RT Network se caracteriza por el uso de tecnologías modernas tanto en el desarrollo de software como en los servicios de videovigilancia que ofrece la empresa. Durante la pasantía, el sistema interno se encontraba en un proceso de actualización tecnológica, dejando atrás un sistema anterior limitado y avanzando hacia una arquitectura más moderna y escalable.

En el frontend, se trabajó con React 19 y TypeScript, herramientas que permiten construir interfaces rápidas, seguras y basadas en componentes reutilizables. El uso de bibliotecas como React Hook Form, Zod, Tailwind CSS, Lucide React y Recharts facilitó la creación de formularios validados, un diseño responsivo y una experiencia de usuario más moderna.

Por otro lado, el backend del sistema se desarrollaba en paralelo, utilizando tecnologías distintas al sistema antiguo. La empresa construyó una API REST moderna basada en Node.js, empleando frameworks como NestJS y Express, y una base de datos MySQL gestionada mediante Sequelize como ORM. Este backend incluía:

- Endpoints CRUD para los diferentes módulos
- Autenticación mediante JSON Web Tokens (JWT)
- Validación de datos con express-validator
- Configuración de CORS para permitir la comunicación segura con el frontend

El frontend que desarrollé consumía directamente estos endpoints, permitiendo integrar la lógica del sistema con las interfaces que construí.

Además, RT Network utiliza tecnologías actualizadas en el área de videovigilancia, como cámaras IP de alta definición, DVR/NVR, cableado estructurado y sistemas de monitoreo remoto en tiempo real.

El proceso de trabajo técnico durante la pasantía incluía análisis de requerimientos, diseño UX/UI, creación de componentes, pruebas de integración con las APIs,

validación de datos y ajustes visuales y funcionales hasta completar cada módulo del sistema.

Situación Administrativa

RT Network S.A.S. de C.V., fundada en 2022, opera como una empresa privada dedicada a la prestación de servicios tecnológicos en la región oriental del país. Su estructura organizativa está orientada hacia la eficiencia operativa, encabezada por la Gerencia General, que supervisa la ejecución de los proyectos y define la estrategia empresarial.

La empresa divide su operación en dos unidades técnicas:

1. **Unidad de Desarrollo de Software**, responsable del diseño, construcción, pruebas e implementación de aplicaciones web.
2. **Departamento Técnico de Videovigilancia**, encargado de instalaciones, mantenimiento preventivo y correctivo, soporte especializado y consultoría en seguridad electrónica.

En términos de gestión de proyectos, la empresa mantiene una comunicación directa con sus clientes mediante levantamiento de requerimientos, presentaciones de avances y capacitaciones al finalizar los proyectos. Asimismo, se lleva un control administrativo basado en:

- asistencia,
- asignación de responsables internos,
- seguimiento técnico,
- control de versiones,
- documentación de procesos.

Esta estructura permite asegurar calidad, cumplimiento de plazos y continuidad de servicio, posicionando a RT Network como un proveedor confiable en el mercado salvadoreño.

Metodología

La metodología aplicada durante mi pasantía profesional en RT Network S.A.S. de C.V. se basó en un enfoque integral orientado al desarrollo frontend moderno,

fundamentado en principios de programación funcional y arquitectura basada en componentes. Este enfoque resultó adecuado para el proceso de modernización tecnológica que la empresa estaba llevando a cabo, especialmente considerando la transición desde un sistema heredado desarrollado en PHP hacia una estructura más escalable y mantenible.

Enfoque metodológico general

Trabajé utilizando React 19 como framework principal, debido a su arquitectura basada en componentes que permite mantener una separación clara entre la lógica de presentación y la lógica de negocio, promoviendo modularidad, reutilización y mantenibilidad. Como señala Zuraida (2023), React es una de las bibliotecas más utilizadas para la construcción de interfaces modernas debido a su capacidad de escalabilidad. Asimismo, su enfoque componentizado facilita una mejor organización y encapsulación del código, lo cual simplifica la gestión y el mantenimiento de aplicaciones complejas, tal como lo affirma Opere (2024).

Este enfoque fue particularmente útil para el desarrollo del sistema interno de RT Network, ya que permitió construir módulos independientes como gestión de productos, usuarios, proveedores, cajas, categorías y subcategorías, sin generar dependencias rígidas entre ellos.

Uso de TypeScript y su relevancia metodológica

El lenguaje principal utilizado fue TypeScript, adoptado por su tipado estático y robustez en la detección de errores. La decisión de utilizarlo se fundamenta en su capacidad para prevenir fallos en tiempo de compilación, mejorando así la calidad del software. SitePoint (2024) destaca cómo TypeScript introduce herramientas avanzadas de desarrollo, tales como autocompletado y refactorización segura.

De acuerdo con ResearchGate (2024), el tipado estático ayuda a disminuir errores en producción y reduce significativamente el tiempo de depuración. Además, al ser un superconjunto tipado de JavaScript, TypeScript permite una mayor escalabilidad y mantenibilidad del código (GeeksforGeeks, 2025), factores esenciales en aplicaciones empresariales como la desarrollada durante la pasantía.

Integración con el backend (desarrollado en paralelo)

El desarrollo del frontend se realizó en sincronía con un backend que la

empresa estaba construyendo y mejorando en paralelo. Esto implicó trabajar con:

- APIs RESTful, algunas existentes y otras desarrolladas durante el proyecto.
- Validaciones tipadas con Zod para asegurar integridad de datos.
- Pruebas funcionales de cada endpoint previo a su integración.
- Coordinación con el desarrollador backend para evitar inconsistencias en modelos y estructuras.

Este modelo permitió un desarrollo incremental donde cada módulo fue probado individualmente antes de incorporarlo al flujo general del sistema

Las herramientas utilizadas fueron:

Tabla .4 Marco tecnológico utilizado en el proyecto

Tecnología	**Versión**	**Descripción**
React	19	Biblioteca de JavaScript para construir interfaces de usuario mediante componentes funcionales reutilizables y reactivos.
TypeScript	5.x	Superset de JavaScript que añade tipado estático, permitiendo detección temprana de errores y mayor seguridad en el código.
Tailwind CSS	3.x	Framework de CSS basado en utilidades que permite diseñar interfaces responsivas directamente en el HTML mediante clases predefinidas.
Git/GitHub	Git: 2.43.0	Sistema de control de versiones distribuido y plataforma de colaboración para gestión de código fuente y trabajo en equipo.

Nota: Las tecnologías mostradas representan un stack de desarrollo

frontend moderno, centrado en React 19 con TypeScript, que sirvió como base para construir interfaces de usuario funcionales, seguras, responsivas y con una experiencia de usuario optimizada.

La selección de este stack tecnológico no fue arbitraria, sino que responde a las mejores prácticas de la industria del desarrollo web moderno. Como señala ResearchGate (2022), React ganó popularidad principalmente debido a su naturaleza declarativa y basada en componentes, permitiendo crear componentes que manejan su propio estado y construir interfaces de usuario más complejas estructurando estos componentes juntos, donde los componentes de React son fragmentos independientes de código que permiten la reutilización.

El control de versiones mediante Git y GitHub constituyó un elemento fundamental de la metodología de trabajo. Según Atlassian (s.f.), el control de versiones, también conocido como control de código fuente, es la práctica de rastrear y gestionar cambios en el código de software; los sistemas de control de versiones son herramientas de software que ayudan a los equipos de software a gestionar cambios en el código fuente a lo largo del tiempo, siendo especialmente útiles para equipos DevOps ya que les ayudan a reducir el tiempo de desarrollo y aumentar las implementaciones exitosas.

La implementación de prácticas de control de versiones garantizó la trazabilidad y el trabajo colaborativo. Como menciona GitLab (s.f.), usar las mejores prácticas de Git ayuda a los equipos de desarrollo de software a satisfacer las demandas de cambios rápidos en la industria combinados con la creciente demanda de los clientes por nuevas características; Git, como pilar del desarrollo de software moderno, ofrece un conjunto de herramientas y características poderosas diseñadas para optimizar los ciclos de desarrollo, mejorar la calidad del código y fomentar la colaboración entre los miembros del equipo.

Durante la pasantía, el uso de Git permitió mantener un historial completo

de cambios, facilitar la colaboración con el equipo de backend y garantizar la trazabilidad del código desarrollado, implementando prácticas como commits atómicos, ramas de características y
revisiones de código mediante pull requests.

Resultados y Discusión

El trabajo de desarrollo frontend realizado durante la pasantía se organiza en tres bimestres, cada uno centrado en la construcción progresiva de la interfaz de usuario y la consolidación funcional del sistema. Cada fase representó un avance significativo en la modernización del sistema interno de RT Network S.A.S. de C.V.

Fase 1: Maquetación de Interfaces (abril – junio 2025)

El primer bimestre correspondió a una etapa fundamental de construcción visual y estructural del sistema. Durante esta fase establecí la base del frontend, enfocándome en definir la arquitectura en React 19, construir componentes reutilizables y diseñar las interfaces que posteriormente serían conectadas con la API del backend.

Implementación del sistema de navegación y componentes base

El primer paso consistió en definir la estructura general de la aplicación. Para ello desarrollé desde cero el componente Sidebar, el cual se convirtió en el elemento central de navegación del sistema. Este componente no solo funciona como un menú lateral, sino como la guía principal para acceder a los módulos más importantes del sistema: gestión de productos, proveedores, usuarios, cajas y categorías.

El Sidebar fue diseñado siguiendo principios de modularidad y experiencia de usuario, estableciendo una línea gráfica coherente para el resto de la aplicación.

Maquetación de módulos CRUD: construcción de la interfaz de gestión

Una vez definida la estructura de navegación, procedí a construir la maquetación visual de los módulos CRUD principales. En esta fase el trabajo se centró únicamente en la interfaz de usuario, sin integrar todavía la lógica ni la comunicación con el backend, preparando así el terreno para su futura funcionalidad.

