TRABAJO DE GRADO

Opción Proyecto de Grado

FourShop: Diseño y desarrollo de una aplicación móvil para la compra y venta de productos, implementada con una arquitectura desacoplada (Django/Kotlin)

Corporación Universitaria Remington.

Facultad de ingenierías

Ingeniería de Sistemas.

Jose Luis Agredo Hoyos

Kevin Sánchez Silva

Docente: Gloria Amparo Lora Patiño

Opción Proyecto de Grado

2025

Tabla de Contenido

Lista de tablas

Lista de ilustraciones

Resumen

FourShop es una aplicación móvil de comercio electrónico (E-commerce) diseñada para impulsar a pequeños emprendimientos que buscan darse a conocer en el mercado digital, ofreciendo una plataforma de fácil acceso donde dichos emprendimientos pueden exhibir, gestionar y vender sus productos de manera sencilla. Este sistema permite a los vendedores registrarse, administrar su inventario, establecer precios y aplicar descuentos a los productos previamente registrados y por otro lado los usuarios cuentan con un catálogo variado, un carrito de compras y la posibilidad de finalizar pedidos con un resumen detallado de su compra.

El desarrollo de la aplicación se llevó a cabo bajo la metodología ágil Scrum, mediante sprints cortos, seguimiento y priorizando las necesidades tanto de vendedores como de compradores. La arquitectura tecnológica integra un Backend de desarrollo en Django (Python), para la gestión lógica de negocio y base de datos, junto con una interfaz móvil creada en Android Studio con Kotlin, para garantizar una experiencia fluida y confiable.

La interfaz de usuario está diseñada con base en dos perfiles, uno como vendedor, que ofrece un panel de control para la gestión de los productos, pedidos y ventas, mientras que el perfil usuario brinda una navegación intuitiva, la opción de seleccionar las cantidades de productos antes de añadir al carrito, resumen y visualización de compras realizadas.

Palabras clave: Django, Kotlin, Interfaz, Gestión, Intuitiva.

Introducción

El Comercio electrónico se ha consolidado como uno de los pilares de la economía digital en el siglo XXI, ofreciendo nuevas oportunidades de crecimiento a negocios de todos los tamaños. Sin embargo, para los pequeños emprendimientos acceder a plataformas tradicionales los limitan debido a los altos costos y poca visibilidad en el mercado lo cual reduce sus posibilidades de expansión. En Colombia donde la mayor parte de iniciativas comerciales provienen de pequeñas empresas, esta situación se convierte en un obstáculo significativo en términos de competitividad.

Debido a lo anterior surge FourShop, una aplicación de comercio electrónico enfocada en pequeños emprendimientos que desean darse a conocer y conectar de una manera más rápida con sus clientes. La plataforma ofrece un entorno sencillo y de fácil manejo para la publicación de productos, gestión de inventario y administrar los pedidos; donde a su vez les brinda a los usuarios una experiencia de compra práctica, segura e intuitiva.

En este trabajo se documenta el proceso completo análisis, diseño, desarrollo y validación de FourShop, donde el sistema se desarrolló bajo la metodología ágil Scrum, utilizando Django (Python) para el Backend y Android Studio con Kotlin para la interfaz móvil, lo que asegura un producto escalable y alineado con las necesidades reales de los vendedores y compradores.

Marco teórico

El comercio electrónico, también conocido como e-Commerce (electronic commerce o e-Commerce), consiste en la compra y venta de productos o de servicios a través de medios electrónicos, tales como Internet y otras redes informáticas. Originalmente el término se aplicaba a la realización de transacciones mediante medios electrónicos tales como el Intercambio electrónico de datos, sin embargo, con el advenimiento de la Internet y la World Wide Web a mediados de los años 90 comenzó a referirse principalmente a la venta de bienes y servicios a través de Internet, usando como forma de pago medios electrónicos, tales como las tarjetas de crédito. (Balado, 2005)

La mayor parte del comercio electrónico consiste en la compra y venta de productos o servicios entre personas y empresas, sin embargo, un porcentaje considerable del comercio electrónico consiste en la adquisición de artículos virtuales (software y derivados en su mayoría), tales como el acceso a contenido Premium de un sitio web. (Trejo, 2019)

Tipos de comercio electrónico

Existen varias maneras de clasificar las distintas formas de comercio electrónico atendiendo al tipo de relación que se establece entre el comprador y el vendedor, a continuación, se mencionan los tipos más comunes en el comercio electrónico.

Tipo de Comercio Electrónico	Descripción
B2B o **business to business**	Se refiere al comercio electrónico entre empresas. Abarca, tanto el comercio electrónico de bienes o servicios, como las transacciones de información relacionada con procesos comerciales entre empresas, lo que han llamado e-business.

	Es una evolución de los procesos de intercambio electrónico de datos, ya existentes antes de la generalización en el empleo de internet como plataforma para realizar negocios.
B2C o business to Consumer	Se refiere al comercio electrónico entre empresas y consumidores finales. Es lo que normalmente todo el mundo entiende por comercio electrónico.
C2C o Consumer to Consumer	Se refiere al comercio electrónico entre consumidores finales en donde unos actúan como vendedores y otros como compradores. El ejemplo típico con las subastas a través de internet en sitios como eBay.
C2B o Consumer to business	Se refiere a comercio electrónico entre consumidor y empresas, en las que el consumidor, o un grupo de ellos, emplean internet para conseguir productos a mejores precios o con mejores condiciones. La forma usual de funcionamiento es una especie de puja en la que los usuarios hacen una petición y las empresas hacen las ofertas.

Tabla 1: Tipos de comercio electrónico

Nota. Adaptado de La nueva era del comercio: el comercio electrónico (Balado, 2005)

Ventajas y desventajas de comercio electrónico

COMERCIO ELECTRONICO	
VENTAJAS	**DESVENTAJAS**
La variedad de productos en un solo lugar le permite al consumidor comparar productos y ofertas de manera más fácil y le asegura una reducción de costos en el proceso de búsqueda.	La mayor desventaja proviene de la falta de confianza en los mecanismos tecnológicos, que es uno de los principales obstáculos para el desarrollo del comercio electrónico.
Los costos del vendedor online se reducen y por ende puede ofrecer mejores precios y otros beneficios, lo cual se traduce en beneficios al consumidor.	Una de las desventajas a los que se enfrenta el consumidor electrónico es en lo que tiene que ver con la idónea selección el producto, ya que el consumidor se expone a lo que realmente quiere y lo que realmente obtiene.
La accesibilidad a cualquier tienda online permite la democratización del consumo, ya que el acceso a ellas y su éxito depende de la versatilidad del sitio web, de su facilidad de uso independientemente de su apariencia y su estrato.	El bajo acceso de las personas a las tic, así como el desconocimiento de la forma en cómo funcionan, lo cual puede generar desconfianza de si está realizando negocios con determinada persona y no con otra o si dicha persona existe.

Tabla 2:Ventajas y desventajas en el comercio electrónico

Nota. Adaptado de Protección al consumidor electrónico en Colombia (Borrero, 2021)

Tecnologías usadas dentro del proyecto

Django: Es un framework web del lado del servidor creado en Python y de alto nivel, que promueve un desarrollo rápido, un diseño limpio, funcional, modular y escalable. Al ser un framework web con características básicas, podemos esperar encontrar en él componentes como autenticación, manejo y validación de formularios, panel administrativo, recursos rest, manejo y enlaces de rutas y muchas otras funciones más. (Yoris, 2024)

Visual Studio Code: Es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y web. Incluye soporte técnico para la depuración, control integrado GIT, resaltado de sintaxis en colores y la identacion, este aspecto hace referencia a la sangría que existe perfectamente demarcada en cada línea de programación inteligente de código (autocompletado de fragmentos de código, refactorización de código y ajuste de la sintaxis o formas de escritura en algunos casos limitado). Es un editor de código abierto, libre y multiplataforma. (milaulas.com, 2019)

Python: Es un lenguaje de programación de alto nivel e interpretado, orientado a objetos y con una sintaxis muy limpia, amigable, fácil de aprender y de gran legibilidad con una gran cantidad de paquetes, módulos, librerías y frameworks a nuestra disposición lo hacen muy atractivo para el desarrollo rápido de aplicaciones. (Yoris, 2024)

Android Studio: Es un sistema operativo al igual que Windows, Linux entre otros. Por lo tanto, tiene el control total del dispositivo que lo contiene, así que cuando se desarrolla una aplicación, se está desarrollando para el sistema operativo y se puede tener control sobre ciertos elementos que generan una experiencia más interesante y agradable para el usuario si se hace de buena forma, en caso contrario se podría estar generando una

de las peores experiencias de usuario y afectando el rendimiento del dispositivo. (Castillo, 2017)

Kotlin: Es un lenguaje de programación creado en 2010 por JetBrains, la cual es una alternativa a Java, que suple varios de los problemas más habituales que los programadores se encuentran en dicho lenguaje, además Kotlin es un lenguaje oficial para el desarrollo en Android, por lo que su uso se ha disparado y actualmente las empresas buscan desarrolladores Kotlin ya que es un lenguaje superior a Java, debido a que es seguro contra nulos, ahorra código y es de fácil uso. (Arístides, 2018)

SQLite: Es un paquete de software de dominio público que proporciona un sistema de gestión de bases de datos relacionales. Los sistemas de bases de datos relacionales se utilizan para almacenar registros definidos por el usuario en tablas grandes, además del almacenamiento y la gestión de bases de datos, un motor de base de datos puede generar informes y resumir datos. SQLite no requiere un servidor o sistema separado para operar. La biblioteca SQLite accede directamente a sus archivos de almacenamiento, tiene configuración cero ya que es sin servidor, toda la instancia de base de datos reside en un solo archivo multiplataforma y no requiere administración y la compilación predeterminada ocupa menos de un megabyte de código y requiere solo unos pocos megabytes de memoria. (Kreibich, 2010)

DBeaver: Es un entorno profesional para trabajar con bases de datos, usada por desarrolladores que necesitan gestionar información de forma visual y eficiente, la cual permite ver, editar y analizar datos almacenados en bases como: SQLite, MySQL, PostgreSQL entre otras. (Kreibich, 2010)

Metodologías de desarrollo ágil

Las metodologías de desarrollo agiles están planteadas para afrontar el problema de los requerimientos inciertos o variables de las entregas tempranas, que tienen como objetivo satisfacer los requerimientos esenciales del cliente. La metodología ágil que se seleccione depende de cada proyecto, según las necesidades particulares, estas están especialmente indicadas en proyectos con requisitos poco definidos o cambiantes. Estas metodologías se aplican bien en equipos pequeños que resuelven problemas concretos, lo que no está reñido con su aplicación en el desarrollo de grandes sistemas, ya que con una correcta modularización de los mismos es fundamental para su exitosa implantación. Dividir el trabajo en módulos abordables minimiza los fallos y el coste. (Dámaris, 2007)

Ventajas

- Capacidad de respuesta a cambios de requisitos a lo largo del desarrollo
- Entrega continua y en plazos breves de software funcional
- Trabajo conjunto entre el cliente y el equipo de desarrollo
- Importancia de la simplicidad, eliminando el trabajo innecesario
- Atención continua a la excelencia técnica y al buen diseño
- Mejora continua de los procesos y el equipo de desarrollo

Extreme Programming – XP

XP es la primera metodología ágil y la que dio conciencia al movimiento actual de metodologías agiles. De la mano de Kent Beck, XP ha conformado un extenso grupo de seguidores en todo el mundo, disparando una gran cantidad de libros a los que dio comienzo el mismo Beck. XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y proporcionando un buen clima de trabajo. Se basa en la realimentación continua entre el

cliente y el equipo de desarrollo, comunicación fluida entre los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. (Dámaris, 2007)

Scrum

El termino Scrum en la gestión del proyecto se describe como "una estrategia flexible y logística de desarrollo de productos, donde un equipo de desarrollo trabaja como una unidad para alcanzar un objetivo común", Scrum adopta plenamente los principios de los métodos agiles de desarrollo y los incorpora a la gestión de proyectos. Se centra en la mejora de la capacidad del equipo de desarrollo para observar y adaptarse a las nuevas exigencias. (Blokehead, 2016)

Estado del arte

En la actualidad, las aplicaciones móviles de comercio electrónico han transformado significativamente la manera en que los usuarios compran y venden productos. Plataformas como Mercado Libre, Rapi, Shopee o Amazon, han facilitado la conexión entre vendedores y compradores mediante diversos sistemas seguros y eficientes. Sin embargo, estas soluciones están orientadas a grandes comercios o marcas ya consolidadas, dejando de lado las micro, pequeñas y medianas empresas que requieren herramientas más simples y de fácil acceso.

Teniendo en cuenta este contexto surge FourShop, una aplicación móvil enfocada en potenciar la participación de las MiPymes en el comercio electrónico mediante un sistema funcional, seguro y que se adapta a las necesidades específicas que estas empresas.

A continuación, se presenta un análisis de las principales plataformas de referencia.

Aplicación	descripción	Fortalezas	Debilidades
Mercado Libre	Es una de las plataformas lideres en comercio electrónico en América Latina. Permite la Compra y venta de productos nuevos y usados, ofreciendo herramientas de pago, envió y reputación de vendedores.	Amplia base de usuarios, sistemas de pago integrados (Mercado Pago) y seguridad en las transacciones.	Interfaz compleja para nuevos usuarios y altas comisiones para pequeños vendedores.
Rappi	Inicialmente enfocada en entregas a domicilio, ha expandido su oferta al comercio electrónico con la venta de productos, servicios	Envíos rápidos, integración con múltiples servicios (supermercados,	No está diseñada específicamente para pequeños emprendedores, y su

	financieros y membresías	farmacias, restaurantes).	uso depende de la ubicación geográfica.
Shopee	Es una ampliación internacional con presencia en América Latina que combina comercio electrónico y recompensas.	Interfaz sencilla, promociones frecuentes y métodos de pago variados.	Dependencia de proveedores internacionales, tiempos de entrega extensos y escaso soporte local.
Amazon	Es una plataforma global de comercio electrónico que ofrece una amplia variedad de productos, servicios de suscripción y logística avanzada.	Alto nivel de confianza, soporte al cliente eficiente y velocidad de envíos.	Competencia elevada para nuevos vendedores y altos costos por comisión o membresía.

Tabla 3: Análisis de competencia

Este análisis comparativo evidencia que las actuales aplicaciones de comercio electrónico cubren ampliamente el mercado global, pero presentan limitaciones para las pequeñas empresas o vendedores independientes que quieren iniciar en el e-commerce.

FourShop

FourShop busca diferenciarse al enfocarse específicamente en micro, pequeñas y medianas empresas colombianas. La aplicación integrara una interfaz móvil intuitiva y moderna, con herramientas de gestión de productos e inventarios, funcionalidades como notificaciones, una mejor comunicación entre comprador y vendedor, y un seguimiento de estado de pedidos.

Además, FourShop prioriza la usabilidad, seguridad, y bajo costo operativo, facilitando la digitalización de negocios locales que tradicionalmente no tienen acceso a plataformas complejas o costosas.

Planteamiento del problema

En los últimos años, el comercio electrónico ha venido posicionándose como una herramienta estratégica fundamental para la reactivación económica y el crecimiento sostenible de las micro, pequeñas y medianas empresas (MiPymes) en Colombia. Según cifras del Departamento Administrativo Nacional de Estadística (DANE), las Pymes representan más del 90% del tejido empresarial colombiano y generan alrededor del 80% del empleo y cerca del 35% del Producto Interno Bruto (PIB) del país, constituyendo el motor de la economía. (Arango, 2019)

La transformación digital es una urgencia para las empresas colombianas que buscan mantenerse competitivas en el siglo XXI. Aunque las grandes compañías han avanzado más en este proceso, las MiPymes que representan el 95% del tejido empresarial enfrentan barreras estructurales y culturales que dificultan su adopción.

Entre los principales retos que se destacan es la baja conectividad, especialmente en zonas rurales, falta de mentalidad digital, la escasa inversión en capacitación tecnológica y la resistencia al cambio. Según Mario Castaño, director técnico de Cintel, el índice de Madurez de Transformación Digital Empresarial fue de 51,5% en 2023. Pese al impulso que trajo la pandemia, solo el 7% de las iniciativas digitales fueron exitosas, lo que llevo a muchas empresas a replantear sus estrategias.

Las MiPymes son clave para el futuro digital del país, pero enfrentaran dificultades que trascienden la tecnología, las limitaciones de infraestructura se suman a la falta de cultura digital, desconocimiento de proveedores tecnológicos y escasez de talento especializado. (Semana, 2025)

Lo que nos lleva a pensar que, aunque el comercio electrónico crece y el uso de las apps para compras está aumentando el Colombia, muchas microempresas o pequeños emprendimientos no logran capturar ese mercado móvil, debido a eso surge la pregunta ¿Cómo desarrollar una aplicación móvil que ayude a mejorar a las MiPymes a competir de manera positiva en el mercado digital?

Objetivos

Objetivo General

Desarrollar una aplicación móvil que potencie la participación de las micro, pequeñas y medianas empresas de Colombia en el comercio electrónico, brindando una buena gestión de productos, ventas y clientes a través de una plataforma segura de fácil acceso y orientada a mejorar la experiencia de compra el usuario.

Objetivos específicos

Analizar la información recolectada con el fin de identificar mejoras que incrementen su usabilidad, seguridad y efectividad comercial.

Diseñar una interfaz móvil intuitiva y funcional que les permita a las empresas registrar, administrar y promocionar sus productos de manera eficiente para mejorar la experiencia de compra y venta.

Construir una versión funcional del prototipo que permita validar la usabilidad, el diseño visual y la interacción del usuario según su rol, garantizando que las operaciones de compra, venta y gestión de productos se realicen de manera intuitiva y eficiente.

Evaluar el desempeño y la usabilidad de la aplicación mediante pruebas piloto con vendedores y clientes, con el fin de identificar posibles mejoras en su funcionalidad y experiencia de uso.

Alcance

El alcance del presente trabajo se define como el diseño y desarrollo de una aplicación móvil de comercio electrónico (B2C), orientada a potenciar la participación de micro, pequeñas y medianas empresas en el entorno digital, demostrando la viabilidad técnica t funcional de una aplicación móvil que permita a los vendedores gestionar sus productos y ventas, y a los clientes realizar compras de manera sencilla, segura e intuitiva.

1. **Alcance en la fase de análisis y diseño**

    - **Definición del modelo de negocio:** Se identificarán los roles principales (comprador y vendedor), sus permisos y las interacciones dentro del sistema.
    - **Diseño arquitectónico y funcional:** Se elaborarán diagramas de casos de uso que representen las operaciones principales de la aplicación, un modelo relacional de la base de datos (SQLite) con las entidades necesarias para la gestión de usuarios, productos, gestión de productos y ventas.
    - **Definición de requerimientos:** Se especificarán los requerimientos funcionales y no funcionales del sistema, asegurando que cumpla con criterios de seguridad, usabilidad y rendimiento.

2. **Alcance en la fase de desarrollo e implementación**

    - **Desarrollo del Backend (Django):** Se implementará la lógica de negocio, la gestión de usuarios y autenticación, control de productos y pedidos; Se crearán endpoints mediante una API REST para la comunicación con la aplicación móvil.

- **Desarrollo del Frontend móvil (Kotlin / Android Studio):** Se desarrollará la aplicación móvil para Android, con una interfaz de usuario sencilla e intuitiva.
- **Gestión de datos y almacenamiento (SQLite):** Se integrará una base de datos ligera y eficiente para el manejo local de datos temporales, sincronizados con el servidor.
- **Interconexión Backend-Frontend:** Se garantizará una comunicación segura mediante peticiones HTTP y formato JSON, asegurando consistencia en el intercambio de información entre el servidor Django y la aplicación móvil.

3. Alcance en la fase de pruebas y validación

- **Pruebas funcionales:** Se realizarán pruebas unitarias y de integración para validar las operaciones principales (registro de usuario, inicio de sesión, gestión de productos, agregar productos al carrito, crear pedidos.
- **Validación de la comunicación:** Se verificará la estabilidad en la conexión entre la aplicación móvil y el servidor, comprobando el envío y recepción correcta de datos mediante API REST.

Actividades para resolver el objetivo específico numero 1

Para desarrollar el objetivo específico numero 1: “Analizar la información recolectada con el fin de identificar mejoras que incrementen su usabilidad, seguridad y efectividad comercial”, se consideró fundamental la etapa de análisis y levantamiento de información correspondiente a la fase inicial del ciclo de vida en el desarrollo de software.

En esta etapa se busca comprender las necesidades de los usuarios finales y los procesos que el sistema debe optimizar, de tal manera que la construcción de la ingeniería de requisitos constituya la base para garantizar que el producto desarrollado responda de manera adecuada a las expectativas tanto de los clientes como de los vendedores que van a hacer uso de la aplicación.

Para cumplir con este objetivo, se llevó a cabo un proceso de recolección de información mediante encuestas dirigidas a los usuarios, tanto compradores como vendedores, con el fin de conocer su experiencia en el uso de aplicación similares, sus preferencias en cuanto a navegación y la facilidad de compra y venta.

Encuesta

A continuación, encontrara la encuesta que se formuló para la recopilación de información, también las puede visualizar en el siguiente enlace.

Enlace de la encuesta: https://forms.office.com/r/95iuipPF8K

1. ¿Con qué frecuencia utiliza aplicaciones móviles para comprar o vender productos? *

