UNIVERSIDAD PÚBLICA DE EL ALTO

CARRERA INGENIERÍA DE SISTEMAS

PROYECTO DE GRADO

“APLICACIÓN MÓVIL PARA LA COMERCIALIZACIÓN DE PRODUCTOS MEDIANTE SERVICIOS EN LA NUBE”

CASO: EMPRESA IMPORTADORA TELAZA

Para Optar al Título de Licenciatura en Ingeniería de Sistemas

MENCIÓN: INFORMÁTICA Y COMUNICACIONES

Postulante: Univ. Wilmer Cruz Quispe

Tutor Metodológico: Lic. Ing. Dionicio Henry Pacheco Ríos

Tutor Revisor: M. Sc. Lic. Ing. Neil Ramiro Gonzales Burgoa

Tutor Especialista: M. Sc. Lic. Ing. Dulfredo Villca Lázaro

EL ALTO – BOLIVIA

2023

DECLARACIÓN JURADA DE AUTENTICIDAD Y RESPONSABILIDAD

Yo, **WILMER CRUZ QUISPE** estudiante con **C.I. 8402978LP** mediante la presente declaro de manera pública que la propuesta del **TRABAJO DE GRADO** titulada **"APLICACIÓN MÓVIL PARA LA COMERCIALIZACIÓN DE PRODUCTOS MEDIANTE SERVICIOS EN LA NUBE"** es original, siendo resultado de mi trabajo personal y no constituye una copia o replica de trabajos similares elaborados.

Autorizo la publicación del resumen de mi propuesta en internet y me comprometo a responder a todos los cuestionamientos que se desprenden de su lectura.

Asimismo, me hago responsable ante la universidad o terceros, de cualquiera irregularidad o daño que pudiera ocasionar, por el incumplimiento de lo declarado.

De identificarse falsificación, plagio, fraude, o que el **TRABAJO DE GRADO** haya sido publicado anteriormente; asumo las consecuencias y sanciones que de mi acción se deriven, responsabilizándome por todas las cargas legales que se deriven de ello sometiéndome a las normas establecidas y vigentes de la Carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto.

El Alto, diciembre de 2023.

Wilmer Cruz Quispe
C.I. 8402978 LP.
wilmer.ck3@gmail.com

Dedicatoria

Este Trabajo de Grado está dedicado a aquellos que han sido fuentes invaluables de apoyo e inspiración durante este viaje académico.

A mis padres, por su amor incondicional, sacrificios y constante aliento a lo largo de los años. Su apoyo ha sido la fuerza impulsora detrás de cada logro.

A mis hermanos y seres queridos, por su paciencia, comprensión y palabras alentadoras que nos han sostenido en los momentos desafiantes.

Agradecimientos

A Dios, por permitirme llegar a este momento tan especial, por darme salud, fuerza y sabiduría para continuar y alcanzar mis objetivos.

Gracias al Lic. Ing. Dionicio Henry Pacheco Ríos. Tutor metodológico, por las palabras de aliento y guía profesional para la culminación y defensa del Trabajo de Grado.

A el M. Sc. Lic. Ing. Neil Ramiro Gonzales Burgoa. Tutor Revisor, por brindarme el apoyo incondicional, tiempo, dedicación y por la comprensión, para culminar el presente Trabajo de Grado.

A el M. Sc. Lic. Ing. Dulfredo Villca Lázaro. Tutor especialista por el voto de confianza y la orientación específica en la realización del Trabajo de Grado.

A la Carrera Ingeniería de Sistemas de la Universidad Pública de El Alto, Institución de educación superior donde adquirí conocimientos y formación que me impulsan a ser mejor cada día.

Índice General

Índice de Tablas

Índice de Figuras

Resumen

El presente proyecto de grado se enfoca en el desarrollo de una aplicación móvil dedicada a optimizar el proceso de comercialización de productos, gestión de ventas y pedidos. El objetivo principal es ofrecer a los usuarios una plataforma intuitiva y eficiente que simplifique la experiencia del usuario y mejore la administración de pedidos tanto para clientes como para el personal de ventas a través de un catálogo digital que permita a los usuarios explorar de manera fácil y atractiva los productos disponibles, con herramientas para la administración eficiente de la base de clientes, incluyendo perfiles, historial de compras y con las medidas de seguridad robustas para proteger la información del cliente y garantizar el acceso autorizado.