Módulo de Gestión de Cajas:

Se desarrolló la estructura visual completa del módulo de Cajas, incluyendo:

Formulario de registro con campos como nombre de caja y monto inicial.

Vista de listado en tabla para mostrar los registros creados.

Botones de edición y eliminación, creados inicialmente solo como elementos visuales.

Figura 2. Formulario del módulo de cajas

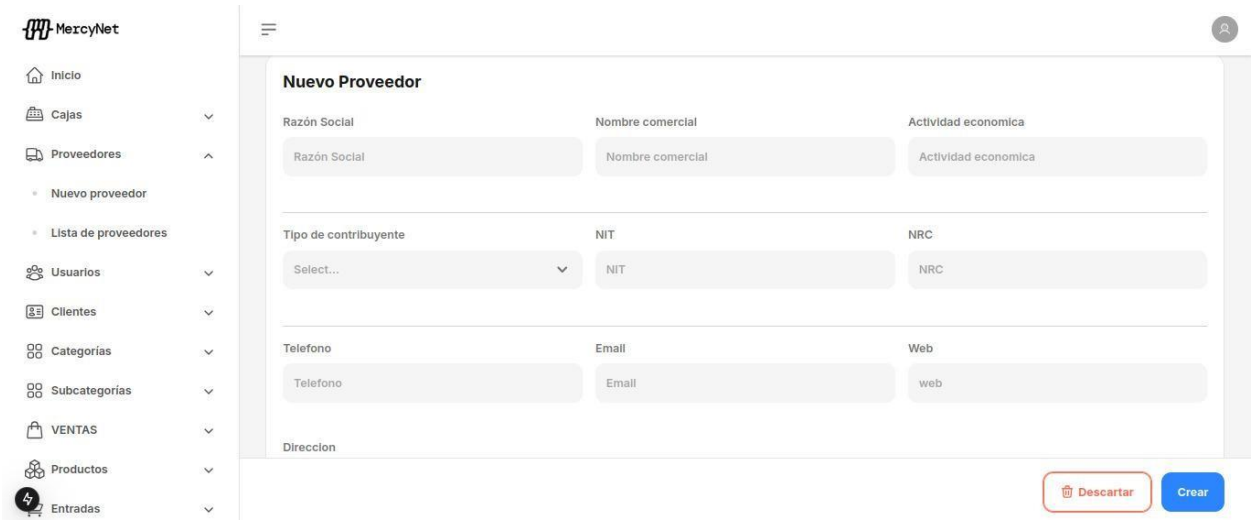
Nota: Interfaz del módulo de cajas que contiene en formulario con los campos esenciales (nombre de caja y monto)

Módulo de Proveedores:

Se diseñó y estructuró el módulo CRUD de proveedores, desarrollando todas las vistas necesarias para la gestión completa de esta entidad. En esta etapa construí la maquetación del formulario de registro, el cual incluía los campos esenciales para la captura de información básica de los proveedores. Asimismo, se implementó una tabla dinámica para el listado de registros, permitiendo visualizar de forma ordenada los datos ingresados.

Durante el proceso de diseño, surgieron nuevos requerimientos por parte de la empresa, por lo que el formulario fue ampliado con campos adicionales relevantes para la operación interna. Esta adaptación evidenció la capacidad del sistema para ajustarse a necesidades cambiantes y permitió planificar una estructura flexible para la futura integración con el backend.

Figura 3. Formulario módulo de proveedores



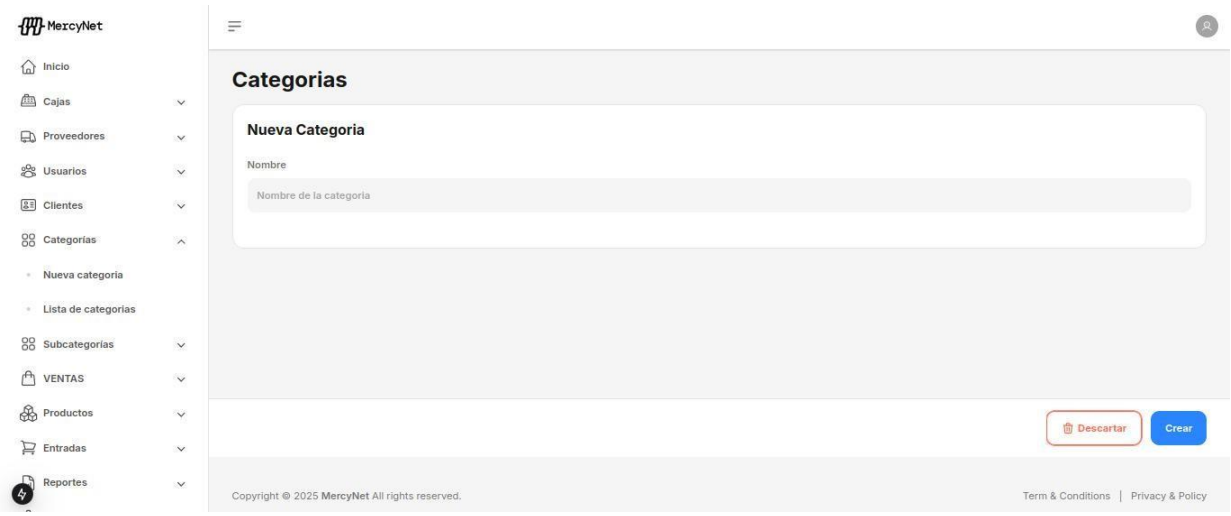
Nota: Interfaz de usuario del módulo de gestión de proveedores La vista muestra el formulario de registro con campos de captura de datos.

Modulo categorías y subcategorías:

Se realizó la maquetación del módulo destinado a la gestión jerárquica de categorías y subcategorías, un componente esencial para la correcta organización de los productos dentro del sistema. Esta etapa incluyó la construcción de formularios independientes para el registro tanto de categorías principales como de sus subcategorías asociadas, asegurando una estructura visual clara y coherente.

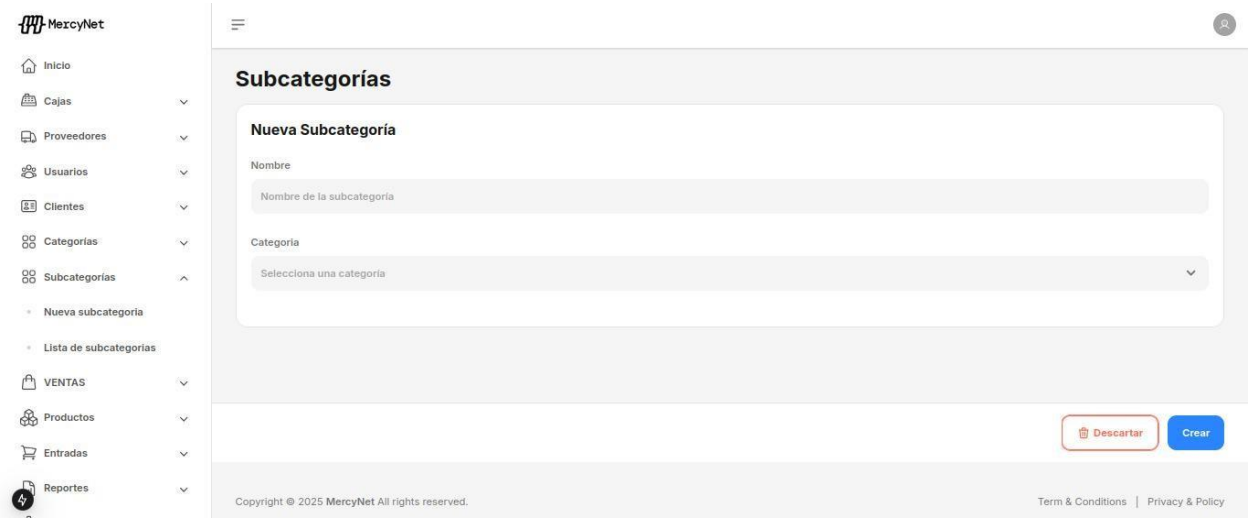
Asimismo, se diseñó una vista de listado que permite visualizar la relación entre cada categoría y sus subcategorías correspondientes, facilitando la gestión de la estructura interna del inventario. Esta maquetación sentó las bases para una futura integración dinámica con el backend, donde dicha jerarquía se administrará de forma automática a través de la API.

Figura 4. Interfaz del Módulo de Gestión de Categorías



Nota: Interfaz del módulo de categorías que permite organizar los productos del sistema, incluye formulario de registro

Figura 5. Formulario módulo de subcategorías

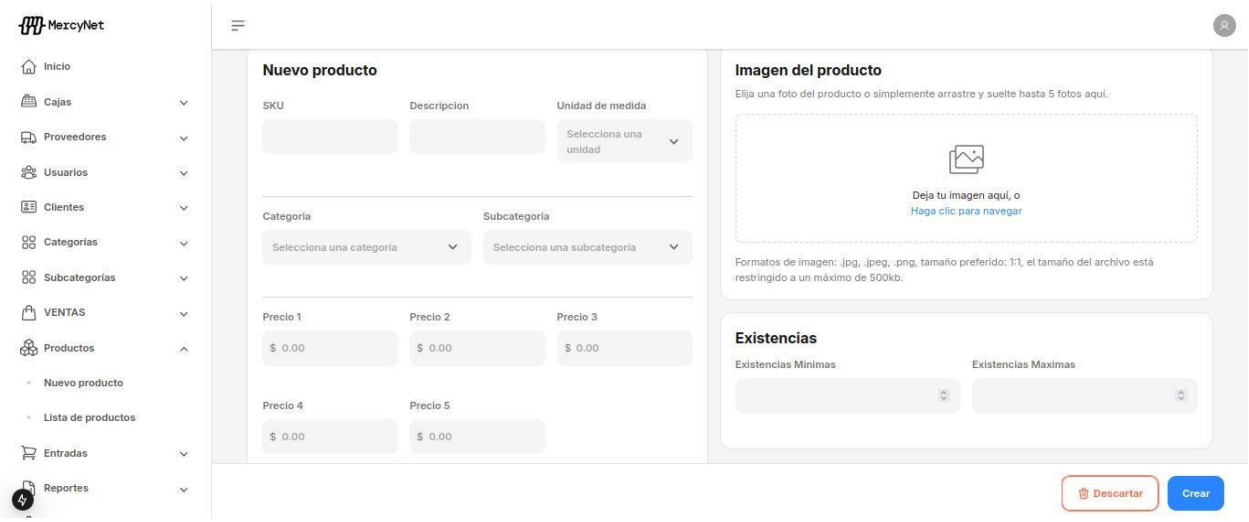


Nota: Interfaz del módulo de subcategorías que permite crear clasificaciones específicas dentro de cada categoría principal.