- ◯ Todos los días
- ◯ Varias veces por semana
- ◯ Ocasionalmente
- ◯ Casi nunca

2. ¿Cuál es su rol principal al usar plataformas de comercio electrónico? *

- ◯ Comprador
- ◯ Vendedor
- ◯ Ambos

3. ¿Qué tan fácil le resulta navegar en una aplicación de compras en línea? *

- ◯ Muy fácil
- ◯ Fácil
- ◯ Difícil
- ◯ Muy difícil

Ilustración 1:Preguntas encuesta 1 a 3

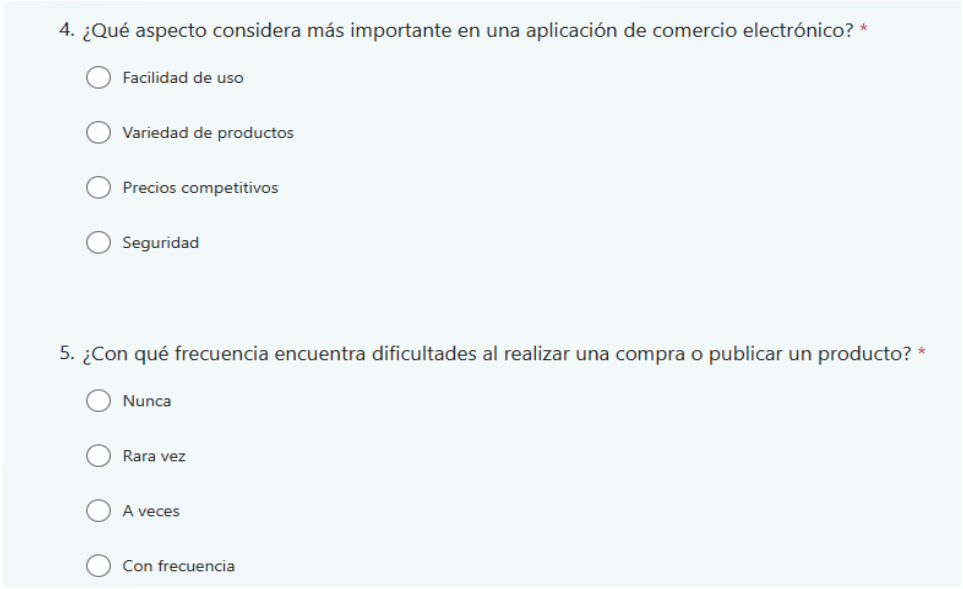


Ilustración 2: Preguntas encuesta 4 a 5

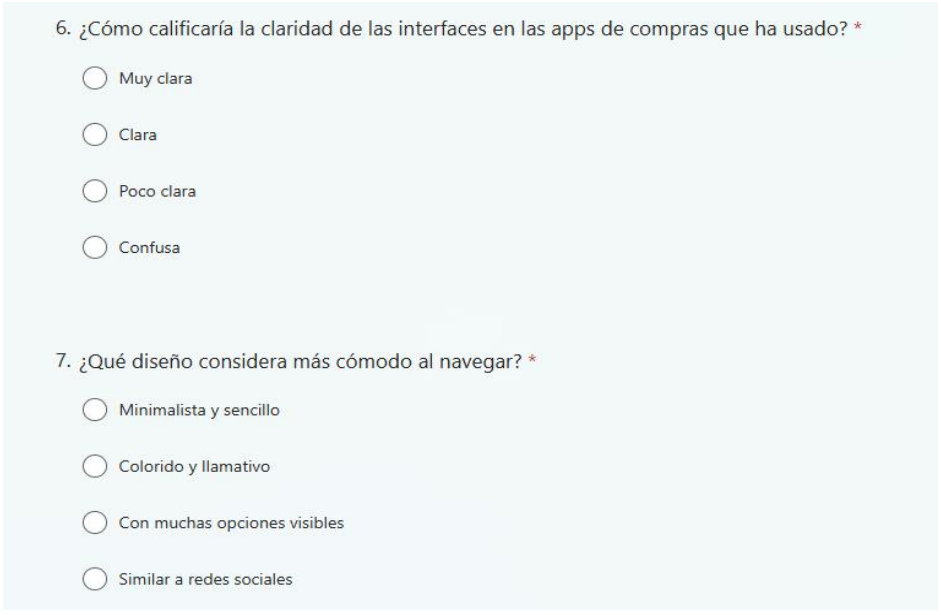


Ilustración 3: Preguntas encuesta 6 a 7

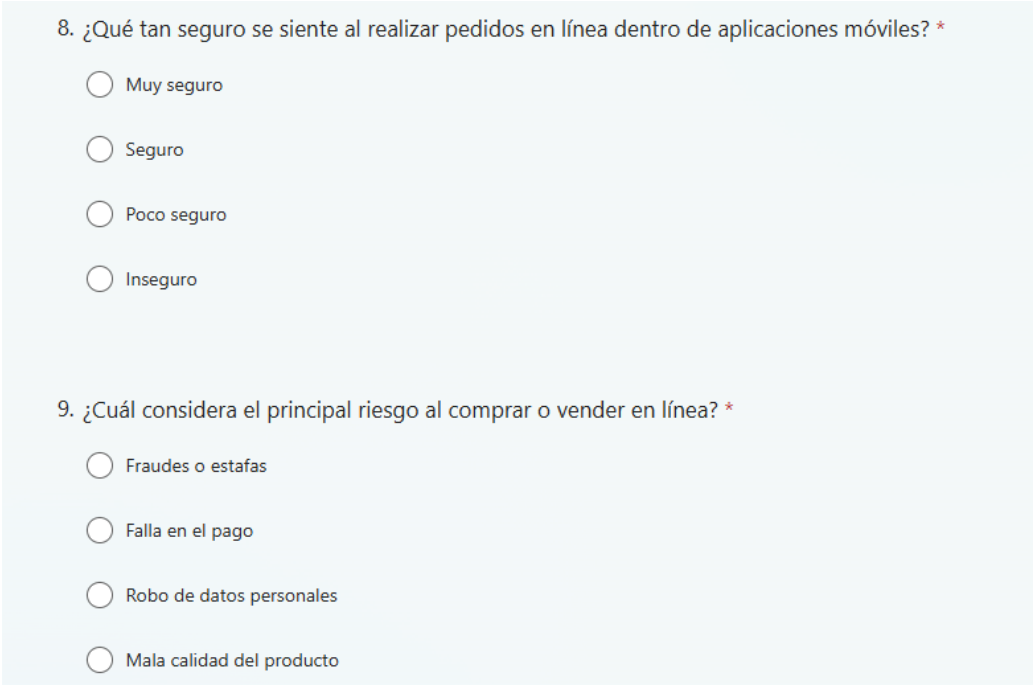


Ilustración 4: Preguntas encuesta 8 a 9

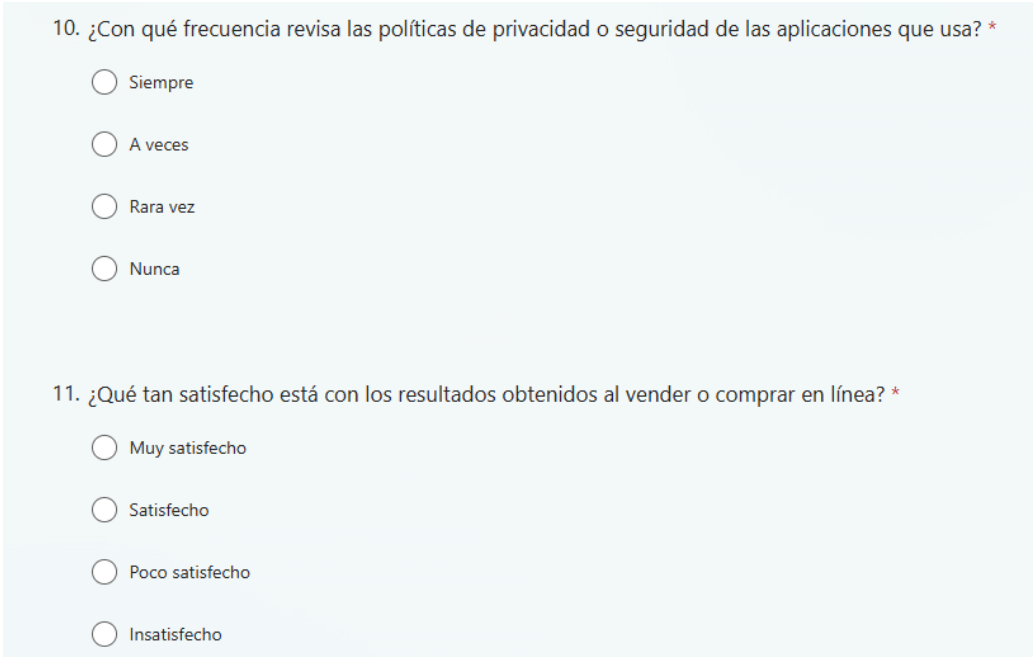


Ilustración 5: Preguntas encuesta 10 a 11

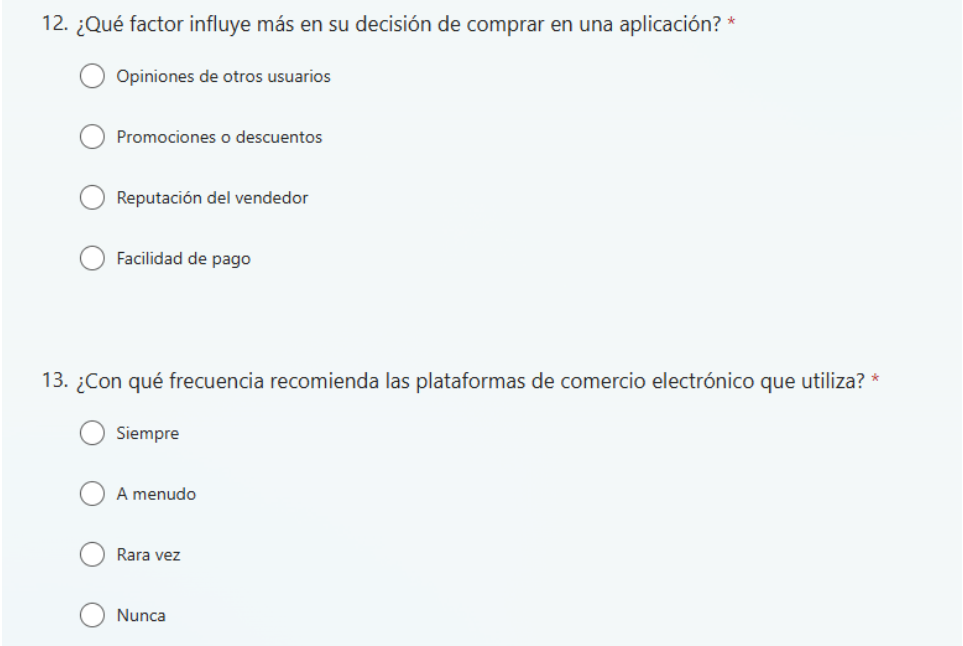


Ilustración 6: Preguntas encuesta 12 a 13

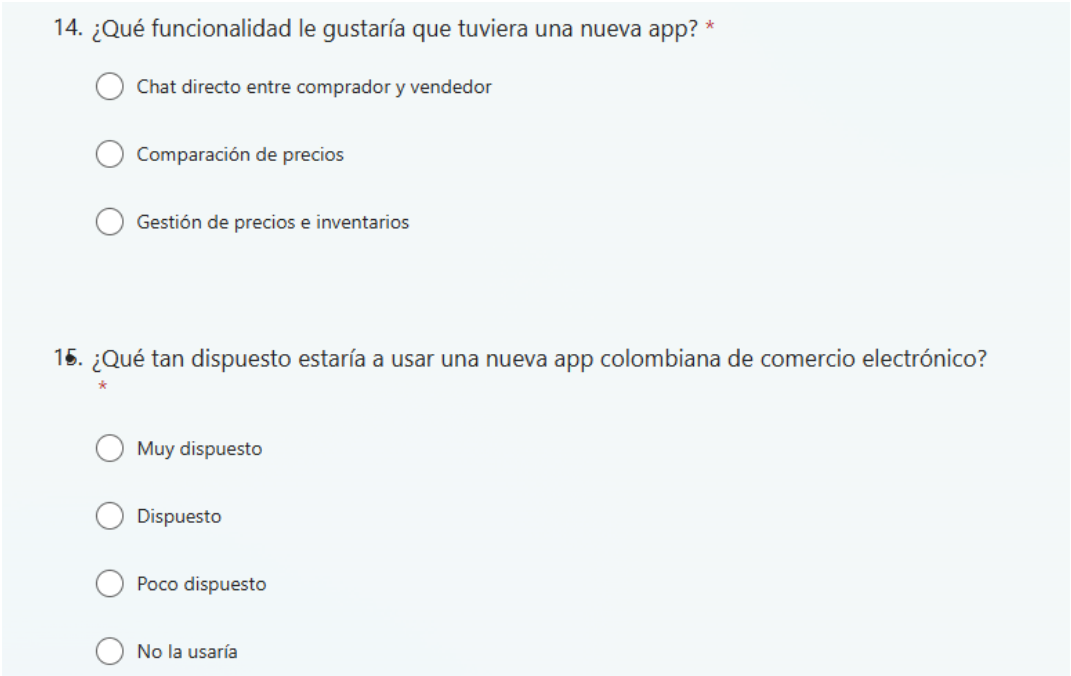


Ilustración 7: Preguntas encuesta 14 a 15

Resultados de la encuesta

La encuesta aplicada a 20 personas revela una diversidad en la relación con el comercio electrónico, donde se observa que un 40% de los participantes afirmo usar aplicaciones móviles varias veces por semana, mientras que un 15% adicional lo hace todos los días, lo que representa un 55% de usuarios activos, esto evidencia que más de la mitad de las personas encuestadas integran el comercio electrónico como parte habitual de sus rutinas de compra o venta de productos.

Por otra parte, un 30% manifestó usarlas ocasionalmente, lo cual da a entender que es un grupo de usuarios potenciales que podrían aumentar su frecuencia de uso si la aplicación les logra ofrecer una mejor experiencia y confianza. Finalmente tenemos un 15% que indico casi nunca emplear este tipo de plataformas, el cual representa un segmento menos familiarizado o con posibles barreras tecnológicas o que prefieren optar por el comercio tradicional.

Ilustración 8: Resultado encuesta pregunta 1

Por otra parte, los resultados reflejan que la mayoría de los participantes se desempeñan principalmente como compradores dentro de las plataformas de comercio electrónico, representando el 60% del total de los encuestados, lo cual evidencia que las aplicaciones móviles de este tipo son utilizadas en su mayoría como medios de consumo más que de comercialización directa.

Luego tenemos un 25% de los encuestados que indico cumplir ambos roles, tanto vendedor como comprador, lo cual aumenta la existencia de usuarios con mayor interacción dentro del ecosistema digital, aprovechando el uso de las plataformas para generar un intercambio comercial bidireccional. Por último, está el 15% restante que se identifica únicamente como vendedor, que, aunque es minoritario constituye un grupo relevante para el desarrollo de funcionalidades dentro de la aplicación, como la gestión de inventarios, el seguimiento de ventas.

Ilustración 9: Resultado encuesta pregunta 2

La mayoría de los encuestados (85%) considera que la navegación en aplicaciones de compras en línea es fácil o muy fácil, lo que evidencia una buena percepción de usabilidad, donde solo un 15% manifestó tener dificultades en la navegación dentro de una aplicación, lo que se podría relacionar con factores como experiencias con diseños poco intuitivos o poca familiaridad con la tecnología. Lo cual nos da como resultado que las aplicaciones deben mantener una estructura clara y sencilla para garantizar una experiencia positiva.

Ilustración 10: Resultado encuesta pregunta 3

Por otro lado, tenemos que el 60% de los participantes considera que la facilidad de uso es el aspecto más relevante en una aplicación, confirmando la importancia de la experiencia de usuario (UX) en el éxito de una aplicación de comercio electrónico. Aunque la variedad de productos también tuvo un valor significativo (30%). La prioridad sigue siendo la interacción intuitiva y fluida, lo que resalta que un diseño simple, organizado y rápido es la clave para la satisfacción del usuario.

Ilustración 11: Resultado encuesta pregunta 4

En términos de dificultad, el 80% de los encuestados afirma que rara vez o nunca tiene dificultades, lo que nos da a entender que las plataformas actuales resultan funcionales y eficientes. Sin embargo, el 20% que si presenta inconvenientes representa una oportunidad de mejora, en el registro o publicación de productos para garantizar un flujo de compra libre de errores que pueda aumentar la retención de usuarios.

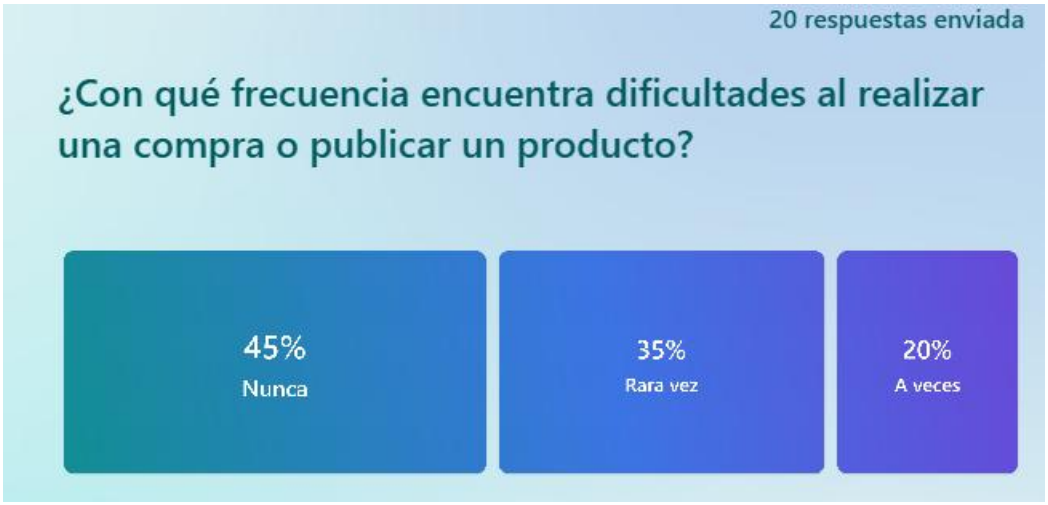


Ilustración 12: Respuesta encuesta pregunta 5

En la claridad de las interfaces, el 90% de los encuestado percibe que las interfaces de las aplicaciones son claras o muy claras, lo que refleja un diseño visualmente comprensible y bien estructurado en la mayoría de las aplicaciones. Solo un 10% indica confusión, lo que sugiere una coherencia visual y la jerarquía de información están siendo bien aplicadas, aunque se pueden realizar mejoras en iconografía o navegación secundaria.

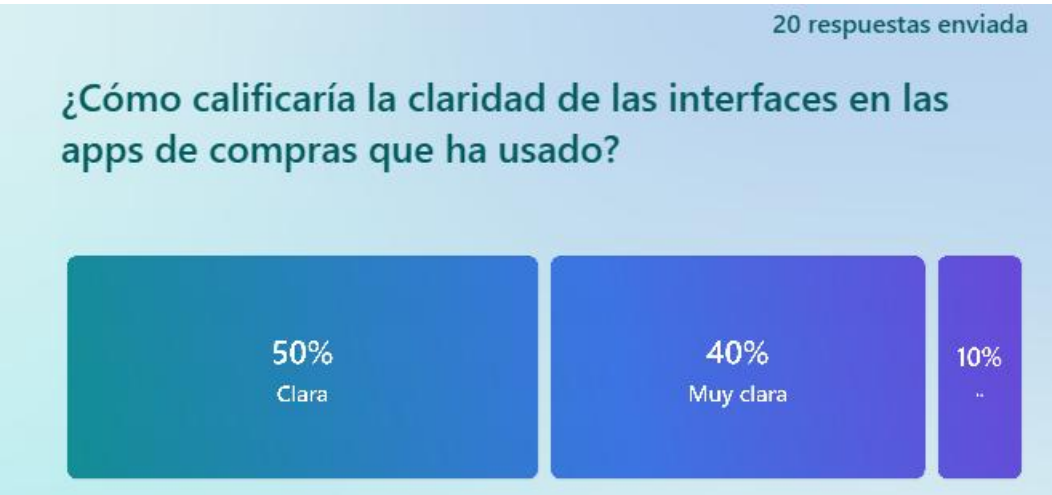


Ilustración 13: Respuesta encuesta pregunta 6

Respecto al diseño, los resultados evidencian con un 70% una clara preferencia de los usuarios por las interfaces con múltiples opciones visibles, esto sugiere que los participantes prefieren más la disponibilidad inmediata de funciones y la visibilidad de las secciones principales.

El 15% de los encuestados optó por un diseño minimalista y sencillo, lo que indica que prefieren interfaces limpias y sin tantos elementos en pantalla, dando prioridad a la simplicidad que, a la cantidad de opciones, Por su parte un 10% prefiere un diseño colorido y llamativo mientras que solo un 5% se inclina por un estilo similar al de las redes sociales. Dando a entender que la mayoría prefieren una funcionalidad practica por encima de lo estético o familiaridad con otros entornos digitales.

Ilustración 14: Resultado encuesta pregunta 7

En términos de seguridad, el 75% de los encuestados manifiesta sentirse seguro o muy seguro al realizar pedidos en línea, lo que refleja un alto nivel de confianza en las plataformas digitales y en los mecanismos de pago implementados. Un 25% manifiesta sentirse poco seguro lo cual puede deberse a la desconfianza que les pueda causar el uso de las aplicaciones o experiencias previamente vividas. Este resultado demuestra que las medidas de seguridad percibidas (cifrado de datos, autenticación o sistemas de pago reconocidos) están generando tranquilidad entre los usuarios. Solo un 5% que aun demuestra inseguridad representa una minoría que podría ser abordada mediante mayor transparencia en los procesos de compra y comunicación sobre políticas de seguridad.

Ilustración 15: Resultado encuesta pregunta 8

En términos de riesgo los encuestados se identifican con dos factores principales, el primero es fraudes o estafas con un 40% y mala calidad de los productos con un 40% respectivamente. Esto evidencia que, aunque confían en la seguridad técnica, todavía existen dudas sobre la confiabilidad de los vendedores y la veracidad de los productos ofrecidos. Los porcentajes menores como fallas en el pago o robo de datos reflejan una buena percepción de la seguridad tecnológica, pero también la necesidad de reforzar la reputación y transparencia de los comercios digitales.

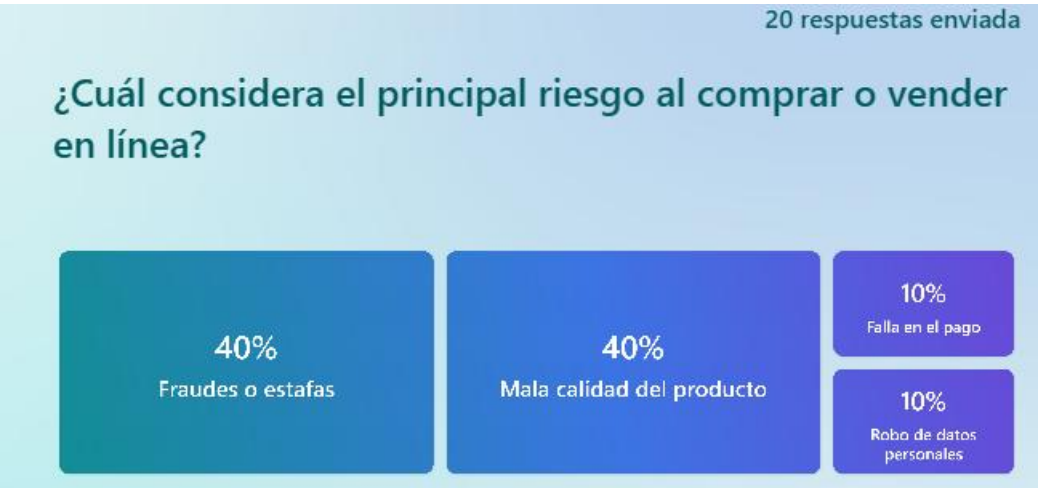


Ilustración 16: Resultado encuesta pregunta 9

En los resultados de la encuesta se vio reflejado que el 50% de los participantes nunca revisa las políticas de privacidad o seguridad, y un 35% solo lo hace ocasionalmente. Esto refleja una baja cultura de seguridad digital entre los usuarios, quienes priorizan la funcionalidad sobre la lectura de términos legales. Esta tendencia sugiere que las aplicaciones deben implementar métodos más visuales o simplificados para informar sobre sus medidas de protección, promoviendo la confianza sin requerir lectura extensa de políticas.

Ilustración 17: Resultado encuesta pregunta 10

En términos de satisfacción el 90% de los usuarios se declara satisfecho o muy satisfecho con sus experiencias de compra o venta, lo que indica que las plataformas están cumpliendo con las expectativas en términos de funcionalidad, rapidez y cumplimiento de pedidos. El 10% que refiere baja satisfacción puede deberse a casos específicos de mala experiencia, como demoras en la entrega o problemas con la calidad del producto. Este resultado demuestra una alta aceptación del comercio electrónico.

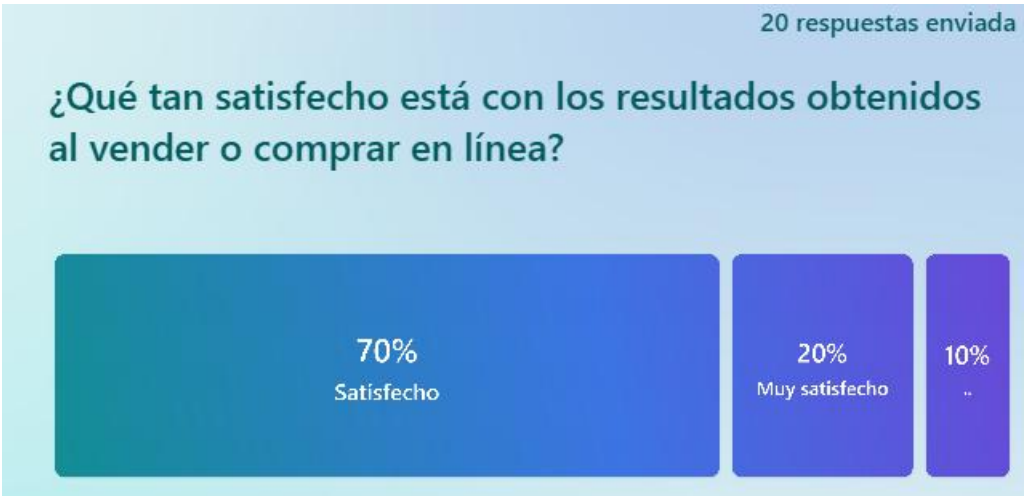


Ilustración 18: Resultado encuesta pregunta 11

En términos de influencia al momento de realizar compras en una aplicación los resultados muestran que el 65% de los participantes son influenciados principalmente por la existencia de promociones o descuentos, lo que da a entender que los compradores priorizan el ahorro económico y los incentivos comerciales a la hora de elegir una aplicación de comercio electrónico a la hora de realizar las compras.

Luego hay un 30% de los encuestados que señalo la facilidad de pago como un aspecto clave, lo cual indica la importancia de contar con métodos de pago agiles y seguros para poder mejorar la experiencia del usuario. Por último, hay un 5% restante que tiene en cuenta las opiniones de otros usuarios lo que indica que las opiniones externas no son decisivas en la mayoría de los casos.

Ilustración 19: Resultado encuesta pregunta 12

Los resultados obtenidos muestran una división en la percepción de satisfacción y fidelidad hacia las plataformas de comercio electrónico, donde un 45% sumando (siempre y a menudo) recomienda activamente las plataformas que usa, mientras que el 55% sumando (rara vez o nunca) no lo hacen con frecuencia. Esto sugiere que, aunque las experiencias generales son positivas, aún hay un margen para fortalecer la lealtad y el sentido de recomendación. Donde las razones principales podrían estar relacionadas con aspectos como la atención al cliente, tiempos de entrega o falta de funciones que mejoren la interacción con el vendedor.

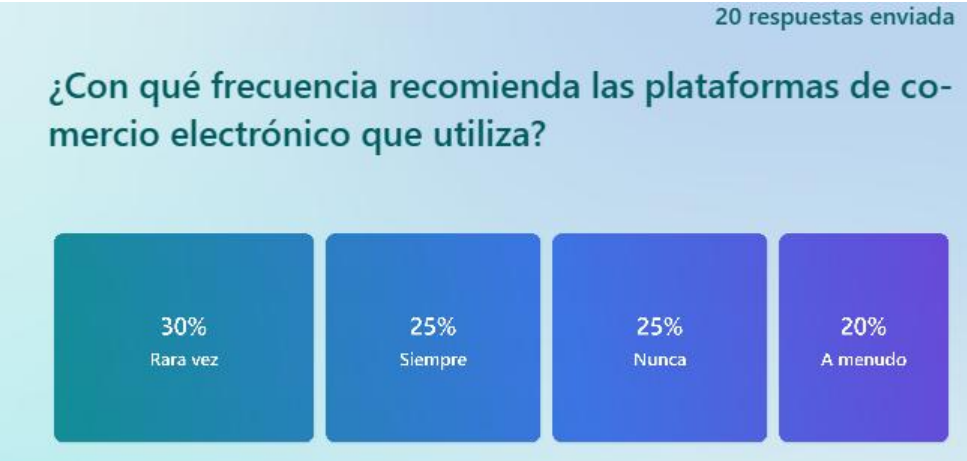


Ilustración 20: Resultado encuesta pregunta 13

Cuando se les pregunto a los encuestados sobre que les gustaría que tuviera una aplicación, el 40% priorizo la implementación de un chat directo entre vendedor y comprador, lo que indica una clara necesidad de comunicación fluida e inmediata dentro de las plataformas. Esto reforzaría la confianza, resolvería dudas antes de la compra y podría reducir casos de insatisfacción.

Por otro lado, un 35% considera importante la comparación de precios, reflejando el interés por obtener la mejor oferta y facilitar decisiones de compra informadas. Finalmente, el 25% que eligió gestión de precios e inventarios, lo que sugiere que una parte de los encuestados (probablemente vendedores) busca herramientas administrativas más eficientes para llevar un mejor control de sus productos.

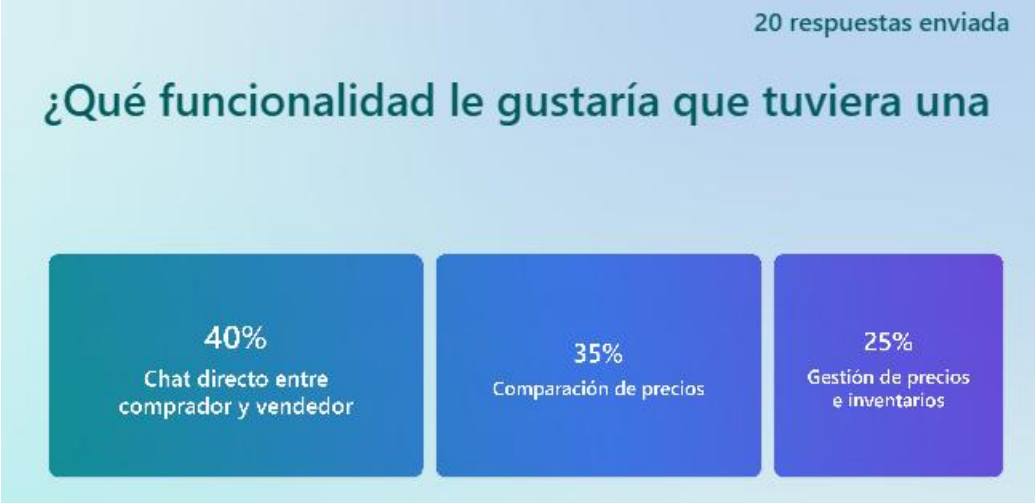


Ilustración 21: Resultado encuesta pregunta 14

Asimismo, la disposición para el uso de una nueva aplicación, la encuesta nos muestra un resultado de que el 80% de los participantes (sumando “dispuesto” y “muy dispuesto”) muestran una actitud positiva hacia el uso de una nueva aplicación de comercio electrónico, lo cual refleja una alta receptividad del mercado nacional hacia alternativas locales, lo que representa una oportunidad importante para el desarrollo e implementación de aplicaciones locales.

El 20% restante de los participantes que se declaró poco dispuesto, se puede interpretar como un grupo reservado o con desconfianza frente a nuevas plataformas, posiblemente por experiencias previas o preferencia por el uso de aplicaciones internacionales ya consolidadas.

Ilustración 22: Resultado encuesta pregunta 15

Conclusión de la encuesta

Los resultados de la encuesta aplicada reflejan una tendencia clara hacia la preferencia por aplicaciones de comercio electrónico que sean fáciles de usar, visualmente claras y seguras, de igual manera la mayoría de los participantes manifestó que navegar en una aplicación de compras resulta “fácil” o “muy fácil”, lo que da a entender que la claridad de la interfaz es un factor determinante para una experiencia positiva.

Estos hallazgos justifican la decisión de diseño de priorizar una interfaz limpia, intuitiva y con navegación simplificada, donde los elementos visuales y las funciones (búsqueda, compra, carrito y contacto) sean de rápido acceso y comprensión inmediata. En cuanto a seguridad y confianza a pesar de que la mayoría de los participantes indico sentirse seguro al comprar en línea, sigue habiendo temores relacionados con fraude y calidad del producto, lo que lleva a la implementación de protocolos de autenticación seguros y el despliegue visible de la información sobre la protección de datos del usuario. Para la comunicación y control dentro de la aplicación se ve reflejada la importancia de una comunicación directa implementando herramientas de interacción en tiempo real, junto con módulos de comparación y gestión de productos, con el fin de fortalecer la transparencia y la confianza entre ambas partes.

La encuesta evidencia que las decisiones de diseño de FourShop deben centrarse en:

- Usabilidad y simplicidad.
- Claridad visual.
- Confianza y seguridad.
- Interactividad.
- Fidelización y personalización.

Identificación de usuarios claves

En este proyecto de una aplicación móvil de comercio electrónico orientada a fortalecer la participación de las micro, pequeñas y medianas empresas colombianas, se identifican tres grupos principales de usuarios, donde cada uno desempeña un papel esencial dentro del ecosistema digital de compras y ventas, contribuyendo al funcionamiento, crecimiento y sostenibilidad de la plataforma.

ID	Actor	Descripción
A-001	**Administrador**	Encargado de la gestión técnica y operativa de la plataforma, supervisa la correcta funcionalidad de los servicios, gestiona usuarios y productos, resuelve incidencias y garantiza la integridad de los datos en el sistema.

A-002	**Comprador**	Usuario que utiliza la aplicación para explorar, comparar y adquirir productos ofrecidos por vendedores registrados, busca una experiencia de compra sencilla, segura y personalizada. Este actor representa el público general que impulsa la demanda dentro del sistema.
A-003	**Vendedor**	Persona o empresa que publica y gestiona sus productos dentro de la aplicación, es el eje económico del sistema, ya que a través de la aplicación puede promocionar sus artículos, gestionar inventarios, recibir pedidos y aumentar su visibilidad digital.

Tabla 4: Los actores del sistema

Requisitos de usuario

ID	Requisito	Descripción
RU-001	**Gestión de usuarios y autenticación**	La aplicación debe permitir el registro de nuevos usuarios (compradores y vendedores) mediante usuario, correo electrónico y contraseña, así como el inicio y cierre de sesión. Además, debe ofrecer la recuperación de contraseña a través del correo electrónico. Los roles y permisos se le asignaran automáticamente según el tipo de usuario (comprador o vendedor).