Para el desarrollo del proyecto se utilizó la metodología Mobile-D, que propone un modelo de proceso ágil, permitiendo adaptaciones continúas basadas en retroalimentación.

Para la conclusión del desarrollo de la aplicación Android se utilizó como herramienta primordial el lenguaje de programación Dart con la Framework de Flutter, con el gestor de base de datos PostgreSQL y con la ayuda principal del servidor desarrollado en NodeJS, con servicios de Google y Firebase para la función correcta de la aplicación.

Palabras Claves: Node.js, PostgreSQL, Flutter y Dart, Autenticación, Desarrollo Ágil, la Nube.

Abstract

The present graduation project focuses on the development of a mobile application dedicated to optimizing the process of product marketing, sales management, and orders. The main objective is to provide users with an intuitive and efficient platform that simplifies the user experience and improves order management for both customers and sales personnel through a digital catalog that allows users to explore products easily and attractively. It includes tools for efficient customer base management, including profiles, purchase history, and robust security measures to protect customer information and ensure authorized access.

For the project development, the Mobile-D methodology was employed, which proposes an agile process model, allowing continuous adaptations based on feedback.

For the completion of the Android application development, the primary tools used were the Dart programming language with the Flutter framework, the PostgreSQL database manager, and the main server developed in NodeJS, with Google and Firebase services for the proper functioning of the application.

Keywords: Node.js, PostgreSQL, Flutter and Dart, Authentication, Agile Development, Cloud.

CAPITULO I

MARCO PRELIMINAR

1.1. Introducción

Las nuevas Tecnologías de la Información y Comunicación han favorecido la aparición de nuevos canales de venta, el comercio electrónico ha revolucionado el mercado digital para las empresas como para los consumidores y se ha convertido en una de las principales actividades de la economía mundial debido a la globalización en la Red, que ha permitido la apertura de negocios en todo el mundo durante las 24 horas del día aumentando las posibilidades de éxito de todo negocio, influyendo en la manera como los bienes y servicios son comercializados, reduciendo costos y mejorando la productividad. El Internet ha llegado a ser algo indispensable en los hogares es así por medio de dispositivos electrónicos como tabletas, smartphones, computadoras, entre otros medios digitales pueden acceder a varios servicios como es redes sociales, contenido multimedia, aplicaciones y entre otros. El crecimiento del comercio electrónico en los distintos países no solo depende de su infraestructura tecnológica sino también de la velocidad en que se eliminen las barreras en los consumidores.

Con relación a las compras realizadas vía Internet en Bolivia, en primer lugar, se encuentra comida y bebida, seguido de suscripciones, ropa, tecnología, mensualidades, entre otros. Las formas en las que realizaron los pagos fueron 46% por transferencia bancaria, 31% pago en físico al momento de realizarse la entrega, 22% por pasarelas de pago (Sánchez, 2021). Aunque la industria del comercio electrónico en el país todavía es pequeña, las razones pueden ser diversas desde la falta de un adecuado sistema de pago, la desconfianza que genera por la informalidad, como las principales limitaciones que obstruyen su desarrollo, a pesar de todo esto el comercio electrónico va ir ganando fuerza debido al crecimiento tecnológico, y la demanda de este servicio será necesario.

Es por esta razón, que los sistemas de información, forman parte esencial de las instituciones empresariales del país, encontrándose en pleno desarrollo y crecimiento, sobre todo en el sector del comercio, donde se ven impulsadas a tener o desarrollar su propia aplicación, para sacar ventajas del servicio electrónico y así llegar

a toda la población y agilizar los procesos. Las herramientas móviles también se convierten en un canal de comunicación con los clientes en cualquier momento, y de forma instantánea, podrán acceder a toda la información con tan solo poseer un dispositivo móvil, el proyecto se enfoca en desarrollar una aplicación móvil Android con un diseño UX/UI, donde se utilizará la metodología ágil Mobile-D, debido a las condiciones y requerimientos planteados por la empresa Importadora y Distribuidora Telaza, Por lo detallado la empresa al ver las ventajas que tiene una aplicación móvil basada en la Framework de Flutter, decide implementar una aplicación móvil para mejorar la administración de la comercialización de los productos, tomando en cuenta la necesidad de reducir pérdidas de información en la gestión administrativa del comercio y disminuir tiempo de atención al cliente.