Módulo de productos:

Se desarrolló la maquetación del formulario de gestión de productos, uno de los módulos más completos del sistema. Este trabajo abarcó la creación de un formulario detallado con los campos principales necesarios para la gestión de inventario (nombre, descripción, precio, categoría, entre otros), así como una vista de listado en tabla que contemplaba opciones de búsqueda y acciones de edición y eliminación. La estructura se diseñó pensando en la escalabilidad y la facilidad de uso para los operadores del sistema.

Figura 6. Módulo de administración de productos



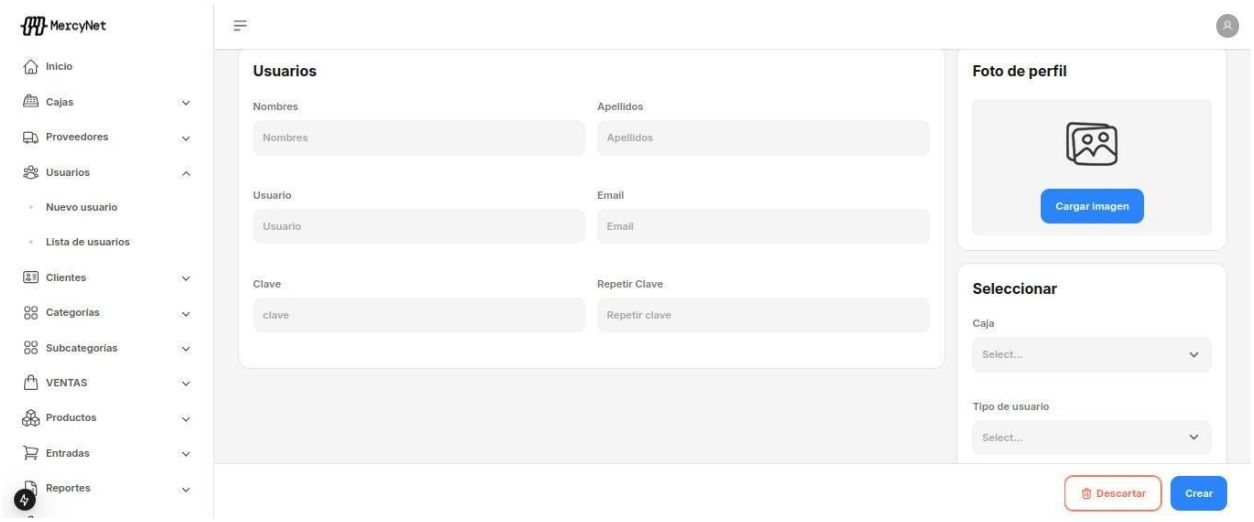
Nota: interfaz del módulo de productos donde se registra y gestiona el inventario del sistema. Incluye formulario con los datos principales del producto (nombre, descripción, precio, categoría)

Módulo de Usuarios:

Como parte del sistema de administración, se desarrolló la maquetación del módulo CRUD de usuarios utilizando React 19. Este componente, considerado crítico para la gestión integral del sistema, incluyó la estructuración de formularios con los campos principales para la creación y edición de usuarios, tales como nombre, correo electrónico, rol y contraseña, anticipando la posterior implementación de validaciones para garantizar la correcta captura y seguridad de la información.

Asimismo, se diseñó un listado dinámico de usuarios con la estructura necesaria para soportar operaciones completas de gestión, incluyendo edición y eliminación, manteniendo coherencia con la arquitectura visual del sistema y los principios de usabilidad establecidos.

Figura 7. Módulo de gestión de usuarios



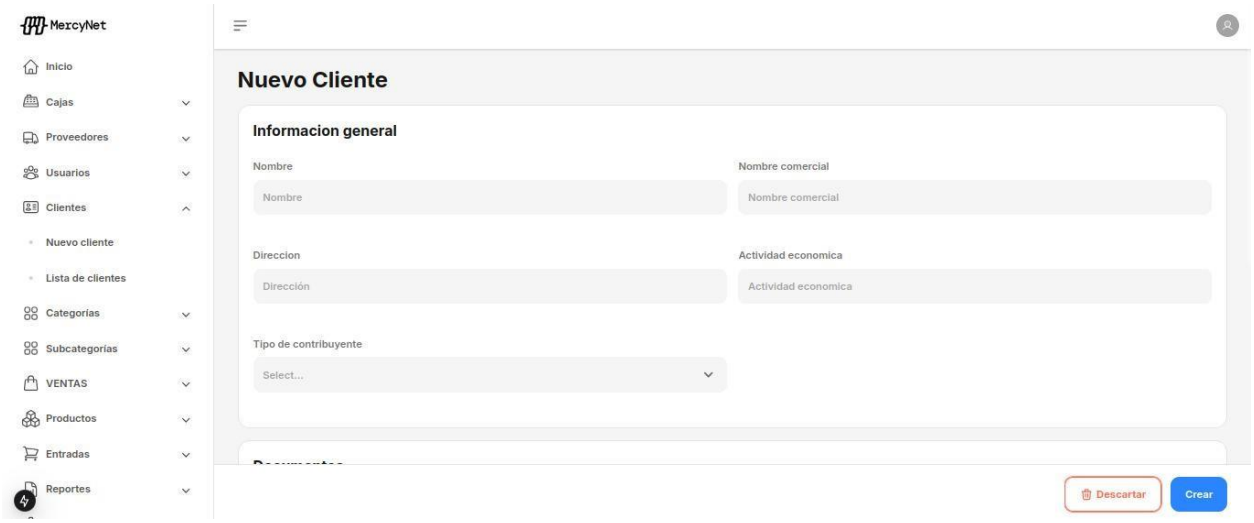
Nota: Interfaz del módulo de usuarios que permite administrar quiénes tienen acceso al sistema. Incluye formulario de registro con datos básicos (nombre, correo, rol, contraseña)

Módulo de Clientes:

Se completó la fase de maquetación con el desarrollo visual del módulo de clientes, definiendo la estructura y distribución de la interfaz para el registro y edición de la información de los clientes. El diseño se centró en mantener coherencia con la guía de estilos del sistema, asegurando uniformidad visual, claridad en la presentación de datos y facilidad de uso para los operadores.

Asimismo, la vista fue preparada para su futura integración con los servicios de datos del backend, permitiendo la conexión con los procesos de almacenamiento, consulta y actualización de información de manera dinámica.

Figura 8. Módulo de registro de clientes



Nota: Interfaz del módulo de clientes donde se registra la información de las personas que realizan compras. Incluye un formulario de captura de datos.

Desarrollo de Diseño UX/UI: Proyectos Externos

Paralelamente al desarrollo del sistema interno, se participó en proyectos de diseño de experiencia de usuario para clientes externos de la empresa, lo que permitió aplicar y fortalecer competencias en diseño web moderno, orientado a la usabilidad, accesibilidad y presentación visual profesional.

Landing Page para Comedor:

Se diseñó una landing page profesional para un servicio de comedor, con el objetivo de ofrecer una interfaz atractiva, clara y funcional para los usuarios. El trabajo se centró exclusivamente en el desarrollo UX/UI, considerando principios de jerarquía visual, organización de la información, selección de paleta de colores y diseño responsivo.

Figura 9. Prototipo de diseño UX/UI para landing page de comedor

Nota: Prototipo visual desarrollado durante la primera fase de la pasantía, que muestra la aplicación de principios de diseño UX/UI.

Soporte Técnico y Mantenimiento

Además del desarrollo frontend, se atendieron responsabilidades relacionadas con soporte técnico, las cuales fortalecieron las competencias en el área de infraestructura tecnológica. Se ejecutaron actividades de mantenimiento preventivo y correctivo en equipos informáticos, específicamente en impresoras y computadoras.

Estas tareas incluyeron limpieza interna de componentes, actualización de software, verificación de conectividad de red, diagnóstico de fallas técnicas y resolución de problemas básicos, contribuyendo de manera directa al mantenimiento operativo de los recursos tecnológicos de la empresa y al correcto funcionamiento de las labores diarias.

Fase 2: Integración Funcional y Consumo de Apis (Julio - agosto 2025)

Desarrollo Funcional del Módulo de Cajas mediante Consumo de API

Durante la segunda fase de la pasantía se realizó la integración funcional completa del módulo de gestión de cajas mediante el consumo directo de los servicios REST proporcionados por el backend. Es importante destacar que el desarrollo del backend se llevó a cabo de forma paralela al desarrollo frontend, por lo que los endpoints fueron siendo proporcionados de manera progresiva conforme avanzaba la implementación servidor–cliente. Esta dinámica permitió validar continuamente la comunicación entre ambas capas del sistema y realizar ajustes de forma iterativa.