RU-002	**Gestión de productos y catalogo**	Los vendedores deben poder registrar, editar y eliminar productos desde su panel de control, incluyendo información como nombre, descripción, precio, categoría e imagen. Los compradores podrán visualizar los productos organizados por categorías, buscar y filtrar según sus preferencias.
RU-003	**Gestión del carrito de compras y pedido**	El sistema debe permitir que los compradores agreguen productos al carrito, modifiquen las cantidades y realicen el pedido. Al finalizar el pedido, se debe generar un resumen del pedido y enviarse la información al Backend para su procesamiento. El carrito se vaciará automáticamente después de un pedido realizado.
RU-004	**Gestión de vendedores y panel administrativo**	Los usuarios con rol de vendedor deben contar con un panel que les permita gestionar sus productos, revisar pedidos recibidos y consultar estadísticas de ventas. El administrador podrá supervisar las actividades generales, mantener la base de datos actualizada y resolver incidencias.
RU-005	**Interfaz de usuario amigable y adaptable**	La aplicación debe ofrecer una interfaz intuitiva, moderna y de fácil navegación, inspirada en plataformas de comercio electrónico reconocidas, adaptándose

		correctamente a distintos tamaños de pantalla en dispositivos Android.
RU-006	**Notificaciones y comunicación**	La aplicación debe permitir el envío de notificaciones al usuario en eventos importantes, como confirmación de pedidos, cambios de estado.

Tabla 5: Requisitos de usuario

Requisitos funcionales (RF)

ID Requisito	Nombre del requisito	Descripción del requisito	Actor
RF-001	**Crear usuario**	El sistema debe permitir el registro de nuevos usuarios, solicitando información básica como nombre, correo electrónico, contraseña y rol (comprador o vendedor).	Comprador, vendedor
RF-002	**Iniciar sesión**	El sistema debe permitir a los usuarios ingresar con su nombre de usuario y contraseña previamente registrados	Comprador, vendedor
RF-003	**Recuperar contraseña**	El sistema debe permitir la recuperación de la contraseña mediante un código enviado al correo electrónico registrado por el usuario.	Comprador, vendedor

RF-004	**Registrar producto**	El sistema debe permitir a los usuarios con rol vendedor agregar nuevos productos proporcionando nombre, descripción, precio, existencias, categoría, subcategoría e imagen	Vendedor
RF-005	**Editar o eliminar producto**	El sistema debe permitir a los vendedores editar o eliminar productos previamente registrados.	Vendedor
RF-006	**Visualizar catálogo de productos**	El sistema debe permitir a los compradores ver el listado completo de productos, organizados por categorías y subcategorías y aplicar filtros de búsqueda.	Comprador
RF-007	**Agregar producto al carrito**	El sistema debe permitir que los usuarios con rol de comprador o vendedor agreguen productos al carrito, ajusten la cantidad y puedan eliminarlos si lo desean.	Comprador, vendedor
RF-008	**Procesar pedido**	El sistema debe permitir que el comprador finalice el pedido, generando un resumen del pedido y enviando los datos al Backend para su registro.	Comprador

RF-009	**Visualizar historial de pedidos**	El sistema debe permitir que los compradores consulten el historial de sus compras realizadas y los vendedores puedan visualizar los pedidos recibidos	Comprador, vendedor
RF-010	**Acceder al panel de vendedor**	El sistema debe permitir a los usuarios con rol vendedor acceder a un panel donde puedan gestionar sus productos, pedidos y estadísticas de ventas	Vendedor
RF-011	**Mostrar interfaz adaptativa**	El sistema debe adaptar automáticamente la interfaz a distintos tamaños de pantalla, manteniendo una experiencia visual optima.	Comprador, vendedor
RF-012	**Enviar notificaciones**	El sistema debe permitir enviar notificaciones al usuario cuando se confirme un pedido o se actualice su estado.	Comprador, vendedor
RF-013	**Buscar productos**	El sistema debe permitir realizar búsquedas de productos por nombre, categoría, subcategoría o palabra clave dentro del catálogo.	Comprador, vendedor
RF-014	**Gestionar perfil de usuario**	El sistema debe permitir que los usuarios consulten y modifiquen su información personal registrada.	Comprador, vendedor

Tabla 6: Requisitos funcionales

Requisitos no funcionales (RNF)

ID Requisito	Descripción	Tipo
RNF-001	La aplicación debe estar disponible las 24 horas del día, los 7 días de la semana, garantizando acceso continuo a las funciones de compra y venta.	Confiabilidad
RNF-002	El sistema debe tener un tiempo de respuesta inferior a 3 segundos al cargar pantallas o procesar acciones como agregar productos al carrito o iniciar sesión.	Rendimiento
RNF-003	La aplicación debe ser compatible con al menos el 90% de los dispositivos Android desde la versión 8.0 (Oreo) en adelante.	Compatibilidad
RNF-004	La interfaz gráfica deber ser intuitiva, moderna y fácil de usar, permitiendo a los usuarios navegar sin necesidad de entrenamiento previo.	Usabilidad
RNF-005	La aplicación debe proteger la información personal de los usuarios mediante protocolos de encriptación.	Seguridad
RNF-006	Los datos sensibles como contraseñas y tokens de autenticación no deben almacenarse como texto	Seguridad

RNF-007	La aplicación debe mantener la sesión iniciada del usuario durante un periodo de tiempo definido, evitando cierres inesperados	Usabilidad / Seguridad
RNF-008	El Backend debe soportar una cantidad considerable de usuarios concurrentes sin degradar el rendimiento.	Escalabilidad
RNF-009	En caso de pérdida de conexión a internet, la aplicación debe mostrar un mensaje claro y permitir reintentos automáticos	Confiabilidad / Experiencia de usuario
RNF-010	Las actualizaciones de productos, carritos y pedidos deben reflejarse en tiempo real o con un retraso máximo de 5 segundos	Rendimiento
RNF-011	La aplicación debe implementar validaciones visuales y mensajes de error claros para guiar al usuario ante entradas incorrectas	Usabilidad
RNF-012	La base de datos debe garantizar integridad referencial entre las tablas	Confiabilidad / Mantenibilidad

RNF-013	El sistema debe estar diseñado con arquitectura modular que facilite la incorporación de nuevas funcionalidades	Mantenibilidad / Escalabilidad
RNF-014	La interfaz debe adaptarse automáticamente a diferentes resoluciones de pantalla (Responsive desing)	Compatibilidad / Usabilidad
RNF-015	La aplicación debe consumir un bajo nivel de recursos del dispositivo, optimizando uso de memoria y batería	Eficiencia
RNF-016	El diseño de la interfaz debe mantener una consistencia visual entre las distintas secciones de la aplicación	Usabilidad / Diseño
RNF-017	La aplicación debe permitir la actualización automática de la información del usuario y sincronización con el servidor al iniciar sesión	Confiabilidad
RNF-018	La aplicación debe garantizar que el tiempo de carga inicial no supere los 5 segundos	Rendimiento
RNF-019	La aplicación debe permitir comunicación fluida con el Backend mediante API REST, con respuestas JSON estandarizadas.	Integrabilidad

RNF-020	La aplicación debe ser evaluada con pruebas de usabilidad antes de su publicación final	Usabilidad / Calidad

Tabla 7: Requisitos no funcionales

Escenarios de los casos de uso

Nombre	**Crear usuario**	
Descripción	Permite registrar un nuevo usuario en la aplicación, ingresando información básica como nombre, correo electrónico, contraseña y tipo de usuario (comprador o vendedor). Según el rol seleccionado, el sistema ajustara las opciones y accesos disponibles dentro de la plataforma.	
Actor	Comprador, vendedor.	
Precondiciones	El usuario debe contar con un dispositivo Android con conexión a internet y un correo electrónico valido.	
Flujo básico		
Pasos	**Actor**	**Sistema**
1	El usuario selecciona la opción “Registrarse” en la pantalla de inicio.	

2		El sistema muestra el formulario de registro con los campos: nombre, correo electrónico, contraseña.
3	El usuario completa los campos y selecciona su rol: Comprador o vendedor.	
4	El usuario da clic en el botón “Registrarse”	
5		El sistema valida que todos los campos requeridos estén diligenciados correctamente y que el correo no se encuentre registrado previamente.
6		Si los datos son válidos, el sistema registra al nuevo usuario en la base de datos.
7		El sistema redirige automáticamente a la pantalla de “iniciar de sesión” para que el usuario acceda con sus nuevas credenciales

8	El usuario inicia sesión con el nombre de usuario y contraseña registrados	
9		El sistema verifica las credenciales y redirige al panel principal correspondiente a su rol (comprador o vendedor).
Flujo alternativo		
Pasos	**Actor**	**Sistema**
1		El sistema detecta que uno o más campos obligatorios no han sido diligenciados.
2	El sistema muestra un mensaje de advertencia indicando los errores específicos.	
3	El usuario corrige la información y vuelve a hacer clic en “Registrarse”.	
4		El sistema valida nuevamente los datos. Si todo está correcto, continua con el registro exitoso
Postcondiciones	Se crea un nuevo usuario registrado correctamente en el sistema con su rol correspondiente (comprador o vendedor)	
RF asociado	RF-001 – Crear usuario	

Tabla 8: Escenario de casos de uso (Crear usuario)

Nombre	**Iniciar sesión**	
Descripción	Permite a los usuarios registrados acceder a la aplicación mediante nombre de usuario y contraseña.	
Actor	Comprador, vendedor.	
Precondiciones	El usuario debe haberse registrado previamente y disponer de conexión a internet.	
Flujo básico		
Pasos	**Actor**	**Sistema**
1	El usuario abre la aplicación e ingresa su nombre de usuario y contraseña	
2	El usuario da clic en “Iniciar sesión”.	
3		El sistema valida las credenciales ingresadas.
4		Si los datos son correctos, el sistema redirige al panel principal según el rol (Comprador o vendedor).
5		El sistema muestra un mensaje de bienvenida.

Flujo alternativo		
Pasos	**Actor**	**Sistema**
1		El sistema detecta credenciales incorrectas o usuario inexistente.
2		El sistema muestra un mensaje de error indicado "usuario o contraseña incorrecta"
3	El usuario puede reintentar el inicio de sesión o recuperar contraseña.	
Postcondiciones	El usuario accede a su panel principal según su rol	
RF Asociado	RF-002 – Iniciar sesión	

Tabla 9: Escenario de casos de uso (Iniciar sesión)

Nombre	Recuperar contraseña
Descripción	Permite recuperar el acceso mediante un Código enviado al correo registrado
Actor	Comprador, vendedor.
Precondiciones	El usuario debe tener un correo electrónico asociado a una cuenta valida

Flujo básico		
Pasos	**Actor**	**Sistema**
1	El usuario selecciona “¿olvido su contraseña?”	
2	El usuario ingresa su correo electrónico	
3		El sistema valida que el correo este registrado
4		El sistema envía un código de verificación al correo
5	El usuario ingresa el código recibido y restablece la contraseña	
6		El sistema actualiza la contraseña
Flujo alternativo		
Pasos	**Actor**	**Sistema**
1		Si el sistema no está registrado, el sistema muestra un mensaje “correo no encontrado”

2	El usuario puede intentar nuevamente con otro correo	
Postcondiciones	El usuario puede acceder nuevamente con la nueva contraseña	
RF Asociado	RF-003 – Recuperar contraseña	

Tabla 10: Escenario de casos de uso (Recuperar contraseña)

Nombre	**Registrar producto**	
Descripción	Permite a los vendedores registrar nuevos productos dentro de la aplicación, ingresando nombre, descripción, precio, existencias, categoría, subcategoría e imagen.	
Actor	vendedor.	
Precondiciones	El usuario debe tener una cuenta activa con rol vendedor.	
Flujo básico		
Pasos	**Actor**	**Sistema**
1	El vendedor accede al panel principal y selecciona la opción “Vender”.	
2		El sistema muestra el formulario con los campos requeridos

3	El vendedor llena la información del producto e incluye una imagen.	
4	El usuario da clic en “Crear producto”	
5		El sistema valida los datos ingresados.
6		El producto se registra correctamente y aparece en el listado de productos del vendedor.
Flujo alternativo		
Pasos	**Actor**	**Sistema**
1		El sistema detecta campos incompletos o formato invalido.
2		El sistema muestra un mensaje de error solicitando la corrección de los datos.
3	El usuario corrige la información y vuelve a dar clic en “Crear producto”	
Postcondiciones	El producto queda disponible en la tienda del vendedor.	

RF Asociado	RF-004 – Registrar producto

Tabla 11: Escenario casos de uso (Registrar producto)

Nombre	**Editar / Eliminar producto**	
Descripción	Permite a los vendedores modificar o eliminar productos registrados.	
Actor	Vendedor.	
Precondiciones	El vendedor debe tener productos registrados	
Flujo básico		
Pasos	**Actor**	**Sistema**
1	El vendedor accede a su lista de productos	
2	El vendedor selecciona un producto y elige la opción “editar” o “eliminar”	
3	El vendedor luego de “editar o “Eliminar” da clic en “guardar”	
4		El sistema actualiza o elimina el registro en la base de datos
5		El sistema muestra un mensaje de confirmación

Flujo alternativo		
Pasos	**Actor**	**Sistema**
1		Si el sistema detecta un error de conexión o el producto no existe, el sistema muestra un mensaje de error
Postcondiciones	El producto se actualiza o se elimina correctamente	
RF Asociado	RF-005 – Editar / Eliminar producto	

Tabla 12:Escenario de casos de uso (Editar / Eliminar producto)

Nombre	**Visualizar catálogo de productos**	
Descripción	Permite al comprador ver productos por categoría, subcategoría o mediante filtros de búsqueda	
Actor	Comprador.	
Precondiciones	El usuario debe estar autenticado o tener conexión a internet	
Flujo básico		
Pasos	**Actor**	**Sistema**
1	El usuario ingresa al catálogo de productos	

2	El usuario realiza búsquedas según la categoría de interés o aplicando filtración de búsqueda	
3		El sistema consulta la base de datos y muestra los resultados de búsqueda
Postcondiciones	El usuario visualiza los productos disponibles	
RF Asociado	RF-006 – Visualizar catálogo de productos.	

Tabla 13: Escenario casos de uso (Visualizar catálogo de productos)

Nombre	**Agregar productos al carrito**	
Descripción	Permite a los usuarios agregar, modificar o eliminar productos de su carrito	
Actor	Vendedor, Comprador.	
Precondiciones	El usuario debe haber iniciado sesión	
Flujo básico		
Pasos	**Actor**	**Sistema**
1	El usuario selecciona un producto	
2	Da clic en “Agregar al carrito”	

3		El sistema agrega el producto y actualiza el total
Flujo alternativo		
1		Si no hay Stock disponible, el sistema muestra un mensaje "Producto agotado"
Postcondiciones	El producto se agrega al carrito y se guarda temporalmente	
RF Asociado	RF-007 – Agregar productos al carrito.	

Tabla 14: Escenario casos de uso (Agregar productos al carrito)

Nombre	**Procesar pedido**	
Descripción	Permite al comprador confirmar la compra de los productos seleccionados, generando un pedido que será enviado al vendedor correspondiente.	
Actor	Comprador.	
Precondiciones	El usuario debe haber iniciado sesión y tener productos en el carrito.	
Flujo básico		
Pasos	**Actor**	**Sistema**
1	El comprador ingresa la dirección y da clic en el icono "Ir a pagar (Contraentrega)"	

2		El sistema muestra la pantalla de resumen del pedido con los datos del usuario, dirección de envío y total.
3	El usuario confirma la información y da clic en “Finalizar compra”	
4		El sistema registra el pedido y muestra un mensaje de éxito.
5		Se vacía el carrito automáticamente.
6		El sistema notifica al vendedor y al comprador correspondiente sobre la nueva orden realizada.
Postcondiciones	El pedido queda registrado y listo para ser procesado por el vendedor.	
RF Asociado	RF-008 – Procesar pedido.	

Tabla 15: Escenario casos de uso (Procesar pedido)

Nombre	**Visualizar historial de pedidos**
Descripción	Permite a compradores y vendedores ver sus pedidos realizados o recibidos.
Actor	Comprador, vendedor.
Precondiciones	El usuario debe haber iniciado sesión y tener pedidos registrados

Flujo básico		
Pasos	**Actor**	**Sistema**
1	El accede a “Mis compras”	
2		El sistema consulta y muestra la lista de pedidos.
3	El usuario puede seleccionar un pedido para ver el detalle	
Postcondiciones	El usuario visualiza correctamente la información de sus pedidos.	
RF Asociado	RF-009 – Visualizar historial de pedidos	

Tabla 16: Escenario casos de uso (Visualizar historial de pedidos)

Nombre	**Acceder al panel vendedor**	
Descripción	Permite al vendedor gestionar productos.	
Actor	vendedor.	
Precondiciones	El usuario debe tener rol vendedor.	
Flujo básico		
Pasos	**Actor**	**Sistema**
1	El vendedor inicia sesión	

2		El sistema valida el rol y lo redirige al panel vendedor.
3	El vendedor da clic en mis publicaciones	
4		El sistema muestra los productos registrados por el vendedor
5	El vendedor puede gestionar sus productos, actualizar Stock, ofertas, editar productos o eliminarlos	
6		El sistema actualiza los cambios realizados por el vendedor
Postcondiciones	El vendedor puede administrar su panel	
RF Asociado	RF-010 – Acceder al panel vendedor.	

Tabla 17: Escenario casos de uso (Acceder al panel vendedor)

Nombre	Mostrar interfaz adaptativa
Descripción	El sistema adapta automáticamente la interfaz según el tamaño y resolución de la pantalla.
Actor	Comprador, vendedor.
Precondiciones	El usuario debe tener la aplicación instalada en un dispositivo Android compatible.

Flujo básico		
Pasos	**Actor**	**Sistema**
1	El usuario abre la aplicación	
2		El sistema detecta la resolución del dispositivo
3		El sistema ajusta automáticamente los componentes visuales
Postcondiciones	La aplicación se adapta visualmente al dispositivo.	
RF Asociado	RF-011 – Mostrar interfaz adaptativa.	

Tabla 18: Escenario casos de uso (Mostrar interfaz adaptativa)

Nombre	**Enviar notificaciones**
Descripción	Permite al sistema enviar alertas sobre pedidos y actualizaciones.
Actor	Comprador, vendedor.
Precondiciones	El usuario debe tener habilitada las notificaciones.

Flujo básico		
Pasos	**Actor**	**Sistema**
1		El sistema detecta cambios en el estado del pedido
2		El sistema envía una notificación al usuario.
3	El usuario visualiza la notificación y accede al detalle	
Postcondiciones	El usuario recibe correctamente las notificaciones	
RF Asociado	RF-012 – Enviar notificaciones.	

Tabla 19: Escenario casos de uso (Enviar notificaciones)

Nombre	**Buscar productos**	
Descripción	Permite buscar productos por nombre o palabra clave.	
Actor	Comprador, vendedor.	
Precondiciones	El usuario debe estar registrado y tener conexión a internet	
Flujo básico		
Pasos	**Actor**	**Sistema**
1	El usuario escribe una palabra clave en la barra de búsqueda	

2		El sistema busca coincidencias en la base de datos.
3		El sistema muestra los resultados filtrados
Postcondiciones	El usuario encuentra los productos deseados	
RF Asociado	RF-013 – Buscar productos.	

Tabla 20: Escenario casos de uso (Buscar productos)

Nombre	**Gestionar perfil de usuario**	
Descripción	Permite consultar p modificar la información personal del usuario.	
Actor	Comprador, vendedor.	
Precondiciones	El usuario debe haber iniciado sesión	
Flujo básico		
Pasos	**Actor**	**Sistema**
1	El usuario accede a su perfil	
2	El usuario realiza cambios en nombre, contraseña o correo	
3		El sistema valida y actualiza los datos

4		El sistema muestra un mensaje de confirmación
Postcondiciones	Los datos personales del usuario se actualizan correctamente	
RF Asociado	RF-014 – Gestionar perfil de usuario.	

Tabla 21: Escenario casos de uso (Gestionar perfil de usuario)

Diagramas modelo de casos de uso

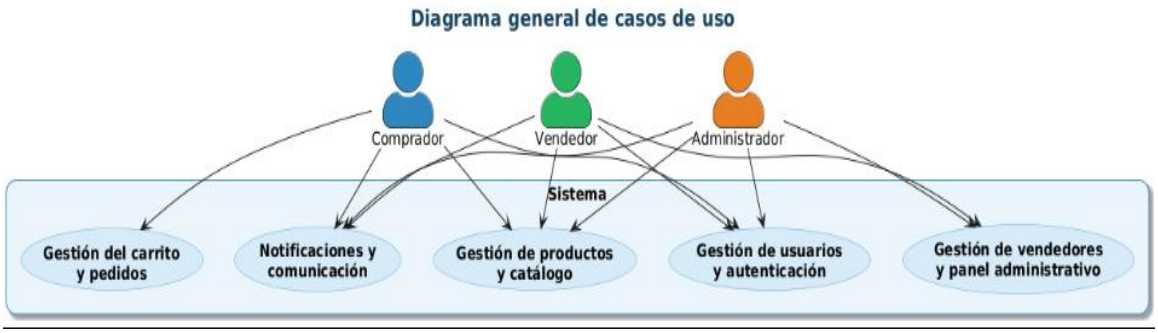


Ilustración 23: Diagrama general de casos de uso

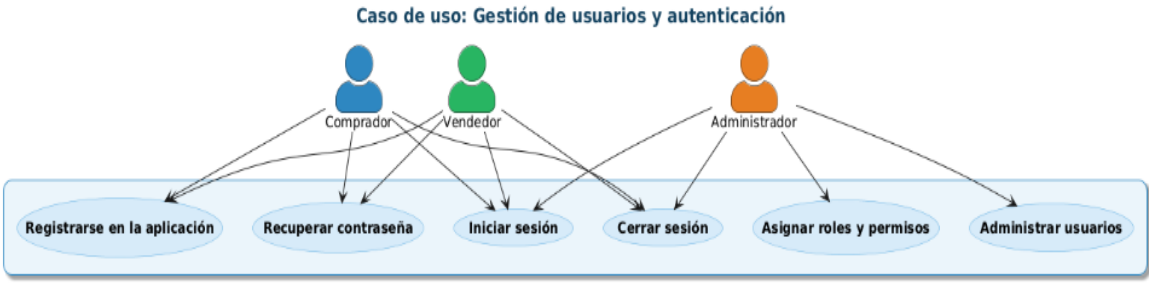


Ilustración 24: Caso de uso (Gestión de usuario y autenticación)

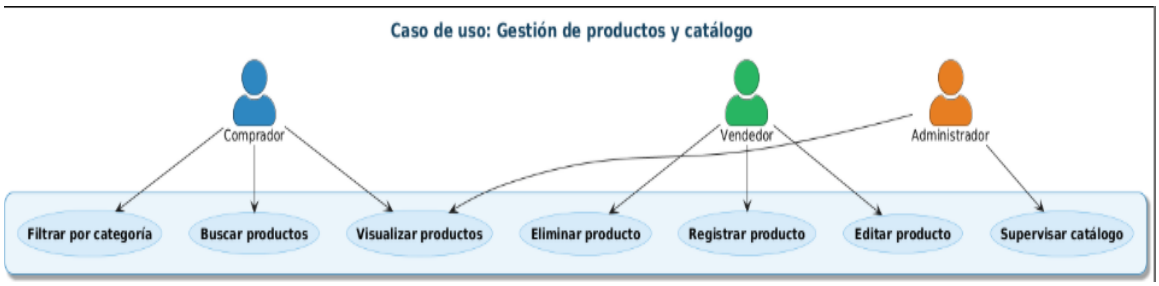


Ilustración 25: Caso de uso (Gestión de productos y catalogo)

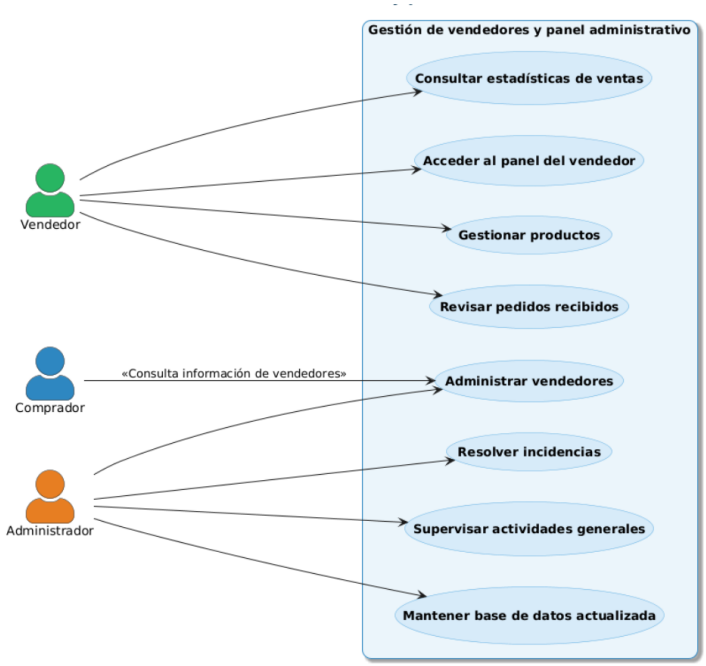


Ilustración 26: Caso de uso (Gestión de vendedores y panel administrativo)

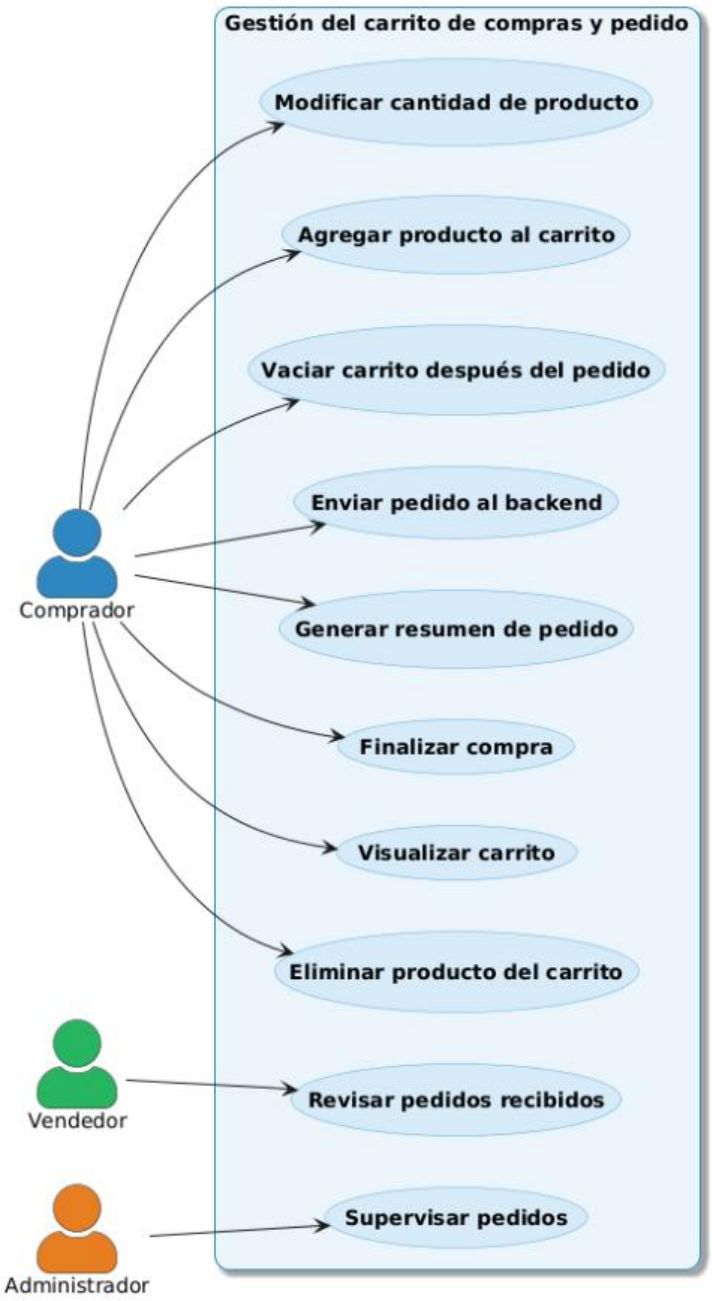


Ilustración 27: Caso de uso (Gestión de carrito y pedido)

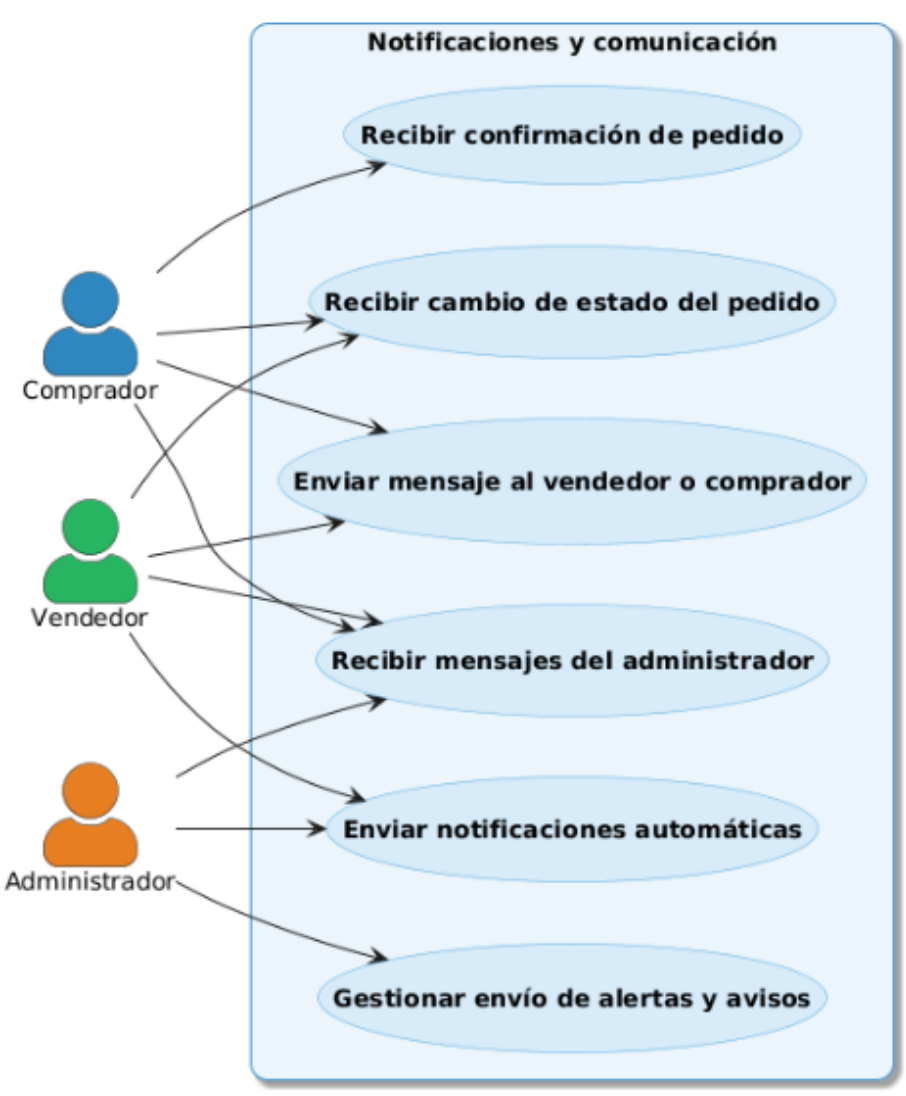


Ilustración 28: Caso de uso (Notificaciones y comunicación)

Diagrama de clases

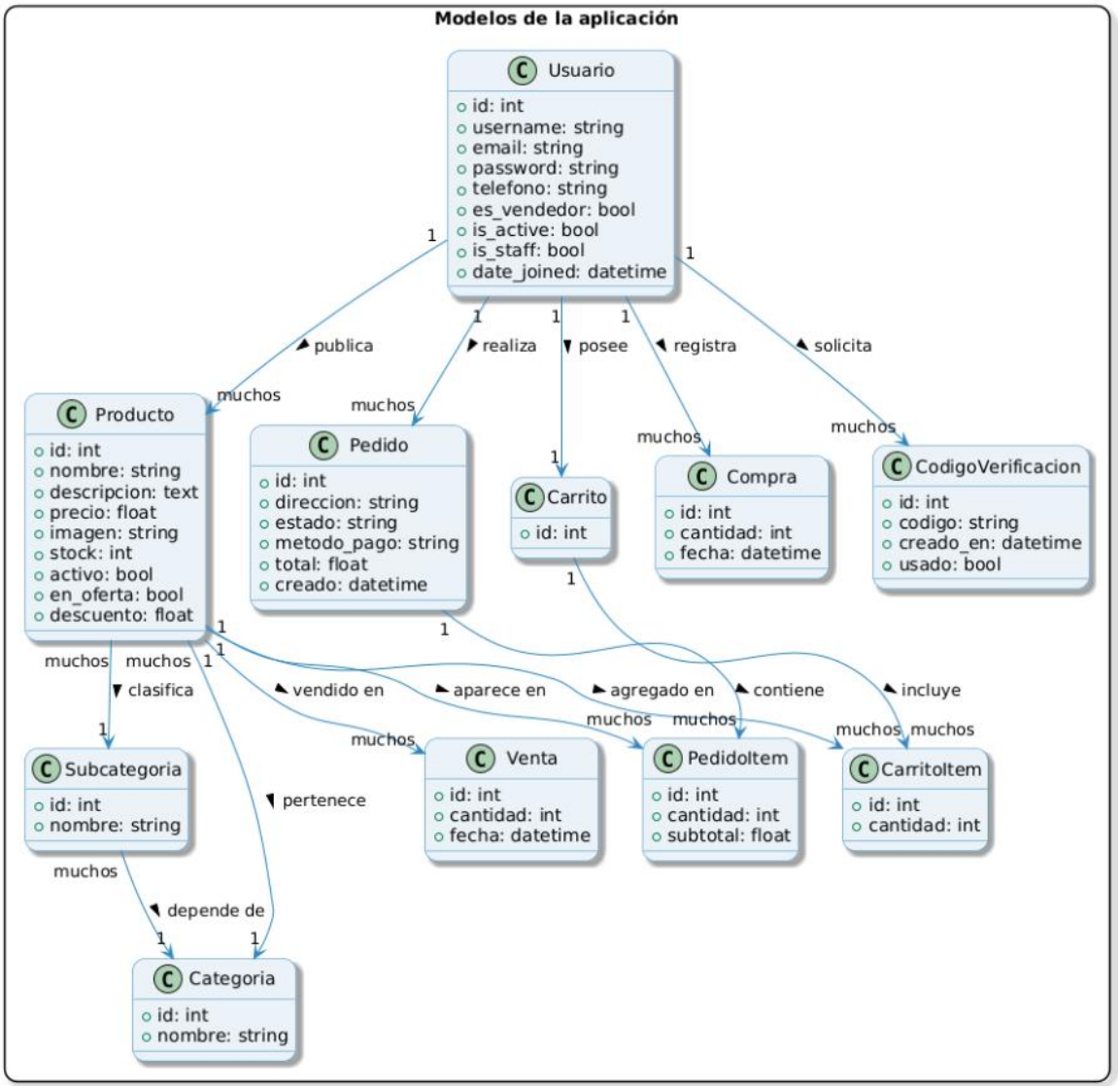


Ilustración 29: Modelo de clases

Modelo entidad relación

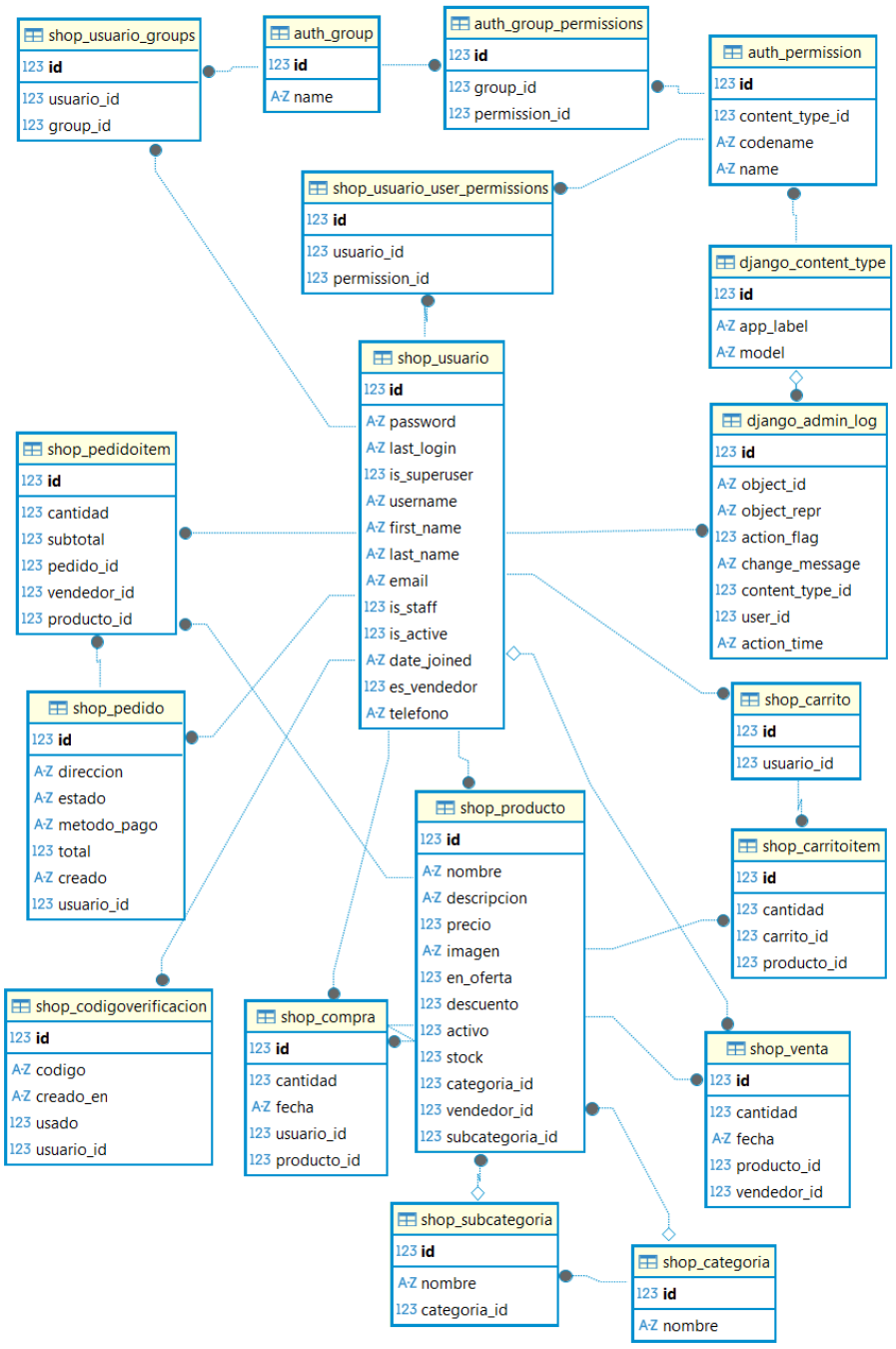


Ilustración 30: Modelo entidad relación

Metodologia

En esta sección se describe la metodología empleada para el desarrollo de la aplicación móvil FourShop, así como las estrategias adoptadas para garantizar un proceso de construcción eficiente, flexible y orientado al usuario, con el fin de asegurar que el producto final cumpla con los requisitos funcionales, técnicos y de usabilidad definidos en las fases de análisis y diseño.

Metodología de trabajo ágil

Para el desarrollo de la aplicación FourShop se seleccionó la metodología ágil Scrum, ya que esta se adapta perfectamente a proyectos de desarrollo de software donde los requerimientos pueden evolucionar con el tiempo. Scrum permite construir la aplicación en pequeñas etapas llamadas Sprints, las cuales se planifica, desarrolla, prueba y revisa una parte del producto funcional. Este enfoque favorece la retroalimentación temprana y continua, mejorando así la calidad y brinda una alineación con los objetivos del proyecto.

Roles del proyecto

Aunque el equipo de desarrollo está conformado por dos personas, se asumieron roles definidos por la metodología Scrum para garantizar una organización efectiva durante todo el ciclo de vida del proyecto.

Product Owner: Ambos autores son responsables de definir y priorizar los requerimientos del producto. Se realizo un análisis de las necesidades de los usuarios (comprador y vendedor) y referentes del comercio electrónico para establecer

funcionalidades prioritarias de FourShop, como la gestión de los productos, autenticación, carrito de compras y panel de vendedor.

Scrum Master: Ambos autores supervisan el cumplimiento de la metodología, asegurando que no existan impedimentos técnicos ni de gestión que afecten el progreso, manteniendo una comunicación fluida y documentando los avances de cada Sprint.

Equipo de desarrollo: Ambos autores son los encargados de desarrollar todo lo anteriormente diseñado, es decir construir la aplicación teniendo en cuenta todas las indicaciones y requerimientos y las pruebas funcionales.

Los Sprints

Para este proyecto se definió una duración de dos semanas por cada Sprint. Al inicio de cada ciclo, se planifican las tareas y funcionalidades que serán desarrolladas, priorizando aquellas que aporten mayor valor visual final.

Durante la ejecución de cada Sprint, se llevan a cabo reuniones de revisión dos veces por semana para responder a las preguntas esenciales de la metodología Scrum:

- ¿Qué se hizo desde la última reunión?
- ¿Qué se hará a continuación?
- ¿Existe algún obstáculo que impida avanzar?

Al finalizar cada Sprint se realizan las siguientes actividades:

Sprint Review: Se revisa lo desarrollado durante el Sprint, verificando que cumpla con los requisitos establecidos. Se evalúa su funcionamiento, integración y se documentan posibles mejoras.

Sprint Retrospective: Se analizan los aciertos y desafíos del Sprint para mejorar el proceso en el siguiente ciclo, buscando optimizar la comunicación, tiempos de desarrollo y calidad del código.

GitHub: Todo el código fuente se gestiona mediante la plataforma GitHub, permitiendo mantener un historial de cambios, control de versiones y una copia de seguridad del proyecto.

Despliegue: Una vez que las funcionalidades del Sprint son aprobadas y probadas, se realiza su despliegue en un entorno de prueba. Al finalizar cada Sprint despliega para el uso en el prototipo.

Resumen de la metodología

El uso de Scrum permitió desarrollar la aplicación móvil de manera ordenada, adaptable y eficiente, asegurando que cada etapa de desarrollo generara una versión funcional del producto, gracias a las revisiones periódica y a la comunicación constante entre las fases, donde se logró construir una aplicación estable, escalable y centrada en las necesidades de los usuarios, contribuyendo al fortalecimiento del comercio electrónico de las MiPymes.

Herramientas de desarrollo necesarias

Recurso	Descripción	Cantidad	Obtenido	
			SI	NO
Computadoras para desarrollo	Equipos de cómputo con entornos configurados para programación en Visual Studio	2	X	

	Code, Android Studio y ejecución de servidores Django.			
Smartphones (Android 8.0+)	Dispositivos utilizados para pruebas del prototipo de la aplicación móvil.	2	X	
Router Wi-Fi	Red local utilizada para las pruebas de conexión entre el frontend móvil y el Backend Django.	1	X	
Software				
Android Studio	Entorno de desarrollo integrado (IDE) utilizado para construir la aplicación móvil nativa en Kotlin con Jetpack Compose.	2	X	
Visual Studio Code	Editor utilizado para el desarrollo del Backend con Django.	2	X	
Django + Django REST Framework	Framework Backend basada en Python para la creación de la API REST que gestiona usuarios, productos, pedidos y autenticación.	1	X	
SQLite	Sistema de gestión de base de datos utilizado para almacenar información de usuarios, productos y pedidos.	1	X	

GitHub	Plataforma empleada para el control de versiones, respaldo y colaboración en el desarrollo del proyecto.	2	X	

Tabla 22: Herramientas necesarias para el desarrollo

Costos del proyecto

Recurso	Cantidad	Costo unitario	Costo total
Asus vivobook 12th Gen Intel® Core™ i5-1235U Sistema operativo Windows Pantalla de 15.6 Pulgadas 	2	$2.379.900	$4.759.800
Celular Redmi 10C Gris Oscuro Sistema operativo Android Memoria interna 128GB Memoria RAM 4.0 + 2.0 GB Cámara de 50Mpx Tamaño de la pantalla 6.71”	2	$939.000	$1.878.000

			
Router Wi-Fi 	1	$95.000	$95.000
Software			
Android Studio	2	$0	$0
Visual Studio Code	2	$0	$0
Django + Django REST Framework	1	$0	$0
SQLite	1	$0	$0
GitHub	2	$0	$0
Subtotal	$6.732.800		

Tabla 23: Costos del proyecto (Hardware y Software)

Talento Humano

Cargo	Cantidad	Detalle	Costo estimado
Diseñador	1	60h x $25.000/h	$1.500.000
Desarrolladores	2	200h x $20.000/h	$4.000.000
Documentación y pruebas	2	40h x $25.000/h	$1.000.000
Subtotal	$6.500.000		

Tabla 24: Costos del proyecto (Talento Humano)

Costo estimado del desarrollo

Descripción	Valor
Hardware y Software	$6.732.800
Talento humano	$6.500.000
Total	$13.232.800

Tabla 25: Costo estimado del proyecto

Historias de usuario

No de historia	RF relacionado	Descripción
1	RF-001	Como nuevo usuario, requiere registrarse con nombre, correo electrónico, contraseña y rol (comprador o vendedor).
2	RF-002	Como usuario registrado, inicia sesión con nombre y contraseña, para acceder a la cuenta y utilizar la aplicación según su rol.

3	RF-003	Como usuario, para recuperar la contraseña se le enviara al correo un código de verificación para restablecer el acceso a la cuenta.
4	RF-004	Como vendedor, para registrar nuevos productos relacionando nombre, descripción, precio, existencias, categoría, subcategoría y una imagen, para que estén disponibles en la tienda.
5	RF-005	Como vendedor, puede editar o eliminar sus productos registrados, para mantener actualizada la información.
6	RF-006	Como comprador, puede ver el catálogo completo de productos organizados por categorías, con opciones de búsqueda y filtrado, para encontrar fácilmente el producto que desea comprar.
7	RF-007	Como comprador, puede agregar productos al carrito, ajustar las cantidades o eliminarlos, gestionando sus compras antes de finalizar el pedido.
8	RF-008	Como comprador, puede finalizar su compra y generar un resumen del pedido, para confirmar y enviar la información al Backend para su registro.
9	RF-009	Como comprador o vendedor, puede consultar el historial de sus pedidos realizados o recibidos, para tener un registro de pedidos.
10	RF-010	Como vendedor, puede acceder a un panel administrativo, donde puede gestionar sus productos, revisar pedidos.

11	RF-011	Como usuario, tendrá una interfaz que se adapte automáticamente al tamaño de la pantalla de su dispositivo, para disfrutar de una experiencia visual cómoda y consistente.
12	RF-012	Como usuario puede recibir notificaciones cuando se confirme o actualice el estado de su pedido, para estar informado en tiempo real.
13	RF-013	Como usuario, puede buscar productos por nombre, categoría o palabra clave, para encontrar fácilmente lo que necesita.
14	RF-014	Como usuario, puede consultar y modificar su información personal, para mantener sus datos actualizados dentro de la aplicación.

Tabla 26: Historias de usuario

Resumen de riesgos

Durante la fase de análisis se les realiza un estudio a los objetivos planteados en el proyecto, los usuarios, requerimientos funcionales y no funcionales y el alcance que va a tener la aplicación. Por lo tanto, es fundamental identificar los posibles riesgos que puedan afectar la ejecución del proyecto, donde la aplicación móvil lo que busca es ofrecer un entorno de comercio electrónico, con funciones de autenticación, gestión de productos, carrito de compras y pedidos, con base en esos aspectos se deben considerar algunos posibles riesgos que se pueden presentar.

1. **Requerimientos incompletos:** No definir con claridad las funciones como roles de usuario o gestión del carrito, conllevaría a correcciones en fases posteriores.
2. **Modificación del alcance:** Si durante el transcurso de la realización del proyecto se agregan nuevas funcionalidades sin la planificación requerida, los tiempos y costos del proyecto podrían incrementarse.
3. **Análisis de los usuarios finales:** Si no se analizan las necesidades de los roles que van a interactuar con la aplicación, podría no resultar intuitiva o funcional.

4. **Riesgos tecnológicos:** No analizar correctamente la compatibilidad entre el frontend y el Backend podría generar problemas a futuro.
5. **priorización de funcionalidades:** Si el equipo se enfoca son cosas menos importantes en lugar del flujo principal (compra, pago y pedido) podrían generar retrasos en el desarrollo de la aplicación.
6. **Estimación incorrecta de tiempo y recursos:** si no se analiza bien la complejidad de la aplicación puede generar que el proyecto tome más tiempo del esperado inicialmente.

Conclusión para el objetivo numero 1

El cumplimiento del primer objetivo permitió consolidar una base sólida para el desarrollo de la aplicación móvil, enfocada en optimizar la experiencia del usuario y mejorar la gestión comercial dentro del entorno digital. A través de la información recolectada, el análisis e implementación de esta, se lograron identificar y aplicar mejoras significativas en la usabilidad y funcionalidad del sistema.

También se evidencio la importancia de una interfaz moderna, adaptable y de fácil acceso, que permita a compradores y vendedores interactuar de manera intuitiva con el sistema. Como resultado, se definieron los requisitos funcionales y no funcionales que guiaron a la construcción del proyecto, garantizando que la aplicación responda a los criterios previamente establecidos.

Actividades para desarrollar el objetivo especifico numero2

Para desarrollar el objetivo específico numero 2: “Diseñar una interfaz móvil intuitiva y funcional que les permita a las empresas registrar, administrar y promocionar sus productos de manera eficiente para mejorar la experiencia de compra y venta”, se tomó como base la información obtenida en la fase de análisis, la cual permitió comprender las necesidades tanto de los compradores como de los vendedores. Se procedió al diseño de una interfaz moderna y de fácil uso, trabajando en la organización visual de componentes clave como catálogo de productos, carrito de compras, panel de vendedor, promoviendo una experiencia de compra y venta más ágil, segura y agradable para todos los usuarios.

Diccionario de datos

Tabla: Usuario						
descripción	En esta tabla se guardará toda la información de los usuarios (comprador, vendedor, administrador)					
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto incrementable**	**Tipo de campo**	**descripción**
Id	Int	4	No	Si	Clave primaria	Identificador único del usuario
password	Varchar	255	No	No	Campo normal	Contraseña encriptada del usuario
Last_login	Datetime	____	Si	No	Campo normal	Fecha y hora del último inicio de sesión
Is_superuser	Bool	____	No	No	Campo lógico	Indica si el usuario tiene permisos de superadministrador

Username	Varchar	150	No	No	Campo único	Nombre de usuario usado para la autenticación
First_name	Varchar	150	Si	No	Campo normal	Primer nombre del usuario
Last_name	Varchar	150	Si	No	Campo normal	Apellido del usuario
Email	Varchar	254	No	No	Campo único	Correo electrónico del usuario
Is_staff	Bool	___	No	No	Campo lógico	Indica si el usuario puede acceder al panel administrativo
Is_active	Bool	___	No	No	Campo lógico	Define si la cuenta del usuario esta activa
Date_joined	Datetime	___	No	No	Campo normal	Fecha y hora en la que el usuario fue registrado
Es_vendedor	Bool	___	No	No	Campo lógico	Define si el usuario tiene el rol vendedor
teléfono	Varchar	20	Si	No	Campo normal	Número de teléfono de contacto

Tabla 27: Diccionario de datos (Tabla Usuario)

Tabla: Categoría						
descripción	En esta tabla se guardan las categorías principales de los productos					
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto incrementable**	**Tipo de campo**	**descripción**
id	Int	4	No	Si	Clave primaria	Identificador único de la categoría
nombre	Varchar	50	No	No	___	Nombre de la categoría

Tabla 28: Diccionario de datos (Tabla Categoría)

Tabla: Subcategoría							
descripción	En esta tabla se guardan todas las subcategorías asociadas a cada categoría						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria		Identificador único de la subcategoría
nombre	Varchar	50	No	No	___		Nombre de la subcategoría

Categoria_id	Bigint	8	No	No	Clave foránea	categoría	Identificador de la categoría a la que pertenece la subcategoría

Tabla 29: Diccionario de datos (Tabla Subcategoría)

Tabla: Producto							
descripción	En esta tabla se guardan los productos registrados por los vendedores						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único del producto
nombre	Varchar	100	No	No	___		Nombre del producto
descripción	Text	___	No	No	_____	_____	descripción del producto
precio	Decimal	10,2	No	No	____	____	Precio del producto

Imagen	Varchar	255	No	No	___	___	Ruta o url de la imagen del producto
En_oferta	Bool	1	No	No	___	___	Indica si el producto está en oferta
descuento	Integer unsigned	4	No	No	___	___	Porcentaje de descuento si aplica
activo	Bool	1	No	No	___	___	Indica si el producto está activo para la venta
stock	Integer unsigned	4	No	No	___	___	Cantidad disponible del producto
Categoria_id	Bigint	8	No	No	Clave foránea	categoría	Relaciona el producto con su categoría

Vendedor_id	Bigint	8	No	No	Cave foránea	Usuario	Relaciona el producto con el vendedor que lo publico
Subcate_id	Bigint	8	No	No	Clave foránea	Subcate	Relaciona el producto con su subcategoría

Tabla 30: Diccionario de datos (Tabla Producto)

Tabla: Carrito							
descripción	En esta tabla se guarda el carrito del usuario						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria		Identificador único del carrito
Usuario_id	Bigint	8	No	No	Clave foránea	usuario	Relaciona el carrito con el usuario

Tabla 31: Diccionario de datos (Tabla Carrito)

Tabla: CarritoItem							
descripción	En esta tabla se guardan los productos agregados al carrito de un usuario						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	____	Identificador único del CarritoItem
cantidad	Integer unsigned	4	No	No	___	_____	Cantidad de productos agregados al carrito
Carrito_id	Bigint	8	No	No	Clave foránea	Carrito	Identificador del carrito asociado
Producto_id	Bigint	8	No	No	Clave foránea	producto	Identificador del producto agregado

Tabla 32: Diccionario de datos (Tabla Carritoitem)

Tabla: Pedido							
descripción	En esta tabla se registran los pedidos realizados por los usuarios						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único del pedido
dirección	Varchar	100	No	No	___	___	dirección de envío del pedido
estado	Varchar	50	No	No	___	___	Estado actual del pedido (pendiente, enviado, entregado)
Método pago	Varchar	50	No	No	___	___	Método de pago (contra entrega)
total	Decimal	10,2	No	No	___	___	Monto total del pedido

creado	Datetime	___	No	No	___	___	Fecha y hora en que se generó el pedido
Usuario_id	Bigint	8	No	No	Clave foránea	usuario	Usuario que realizo el pedido

Tabla 33: Diccionario de datos (Tabla Pedido)

Tabla: PedidoItem							
descripción	En esta tabla se guardan los productos que componen cada pedido						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único del PedidoItem
cantidad	Integer unsigned	4	No	No	___	___	Cantidad de unidades de producto en el pedido.
subtotal	Decimal	10,2	No	No	___	___	Subtotal del PedidoItem

Pedido_id	Bigint	8	No	No	Clave foránea	Pedido	Pedido al que pertenece el ítem
Vendedor_id	Bigint	8	No	No	Clave foránea	Usuario	Vendedor asociado al producto
Producto_id	Bigint	8	No	No	Clave foránea	producto	Producto incluido en el pedido

Tabla 34: Diccionario de datos (Tabla PedidoItem)

Tabla: Compra							
descripción	En esta tabla se guardan las compras realizadas por los usuarios, con los productos adquiridos y la fecha.						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único de la compra
cantidad	Integer unsigned	4	No	No	___	___	Cantidad de productos comprados

fecha	Datetime	___	No	No	___	___	Fecha en la que se realizó la compra
Usuario_id	Bigint	8	No	No	Clave foránea	Usuario	Usuario que realizo la compra
Producto_id	Bigint	8	No	No	Clave foránea	producto	Producto adquirido

Tabla 35: Diccionario de datos (Tabla Compra)

Tabla: Venta							
descripción	En esta tabla se guardan las ventas realizadas por los vendedores, asociando los productos vendidos, las cantidades y la fecha.						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único de la venta
cantidad	Integer unsigned	4	No	No	___	___	Cantidad de productos vendidos

fecha	Datetime	___	No	No	___	___	Fecha en la que se realizó la venta
Producto_id	Bigint	8	No	No	Clave foránea	producto	Producto que fue vendido
Vendedor_id	Bigint	8	No	No	Clave foránea	usuario	Usuario que realizo la venta

Tabla 36: Diccionario de datos (Tabla Venta)

Tabla: usuario_user_permissions							
descripción	En esta tabla gestiona la relación entre los usuarios y los permisos que tienen asignados dentro del sistema, permitiendo controlar los niveles de acceso.						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único del registro

Usuario_id	Bigint	8	No	No	Clave foránea	usuario	Usuario al que se le asigna el permiso
Permission_id	Bigint	8	No	No	Clave foránea	permissions	Permiso asignado al usuario

Tabla 37: Diccionario de datos (Tabla Usuario_user_permissions)

Tabla: CodigoVerificacion							
descripción	En esta tabla se almacenan los códigos generados para la verificación de usuarios o recuperación de contraseñas, con su estado de uso y fecha						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único del Código
Código	Varchar	variable	No	No	___	___	Código generado para verificación
Creado_en	Datetime	___	No	No	___	___	Fecha y hora en la que se generó el Código

usado	Bool	___	No	No	___	___	Indica si el Código ya se utilizo
Usuario_id	Bigint	8	No	No	Clave foránea	usuario	Usuario al que pertenece el Código

Tabla 38: Diccionario de datos (Tabla CodigoVerificacion)

Tabla: usuario_groups							
descripción	En esta tabla se almacena la relación entre los usuarios y los grupos a los que pertenecen, permite asignar permisos y roles de manera colectiva a los usuarios dentro del sistema.						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único del registro
Usuario_id	Bigint	8	No	No	Clave foránea	usuario	Usuario que pertenece al grupo

Group_id	Bigint	8	No	No	Clave foránea	Auth_group	Grupo al que pertenece el usuario

Tabla 39: Diccionario de datos (Tabla Usuario_groups)

Las tablas que se relacionan a continuación son tablas del sistema de autenticación y administración de Django, creadas automáticamente cuando se ejecutan las migraciones iniciales (Python manage.py migrate). Aunque no son creadas manualmente son necesarias para que la aplicación pueda gestionar usuarios, permisos y roles.