Esta integración permitió enlazar la interfaz desarrollada en React 19 con la base de datos del sistema, garantizando la persistencia, consulta y actualización de la información en tiempo real.

El backend expuso un conjunto de endpoints protegidos mediante autenticación, definidos bajo la ruta base /api/cajas, los cuales fueron consumidos desde el frontend a través de peticiones HTTP utilizando el método fetch. Estos endpoints permitieron ejecutar todas las operaciones CRUD del módulo, asegurando una comunicación segura entre cliente y servidor. Las rutas implementadas fueron:

- **POST /api/cajas**: Permite registrar una nueva caja en el sistema.
- **GET /api/cajas**: Obtiene el listado completo de cajas registradas.
- **GET /api/cajas/:id**: Consulta la información de una caja específica.
- **PUT /api/cajas/:id**: Actualiza los datos de una caja existente.
- **DELETE /api/cajas/:id**: Elimina una caja del sistema.

Todas estas rutas se encuentran protegidas mediante un middleware de autenticación, lo que garantiza que únicamente los usuarios autorizados puedan realizar operaciones sobre los registros, fortaleciendo así la seguridad del sistema.

El modelo de datos utilizado para la gestión de cajas está conformado por los atributos **id**, **nombre_caja** y **efectivo_caja**, los cuales representan de forma estructurada la información financiera básica de cada caja registrada en el sistema. Un ejemplo del formato de los datos enviados y recibidos desde la base de datos es el siguiente:

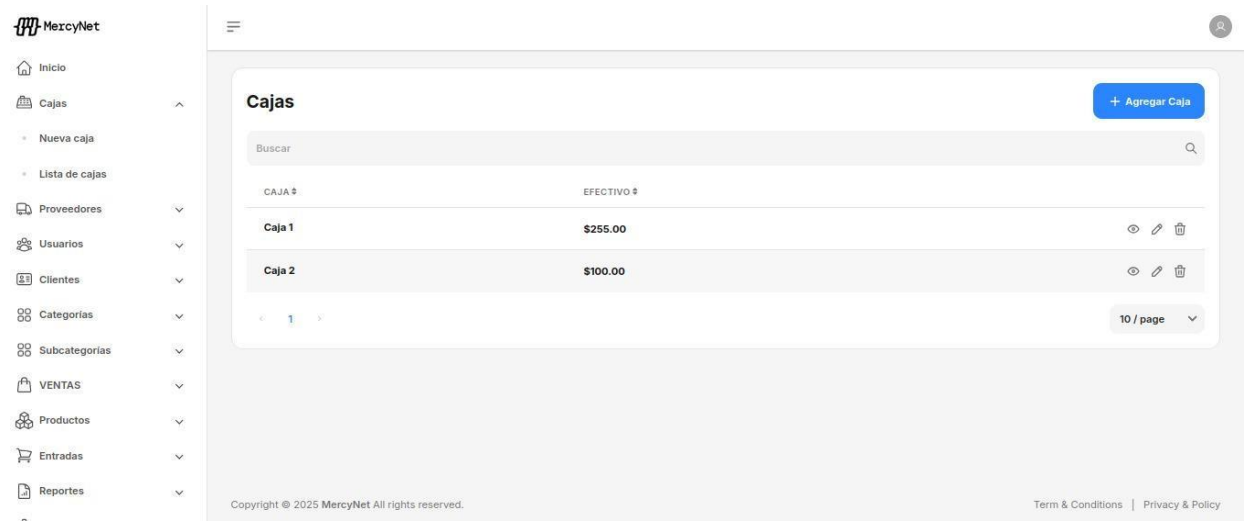
- id: Identificador único de la caja
- nombre_caja: Nombre descriptivo de la caja
- efectivo_caja: Monto de efectivo disponible en la caja

Desde el frontend, al momento de registrar una nueva caja, los datos capturados en el formulario son enviados al backend mediante una petición POST, donde se validan y posteriormente se almacenan en la base de datos. De igual forma, el listado de cajas se obtiene a través de una petición GET, permitiendo mostrar en la interfaz una tabla dinámica con los registros actualizados en tiempo real.

Las opciones de editar y eliminar se implementaron utilizando las peticiones PUT y DELETE respectivamente, mediante el uso del identificador único de cada caja. De esta manera, se logró que cada acción realizada por el usuario en la interfaz tuviera un reflejo inmediato en la base de datos, consolidando un flujo de trabajo completamente funcional.

Esta integración convirtió al módulo de cajas en un componente totalmente operativo dentro del sistema, dejando atrás su condición de prototipo visual desarrollada en la fase anterior. Asimismo, evidenció el trabajo coordinado entre el equipo de frontend y backend, demostrando la correcta comunicación entre el cliente desarrollado en React 19 y el servidor construido sobre tecnologías basadas en Node.js, fortaleciendo la arquitectura cliente–servidor del sistema.

Figura 10. Listado operativo de cajas del sistema



Nota: Interfaz del módulo de cajas con listado de registros y opciones de gestión.

Desarrollo Funcional del Módulo de Proveedores mediante Consumo de API

Durante la segunda fase de la pasantía se realizó la integración funcional completa del módulo de gestión de proveedores mediante el consumo directo de los servicios REST proporcionados por el backend. Es importante señalar que el desarrollo del backend se ejecutó de forma paralela al desarrollo frontend, por lo que los endpoints fueron siendo entregados progresivamente conforme avanzaba la implementación del sistema. Esta metodología permitió validar continuamente la comunicación entre ambas capas y realizar ajustes de manera iterativa.

Esta integración permitió conectar la interfaz desarrollada en React 19 con la base de datos del sistema, asegurando la persistencia, consulta y actualización de la información de los proveedores en tiempo real.

El backend expuso un conjunto de endpoints protegidos mediante autenticación bajo la ruta base /api/proveedores, los cuales fueron consumidos desde el frontend a través de peticiones HTTP utilizando el método *fetch*. Estas rutas permitieron ejecutar todas las operaciones CRUD del módulo, garantizando una comunicación segura entre cliente y servidor. Las rutas implementadas fueron:

- **POST /api/proveedores:** Permite registrar un nuevo proveedor en el sistema.
- **GET /api/proveedores:** Obtiene el listado completo de proveedores registrados.
- **GET /api/proveedores/:id**: Consulta la información de un proveedor específico.
- **PUT /api/proveedores/:id:** Actualiza los datos de un proveedor existente.
- **DELETE /api/proveedores/:id:** Elimina un proveedor del sistema.

Todas estas rutas se encuentran protegidas mediante un middleware de autenticación, lo que garantiza que únicamente los usuarios autorizados puedan realizar operaciones sobre los registros, fortaleciendo la seguridad del sistema y el control de acceso a la información.

El modelo de datos utilizado para la gestión de proveedores está conformado por los atributos: id, razon_social, nombre_comercial, actividad_economica, tipo_contribuyente, NIT, NRC, teléfono, correo electrónico, sitio web y dirección, permitiendo almacenar de manera estructurada toda la información fiscal, comercial y de contacto de cada proveedor. Un ejemplo del formato de los datos intercambiados con la base de datos es el siguiente:

- id: Identificador único del proveedor.
- razon_social: Nombre legal de la empresa o proveedor.
- nombre_comercial: Nombre con el que opera comercialmente.
- actividad_economica: Giro o actividad principal.
- tipo_contribuyente: Clasificación tributaria.
- NIT: Número de Identificación Tributaria.
- NRC: Número de Registro de Contribuyente.
- telefono: Número de contacto.

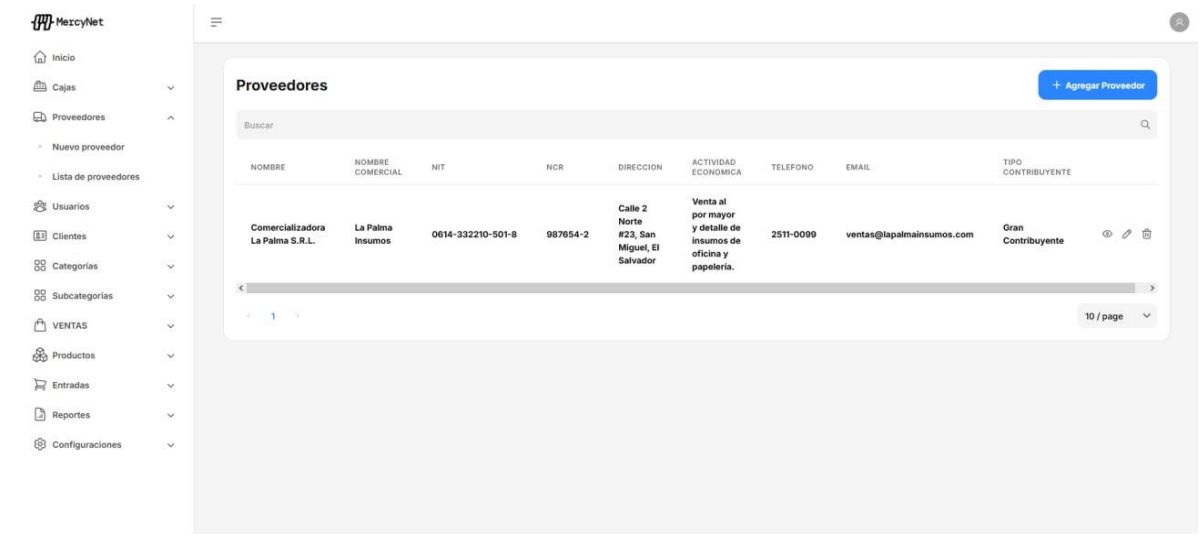
- correo: Correo electrónico del proveedor.
- web: Sitio web.
- direccion: Dirección física del proveedor.