Tabla: auth_group							
descripción	En esta tabla se almacenan los diferentes grupos o roles que pueden tener los usuarios dentro del sistema, como “Administrador, vendedor o cliente”						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único del grupo
name	Varchar	150	No	No	Campo único	___	Nombre del grupo o rol

Tabla 40: Diccionario de datos (Tabla auth_group)

Tabla: auth_permission							
descripción	Contiene los permisos definidos para cada modelo del sistema. Estos permisos determinan las acciones que pueden realizar los usuarios (agregar, editar, eliminar, ver).						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único del permiso
Content_type_id	Int	4	No	No	Clave foránea	Django_ content_ tipe	Tipo de modelo al que pertenece el permiso
codename	Varchar	100	No	No	Campo de texto	___	Código interno del permiso (ej. Add_user)
name	Varchar	255	No	No	Campo de texto		Nombre legible del permiso

Tabla 41: Diccionario de datos (Tabla auth_permission)

Tabla: auth_group_permissions							
descripción	Esta tabla establece la relación entre los grupos y los permisos asignados, permitiendo definir qué acciones puede realizar cada grupo dentro del sistema						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único del registro
Group_id	Int	4	No	No	Clave foránea	Auth_group	Grupo al que pertenece el permiso
Permission_id	Int	4	No	No	Clave foránea	Auth_ permission	Permiso asignado ala grupo

Tabla 42: Diccionario de datos (Tabla auth_group_permissions)

Tabla: django_admin_log							
descripción	Esta tabla registra todas las acciones realizadas por los usuarios administradores dentro del panel de administración de Django, como la creación, modificación o eliminación de registros.						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**

id	Int	4	No	Si	Clave primaria	___	Identificador único del registro
object_id	text	___	Si	No	Campo de texto	___	Identificador del objeto afectado
Object_repr	Varchar	200	No	No	Campo de texto	___	Descripción textual del objeto
Action_flag	Smallint	2	No	No	Campo numérico	___	Tipo de acción (1=Agregar, 2=Modificar)
Change_ message	text	___	Si	No	Campo de texto	___	Mensaje o descripción del cambio realizado
Content_ type_id	Int	4	Si	No	Clave foránea	Django_ content_ type	Tipo de objeto modificado
User_id	Int	4	No	No	Clave foránea	usuario	Usuario que realizo la acción

Action_time	Datetime	___	No	No	Campo fecha	___	Fecha y hora en que se ejecutó la acción.

Tabla 43: Diccionario de datos (Tabla django_admin_log)

Tabla: django_content_type							
descripción	En esta tabla se almacena la relación entre los usuarios y los grupos a los que pertenecen, permite asignar permisos y roles de manera colectiva a los usuarios dentro del sistema.						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único del tipo de contenido
App_label	Varchar	100	No	No	Campo de texto	___	Nombre de la aplicación a la que pertenece el modelo
model	Varchar	100	No	No	Campo de texto	___	Nombre del modelo registrado

Tabla 44: Diccionario de datos (Tabla django_content_type)

Tabla: django_migrations							
descripción	Esta tabla registra todas las migraciones aplicadas al sistema. Permite llevar un control de los cambios estructurales realizados en la base de datos.						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
id	Int	4	No	Si	Clave primaria	___	Identificador único de la migración
app	Varchar	255	No	No	Campo de texto	___	Nombre de la aplicación que genero la migración
name	Varchar	255	No	No	Campo de texto	___	Nombre del archivo de migración aplicado
applied	Datetime	___	No	No	Campo fecha	___	Fecha y hora en que se ejecutó la migración

Tabla 45: Diccionario de datos (Tabla django_migrations)

Tabla: usuario_groups							
descripción	En esta tabla se almacena la información de las sesiones activas de los usuarios en la aplicación. Django la utiliza para mantener la sesión iniciada, guardar datos temporales del usuario (como su estado de autenticación) y permitir que continue navegando son volver a iniciar sesión.						
Campo	**Tipo de dato**	**Tamaño**	**Nulo**	**Auto increm**	**Tipo de campo**	**Tabla relación**	**descripción**
Session_key	Varchar	40	No	No	Clave primaria	___	Identificador único de la sesión
Session_data	text	____	No	No	Campo de texto	____	Contiene los datos de la sesión codificados (ej. Id del usuario y otra información temporal)
Expire_date	Datetime	___	No	No	Clave fecha y hora	___	Fecha y hora de expiración de la sesión.

Tabla 46: Diccionario de datos (Tabla django_session)

Prototipo paso a paso

Bienvenida

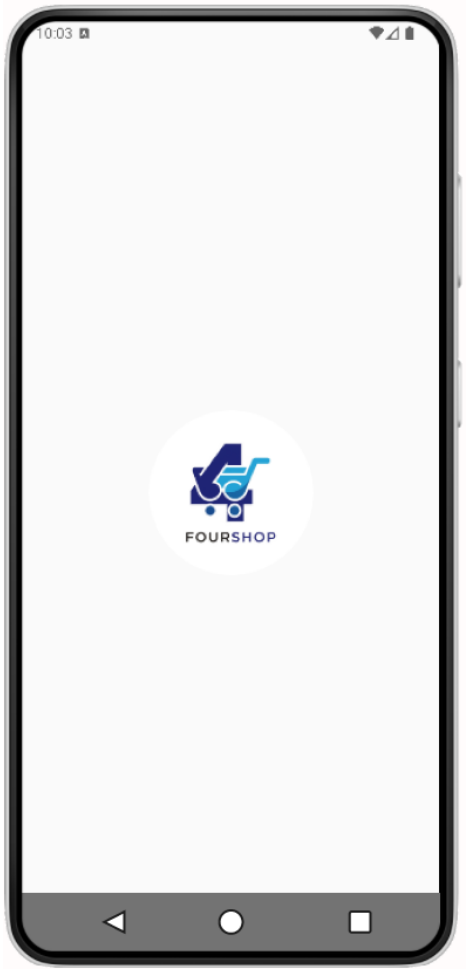


Ilustración 31: Prototipo (Bienvenida)

Ilustración 32: Prototipo (Registrarse)

Crear nuevo usuario

Ilustración 33: Prototipo (Crear cuenta)

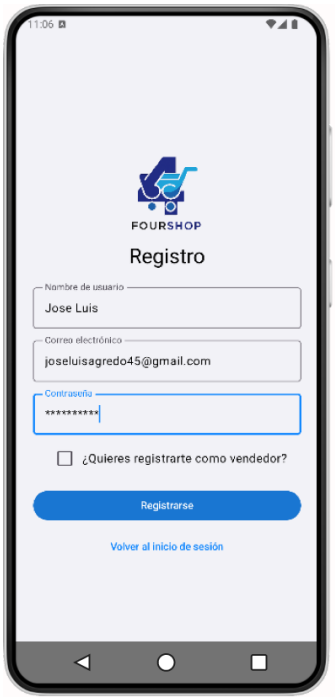


Ilustración 34: Prototipo (Crear usuario)

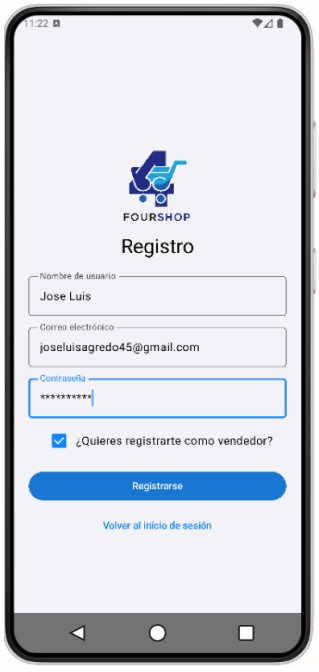


Ilustración 35: Prototipo (Crear vendedor)

Iniciar sesión

Ilustración 36: Prototipo (Iniciar sesión)

Ilustración 37: Prototipo (Inicio_Datos)

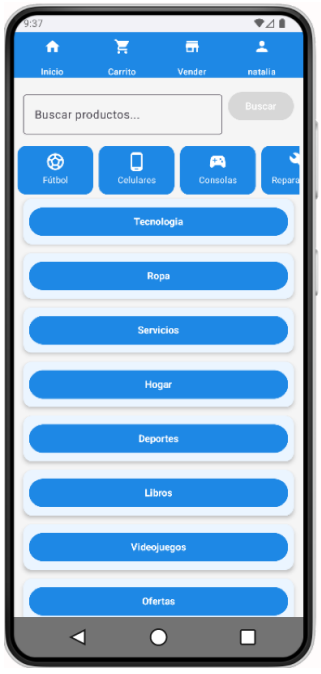


Ilustración 38: Prototipo (Inicio_Vendedor)

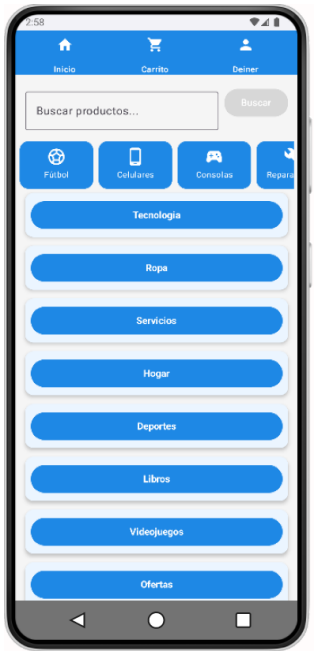


Ilustración 39: Prototipo (Inicio_Usuario)

Recuperar contraseña

Ilustración 40:Prototipo (Recuperar_contraseña)

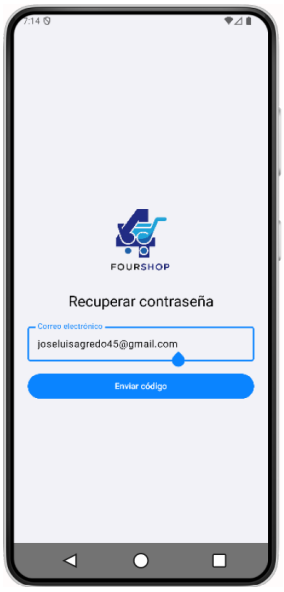


Ilustración 41: Prototipo (Campos_llenos)

Ilustración 42: Prototipo (Codigo_Verificación)

Ilustración 43: Prototipo (Notificación_Código)

Ilustración 44: Prototipo (Código)

Ilustración 45: Prototipo (Crear_Nueva_Contraseña)

Ilustración 46: Prototipo (Nueva_Contraseña)

Categorías y Subcategorías

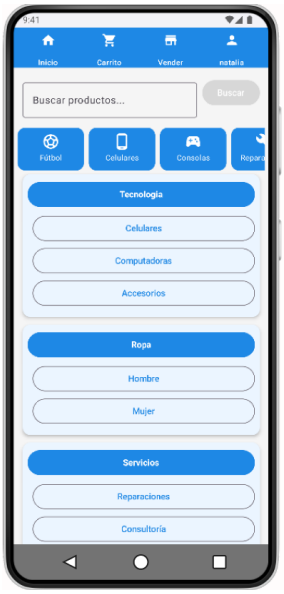


Ilustración 47: Prototipo (Categorias_Subcategorias)

Productos por subcategoría y filtración de productos

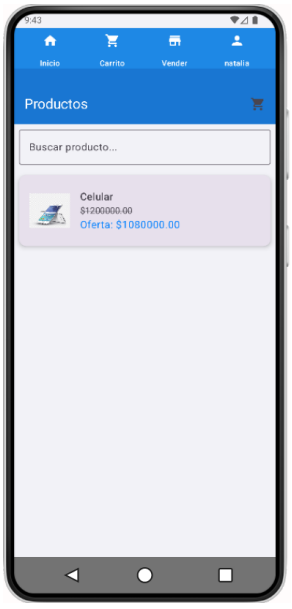


Ilustración 48: Prototipo (Productos)

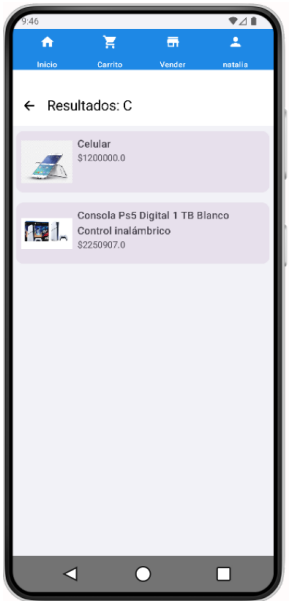


Ilustración 49: Prototipo (Filtrar)

Descripción del producto

Ilustración 50: Prototipo (Descripción_Producto)

Producto agregado al carrito y carrito

Ilustración 51: Prototipo (Agregado_Carrito)

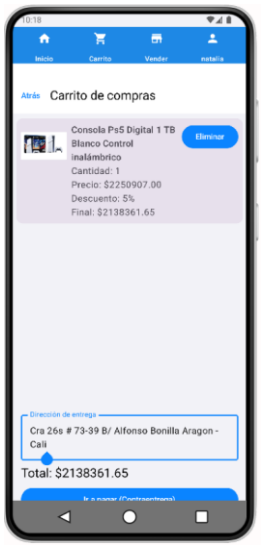


Ilustración 52: Prototipo (Carrito)

Resumen del pedido

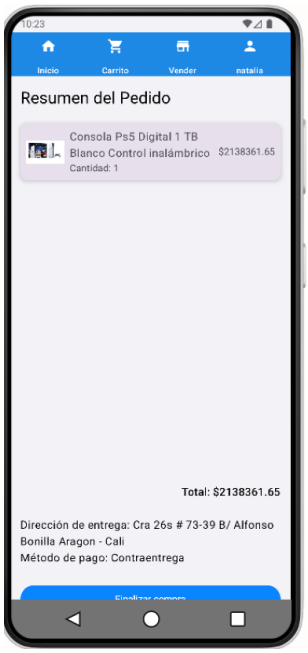


Ilustración 53: Prototipo (Resumen_Pedido)

Registro de compras y ventas

Ilustración 54: Prototipo (Mis Compras)

Ilustración 55: Prototipo (Mis Ventas)

Notificaciones al comprador y vendedor

Ilustración 56: Prototipo (Notificación_Comprador)

Ilustración 57: Prototipo (Notificación_Vendedor)

Perfil

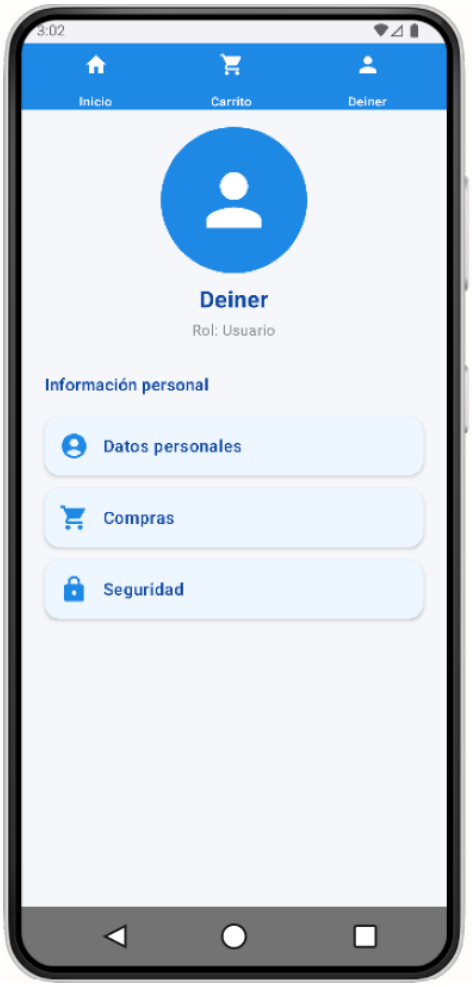


Ilustración 58: Prototipo (Perfil_Usuario)

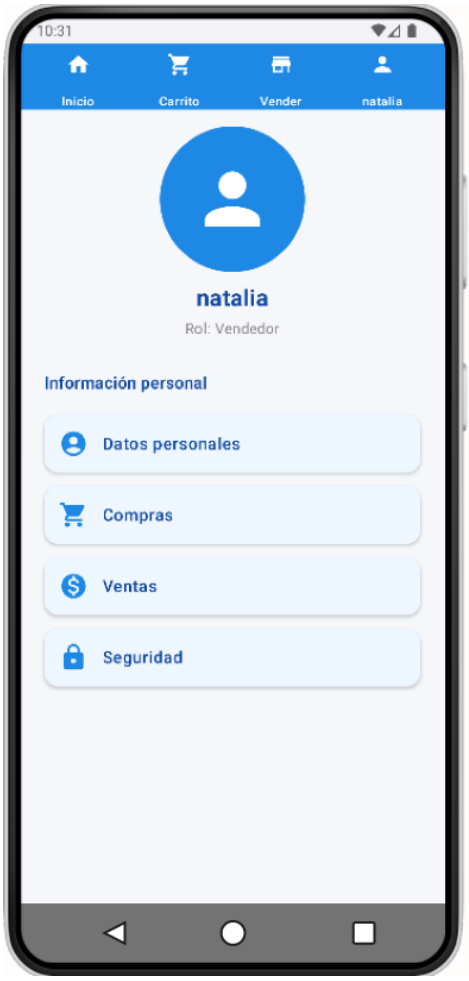


Ilustración 59: Prototipo (Perfil_Vendedor)

Datos personales y seguridad

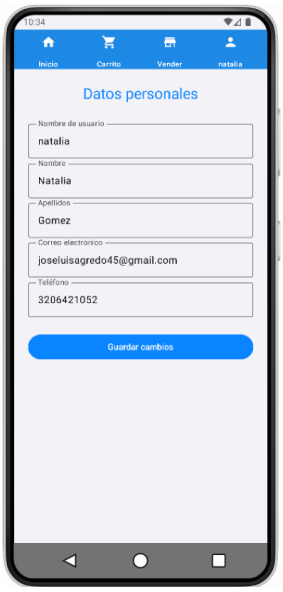


Ilustración 60: Prototipo (Datos_Personales)

Ilustración 61: Prototipo (Datos_Guardados)

Ilustración 62: Prototipo (Seguridad)

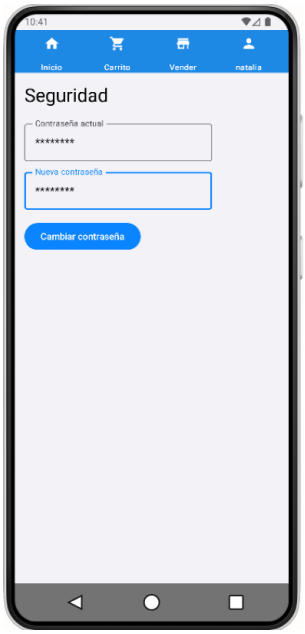


Ilustración 63: Prototipo (Cambio_Contraseña)

Crear producto

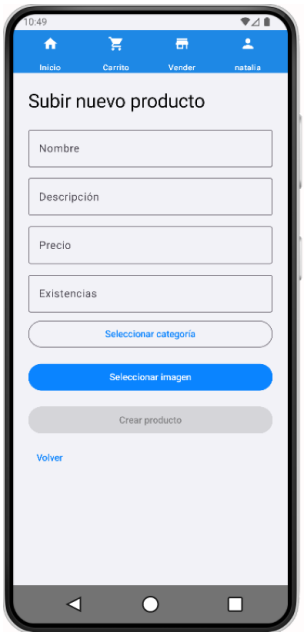


Ilustración 64: Prototipo (Crear_Producto)

Ilustración 65: Prototipo (Datos_Producto)

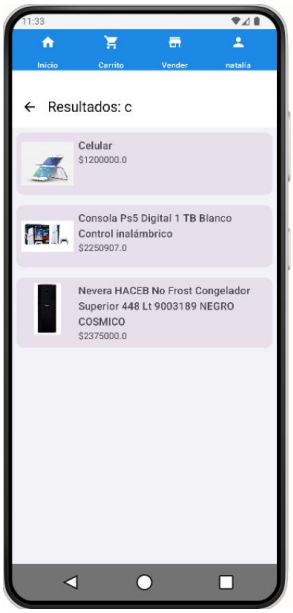


Ilustración 66: Prototipo (Producto_Creado)

Panel administrador (Mis publicaciones)

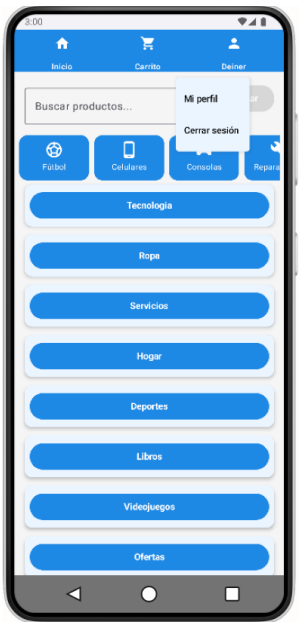


Ilustración 67: Prototipo (Usuario_Sin_Panel)

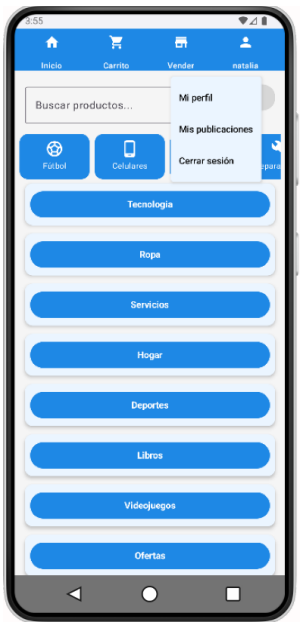


Ilustración 68: Prototipo (Vendedor_Con_Panel)

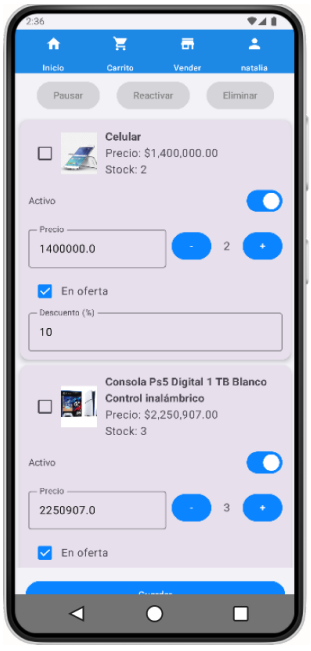


Ilustración 69: Prototipo: (Panel_Administrador)

Arquitectura del sistema

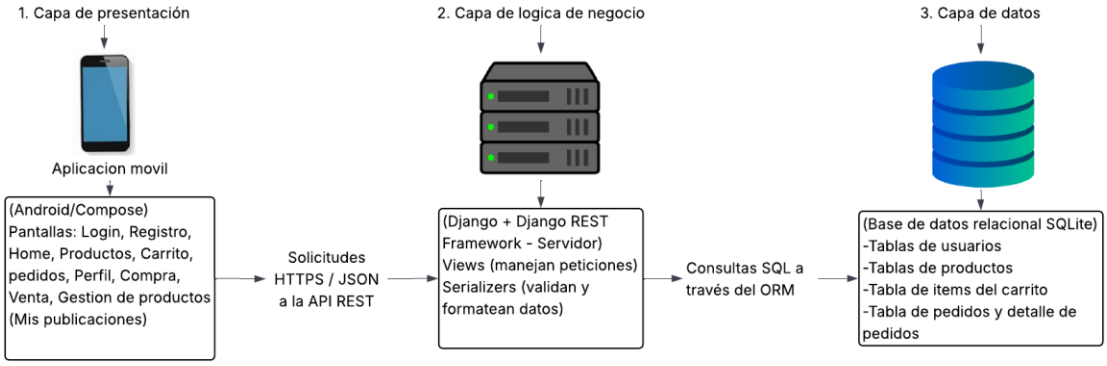


Ilustración 70: Modelo de arquitectura (Capas y componentes)

Diagramas de secuencia

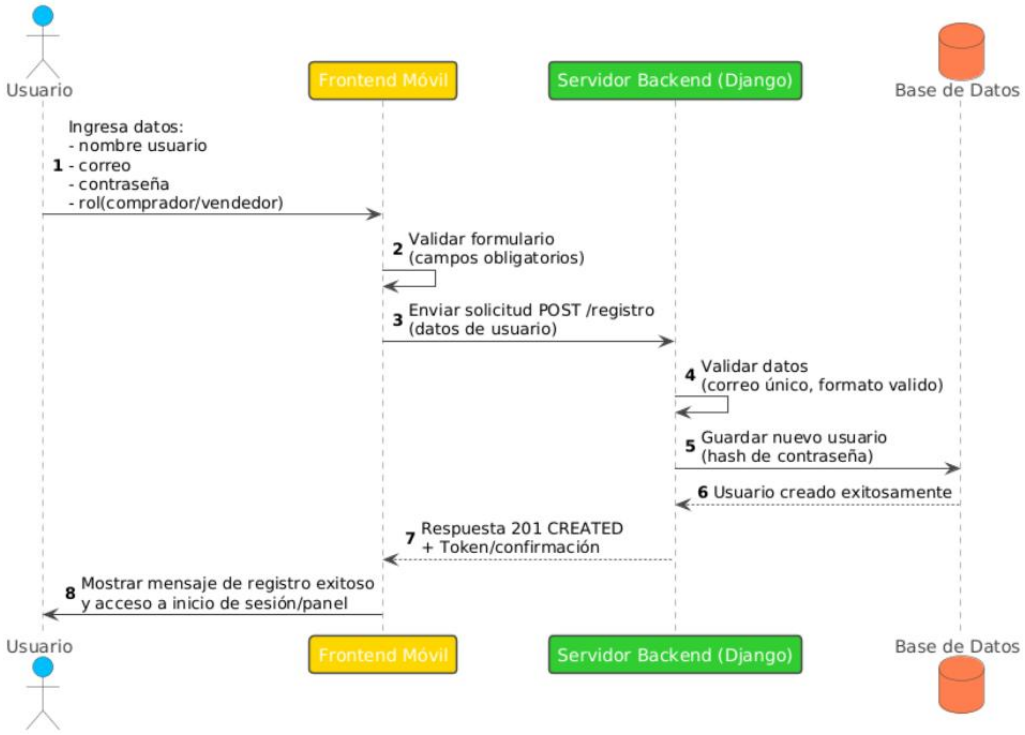


Ilustración 71: Diagrama de flujo (Registro)

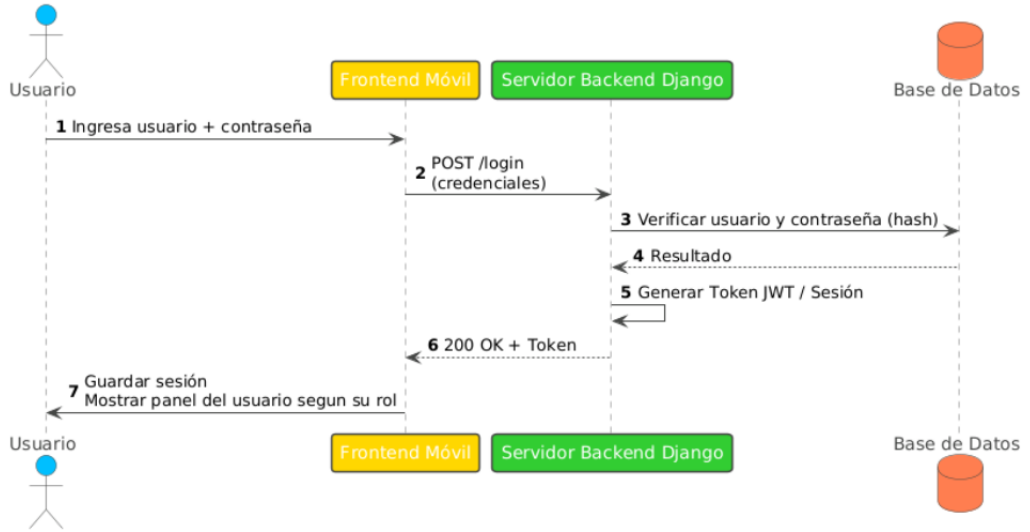


Ilustración 72: Diagrama de flujo (Autenticación/Iniciar sesión)

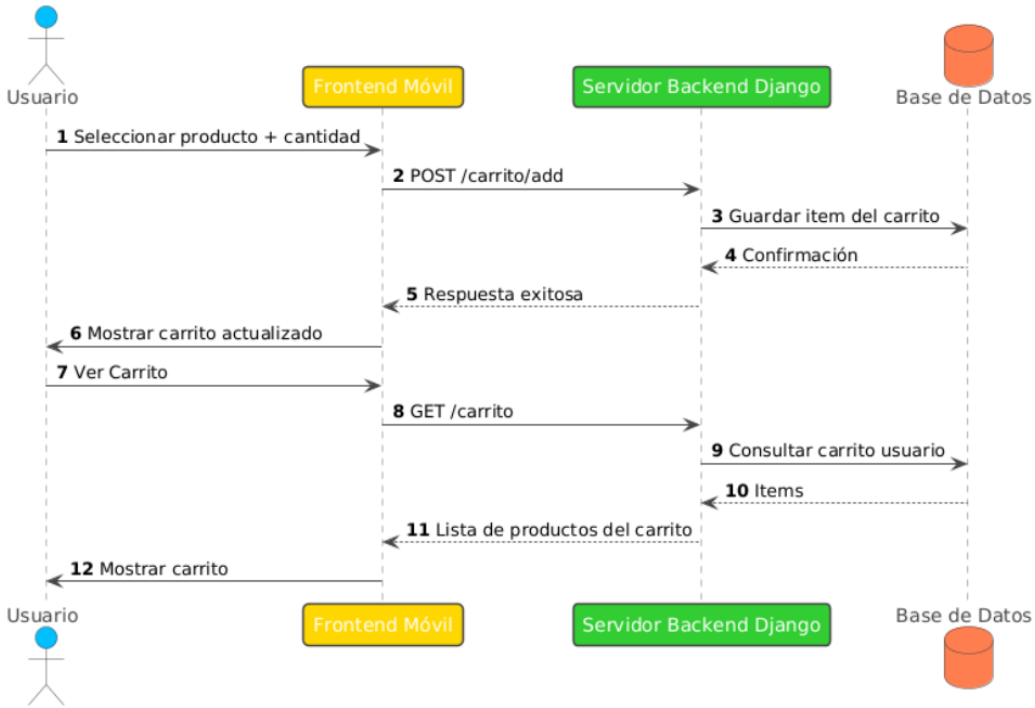


Ilustración 73: Diagrama de flujo (Carrito de compras)

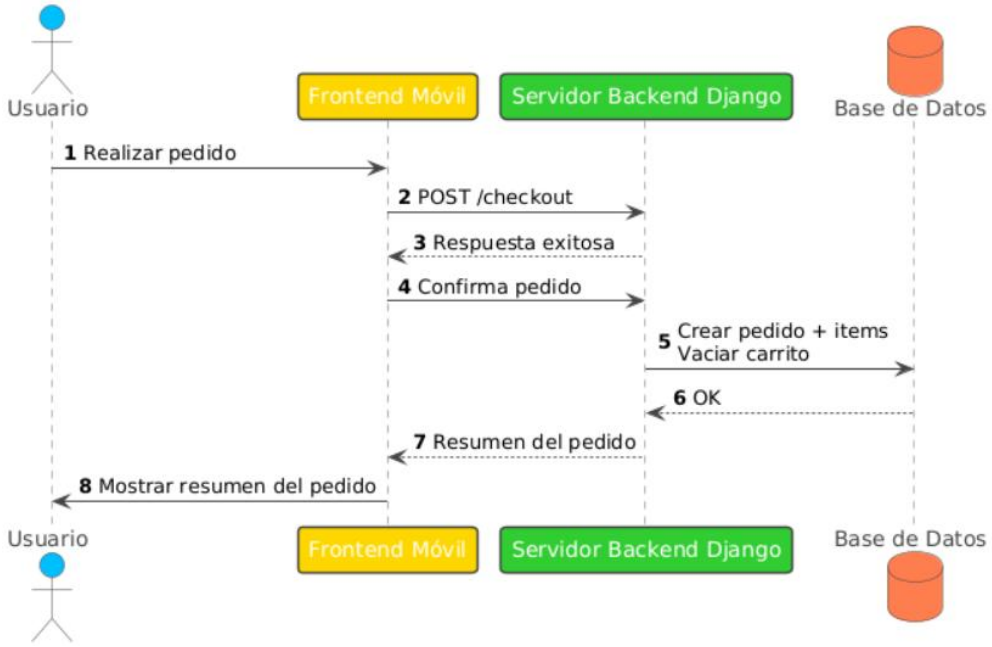


Ilustración 74: Diagrama de flujo (Pedido)

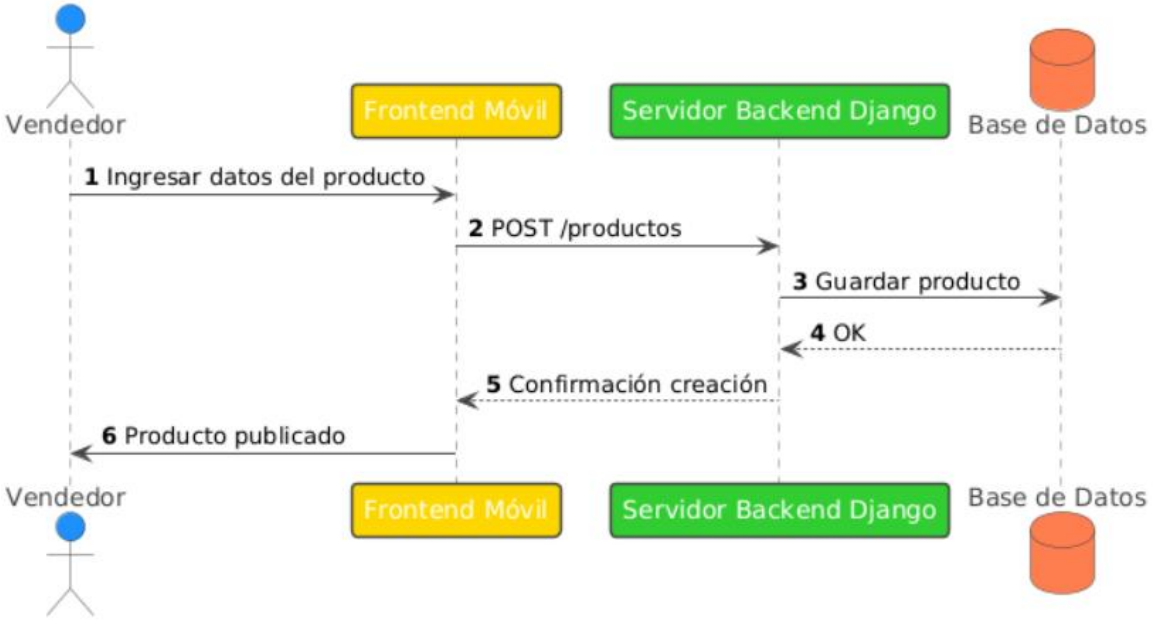


Ilustración 75: Diagrama de flujo (Crear producto)

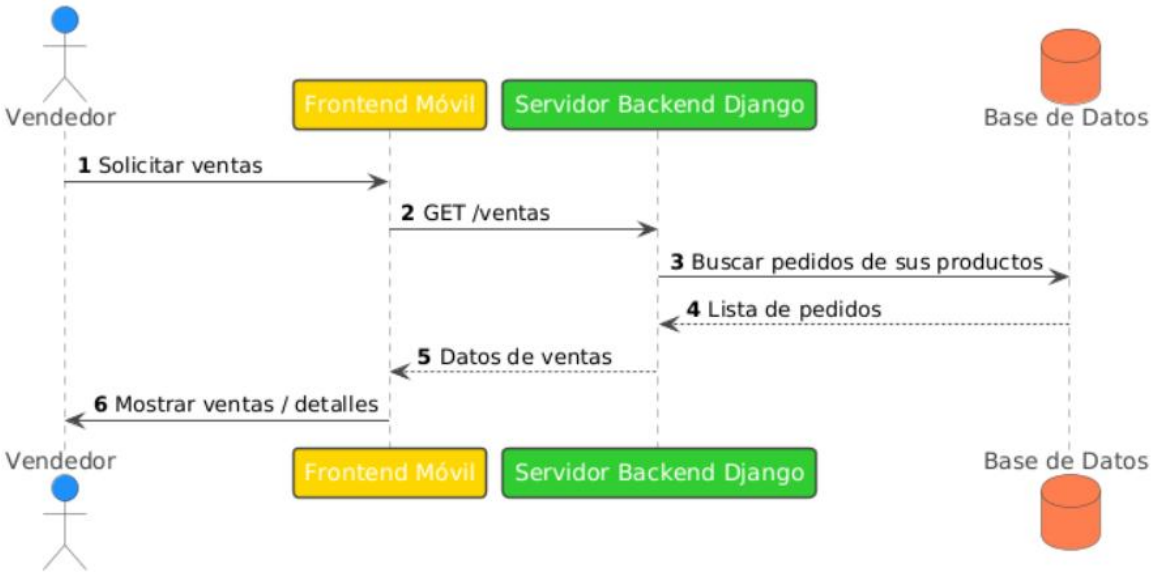


Ilustración 76: Diagrama de flujo (Ventas)

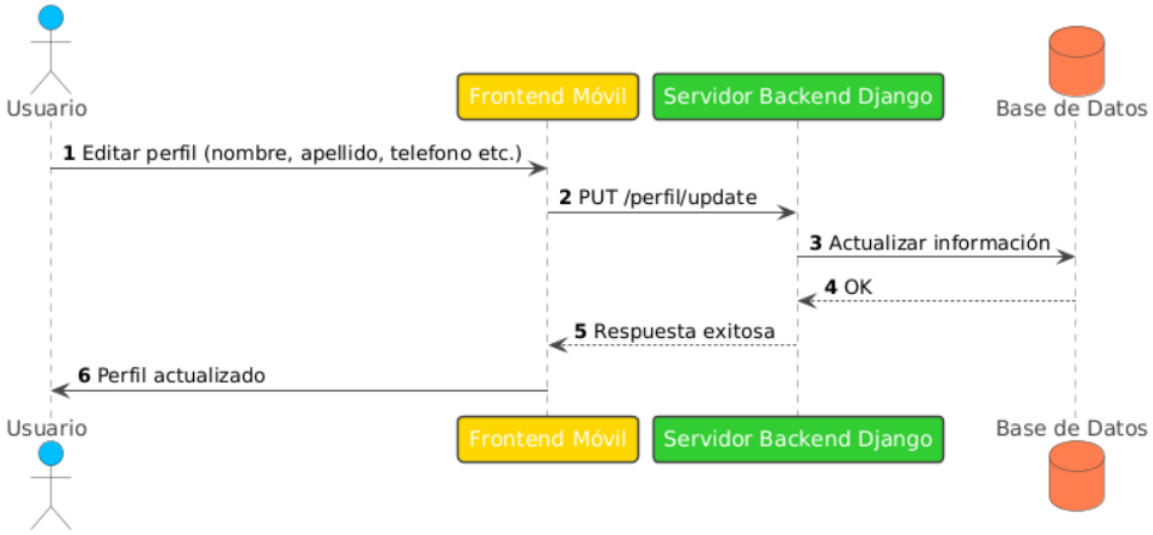


Ilustración 77: Diagrama de flujo (Actualización de Perfil)

Conclusiones del prototipo

Los prototipos juegan un papel muy importante en el desarrollo de la aplicación, ya que al realizar una correcta planificación de como interactuaran los usuarios con el sistema antes de iniciar la implementación final. Verificando su usabilidad y la eficiencia en el flujo de la aplicación, así como la conexión con los requisitos funcionales con la base de datos, garantizando que cada pantalla cumpla con las respectivas funciones según su rol (Comprador o vendedor) dentro del entorno del comercio electrónico.

- Las pantallas de registro e inicio re relacionan directamente con la tabla usuario controlando los permisos según su rol (Comprador/Vendedor).
- La pantalla crear productos se relacionan con las tablas, Producto, Categoría y Subcategoría, reflejando una jerarquía en el catálogo de productos disponibles.
- Las pantallas de carrito y resumen de pedido están asociadas con las tablas Carrito, Pedido y PedidoItem, para la gestión entre los productos seleccionados y la culminación del pedido.
- La pantalla panel vendedor se conecta con las tablas Producto, Venta y usuario, permitiendo administrar stock, precios y ofertas de manera centralizada.
- Las notificaciones se generan tomando el resumen del pedido al momento de finalizar el pedido, donde tanto a la persona que realiza la compra como al vendedor del producto le llega la notificación correspondiente a su rol.

En conclusión, el uso de los prototipos en el desarrollo permite validar la coherencia entre la interfaz de usuario, la lógica funcional, asegurando que la aplicación no solo sea visualmente intuitiva y alineada con los objetivos de negocio planteados, sino que también sea consistente y atractiva.

Actividades para desarrollar el objetivo específico numero 3

Para desarrollar el objetivo específico numero 3: “Construir una versión funcional del prototipo que permita validar la usabilidad, el diseño visual y la interacción del usuario según su rol, garantizando que las operaciones de compra, venta y gestión de productos se realicen de manera intuitiva y eficiente”, se tomó en consideración la información recopilada en la fase de análisis y diseño propuesto para dar inicio al desarrollo de la aplicación siguiendo el ciclo de vida del software.

La construcción de la aplicación se llevó a cabo, integrando una interfaz desarrollada en Android Studio con lenguaje de programación Kotlin mediante Jetpack Compose con los servicios REST del Backend construido en Django REST Framework, asegurando una comunicación eficiente y segura a través de Retrofit y Token Autentication.

Autenticación de usuarios (Login y Registro)

En esta parte del desarrollo se define el flujo de autenticación, que permite al usuario ingresar o crear su cuenta según su rol (Comprador o vendedor).

En este proceso, se implementaron las siguientes funciones claves:

1. **LoginScreen.kt (ui/screens/):** Construye la interfaz del formulario de acceso, gestionando los campos de usuario y contraseña.

```
94              // Botón ingresar
95              Button(
96                  onClick = { viewModel.login(username, password, onLoginSuccess) },
97                  modifier = Modifier.fillMaxWidth(),
98                  enabled = username.isNotBlank() && password.isNotBlank() &&
99                          uiState !is LoginViewModel.State.Loading
100             ) {
101                 Text(
102                     when (uiState) {
103                         is LoginViewModel.State.Loading -> "Cargando..."
104                         else -> "Ingresar"
105                     }
106                 )
107             }
```

Ilustración 78: Desarrollo (Formulario de acceso)

2. **AuthViewModel.kt (ui/viewmodel/):** Gestiona la lógica del Login, usando corutinas y Retrofit para consumir el endpoint /api/Login/ del Backend. Dentro de esta función se envía la solicitud al servidor y se guarda el token JWT en AuthState para mantener la sesión activa.

```
16      // 🔐 LOGIN
17      fun login(username: String, password: String) {
18          viewModelScope.launch {
19              _authState.value = AuthState.Loading
20              try {
21                  val response = authRepository.login(username, password)
                    _authState.value = AuthState.Success(response.access)
23              } catch (e: Exception) {
24                  _authState.value = AuthState.Error("Error de login: ${e.message}")
25              }
26          }
27      }
```

Ilustración 79: Desarrollo (Token_Sesión)

3. **NavRoot.kt:** En este bloque NavHost, donde se usa una condición para restringir el acceso a plantillas especificas según su rol.

```
288                 // Mis publicaciones
289                 if (usuario?.es_vendedor == true) {
290                     composable("mis-publicaciones") {
291                         MisPublicacionesScreen(
292                             productos = productosViewModel.productos,
293                             cargando = productosViewModel.cargando,
294                             error = productosViewModel.error,
295                             onToggleActivo = { id, activo ->
296                                 productosViewModel.actualizarActivo(
297                                     id,
298                                     activo
299                                 )
300                             },
301                             onUpdateStock = { id, stock ->
302                                 productosViewModel.actualizarStock(
303                                     id,
304                                     stock
305                                 )
306                             },
307                             onUpdateOferta = { id, enOferta, descuento ->
308                                 productosViewModel.actualizarOferta(id, enOferta, descuento)
309                             },
310                             onReload = { productosViewModel.cargarMisProductos() },
311                             onPausar = { ids -> productosViewModel.pausarProductos(ids) },
312                             onReactivar = { ids -> productosViewModel.reactivarProductos(ids) },
313                             onEliminar = { ids -> productosViewModel.eliminarProductos(ids) },
314                             onGuardar = {
315                                 navController.navigate("home") {
316                                     popUpTo("mis-publicaciones") { inclusive = true }
317                                 }
318                             },
319                             onUpdatePrecio = { id, nuevoPrecio ->
320                                 productosViewModel.actualizarPrecio(id, nuevoPrecio)
321                             } // ✅ nuevo parámetro agregado
322                         )
323                     }
324                 }
```

Ilustración 80: Desarrollo (Panel_Administrador)

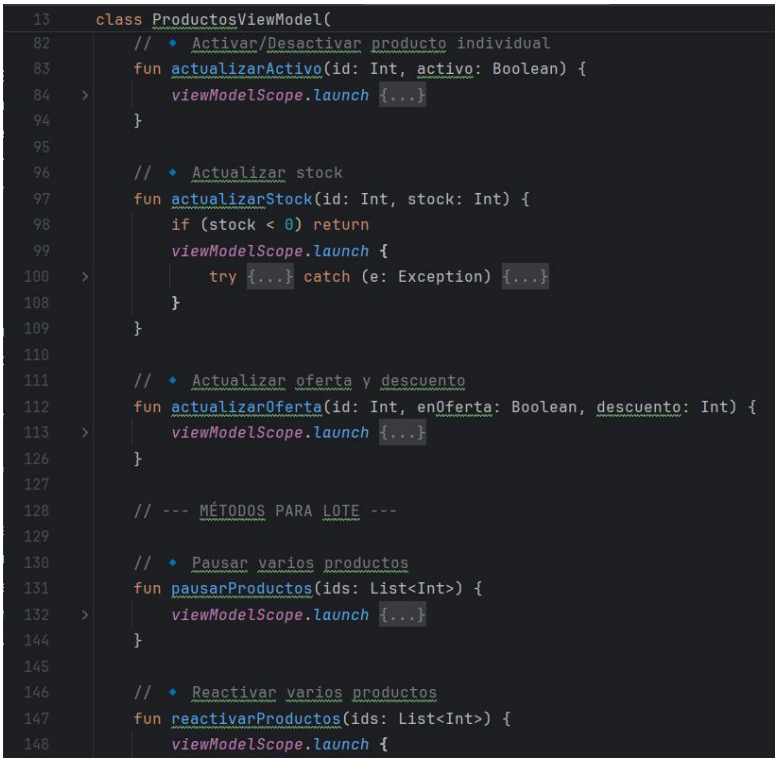
Gestión de productos y publicaciones del vendedor

1. **MisPublicacionesScreen.kt (ui/screens/):** Muestra el listado de productos creados por el vendedor y permite modificarlos. Se utiliza un LazyColum para listar los productos y botones de acción para pausar, reactivar o eliminar.

```
20   fun MisPublicacionesScreen(                                    ⚠1 ✓
57               Column {
82       }
83
84       // --- Lista de productos ---
85       LazyColumn(modifier = Modifier.weight(1f)) {
86           items(productos) { producto ->
87               var descuento by remember { mutableStateOf(producto.descuento) }
88               var precioTexto by remember { mutableStateOf(producto.precio.toString()) }
89               var isChecked by remember { mutableStateOf(false) }
90
91               Card(
92   ›               modifier = Modifier.(...),
95                   elevation = CardDefaults.cardElevation(defaultElevation = 4.dp)
96   ›           ) {...}
218          }
219      }
220
221      // --- Botón Guardar ---
222      Button(
223          onClick = onGuardar,
224          modifier = Modifier
225              .fillMaxWidth()
226              .padding(16.dp)
227      ) {
228          Text("Guardar")
229      }
230  }
```

Ilustración 81: Desarrollo (Gestion_Productos)

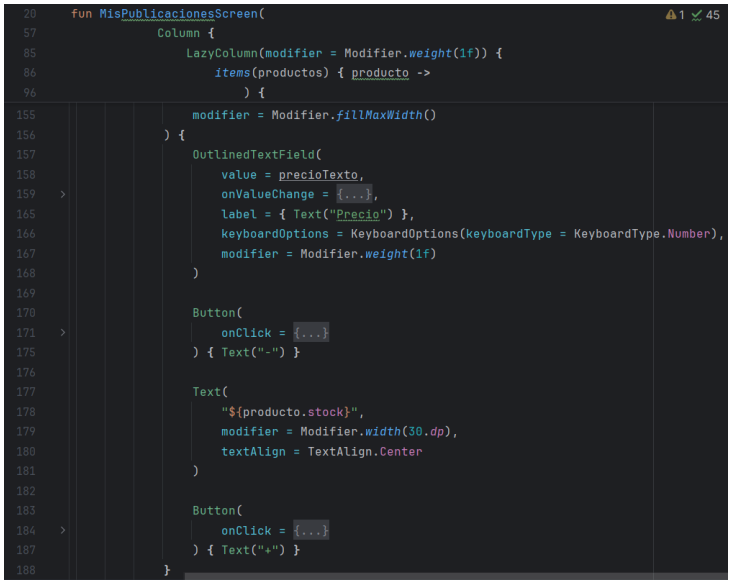
2. **ProductosViewModel.kt:** Contiene las funciones que conectan con el Backend para actualizar la información de cada producto, donde cada función realiza una llamada al endpoint correspondiente (/api/productos/{id}/update/) para reflejar los cambios en tiempo real.



```
13    class ProductosViewModel(
82        // 🔹 Activar/Desactivar producto individual
83        fun actualizarActivo(id: Int, activo: Boolean) {
84            viewModelScope.launch {...}
94        }
95
96        // 🔹 Actualizar stock
97        fun actualizarStock(id: Int, stock: Int) {
98            if (stock < 0) return
99            viewModelScope.launch {
100               try {...} catch (e: Exception) {...}
108           }
109       }
110
111       // 🔹 Actualizar oferta y descuento
112       fun actualizarOferta(id: Int, enOferta: Boolean, descuento: Int) {
113           viewModelScope.launch {...}
126       }
127
128       // --- MÉTODOS PARA LOTE ---
129
130       // 🔹 Pausar varios productos
131       fun pausarProductos(ids: List<Int>) {
132           viewModelScope.launch {...}
144       }
145
146       // 🔹 Reactivar varios productos
147       fun reactivarProductos(ids: List<Int>) {
148           viewModelScope.launch {
```

Ilustración 82: Desarrollo (Actualizaciones_Productos)

3. **Interfaz de edición:** El usuario puede editar detalles de sus productos directamente en la aplicación.



```
20    fun MisPublicacionesScreen(                                    ⚠1 ✓45
57                Column {
85                    LazyColumn(modifier = Modifier.weight(1f)) {
86                        items(productos) { producto ->
96                            ) {
155                       modifier = Modifier.fillMaxWidth()
156                   ) {
157                       OutlinedTextField(
158                           value = precioTexto,
159                           onValueChange = {...},
165                           label = { Text("Precio") },
166                           keyboardOptions = KeyboardOptions(keyboardType = KeyboardType.Number),
167                           modifier = Modifier.weight(1f)
168                       )
169
170                       Button(
171                           onClick = {...}
175                       ) { Text("-") }
176
177                       Text(
178                           "${producto.stock}",
179                           modifier = Modifier.width(30.dp),
180                           textAlign = TextAlign.Center
181                       )
182
183                       Button(
184                           onClick = {...}
187                       ) { Text("+") }
188                   }
```

Ilustración 83: Desarrollo (Editar_Precio)

Carrito de compras y proceso de pedido

1. **CartScreen.kt (ui/screens/):** Muestra los productos añadidos al carrito para procesar el pedido.

Ilustración 84: Desarrollo (Realizar_Pedido)

2. **CartViewModel.kt:** Gestiona la comunicación con los endpoints del Backend para eliminar, confirmar pedidos y vaciar carrito.

Ilustración 85: Desarrollo (Confirmar_Pedido)

Resumen de pedido y confirmación

1. **ResumenPedidoScreen.kt (ui/screens/):** muestra los datos del pedido final antes de confirmar, permite revisar la compra y confirmar con el botón “Finalizar compra”.

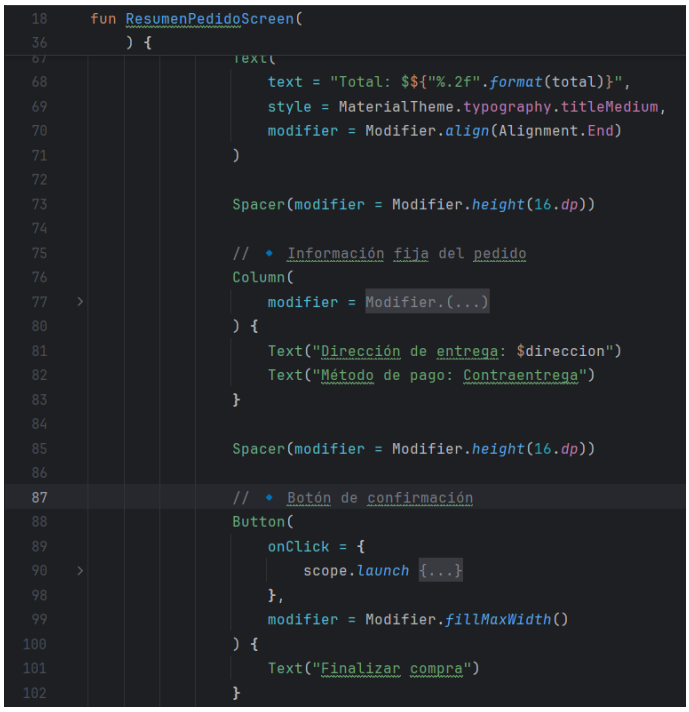


Ilustración 86: Desarrollo (Resumen_Pedido)

2. **AuthService.kt:** Recibe la confirmacion de pedido y consulta el historial de compras.

Ilustración 87: Desarrollo (Obtener_Compras)

Conexión con el Backend Django

El sistema se comunica con la API mediante Retrofit, definiendo endpoints seguros y autenticados.

1. **AuthApi.kt y ProductosApi.kt:** Contienen las rutas y métodos HTTP (GET, POST, PATH, DELETE) que son los que se encargan de interactuar con el Backend y la base de datos.

```
interface AuthApi {

    // ---------- Perfil de Usuario ----------

    @GET("perfil/")
    suspend fun obtenerPerfil(): UserResponse

    @PATCH("perfil/")
    suspend fun actualizarPerfil(@Body body: Map<String, String>): Response<Unit>

    @POST("perfil/verificar-password/")
    suspend fun cambiarPassword(@Body request: PasswordChangeRequest): Response<Unit>
```

Ilustración 88: Desarrollo (Rutas_Api)

2. **RetrofitClient.kt:** Configura el cliente HTTP, base URL y token interceptor.

```
package com.fourshop.mobile.data.api

import ...

object RetrofitClient {
    // Emulador Android Studio -> 10.0.2.2 apunta a localhost del PC
    private const val BASE_URL = "http://10.0.2.2:8000/api/"

    private lateinit var authService: AuthService

    // Inicializa Retrofit con el AuthService
    fun init(context: Context, authApi: AuthApi) {
        authService = AuthService(authApi, context)

        val okHttpClient = OkHttpClient.Builder()
            .addInterceptor(AuthInterceptor(authService))
            .build()

        retrofit = Retrofit.Builder()
            .baseUrl(BASE_URL)
            .client(okHttpClient)
            .addConverterFactory(GsonConverterFactory.create())
            .build()
    }
```

Ilustración 89: Desarrollo (Conexión_Backend)

Estructura general del Backend Django

El Backend se realizó con Django + Django REST Framework (DRF), lo que hace que el servidor ofrezca endpoints API REST, los cuales son los que la aplicación móvil consume mediante peticiones HTTP, usando JSON para enviar y recibir datos.