Desde el frontend, al momento de registrar un nuevo proveedor, los datos capturados en el formulario son enviados al backend mediante una petición POST, donde son validados y posteriormente almacenados en la base de datos. De igual manera, el listado de proveedores se obtiene mediante una petición GET, permitiendo mostrar en la interfaz una tabla dinámica con los registros actualizados en tiempo real.

Las opciones de editar y eliminar se implementaron mediante las peticiones PUT y DELETE, utilizando el identificador único de cada proveedor. De esta forma, cada acción realizada por el usuario en la interfaz tiene un reflejo inmediato en la base de datos, consolidando un flujo de trabajo completamente funcional.

Esta integración convirtió al módulo de proveedores en un componente totalmente operativo dentro del sistema, superando su etapa de prototipo visual desarrollada en la fase anterior. Asimismo, evidenció el trabajo coordinado entre el frontend desarrollado en React 19 y el backend construido sobre Node.js, fortaleciendo la arquitectura cliente–servidor del sistema.

Figura 11. Módulo de gestión de proveedores integrado mediante consumo de API



Desarrollo Funcional del Módulo de Categorías y Subcategorías mediante Consumo de API

Durante la segunda fase de la pasantía se realizó la integración funcional completa del módulo de gestión de categorías y subcategorías mediante el consumo directo de los servicios REST proporcionados por el backend. Al igual que en los demás módulos del sistema, el desarrollo del backend se llevó a cabo de forma paralela al desarrollo frontend, por lo que los endpoints fueron siendo proporcionados progresivamente conforme avanzaba la implementación del sistema, permitiendo validar de manera continua la correcta comunicación entre cliente y servidor.

Esta integración permitió enlazar la interfaz desarrollada en React 19 con la base de datos del sistema, garantizando la persistencia, consulta y actualización en tiempo real de la información correspondiente a las categorías y sus respectivas subcategorías, manteniendo la relación jerárquica entre ambas entidades.

El backend expuso un conjunto de endpoints protegidos mediante autenticación bajo la ruta base /api/categorias para la gestión de categorías y /api/sub-categorias para la gestión de subcategorías. Estos servicios fueron consumidos desde el frontend mediante peticiones HTTP utilizando el método *fetch*, permitiendo ejecutar todas las operaciones CRUD del módulo. Las rutas implementadas para categorías fueron las siguientes:

- **POST /api/categorias:** Permite registrar una nueva categoría en el sistema.
- **GET /api/categorias:** Obtiene el listado completo de categorías registradas.
- **GET /api/categorias/:id:** Consulta la información de una categoría específica.
- **PUT /api/categorias/:id:** Actualiza los datos de una categoría existente.
- **DELETE /api/categorias/:id:** Elimina una categoría del sistema.

Por su parte, para la gestión de subcategorías se implementaron los siguientes endpoints:

- **POST /api/sub-categorias:** Permite registrar una nueva subcategoría.
- **GET /api/sub-categorias:** Obtiene el listado completo de subcategorías**.**
- **GET /api/sub-categorias/:id:** Consulta una subcategoría específica.
- **PUT /api/sub-categorias/:id:** Actualiza los datos de una subcategoría existente**.**
- **DELETE /api/sub-categorias/:id:** Elimina una subcategoría del sistema.
- **GET /api/sub-categorias/categoria/:id_categoria:** Obtiene todas las subcategorías asociadas a una categoría específica.

Todas estas rutas se encuentran protegidas mediante un middleware de autenticación, garantizando que únicamente los usuarios autorizados puedan gestionar la información del sistema.

El modelo de datos utilizado para la gestión de categorías está conformado por los atributos id y nombre, permitiendo identificar y clasificar los productos del sistema de manera ordenada. Para la gestión de subcategorías, el modelo de datos incluye los atributos id, id_categoria, nombre y nombre_categoria, lo que permite establecer la relación directa entre cada subcategoría y su categoría principal.

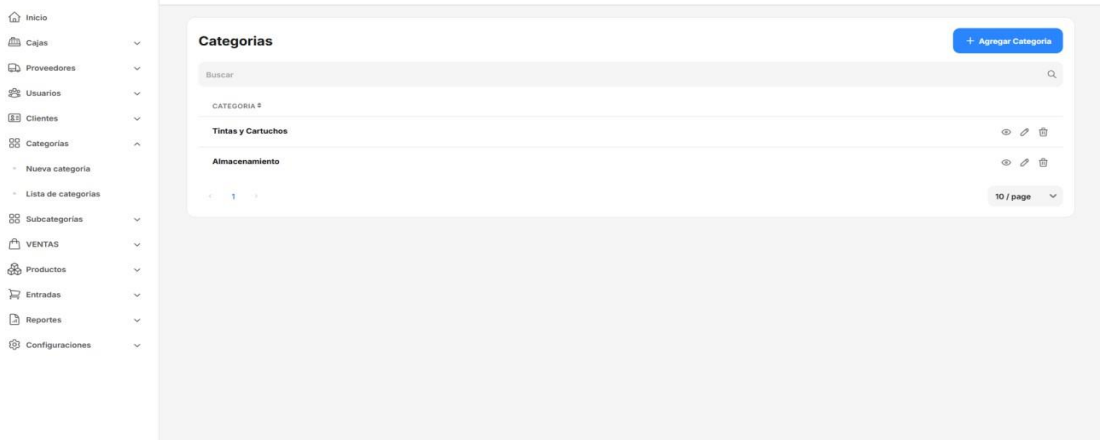
Desde el frontend, al momento de registrar una nueva categoría o subcategoría, los datos capturados en los formularios son enviados al backend mediante una petición POST, donde son validados y posteriormente almacenados en la base de datos. El listado de categorías y subcategorías se obtiene a través de peticiones GET, permitiendo mostrar tablas dinámicas con los registros actualizados en tiempo real.

Las operaciones de edición y eliminación se implementaron mediante las peticiones PUT y DELETE, utilizando el identificador único de cada registro. Además, el uso del endpoint que permite consultar subcategorías por categoría facilitó la visualización

jerárquica de los datos, asegurando una correcta organización de los productos dentro del sistema.

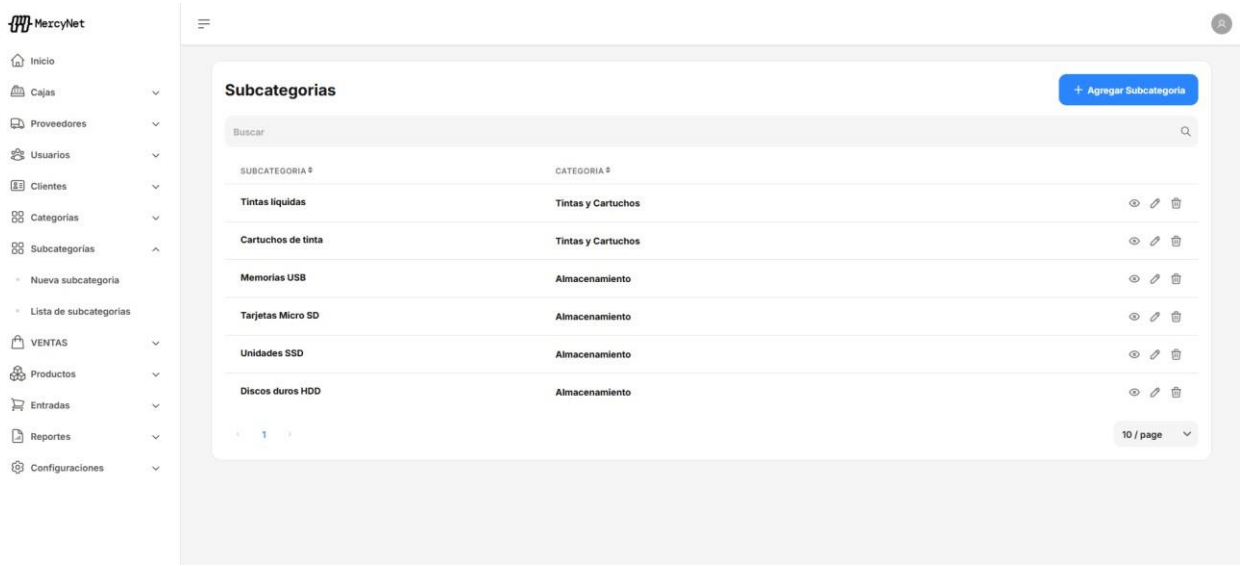
Esta integración convirtió al módulo de categorías y subcategorías en un componente completamente operativo del sistema, superando su etapa de prototipo visual desarrollada en la fase anterior. Asimismo, evidenció la correcta comunicación entre el frontend desarrollado en React 19 y el backend construido sobre Node.js, fortaleciendo la arquitectura cliente–servidor del sistema.

Figura 12. Módulo de gestión de categorías integrado mediante consumo de API



Nota: Interfaz del módulo de categorías para la organización de productos.

Figura 13. Módulo de gestión de subcategorías integrado mediante consumo de API



Desarrollo Funcional del Módulo de Productos mediante Consumo de API

Durante la segunda fase de la pasantía se realizó la integración funcional completa del módulo de gestión de productos mediante el consumo directo de los servicios REST proporcionados por el backend. Al igual que en los demás módulos del sistema, el desarrollo del backend se ejecutó de forma paralela al desarrollo del frontend, por lo que los endpoints fueron suministrados progresivamente conforme avanzaba la implementación del sistema, permitiendo validar de manera continua la correcta comunicación entre ambas capas de la arquitectura.

Esta integración permitió enlazar la interfaz desarrollada en React 19 con la base de datos del sistema, asegurando la persistencia, consulta, actualización y eliminación de la información de los productos en tiempo real, consolidando el núcleo funcional del sistema de inventario.