1. **models.py:** Su función es definir como se guardan los datos en la DB, donde cada clase representa una tabla en la base de datos, y cada atributo representa una columna.

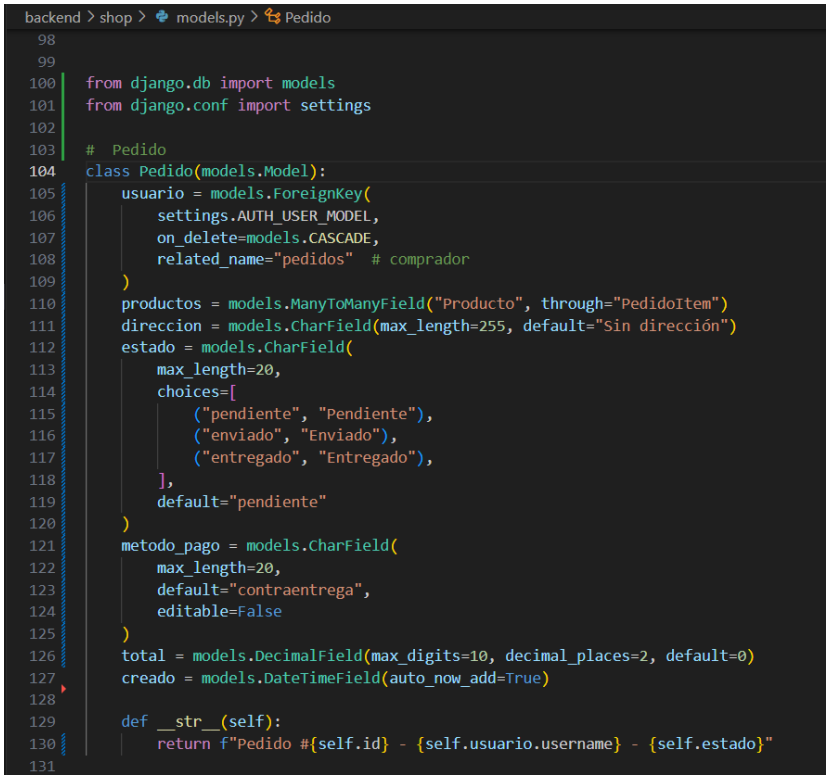


Ilustración 90: Desarrollo (Backend_models.py)

La clase crea la estructura de los datos, Django genera automáticamente las tablas SQL y permite guardar, consultar, actualizar o eliminar objetos. Por ejemplo, cuando la aplicación móvil pide (/api/pedido), Django consulta la tabla pedido y devuelve los datos.

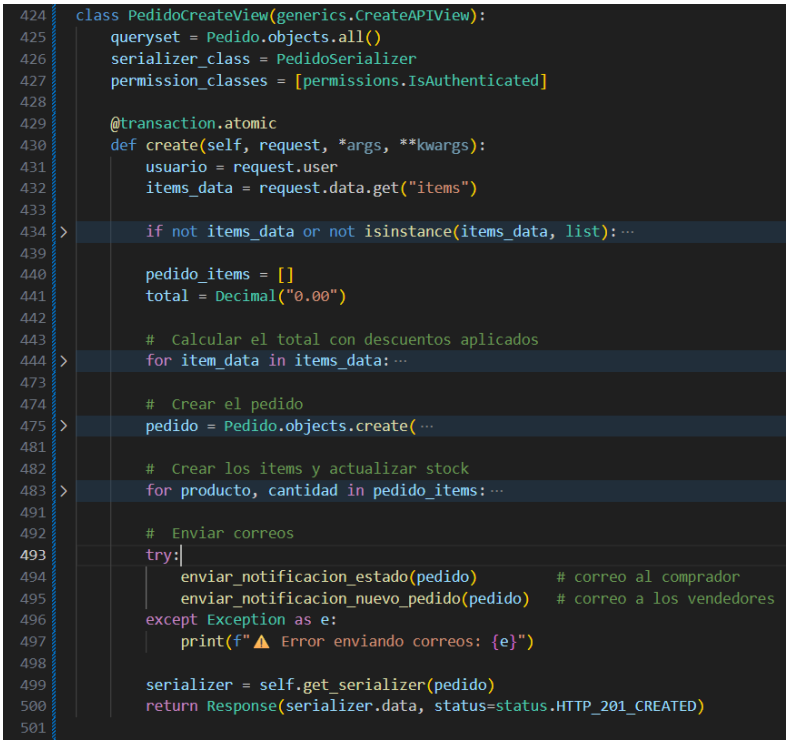
2. **serializaers.py:** Se encarga de convertir los objetos del modelo (Python) a JSON, y viceversa, lo que permite que la aplicación móvil pueda enviar y recibir datos legibles.

```python
# --- Serializer para crear Pedidos ---
class PedidoSerializer(serializers.ModelSerializer):
    items = PedidoItemSerializer(many=True, write_only=True)
    detalle_items = PedidoItemSerializer(source="items", many=True, read_only=True)
    fecha = serializers.DateTimeField(source="creado", read_only=True)

    class Meta:
        model = Pedido
        fields = ["id", "usuario", "direccion", "estado", "metodo_pago", "total", "fecha", "detalle_items", "items"]
        read_only_fields = ["metodo_pago", "estado", "usuario", "total", "fecha"]

    def create(self, validated_data):
        items_data = validated_data.pop("items")
        user = self.context["request"].user
        pedido = Pedido.objects.create(usuario=user, metodo_pago="contraentrega", **validated_data)
        total = 0
        for item in items_data:
            producto = item["producto"]
            cantidad = item["cantidad"]
            precio_final = producto.precio
            if producto.en_oferta and producto.descuento:
                precio_final = precio_final - (precio_final * producto.descuento / 100)
            subtotal = precio_final * cantidad
            total += subtotal

            PedidoItem.objects.create(
                pedido=pedido,
                producto=producto,
                cantidad=cantidad,
                subtotal=subtotal
            )

        pedido.total = total
        pedido.save()
        return pedido
```

Ilustración 91: Desarrollo (Backend_serializers.py)

La clase convierte el pedido a JSON para que la aplicación lo reciba, valida los datos cuando la aplicación envía información al servidor (Crear un pedido); y cuando la aplicación móvil envía datos JSON al Backend, el serializer lo que hace es validar el formato y crea un nuevo objeto en la base de datos.

3. **Views.py:** Contiene la lógica que responde a las peticiones (GET, POST, PATH, DELETE), donde cada vista se asocia a una URL y usa los modelos y serializers para procesar los datos.

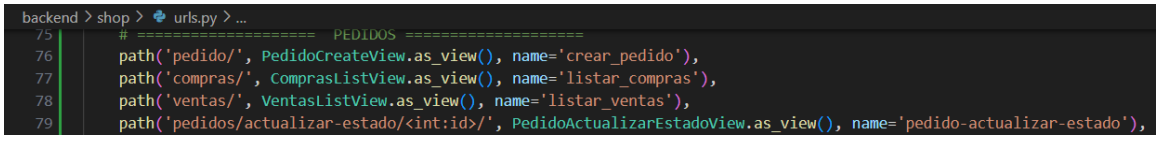


```python
class PedidoCreateView(generics.CreateAPIView):
    queryset = Pedido.objects.all()
    serializer_class = PedidoSerializer
    permission_classes = [permissions.IsAuthenticated]

    @transaction.atomic
    def create(self, request, *args, **kwargs):
        usuario = request.user
        items_data = request.data.get("items")

        if not items_data or not isinstance(items_data, list):···

        pedido_items = []
        total = Decimal("0.00")

        #  Calcular el total con descuentos aplicados
        for item_data in items_data:···

        #  Crear el pedido
        pedido = Pedido.objects.create(···

        #  Crear los items y actualizar stock
        for producto, cantidad in pedido_items:···

        #  Enviar correos
        try:
            enviar_notificacion_estado(pedido)          # correo al comprador
            enviar_notificacion_nuevo_pedido(pedido)    # correo a los vendedores
        except Exception as e:
            print(f"⚠️ Error enviando correos: {e}")

        serializer = self.get_serializer(pedido)
        return Response(serializer.data, status=status.HTTP_201_CREATED)
```

Ilustración 92: Desarrollo (Backend_views.py)

Si la aplicación realiza un GET (api/pedido/), views.py devuelve todos los pedidos, si realiza un POST (api/pedido/), crea uno nuevo (si el usuario tiene rol vendedor), views.py también se encarga de controlar los permisos, la autenticación y la lógica personalizada.

4. **Urls.py:** Es donde se definen las rutas que apuntan a las vistas del Backend y se encarga de conectar las URLs externas con las funciones o clases internas.



```
backend > shop > urls.py > ...
    # ====================  PEDIDOS ====================
    path('pedido/', PedidoCreateView.as_view(), name='crear_pedido'),
    path('compras/', ComprasListView.as_view(), name='listar_compras'),
    path('ventas/', VentasListView.as_view(), name='listar_ventas'),
    path('pedidos/actualizar-estado/<int:id>/', PedidoActualizarEstadoView.as_view(), name='pedido-actualizar-estado'),
```

Ilustración 93: Desarrollo (Backend_urls.py)

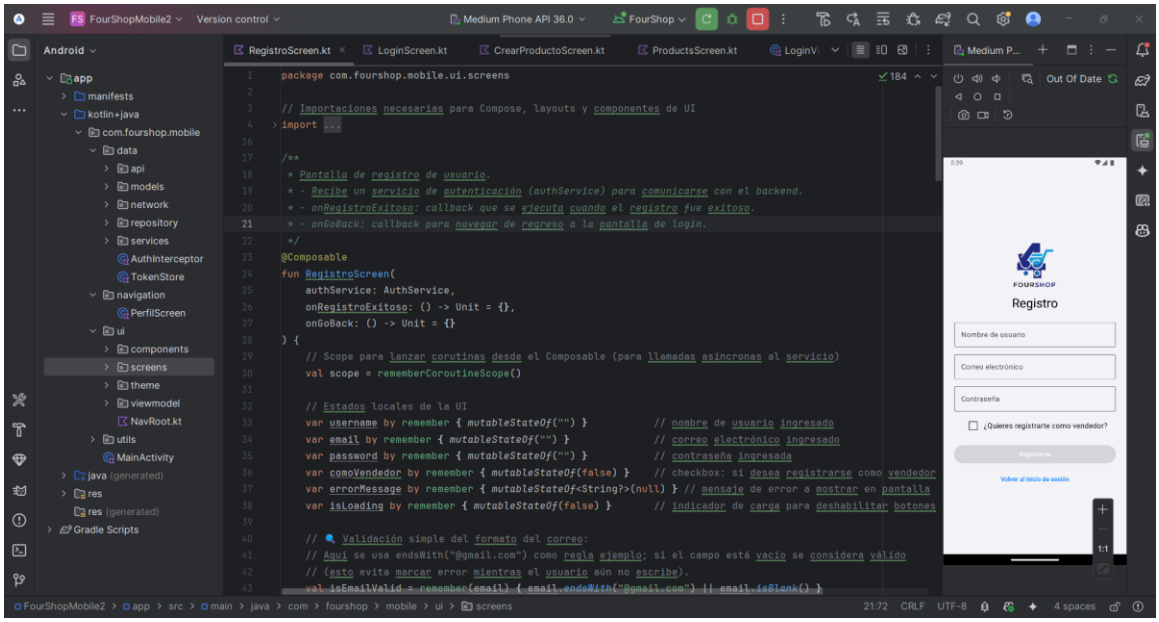
Esta URL apunta a la vista PedidoCreateView, y cuando se realiza la llamada HTTP, esta URL llega al Backend, se redirige a la vista, usa el serializer y se realizan las consultas.

5. **Admin.py:** Permite gestionar los modelos desde el panel de administración de Django.

```
backend > shop > admin.py > ...
1
2   from django.contrib import admin
3   from django.contrib.auth.admin import UserAdmin
4   from .models import Categoria, Subcategoria
5
6   admin.site.register(Categoria)
7   admin.site.register(Subcategoria)
8   from .models import Producto, Usuario, CarritoItem, Carrito, Pedido, PedidoItem, CodigoVerificacion, Venta, Compra
9   class ProductoAdmin(admin.ModelAdmin):
10      list_display = ('nombre', 'categoria', 'subcategoria', 'precio', 'vendedor')
11      list_filter = ('categoria', 'subcategoria')
12      search_fields = ('nombre', 'descripcion')
13
14  class UsuarioAdmin(UserAdmin):
15      list_display = ('username', 'email', 'es_vendedor', 'is_staff', 'is_superuser', 'is_active')
16      list_filter = ('es_vendedor', 'is_staff', 'is_superuser', 'is_active')
17      fieldsets = UserAdmin.fieldsets + (
18          ('Rol', {'fields': ('es_vendedor', 'telefono')}),
19      )
20
21  admin.site.register(Producto, ProductoAdmin)
22  admin.site.register(Usuario, UsuarioAdmin)
23  admin.site.register(CarritoItem)
24  admin.site.register(Carrito)
25  admin.site.register(Pedido)
26  admin.site.register(PedidoItem)
27  admin.site.register(CodigoVerificacion)
28  admin.site.register(Venta)
29  admin.site.register(Compra)
```

Ilustración 94: Desarrollo (Backend_admin.py)

Documentacion del código y estructura de carpetas



```
package com.fourshop.mobile.ui.screens

// Importaciones necesarias para Compose, layouts y componentes de UI
import ...

/**
 * Pantalla de registro de usuario.
 * - Recibe un servicio de autenticación (authService) para comunicarse con el backend.
 * - onRegistroExitoso: callback que se ejecuta cuando el registro fue exitoso.
 * - onGoBack: callback para navegar de regreso a la pantalla de login.
 */
@Composable
fun RegistroScreen(
    authService: AuthService,
    onRegistroExitoso: () -> Unit = {},
    onGoBack: () -> Unit = {}
) {
    // Scope para lanzar corutinas desde el Composable (para llamadas asincronas al servicio)
    val scope = rememberCoroutineScope()

    // Estados locales de la UI
    var username by remember { mutableStateOf("") }          // nombre de usuario ingresado
    var email by remember { mutableStateOf("") }             // correo electrónico ingresado
    var password by remember { mutableStateOf("") }          // contraseña ingresada
    var comoVendedor by remember { mutableStateOf(false) }   // checkbox: si desea registrarse como vendedor
    var errorMessage by remember { mutableStateOf<String?>(null) } // mensaje de error a mostrar en pantalla
    var isLoading by remember { mutableStateOf(false) }      // indicador de carga para deshabilitar botones

    // 🔍 Validación simple del formato del correo:
    // Aquí se usa endsWith("@gmail.com") como regla ejemplo; si el campo está vacío se considera válido
    // (esto evita marcar error mientras el usuario aún no escribe).
    val isEmailValid = remember(email) { email.endsWith("@gmail.com") || email.isBlank() }
```

Ilustración 95: Documentacion y estructura de carpetas (Android Studio)

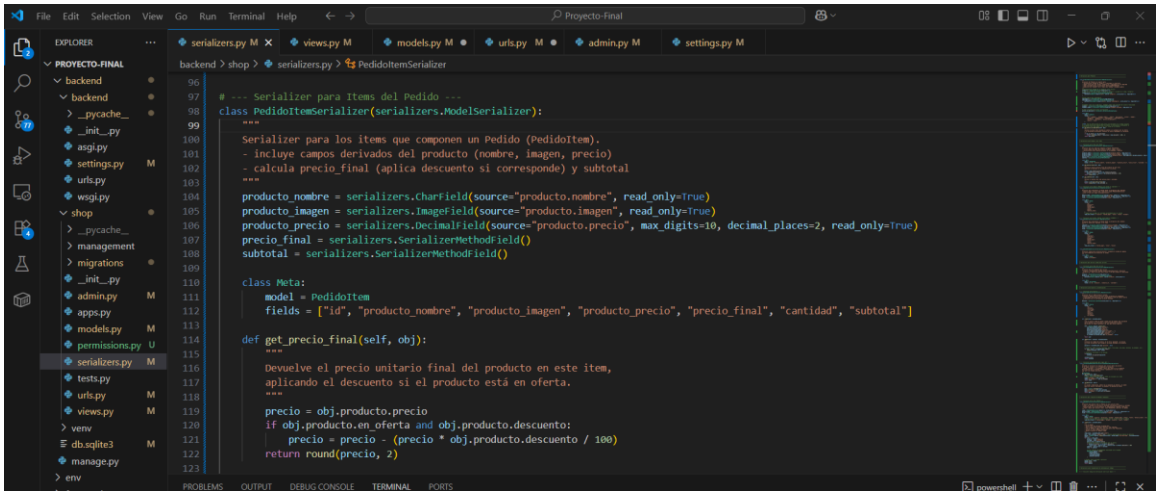


Ilustración 96: Documentacion y estructura de carpetas (Visual Studio Code)

Integración Backend-Frontend

El proceso de integración entre el Backend en Django REST Framework y el Frontend móvil en Android – Jetpack Compose se genera mediante la comunicación por medio de API REST, donde se utilizan peticiones HTTP para lograr el intercambio de datos.

El Frontend permite que el usuario interactúe con la aplicación, donde puede realizar las siguientes acciones:

- Registrarse
- Iniciar sesión
- Ver productos
- Agregar productos al carrito
- Realizar pedidos
- Crear productos (vendedor)
- Gestionar productos (Mis publicaciones)

El Backend se encarga del procesamiento de la lógica de negocio, donde se realizan las siguientes acciones:

- Validar datos
- Gestión de usuarios
- Carrito
- Pedidos
- Consultas de productos
- Acceso a la base de datos

Esta integración es importante debido a que sin ella el Frontend sería solo pantallas estáticas sin posibilidad de navegar a través de ellas y realizar las diferentes acciones que ofrece cada una y el Backend solo sería una máquina de datos sin usuarios que puedan realizar las peticiones.

Manejo de errores y Excepciones en el proyecto

El manejo de errores se implementó tanto en el Backend como en el frontend, asegurando que el usuario reciba mensajes claros cuando ocurre un problema sin que la aplicación se detenga inesperadamente.

En el Backend se usaron excepciones controladas y validaciones de datos mediante los Serializers de Django y así lograr que haya un retorno de códigos de estado adecuados.

- Datos no validos (Código 400)
- No autenticado (Código 401)
- Sin permisos (Código 403)
- Recurso no encontrado (Código 404)
- Error interno en el servidor (Código 500)

En el Frontend se utilizaron boques try/catch y manejo de estados de error.

- Mostrar mensajes al usuarios
- Evitar cierres inesperados de la aplicación

El manejo de excepciones brinda una mayor estabilidad evitando que la aplicación se caiga, mejor experiencia de usuario mediante mensajes claros, seguridad al proteger la información y una depuración más fácil con mensajes ayudan a encontrar el lugar donde se está presentando el problema.

Retos en la fase de diseño y desarrollo

Durante la fase de diseño y desarrollo de la aplicación FourShop, se presentaron diversos desafíos técnicos y de integración que pusieron a prueba nuestra capacidad de planificación, análisis y solución de problemas, pero principalmente el trabajo en equipo.

integración entre Backend y la aplicación móvil.

Un gran desafío fue lograr la comunicación estable y segura entre el Backend desarrollado en Django REST Framework en Visual Studio Code y la aplicación móvil desarrollada en Android Studio con Kotlin + Jetpack Compose, donde se presentaron muchos errores en la sincronización de datos. Donde se tuvo que implementar un control más riguroso al momento de realizar las solicitudes HTTP, para poder asegurar una estructura coherente de los endpoints en Django.

El diseño de una interfaz intuitiva y visualmente atractiva represento un reto importante, ya que debía tener un enfoque centrado en el usuario y el rol que iba a desempeñar en la aplicación, realizando diversas pruebas, ajustando elementos esenciales como colores, la distribución y la navegación, todo con el propósito de lograr una experiencia fluida para el usuario final.

La seguridad y la autenticación de usuarios, fue un reto clave para poder garantizar la seguridad de la información personal de los usuarios, donde se optó por el uso de tokens JWS, para proteger las sesiones del usuario, así como el uso de HTTPS para tener un cifrado de las comunicaciones, el uso de permisos para diferenciar las funciones de los usuarios, evitando accesos no autorizados.

La implementación de notificaciones para recuperar la contraseña y las notificaciones para confirmar la realización del pedido, tanto para el comprador como vendedor, también implico un reto importante debido a que se debía tener claro que datos se debían enviar tanto para el comprador como para el vendedor de tal manera que ambos roles se enteraran de la acción que se estaba llevando a cabo.

Durante la etapa de desarrollo del proyecto, la comunicación y el trabajo en equipo representaron un aspecto fundamental para el éxito de la aplicación, donde a lo largo de la fase de diseño y desarrollo, surgieron retos asociados al trabajo colaborativo, donde la expresión de ideas y estrategias especificas jugaron un papel fundamental para superar cada etapa.

Actividades para desarrollar el objetivo específico numero 4

Para desarrollar el objetivo específico numero 4: “Evaluar el desempeño y la usabilidad de la aplicación mediante pruebas piloto con vendedores y clientes, con el fin de identificar posibles mejoras en su funcionalidad y experiencia de uso”, para verificar la operabilidad y el rendimiento general de la aplicación FourShop, se realizaron pruebas piloto que abarcaron desde registro de usuarios hasta la confirmacion de compras y la gestión de pedidos entre vendedores y compradores mediante notificación por correo electrónico.

Objetivo de las pruebas

Garantizar que la aplicación móvil funcione correctamente en sus módulos principales.

Esto se logra evaluando que:

- Las funcionalidades cumplan con los requerimientos definidos.
- La experiencia de usuario sea estable.
- La comunicación entre el Frontend y el Backend sea confiable.
- Se validen los roles de usuario (vendedor / comprador).

Alcance de las pruebas

Se probará	No se probará
Registro e inicio de sesión	Pasarelas de pago
Listado de productos	Pruebas masivas de rendimiento
Carrito de compras	Compatibilidad con iOS
Pedido	Automatización
Gestión de productos y ventas	Soporte multi-idioma
Validación de roles	

Tabla 47: Alcance de las pruebas

Tipos de pruebas a realizar

Pruebas	descripción
Unitarias	Validar funciones criticas individuales
Integración	Verificar interacción entre módulos
Funcionales	Validar requisitos funcionales principales

Tabla 48: Tipos de pruebas

Criterios de aceptación

Las funcionalidades se considerarán aprobadas si:

- Cumple con el requerimiento
- Interactúa de manera correcta con la API
- Maneja errores sin crashear
- Presenta mensajes adecuados al usuario

Pruebas unitarias

Validación de correo en registro				
descripción	Verificar que el campo email solo acepte correos validos			
Precondiciones	Estar en la pantalla de registro			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PU-001	1.Ingresar un correo invalido. 2.Pulsar botón “Registrarse”	El sistema muestra un mensaje de error: “Por favor ingrese un correo valido (@gmail.com)”	Se mostro el mensaje correctamente	Exitoso

Tabla 49: Prueba unitaria (Validación de correo en registro)

Validación de campos obligatorios				
descripción	Validar que no se permita registrar usuario con campos vacíos			
Precondiciones	Formulario vacío			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PU-002	1.Revisar si el botón "Registrarse" está activo	El botón debe estar deshabilitado.	Se mostro la validación correctamente	Exitoso

Tabla 50: Prueba unitaria (Validación de campos obligatorios)

Marca de rol vendedor				
descripción	Validar que el rol vendedor se envié como boolean en el registro			
Precondiciones	Casilla ¿Quieres registrarte como vendedor? disponible			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PU-001	1.Marcar checkbox 2.Desmarcar checkbox	es_vendedor = True/False en el request	Valor correcto enviado al Backend	Exitoso

Tabla 51: Prueba unitaria (Marca de rol vendedor)

Evidencias pruebas unitarias

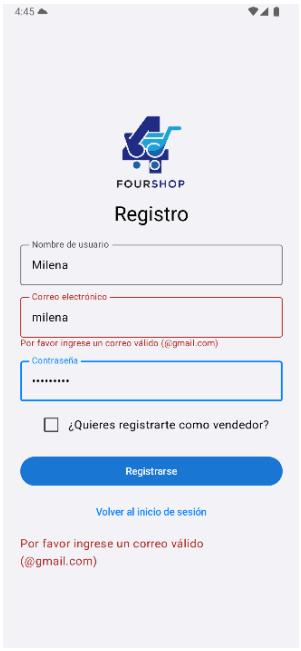


Ilustración 97: Prueba unitaria (validación de correo en registro)

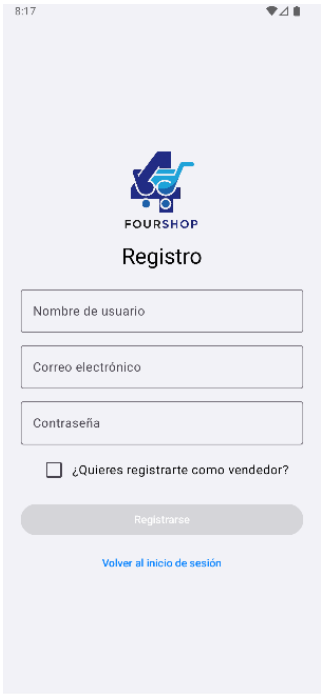


Ilustración 98: Prueba unitaria (validación de campos obligatorios)

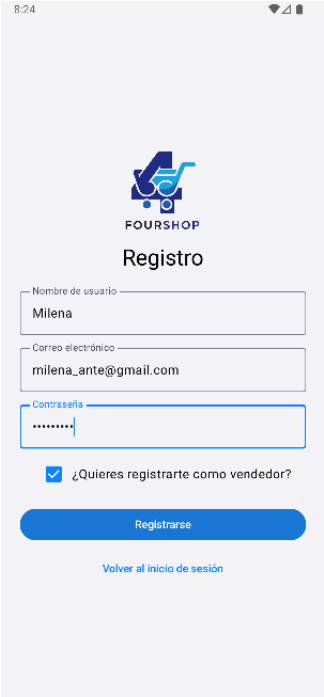


Ilustración 99: Prueba unitaria (checkbox marcado)

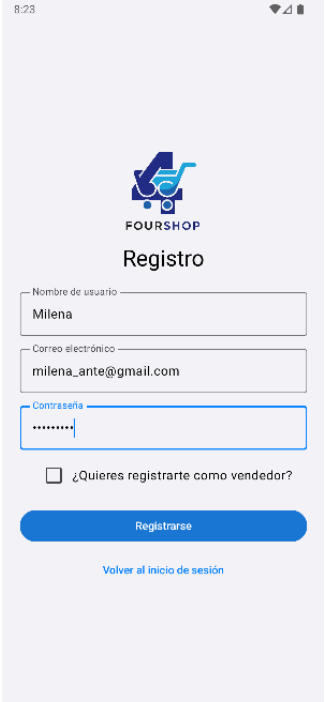


Ilustración 100: Prueba unitaria (checkbox desmarcado)

Pruebas de integración

Registro conectado al Backend				
descripción	Validar que el usuario cree el registro en el Backend			
Precondiciones	Datos validos de registro			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PI-001	1.Registrar usuario 2.Enviar datos al Backend	El Backend retorna 201 Created y usuario registrado	Usuario creado correctamente	Exitoso

Tabla 52: Prueba de integración (Registro conectado al Backend)

Login con JWT				
descripción	Verificar autenticación y recepción del token			
Precondiciones	Usuario registrado			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PI-002	1.Ingresar credenciales 2.Pulsar ingresar	Token recibido y enviado al usuario	Token recibido y guardado	Exitoso

Tabla 53: Prueba de integración (Login con JWT)

Carga de productos desde API				
descripción	Verificar consumo de API para listado de productos			
Precondiciones	Token valido			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PI-003	1.Abrir pantalla de Home	Lista de productos retornada por API	Lista cargo correctamente	Exitoso

Tabla 54: Prueba de integración (Carga de productos desde la API)

Agregar producto al carrito				
descripción	Validar integración entre UI y endpoint carrito			
Precondiciones	Usuario autenticado			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PI-004	1.Clic en "Agregar al carrito" 2.Ver actualización en carrito	API retorna "Producto agregado al carrito" y UI actualiza cantidad	Operación correcta	Exitoso