El backend expuso un conjunto de endpoints protegidos mediante autenticación bajo la ruta base **/api/productos**, los cuales fueron consumidos desde el frontend a través de peticiones HTTP utilizando el método fetch. Estas rutas permitieron ejecutar todas las operaciones CRUD del módulo, así como la consulta especializada por código SKU. Las rutas implementadas fueron las siguientes:

- **GET /api/productos/sku/: sku:** Permite consultar un producto específico mediante su código SKU.
- **POST /api/productos:** Permite registrar un nuevo producto en el sistema.
- **GET /api/productos:** Obtiene el listado completo de productos con la información de sus dependencias.
- **GET /api/productos/:id:** Consulta la información detallada de un producto específico.
- **PUT /api/productos/:id:** Actualiza los datos de un producto existente.
- **DELETE /api/productos/:id:** Elimina un producto del sistema.

Todas estas rutas se encuentran protegidas mediante un middleware de autenticación, garantizando que únicamente los usuarios autorizados puedan gestionar la información del inventario, fortaleciendo así la seguridad del sistema.

El modelo de datos utilizado para la gestión de productos está conformado por múltiples atributos que permiten una administración completa del inventario. Entre los principales campos se encuentran: id, sku, descripción, fecha de vencimiento, id_categoria, id_sub_categoria, unidad de medida, presentación, stock, stock mínimo, stock máximo y precios en diferentes escalas (precio1 a precio5). Esta estructura permite un control detallado de los productos tanto a nivel operativo como comercial.

Desde el frontend, al momento de registrar un nuevo producto, los datos capturados en el formulario son enviados al backend mediante una petición POST, donde son validados y posteriormente almacenados en la base de datos. El listado de productos se obtiene mediante una petición GET, permitiendo mostrar en la interfaz una tabla dinámica con los registros actualizados en tiempo real.

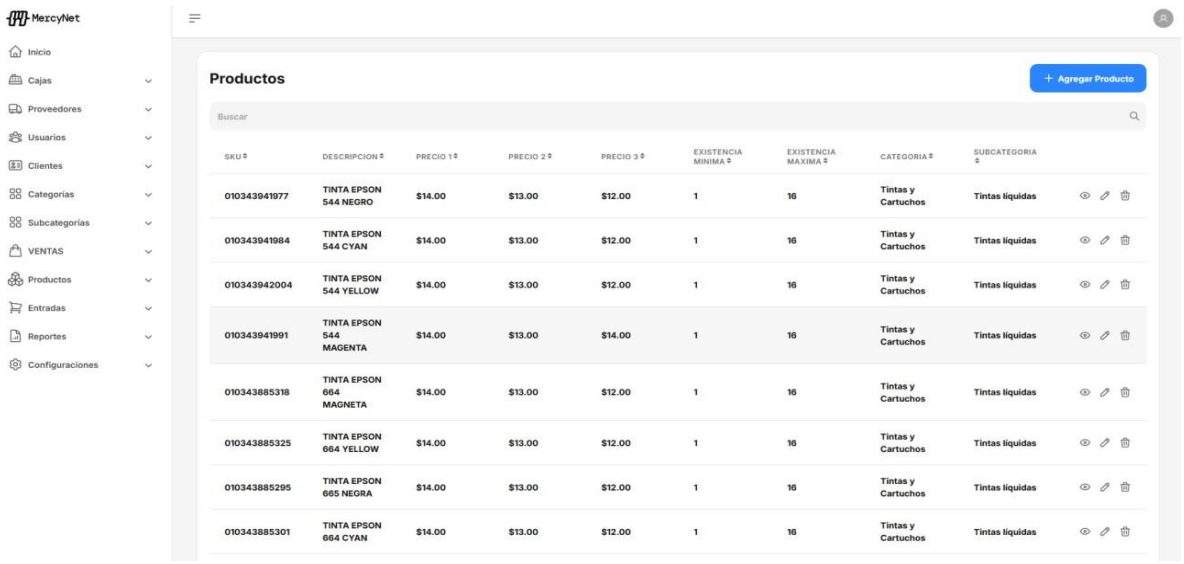
Una funcionalidad relevante implementada en este módulo fue la búsqueda de productos mediante código SKU, utilizando el endpoint específico para este propósito, lo cual agiliza significativamente los procesos de registro de entradas, ventas y consultas rápidas dentro del sistema.

Las opciones de edición y eliminación se implementaron mediante las peticiones PUT y DELETE, utilizando el identificador único de cada producto. De esta forma, cada acción realizada por el usuario en la interfaz tiene un reflejo inmediato en la base de datos, acompañado de confirmaciones de usuario y retroalimentación visual para garantizar una experiencia segura y controlada.

Esta integración convirtió al módulo de productos en uno de los componentes más robustos y estratégicos del sistema, superando su condición de prototipo visual desarrollada en la fase anterior. Asimismo, evidenció el correcto acoplamiento entre el frontend desarrollado en React 19 y el backend construido sobre Node.js, consolidando

la arquitectura cliente–servidor del sistema de inventario.

Figura 14. Módulo de administración de productos



Nota: Interfaz del módulo de productos que muestra el inventario del sistema con barra de búsqueda y opciones de gestión.

Implementación de Validaciones y Manejo de Estado

Un aspecto fundamental del desarrollo funcional del sistema fue la implementación de un sistema robusto de validaciones del lado del cliente, con el objetivo de garantizar la calidad e integridad de la información antes de ser enviada al servidor. Se establecieron mecanismos de validación en todos los formularios de los distintos módulos, previniendo el registro de datos incorrectos, incompletos o inconsistentes.

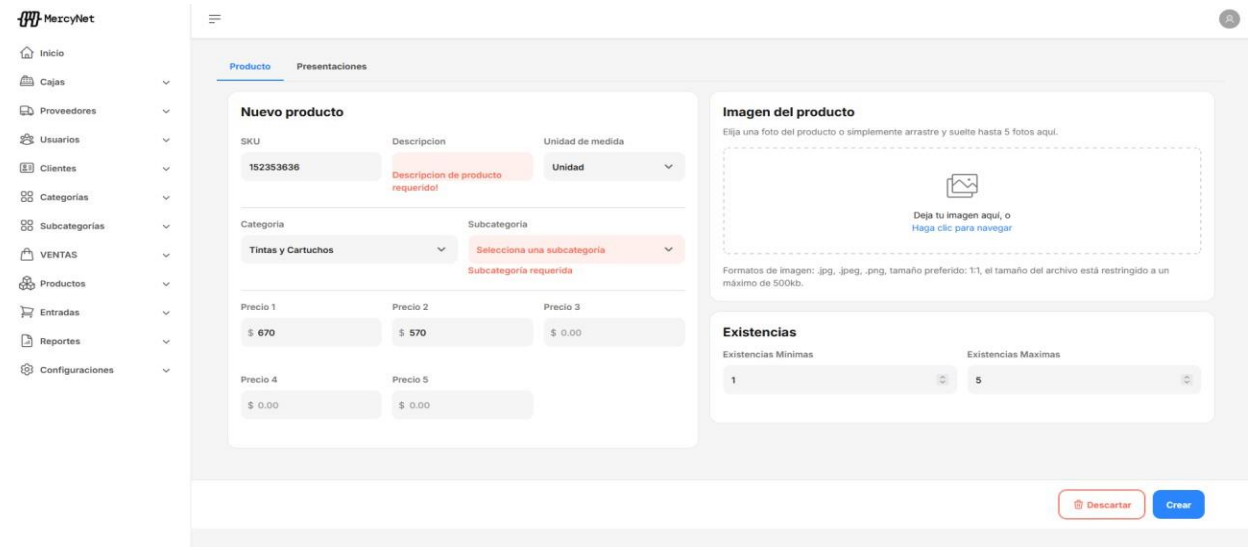
Las validaciones implementadas incluyeron la verificación de campos obligatorios, validación de formatos específicos (como direcciones de correo electrónico, valores numéricos y campos alfanuméricos), control de longitudes mínimas y máximas, así como validaciones personalizadas de acuerdo con las reglas de negocio propias de cada módulo. Todo este proceso permitió reducir errores en la base de datos y mejorar la experiencia del usuario durante la interacción con el sistema.

El manejo de estado de la aplicación se realizó utilizando las capacidades nativas de React 19, mediante el uso eficiente de los hooks useState y useEffect, lo que

permitió gestionar correctamente el ciclo de vida de los datos dentro de cada componente. Se establecieron patrones de actualización de estado que garantizaban la sincronización entre la información presentada en la interfaz y los datos provenientes del servidor.

Asimismo, se implementó retroalimentación visual inmediata al usuario, mostrando mensajes de validación, confirmación de operaciones exitosas y alertas en caso de errores, fortaleciendo la usabilidad y confiabilidad del sistema.

Figura 15. Formulario con validaciones implementadas



Nota: Interfaz de formulario con validaciones en tiempo real durante el ingreso de datos.

Instalación y Mantenimiento de Sistemas de Videovigilancia

Paralelamente al desarrollo de software, se asumieron responsabilidades en el área de infraestructura de seguridad electrónica, lo que permitió ampliar el espectro de competencias técnicas desarrolladas durante la pasantía, integrando conocimientos de redes, hardware y sistemas de videovigilancia.

Instalación de Cámaras de Videovigilancia

Se realizaron instalaciones completas de sistemas de videovigilancia en distintos entornos. Este trabajo técnico incluyó el tendido profesional de cableado de red

estructurado, la instalación física de las cámaras en los puntos estratégicamente definidos, la utilización de herramientas especializadas y la conexión de los dispositivos al sistema central de monitoreo.

Asimismo, se llevó a cabo la configuración inicial de cada cámara, verificando parámetros de red, resolución, ángulo de visión y transmisión de datos. Posteriormente, se comprobó el correcto funcionamiento de cada equipo, asegurando la captura y transmisión de imágenes en tiempo real con la calidad requerida y la cobertura adecuada de las áreas asignadas.

Mantenimiento del Sistema de Vigilancia

También se ejecutaron labores de mantenimiento preventivo y correctivo en sistemas de videovigilancia ya instalados. El mantenimiento preventivo incluyó la limpieza de lentes, verificación de conexiones de red, revisión de fuentes de alimentación y ajustes de posicionamiento. Por su parte, el mantenimiento correctivo contempló el reemplazo de cámaras con fallas, reparación de conexiones de red dañadas y solución de problemas de configuración.

Estas acciones garantizaron la continuidad operativa del sistema de seguridad, minimizando tiempos de inactividad y asegurando el correcto funcionamiento de los equipos de monitoreo.

Fase 3: Optimización, Control de Acceso y Consolidación (septiembre - octubre 2025)

El último bimestre de la pasantía representó la etapa de madurez, optimización y consolidación del sistema frontend. El enfoque principal se centró en el fortalecimiento del rendimiento de la aplicación, la implementación de mecanismos avanzados de seguridad mediante control de acceso por roles, la mejora de la experiencia de usuario a través de ajustes en la interfaz, y la ejecución de pruebas funcionales para asegurar la estabilidad del sistema. Esta fase permitió transformar una aplicación funcional en un sistema robusto, seguro y listo para su uso en un entorno de producción.

Implementación de Control de Roles y Permisos

Uno de los desarrollos más críticos de esta fase fue la integración del sistema de control de acceso basado en roles, una funcionalidad esencial para garantizar la seguridad, integridad de la información y correcta gestión de los usuarios del sistema.

Se implementó de manera completa el control de acceso en el frontend, enlazándolo con el sistema de permisos configurado en el backend. Esta integración permitió restringir las acciones disponibles para cada usuario según su rol asignado, garantizando que únicamente el personal autorizado pudiera acceder a funcionalidades sensibles del sistema.

El desarrollo incluyó la creación de componentes condicionales que muestran u ocultan opciones de la interfaz según los permisos del usuario autenticado. Asimismo, se implementaron guardias de ruta para evitar el acceso directo a secciones restringidas mediante la manipulación de la URL, y se establecieron validaciones adicionales en acciones críticas como la eliminación de registros.

Esta arquitectura de seguridad permitió una gestión más profesional del flujo de usuarios, diferenciando claramente entre administradores, operadores y usuarios con permisos limitados, fortaleciendo así la protección de la información y la correcta operación del sistema.

Implementación de Validaciones Robustas con Zod

Durante esta fase se fortaleció de manera significativa el sistema de validaciones del frontend mediante la integración de bibliotecas especializadas que elevaron los estándares de calidad y seguridad de los datos ingresados por los usuarios.

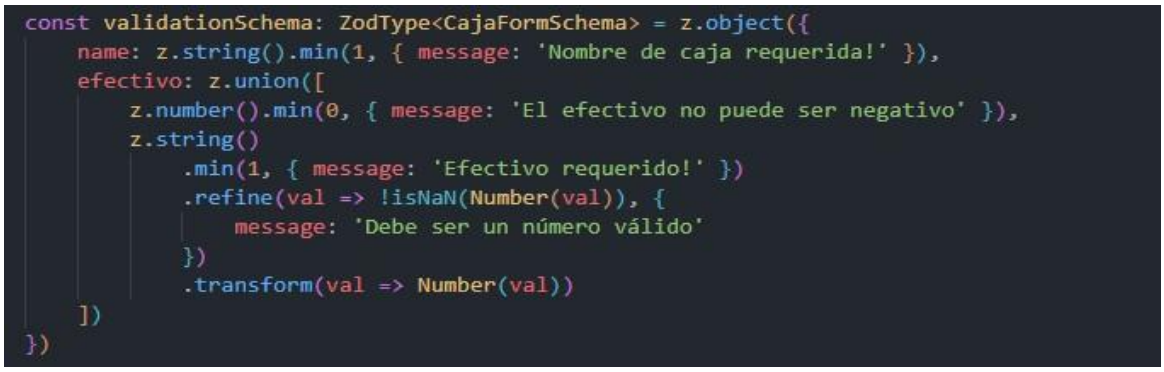
Validación de Esquemas con Zod

Se integró la biblioteca Zod como sistema de validación de esquemas, proporcionando una capa robusta de verificación del lado del cliente. Zod permitió definir esquemas de validación tipados, reutilizables y centralizados, asegurando que los datos cumplieran con las estructuras correctas antes de ser enviados al backend.

Se crearon esquemas de validación para cada una de las entidades del sistema (usuarios, productos, proveedores, cajas, categorías, entre otros), definiendo reglas específicas para cada campo, tales como: tipo de dato, formatos requeridos, longitudes mínimas y máximas, patrones mediante expresiones regulares, validaciones personalizadas y transformaciones automáticas de datos.

La implementación de Zod dio como resultado un sistema de validación robusto, mantenible y con retroalimentación clara al usuario, mostrando mensajes de error específicos que guían la correcta captura de la información y previenen el envío de datos incorrectos o incompletos al servidor.

Figura 16. Esquema de validación con Zod para el módulo de cajas



```
const validationSchema: ZodType<CajaFormSchema> = z.object({
    name: z.string().min(1, { message: 'Nombre de caja requerida!' }),
    efectivo: z.union([
        z.number().min(0, { message: 'El efectivo no puede ser negativo' }),
        z.string()
            .min(1, { message: 'Efectivo requerido!' })
            .refine(val => !isNaN(Number(val)), {
                message: 'Debe ser un número válido'
            })
            .transform(val => Number(val))
    ])
})
```

Nota: Esquema de validación implementado con Zod para el formulario de cajas.

Continuación del Soporte en Videovigilancia

Durante la fase final de la pasantía se mantuvieron de forma activa las responsabilidades en el área de seguridad electrónica y videovigilancia, consolidando las competencias técnicas desarrolladas durante los meses anteriores.

Mantenimiento Preventivo y Correctivo

Se continuó ejecutando el programa de mantenimiento del sistema de videovigilancia, realizando actividades tanto preventivas como correctivas. Entre estas se incluyó el reemplazo de cámaras con fallas técnicas, la reparación de conexiones de red deterioradas, la actualización de firmware de los dispositivos y la optimización de configuraciones de grabación.

Estas acciones garantizaron la continuidad del monitoreo, la estabilidad operativa del sistema de seguridad y la calidad en la vigilancia de las instalaciones, contribuyendo directamente a la confiabilidad del servicio ofrecido por la empresa.

Glosario

API (Application Programming Interface):
Conjunto de reglas que permite la comunicación entre diferentes aplicaciones. En el sistema desarrollado permitió la conexión entre el frontend en React y el backend en Node.js para el intercambio de información.

Backend:
Parte del sistema encargada del procesamiento de datos, la lógica de negocio y la conexión con la base de datos. Fue desarrollado utilizando tecnologías basadas en Node.js.

CRUD:
Conjunto de operaciones básicas para la gestión de datos: Crear (Create), Leer (Read), Actualizar (Update) y Eliminar (Delete).

Endpoint:
Ruta específica de una API que permite acceder a una funcionalidad determinada del sistema mediante una petición HTTP.

Fetch:
Método de JavaScript utilizado para realizar solicitudes HTTP al servidor para consumir servicios de una API REST.

Figma:
Herramienta de diseño digital utilizada para la creación de prototipos de interfaces UX/UI, permitiendo visualizar el diseño antes de su desarrollo funcional.

Frontend:
Parte visual del sistema con la que interactúan los usuarios finales. En este proyecto fue desarrollado utilizando React 19.

Git:
Sistema de control de versiones distribuido que permite gestionar cambios en el código fuente del proyecto.

GitHub:

Plataforma en la nube utilizada para alojar repositorios Git y facilitar el trabajo colaborativo entre desarrolladores.

Hook:

Función especial de React que permite utilizar estado y otras funcionalidades internas dentro de componentes funcionales.

JWT (JSON Web Token):

Mecanismo de autenticación basado en tokens que permite verificar la identidad de los usuarios y proteger las rutas del sistema.

Middleware:

Componente intermedio que se ejecuta entre la solicitud del cliente y la respuesta del servidor, utilizado para validaciones y control de acceso.

Node.js:

Entorno de ejecución de JavaScript del lado del servidor, utilizado para el desarrollo del backend del sistema.

Payload:

Conjunto de datos enviados desde el frontend al backend dentro de una petición HTTP.

React 19:

Biblioteca de JavaScript utilizada para el desarrollo de interfaces de usuario basadas en componentes reutilizables.

REST:

Estilo de arquitectura para el diseño de servicios web que permite la comunicación entre sistemas mediante métodos HTTP.

Rol de Usuario:

Conjunto de permisos asignados a un usuario dentro del sistema según su nivel de acceso y funciones permitidas.

Ruta Protegida:
Sección del sistema a la que solo pueden acceder usuarios autenticados mediante validaciones de seguridad.

useEffect:
Hook de React que permite ejecutar efectos secundarios como consumo de APIs, actualizaciones de datos y control del ciclo de vida de los componentes.

useState:
Hook de React que permite gestionar el estado de los componentes funcionales.

UX/UI:
UX (User Experience) se refiere a la experiencia del usuario al interactuar con el sistema, mientras que UI (User Interface) se enfoca en el diseño visual de la interfaz.

Validación de Datos:
Proceso de verificación de la información ingresada por el usuario para asegurar que cumpla con los requisitos establecidos.

Videovigilancia:
Sistema de monitoreo por cámaras utilizado para la seguridad física de instalaciones.

Zod:
Biblioteca de validación de esquemas utilizada para asegurar que los datos enviados al backend cumplan con la estructura y reglas definidas.