Tabla 55: Prueba de integración (Agregar producto al carrito)

Procesar pedido en el Backend				
descripción	Verificar creación de pedidos con Ítems			
Precondiciones	Carrito con productos			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PI-005	1.Clic en "Ir a pagar (Contraentrega)" 2.Confirmar pedido dando clic en "Finalizar compra"	Pedido creado exitosamente y carrito vacío	Pedido registrado	Exitoso

Tabla 56: Prueba de integración (Procesar pedido en el Backend)

Evidencias pruebas de integración

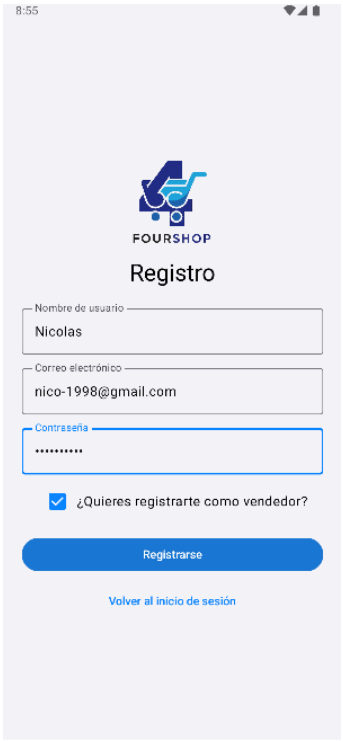


Ilustración 101: Prueba integración (Registro conectado al Backend)

```
(venv) PS C:\Users\sandr\Proyecto-Final\backend> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
October 27, 2025 - 03:53:55
Django version 5.2.7, using settings 'backend.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

WARNING: This is a development server. Do not use it in a production setting. Use a production WSGI or ASGI server inste
ad.
For more information on production servers see: https://docs.djangoproject.com/en/5.2/howto/deployment/
 Usuario registrado: Nicolas
[27/Oct/2025 03:56:09] "POST /api/registro/ HTTP/1.1" 201 83
```

Ilustración 102: Prueba integración (Usuario registrado en el Backend)

```
(venv) PS C:\Users\sandr\Proyecto-Final\backend> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
October 27, 2025 - 04:12:09
Django version 5.2.7, using settings 'backend.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

WARNING: This is a development server. Do not use it in a production setting. Use a production WSGI or ASGI server inste
ad.
For more information on production servers see: https://docs.djangoproject.com/en/5.2/howto/deployment/
Token generado y enviado al usuario Nicolas
[POST] /auth/login/ → 200 OK ✅
[27/Oct/2025 04:12:35] "POST /api/login/ HTTP/1.1" 200 531
[27/Oct/2025 04:12:36] "GET /api/perfil/ HTTP/1.1" 200 126
[27/Oct/2025 04:12:36] "GET /api/categorias/ HTTP/1.1" 200 1266
```

Ilustración 103: Prueba integración (Login con JWT)

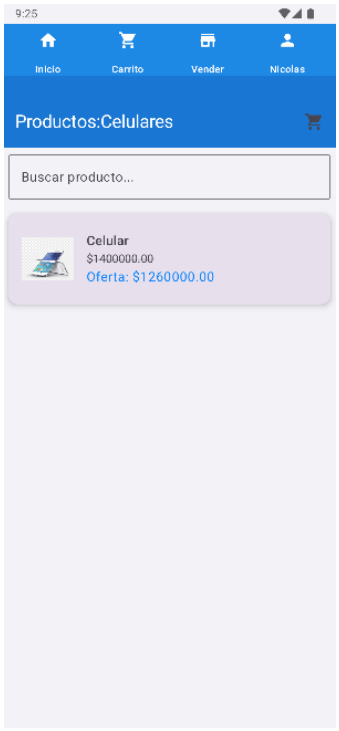


Ilustración 104: Prueba integración (Carga de productos desde API)

```
(venv) PS C:\Users\sandr\Proyecto-Final\backend> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
October 27, 2025 - 04:24:15
Django version 5.2.7, using settings 'backend.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

WARNING: This is a development server. Do not use it in a production setting. Use a production WSGI or ASGI server inste
ad.
For more information on production servers see: https://docs.djangoproject.com/en/5.2/howto/deployment/
Token generado y enviado al usuario Nicolas
[POST] /auth/login/ → 200 OK ✅
[27/Oct/2025 04:24:30] "POST /api/login/ HTTP/1.1" 200 531
[27/Oct/2025 04:24:31] "GET /api/perfil/ HTTP/1.1" 200 126
[27/Oct/2025 04:24:31] "GET /api/categorias/ HTTP/1.1" 200 1266
[27/Oct/2025 04:24:54] "GET /api/productos/subcategoria/1/ HTTP/1.1" 200 562
```

Ilustración 105: Prueba Integración (Carga de productos desde Backend)

Ilustración 106: Prueba integración (Producto agregado al carrito)

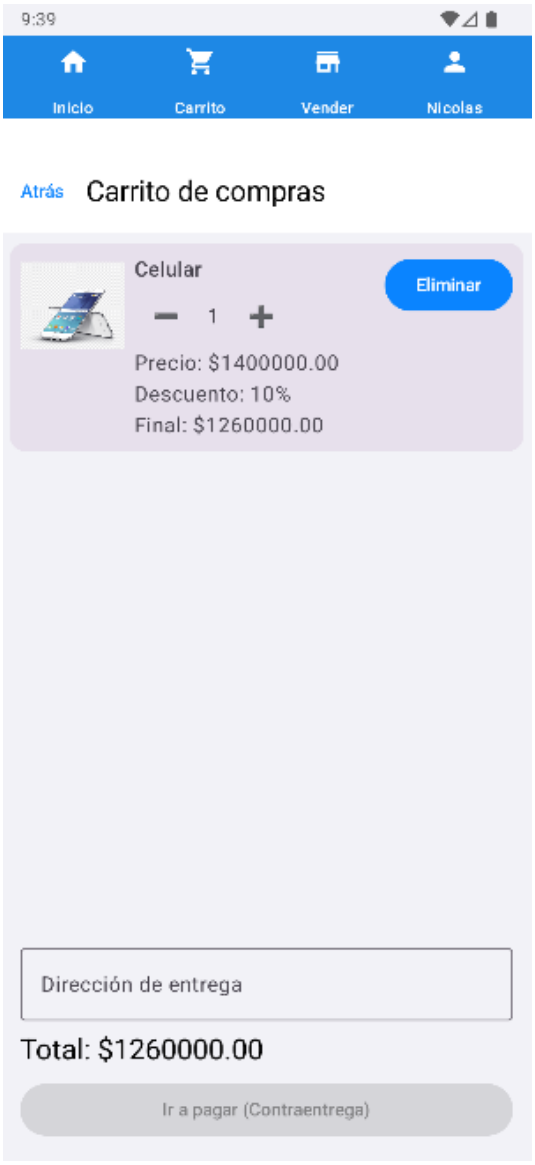


Ilustración 107: Prueba integración (Actualización de cantidad)

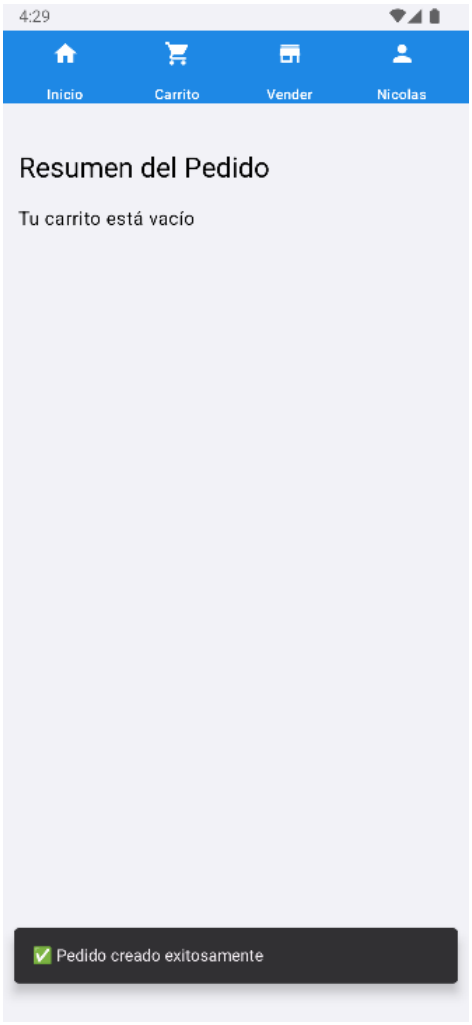


Ilustración 108: Prueba integración (Pedido creado y carrito vacío)

```
Token generado y enviado al usuario Nicolas
[POST] /auth/login/ → 200 OK ✅
[27/Oct/2025 11:28:36] "POST /api/login/ HTTP/1.1" 200 531
[27/Oct/2025 11:28:36] "GET /api/perfil/ HTTP/1.1" 200 126
[27/Oct/2025 11:28:37] "GET /api/categorias/ HTTP/1.1" 200 1266
[27/Oct/2025 11:28:40] "GET /api/productos/subcategoria/20/ HTTP/1.1" 200 532
[27/Oct/2025 11:28:44] "POST /api/carrito/ HTTP/1.1" 201 564
[27/Oct/2025 11:28:46] "GET /api/carrito/ HTTP/1.1" 200 566
[27/Oct/2025 11:29:01] "GET /api/carrito/ HTTP/1.1" 200 566
[27/Oct/2025 11:29:30] "POST /api/pedido/ HTTP/1.1" 201 429
[27/Oct/2025 11:29:30] "GET /api/carrito/ HTTP/1.1" 200 566
[27/Oct/2025 11:29:30] "DELETE /api/carrito/26/ HTTP/1.1" 204 0
[27/Oct/2025 11:29:36] "GET /api/perfil/ HTTP/1.1" 200 126
[27/Oct/2025 11:29:36] "GET /api/categorias/ HTTP/1.1" 200 1266
```

Ilustración 109: Prueba integración (Pedido creado en el Backend)

Pruebas funcionales

Registro completo de usuario vendedor				
descripción	Validar creación de usuario vendedor			
Precondiciones	Datos validos			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PF-001	1.Llenar formulario completo y enviar	Registro exitoso direccione a Login	Funciona correctamente	Exitoso

Tabla 57: Prueba funcional (Registro completo de usuario vendedor)

Error en inicio de sesión invalido				
descripción	Verificar mensaje si falla Login			
Precondiciones	Usuario incorrecto			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PF-002	1.Ingresar credenciales no registradas	Mostrar error 401 Usuario o contraseña incorrectos	Mensaje mostrado	Exitoso

Tabla 58: Prueba funcional (Error en inicio de sesión invalido)

Ver perfil de usuario				
descripción	Mostrar información personal correctamente			
Precondiciones	Usuario logueado			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PF-003	1.Abrir la pantalla de perfil 2. clic en datos personales	Datos reales del usuario	Datos coinciden con el usuario logueado	Exitoso

Tabla 59: Prueba funcional (Ver perfil de usuario)

Gestión de productos				
descripción	Permitir editar un producto ya publicado			
Precondiciones	Ser usuario vendedor			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PF-004	1.Modificar precio y guardar	Cambios reflejados en API y UI	Cambios realizados correctamente	Exitoso

Tabla 60: Prueba funcional (gestión de productos)

Ver ventas del vendedor				
descripción	Visualizar pedidos recibidos			
Precondiciones	Vendedor con ventas			
ID	**Pasos que ejecutar**	**Resultado esperado**	**Resultado obtenido**	**Estado**
PF-005	1.Ir a mis ventas	Mostrar pedidos con totales	Funciona correctamente	Exitoso

Tabla 61: Prueba funcional (Ver ventas del vendedor)

Evidencias pruebas funcionales

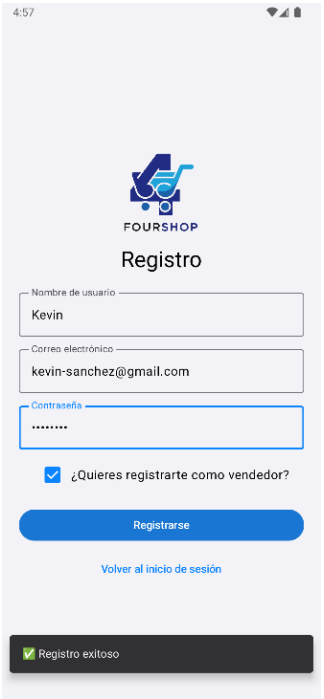


Ilustración 110: Prueba funcional (Registro completo de usuario vendedor)

Ilustración 111: Prueba funcional (Credenciales no registradas)

```
(venv) PS C:\Users\sandr\Proyecto-Final\backend> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
October 27, 2025 - 12:06:07
Django version 5.2.7, using settings 'backend.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

WARNING: This is a development server. Do not use it in a production setting. Use a production WSGI or ASGI server inste
ad.
For more information on production servers see: https://docs.djangoproject.com/en/5.2/howto/deployment/
Unauthorized: /api/login/
[27/Oct/2025 12:06:12] "POST /api/login/ HTTP/1.1" 401 63
```

Ilustración 112: Prueba funcional (Error 401 en el Backend)

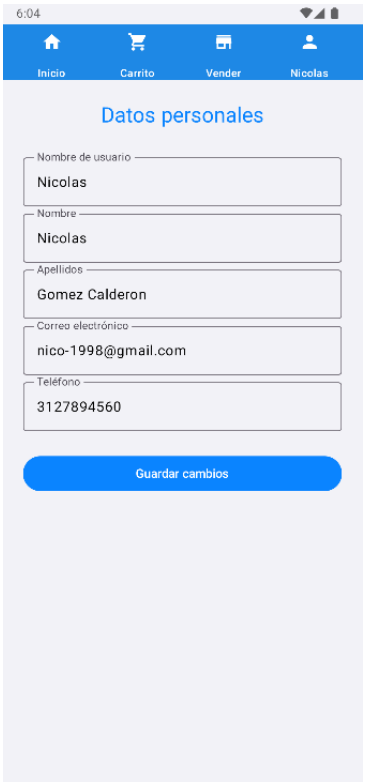


Ilustración 113: Prueba funcional (Ver perfil de usuario)

```
Token generado y enviado al usuario Nicolas
[POST] /auth/login/ → 200 OK ✅
[27/Oct/2025 13:02:43] "POST /api/login/ HTTP/1.1" 200 531
[27/Oct/2025 13:02:43] "GET /api/perfil/ HTTP/1.1" 200 126
[27/Oct/2025 13:02:44] "GET /api/categorias/ HTTP/1.1" 200 1266
[27/Oct/2025 13:02:47] "GET /api/perfil/ HTTP/1.1" 200 126
[27/Oct/2025 13:02:49] "GET /api/perfil/ HTTP/1.1" 200 126
📤 Datos recibidos en PATCH: {'username': 'Nicolas', 'first_name': 'Nicolas', 'last_name': 'Gomez Calderon',
 'email': 'nico-1998@gmail.com', 'telefono': '3127894560', 'password_actual': ''}
✅ Datos actualizados correctamente.
[27/Oct/2025 13:03:38] "PATCH /api/perfil/ HTTP/1.1" 200 155
[27/Oct/2025 13:03:40] "GET /api/perfil/ HTTP/1.1" 200 155
[27/Oct/2025 13:03:43] "GET /api/perfil/ HTTP/1.1" 200 155
```

Ilustración 114: Prueba funcional (Datos de usuario en el Backend)

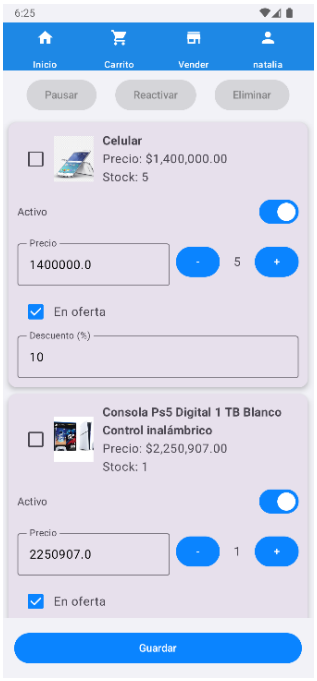


Ilustración 115: Prueba funcional (Precio inicial)

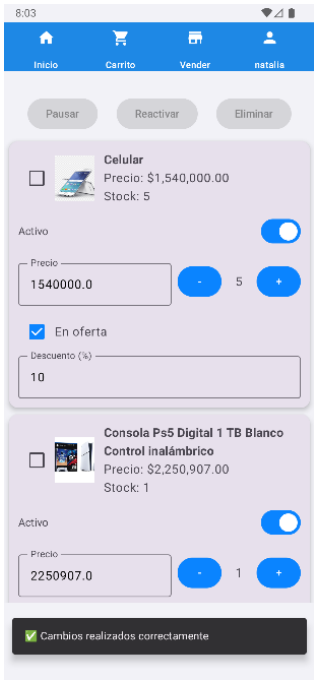


Ilustración 116: Prueba funcional (Precio modificado y guardado)

Ilustración 117: Prueba funcional (Ver ventas del vendedor)

Resumen de pruebas

Tipo	Casos	Exitosos	Fallidos
Pruebas Unitarias	3	3	0
Pruebas Integración	5	5	0
Pruebas Funcionales	5	5	0
TOTAL	13	13	0

Tabla 62: Resumen general de las pruebas

Conclusiones de la ejecución de pruebas

Durante las pruebas realizadas entre vendedores y compradores, se logró evidenciar que la aplicación FourShop cumple satisfactoriamente con los requisitos funcionales establecidos. Los procesos de registro, autenticación, gestión de productos, carrito y pedidos se ejecutaron de manera correcta, realizando la integración con el Backend de manera exitosa y brindando una correcta experiencia de usuario.

En esta fase no solo permitió validar el correcto funcionamiento de la aplicación, sino también comprobar que la etapa de desarrollada cumpliera con lo establecido durante el diseño y análisis de los requisitos, validando que la interacción completa entre los roles del sistema (vendedor y comprador) se llevara a cabo adecuadamente teniendo en cuenta los permisos y accesos de cada rol.

Como resultado, todas las acciones implementadas dentro del sistema durante el proceso de pruebas permitieron asegurar un funcionamiento operativo confiable y dinámico, brindando una experiencia satisfactoria al usuario final.

Conclusión general del desarrollo del proyecto

El desarrollo de la aplicación móvil FourShop represento una gran experiencia integral en la que se logro abordar las etapas del ciclo de vida del software, desde la definición de requisitos, diseño, desarrollo del Backend y el frontend y las pruebas piloto. Este proyecto permitió demostrar como una solución tecnológica puede facilitar a las micro, pequeñas y medianas empresas los procesos de compra y venta de productos en línea, brindando una plataforma de fácil acceso y funcional tanto para vendedores como compradores.

Uno de los aportes mas importantes del proyecto fue la implementación de un entorno digital dinámico, donde los vendedores pueden gestionar sus productos y pedidos, mientras que los compradores cuentan con una experiencia de compras optimizadas a través del carrito de compras intuitivo, una visualización de catálogos dinámica con múltiples opciones de búsqueda, historial de compras y notificaciones por medio de correo electrónico para estar atento al estado de sus pedidos. Todo con el propósito de apoyar a pequeños y medianos vendedores a digitalizar sus servicios, dando un aporte a la modernización del comercio electrónico.

Este proyecto es evidencia de como el trabajo en equipo y la ingeniería de software potencian las capacidades de transformar una idea en una herramienta útil para la sociedad, demostrando que con disciplina, compromiso y aprendizaje continuo es posible generar impactos positivos en la sociedad.

El resultado final es una aplicación sólida, confiable e intuitiva, que aporta beneficios reales a los usuarios al simplificar el proceso de compra y venta de productos. FourShop logra convertirse en una solución que no solo responde a las necesidades de la tecnología actual, sino que también juega un papel importante en el crecimiento económico mediante el impulso al comercio digital.

Conclusiones generales

El desarrollo del proyecto permitió realizar una integración de los objetivos planteados al inicio, logrando así el diseño e implementación de una aplicación móvil funcional orientada principalmente a impulsar la participación de las micro, pequeñas y medianas empresas de Colombia en el comercio electrónico.

Durante todo el proceso, se analizaron de manera rigurosa las necesidades de los usuarios que iban a participar en la aplicación según su rol de preferencia (comprador / vendedor), se diseño una interfaz intuitiva con un enfoque centralizado en la compra y venta de productos y se realizó la construcción de un prototipo totalmente funcional incorporando los mecanismos necesarios como lo son la autenticación, gestión de productos, carrito de compras y registro de pedidos. De igual manera se realizaron pruebas de validación con usuarios para asegurar la usabilidad y efectividad de la aplicación.

A nivel global, este proyecto demuestra que es posible brindar soluciones tecnológicas de fácil acceso, útil y adaptable a todo el contexto comercial de Colombia, logrando así generar un aporte positivo a la digitalización comercial y crecimiento de las MiPymes.

Reflexion del cumplimiento del objetivo general

El objetivo general se cumplió correctamente al lograr la entrega de una plataforma móvil enfocada a la potenciación, respecto a la participación de negocios en el comercio electrónico.

Esta aplicación móvil ofrece:

- Registro y autenticación segura.
- Gestión y administración de productos.
- Un flujo de compra eficiente y dinámico para los clientes.
- Una gestión integrada de ventas y pedidos.

Se tuvo presente que la experiencia de usuario con la aplicación fuera sencilla, rápida y de fácil acceso, favoreciendo tanto a vendedores como compradores en su adaptación a la plataforma sin dificultad.

Análisis de impacto del proyecto

FourShop genera un impacto positivo en diferentes sectores:

1. **Sector económico:** Aumenta las oportunidades de venda para las pequeñas y medianas empresas, sin la intervención de intermediarios costosos.

2. **Sector tecnológico:** Promueve la adopción de herramientas digitales, reduciendo la brecha tecnológica del sector comercial.

3. **Sector social:** Impulsa la visibilidad de los pequeños emprendimientos, contribuyendo a su sostenibilidad y crecimiento a futuro.

4. **Experiencia del usuario:** Permite que compradores puedan acceder a productos de forma más cómoda, rápida y segura sin tener que ir al punto físico.

FourShop logra convertirse en una herramienta de apoyo en la transformación digital del comercio colombiano, contribuyendo a que negocios locales puedan competir en un mercado que día tras día es más exigente y digitalizado.

Trabajos futuros

Aunque se logró el desarrollo de una versión funcional de la aplicación móvil FourShop, existe un conjunto de funcionalidades y mejoras adicionales que pueden implementarse con el fin de fortalecer aún más el sistema y lograr ampliar el alcance en el comercio electrónico en Colombia.

Algunas de las principales líneas de trabajo futuro se encuentran:

1. **Integración de pasarelas de pago en línea:** Actualmente el método principal es Contraentrega. Se plantea la incorporación de servicios como PayU o Nequi para lograr la automatización de los procesos de pago y ofrecerle una mayor comodidad al usuario.

2. **Chat en tiempo real:** Implementar una comunicación directa entre compradores y vendedores para resolver dudas y mejorar la experiencia de compra.

3. **Gestión avanzada de inventario para vendedores:** Implementar alertas de stock bajo, generación de reportes estadísticos y herramientas de apoyo enfocada en decisiones comerciales.

4. **Geolocalización:** Integrar mapas con el fin de estimar costos y tiempos de entrega, con el propósito de facilitar el trabajo en el sector logístico de las MiPymes.

Bibliografía

Arango, A. (2019). *1er Congreso de Seguridad Y Salud en el Trabajo de la Pequeña y Mediana Empresa.* Bogotá: Ministerio del Trabajo. Recuperado el 06 de octubre de 2025, de https://www.mintrabajo.gov.co/prensa/comunicados/2019/septiembre/mipymes-representan-mas-de-90-del-sector-productivo-nacional-y-generan-el-80-del-empleo-en-colombia-ministra-alicia-arango?utm_source

Arístides, G. O. (2018). *Iniciacion a Android en Kotlin. Casos practicos.* España: Ediciones parainfo SA. Recuperado el 02 de octubre de 2025, de https://books.google.com.co/books?hl=es&lr=&id=GVJ1DwAAQBAJ&oi=fnd&pg=PP1&dq=que+es+kotlin&ots=LrKlttJ-t-&sig=YBX2AW9RfedbdV4GPrLC8mnDcQU&redir_esc=y#v=onepage&q=que%20es%20kotlin&f=false

Balado, E. S. (2005). *LA NUEVA ERA DEL COMERCIO: EL COMERCIO ELECTRONICO* (1ra Edicion ed.). Vigo, España: Ideaspropias Edditorial. Recuperado el 30 de Septiembre de 2025, de https://books.google.com.co/books?hl=es&lr=&id=evLz521ZVmAC&oi=fnd&pg=PA1&dq=que+es+comercio+electr%C3%B3nico&ots=ZIEIj8cY7Z&sig=jO-BTnCtid9Tckx5xXQV2fx_Dns&redir_esc=y#v=onepage&q=que%20es%20comercio%20electr%C3%B3nico&f=false

Blokehead, T. (2016). *Scrum - Guia definitiva de practicas agiles escenciales de Scrum.* (B. Incorporated, Ed.) Recuperado el 06 de octubre de 2025, de https://www.google.com.co/books/edition/Scrum_Gu%C3%ADa_definitiva_de_pr%C3%A1cticas_%C3%A1g/T24eDQAAQBAJ?hl=es&gbpv=1&dq=metodologias+de+desarrollo+agil&pg=PT12&printsec=frontcover

Borrero, R. C. (2021). *Proteccion al consumidor electronico en Colombia* (1ra edicion ed.). Villavicencio, Colombia: Ediciones USTA. Recuperado el 02 de octubre de 2025, de

https://www.google.com.co/books/edition/Protecci%C3%B3n_al_consumidor_electr%C3%B3nico_e/i-grEAAAQBAJ?hl=es&gbpv=1&dq=ventajas+y+desventajas+de+los+tipos+de+comercio+electronico+a%C3%B1o+2021+en+espa%C3%B1ol&pg=PT31&printsec=frontcover

Castillo, J. D. (2017). *Desarrollo de aplicaciones android con Android Studio.* Madrid, España: RC Libros. Recuperado el 02 de octubre de 2025, de https://books.google.com.co/books?hl=es&lr=&id=i96LDwAAQBAJ&oi=fnd&pg=PA9&dq=android+studio&ots=kx5mVsSQFw&sig=_04rjGbFFiS8VD1pDOfgXbqutdQ&redir_esc=y#v=onepage&q=android%20studio&f=false

Dámaris, A. C. (2007). *Metodologías Ágiles.* (U. N. Tujillo, Ed.) Trujillo, Perú. Recuperado el 06 de octubre de 2025, de https://d1wqtxts1xzle7.cloudfront.net/53222887/Metodologias_Agiles-libre.pdf?1495404476=&response-content-disposition=inline%3B+filename%3DUniversidad_Nacional_de_Trujillo.pdf&Expires=1759774032&Signature=dAg6j9sBWbp-AaQcPtlK6vnf7FXoLfISNG~m0EWPDSSy7tM8Aj

Kreibich, J. (2010). *Using SQLite.* California, EE.UU: O'Reilly Series. Recuperado el 02 de octubre de 2025, de https://books.google.com.co/books?hl=es&lr=&id=HFIM47wp0X0C&oi=fnd&pg=PR7&dq=qu%C3%A9+es+sqlite&ots=Fj1DiUup4T&sig=Y2Jt4sgyCtHTljkr_8qy-L6zcPw&redir_esc=y#v=onepage&q&f=false

milaulas.com. (octubre de 2019). *apoyoescolar.milaulas.com.* Recuperado el 02 de octubre de 2025, de https://apoyoescolar.milaulas.com/pluginfile.php/675/mod_resource/content/1/Visual%20Studio%20Code.pdf

Semana. (2025). Los retos de la transformacion digital para las empresas colombianas. Recuperado el 06 de octubre de 2025, de https://www.semana.com/mejor-

colombia/articulo/solo-el-7-de-las-iniciativas-digitales-en-colombia-son-exitosas-que-esta-fallando/202501/?utm_source

Trejo, J. M. (2019). *Fundamentos de Negocios Electronicos* (Primera Edicion ed.). Guadalajara, Mexico: Universidad de Guadalajara. Recuperado el 30 de Septiembre de 2025, de https://dca.cucea.udg.mx/sites/default/files/adjuntos/2019_fundamentos_de_negocios_electronicos.pdf

Yoris, A. C. (2024). *Primeros pasos con DJANGO 5.* Recuperado el 02 de octubre de 2025, de https://www.google.com.co/books/edition/Primeros_pasos_con_Django_5/On58EAAAQBAJ?hl=es&gbpv=1&kptab=overview