Conclusiones

La pasantía profesional desarrollada en RT Network S.A.S. de C.V., durante el período comprendido entre abril y octubre de 2025, permitió cumplir de manera satisfactoria los objetivos establecidos en el plan de trabajo, fortaleciendo significativamente las competencias técnicas en el área de desarrollo frontend y contribuyendo de forma directa a la implementación del Sistema de Inventario, Ventas y Facturación de la empresa.

Se consolidó un dominio técnico avanzado en React 19, abarcando aspectos clave como la gestión de estado, la optimización del rendimiento y la correcta arquitectura de componentes. Asimismo, la integración de bibliotecas especializadas como Zod permitió elevar los estándares de calidad del código desarrollado, fortaleciendo los procesos de validación de datos y asegurando la confiabilidad de la información manejada por el sistema.

La implementación de módulos completos bajo el esquema CRUD, mediante el consumo de APIs RESTful, permitió desarrollar competencias sólidas en la comunicación cliente–servidor, el manejo de peticiones asíncronas y la sincronización de datos en tiempo real. Este proceso fue determinante para transformar las interfaces visuales en herramientas plenamente funcionales, alineadas a los procesos operativos de la empresa.

El trabajo realizado en el área de UX/UI, tanto en el sistema de inventario como en proyectos externos, fortaleció la capacidad de diseñar interfaces intuitivas, accesibles y visualmente coherentes. La aplicación de principios de diseño centrado en el usuario permitió equilibrar de forma adecuada la funcionalidad, la estética y la experiencia de uso.

La aplicación de buenas prácticas profesionales, como el uso de Git/GitHub para el control de versiones, la refactorización continua del código y la realización de pruebas funcionales permitió mantener un flujo de trabajo ordenado, documentado y eficiente, facilitando el trabajo colaborativo y el mantenimiento del sistema.

Se alcanzó un 100 % de cumplimiento de los objetivos planteados en el plan de pasantía, desarrollándose con éxito las tres fases establecidas: maquetación de interfaces, integración funcional y optimización del sistema, evidenciando un crecimiento progresivo y resultados concretos en cada etapa del proceso.

En términos generales, la pasantía profesional constituyó una experiencia formativa integral, que permitió aplicar los conocimientos teóricos en un entorno real de trabajo, desarrollar competencias técnicas y profesionales fundamentales, y obtener una visión clara del ejercicio profesional de la Ingeniería de Sistemas Informáticos, estableciendo bases sólidas para el futuro desempeño en el campo de la tecnología.

Recomendaciones

Implementación de Pruebas Automatizadas:

Se recomienda incorporar un framework de pruebas automatizadas, como Jest y React Testing Library, con el fin de garantizar la estabilidad y confiabilidad del Sistema de Inventario, Ventas y Facturación a largo plazo. La aplicación de pruebas unitarias y de integración permitirá disminuir de forma considerable el riesgo de errores durante la incorporación de nuevas funcionalidades o en procesos de refactorización del código existente.

Documentación Técnica Formal:

Aunque el código desarrollado cumple con buenas prácticas de estructura y nomenclatura, se sugiere elaborar una documentación técnica formal que incluya diagramas de arquitectura del sistema, guías de estilo de código y documentación detallada de los componentes. Esto facilitará la incorporación de nuevos desarrolladores, mejorará la mantenibilidad y fortalecerá la continuidad del proyecto a futuro.

Capacitación de Usuarios Finales:

Se recomienda diseñar un programa de capacitación para los usuarios finales del sistema, que contemple la elaboración de manuales de usuario, videos tutoriales y sesiones de entrenamiento presenciales. Estas acciones permitirán un mayor aprovechamiento del sistema, reducirán la curva de aprendizaje y contribuirán a una correcta adopción de la herramienta por parte del personal.

Referencias

Atlassian. (s.f.). What is version control. Git Tutorial. Recuperado el 30 de octubre de 2025, de https://www.atlassian.com/git/tutorials/what-is-version-control

GeeksforGeeks. (2025, 23 de julio). React Component Based Architecture. Recuperado el 30 de octubre de 2025, de https://www.geeksforgeeks.org/reactjs/react-component-based-architecture/

Git. (s.f.). Git documentation. https://git-scm.com/book/en/v2

GitHub. (2025). GitHub. https://github.com/

GitLab. (s.f.). What are Git version control best practices?. Recuperado el 30 de octubre de 2025, de https://about.gitlab.com/topics/version-control/version-control-best-practices/

Kothari, S. (2025, 12 de marzo). What is REST API and Its Architecture?. Medium.

Recuperado el 30 de octubre de 2025, de https://medium.com/@smita.s.kothari/what-is-rest-api- and-its-architecture-90b49aa6c7a9

React Team. (2024). React Documentation. https://react.dev/

ResearchGate. (2022, junio). React: A detailed survey. Indonesian Journal of Electrical Engineering and Computer Science, 26(3), 1710-1717. Recuperado el 30 de octubre de 2025, de https://www.researchgate.net/publication/361085649_React_A_detailed_survey

ResearchGate. (2024, 15 de febrero). The Impact of TypeScript on Developer Productivity: Does Static Typing Really Help?. Recuperado el 30 de octubre de 2025, de https://www.researchgate.net/publication/389650048_The_Impact_of_TypeScript_on_Developer_Productivity_Does_Static_Typing_Really_Help

SitePoint. (2024, 11 de noviembre). An Introduction to TypeScript: Static Typing for the Web. Recuperado el 30 de octubre de 2025, de https://www.sitepoint.com/introduction-to- typescript/

Tailwind Labs. (2024). Tailwind CSS. https://tailwindcss.com/

Theneo. (2024, 2 de octubre). Understanding the Benefits of a REST API. Theneo Blog. Recuperado el 30 de octubre de 2025, de https://www.theneo.io/blog/understanding-the-benefits- of-a-rest-api-803fc

Visual Studio Code. (2021, 3 de noviembre). Visual Studio Code. https://code.visualstudio.com/

Visto Bueno del Tutor de la Institución

Jefe de la Unidad de tecnología de la información

Firma: 

Ing. José Elias Romero Bonilla
CEO y Representante Legal
RT Network S.A.S. de C V

Anexos

Figura 17. Constancia de finalización de la pasantía

San Francisco Gotera, Morazán, 20 de octubre de 2025

Inga. Milagro Alicia González de Reyes
Coordinadora de Procesos de Graduación
Departamento de Ingeniería y Arquitectura
Facultad Multidisciplinaria Oriental

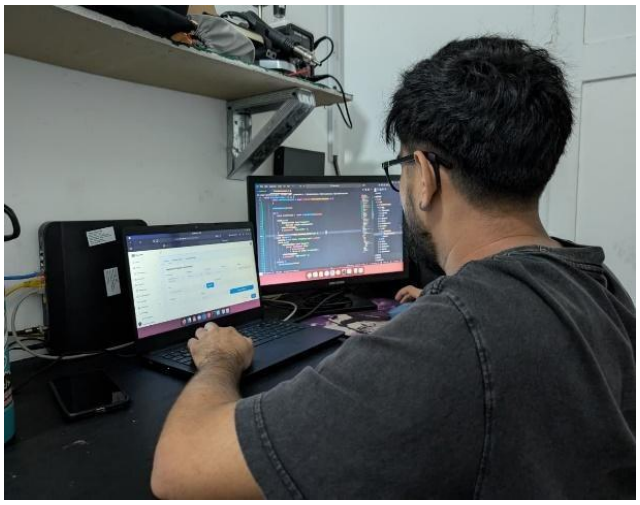
Presente.

El suscrito, **José Elías Romero Bonilla**, de **RT NETWORK S.A.S. DE C.V.**, **HACE CONSTAR** que el Br. **Roberto Hernán Laínez Trejo** ha realizado satisfactoriamente su **Pasantía de Práctica Profesional** en el proyecto **"Desarrollo e Integración de Soluciones de Software"**, habiendo iniciado en la fecha comprendida del **12 de abril de 2025** al **20 de octubre de 2025**, por un total de **seis meses**.

Y para los efectos que el interesado estime convenientes, se firma y sella la presente en **San Francisco Gotera, Morazán**, a las **3:00 p.m.** horas del día **20 de octubre del año 2025**.

F:_____
José Elías Romero Bonilla
Representante Legal / Responsable de Pasantía
RT NETWORK S.A.S. DE C.V.

Figura 18 Desarrollo de interfaz del módulo de entradas

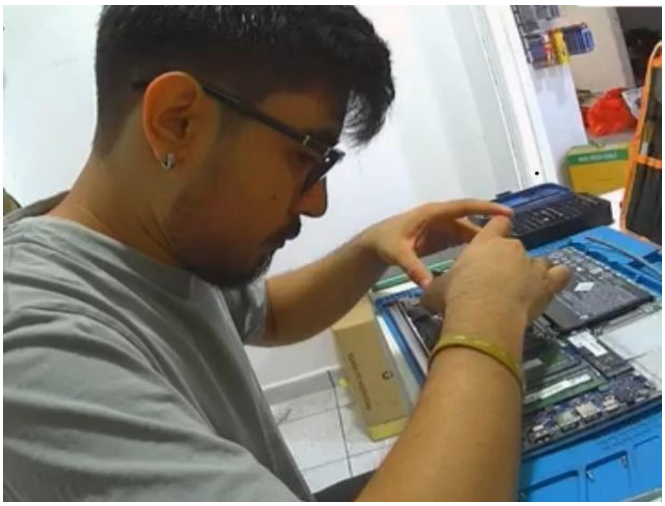


Nota: En ese día de trabajo, se estaba realizando modificaciones al módulo de entradas e implementando el CRUD, para poder registrar nuevas entradas de producto y actualizar su stock.

Figura 19 instalación de cámaras de video vigilancia

Nota: instalación de nuevas cámaras de video vigilancia

Figura 20. Mantenimientos Preventivos



Nota: Se realizo mantenimiento preventivo a equipos, haciendo limpieza de las piezas de componentes y cambio de pasta termina