1.2. Antecedentes de la Investigación

1.2.1. Antecedentes Institucionales

La empresa textil Telaza es una institución u organización social con actores bolivianos, creada con la finalidad de gestionar, desarrollar e impulsar los productos en demanda para la importación y distribución de los mismos, para abastecer a los consumidores a nivel nacional.

De esta manera aumentar la productividad para ser más competitivos y ofrecer precios excelentes en el mercado sin perder la calidad del producto, buscando formas de mejorar y ofrecer soluciones creativas a los contratiempos, demostrando nuevas tendencias en las expectativas de los clientes, para responder a las opiniones, problemas, mensajes y quejas, tomando en cuenta todas las necesidades para mejorar el servicio al cliente. Actualmente distribuye sus productos en la Ciudad de La Paz, Cochabamba, Santa Cruz y la Ciudad de El Alto.

Por lo tanto, la empresa Importadora y Distribuidora Telaza, tiene como:

Misión: Comercializar productos de buena calidad y un excelente precio, importando productos de marcas reconocidas, contando con el personal capacitado y calificado, ofreciendo un buen servicio con el fin de lograr la satisfacción de los clientes.

Visión: Ser la empresa líder en el crecimiento de comercialización e importación de los productos textiles, buscando la competitividad y posicionamiento en el mercado local y generar la rentabilidad que permita expandir la compañía por toda Bolivia.

Objetivo: Lograr un crecimiento continuo y sustentable en la importación y comercialización de los productos textiles, ofreciendo excelentes precios a los clientes que compran continuamente, mostrando nuevas tendencias en los productos.

Además, la empresa textil Telaza en el área de comercialización cuenta con el encargado de atención al cliente en la agencia ubicada en la 16 de julio de la Ciudad de El Alto, pero este es solo el encargado de las ventas y realiza la comercialización de los productos sin tener una información detallada de cada producto y el registro de los mismos se lo realiza de manera manual, pues no existe un control organizado, adecuado de cada producto.

TELAZA tiene a la venta los siguientes productos:

- Cuero vuelto
- Cuero vuelto licra
- Corderoy rígido (mil rayas)
- Corderoy licra (japones)
- Kaki (dril)
- Kaki liso
- Kaki galleta
- Piel de foca
- Piel de oso
- Tela de Peluche

- Piel de vaca
- Cuero ecológico
- Cuerina coreana
- Cuero arrugado
- Cuerina sobaco de elefante
- Paño planchado
- Paño bebe
- Paño liso
- Paño cuadrado
- French Terry
- Chiporro corderito
- Felpa
- Lana bucle

Dentro de la institución se destaca los siguientes problemas respecto a la comercialización de los productos:

Dificultad en el control y gestión de la comercialización de los productos, en donde el repartidor no tiene el detalle de la orden de la compra. El pedido de los productos eventualmente se recibe vía teléfono, la administración y control de la comercialización de los productos de la empresa es rudimentaria. La preparación de pedidos para el consumidor, donde usualmente los operarios elaboran múltiples pedidos de cada tipo de productos, casualmente no se cuenta con un registro relevante de la orden del pedido.

Además, el encargado de ventas, necesita realizar los registros de manera ordenada de las actividades de la comercialización, donde se tiene un informe de manera manual para verificar las ventas y son propensas a una pérdida de información. Cuentan con un registro de los productos y clientes de forma manual anotados en hojas de papel.

En síntesis, en las condiciones actuales existen dificultades en la elaboración de informes de la comercialización de los productos.

Por lo antes mencionado debe aprovechar al máximo las nuevas soluciones tecnológicas para brindar toda la información de los productos para el consumidor.

En la figura 1 se muestra el organigrama de la empresa, con cada una de sus unidades dependientes, ahora bien, las unidades donde la aplicación Android tomara el correspondiente funcionamiento son las siguientes:
El departamento de ventas, compras y almacenes.

Figura 1

Organigrama Institucional

Nota. Direccion General de la Empresa TELAZA.

1.2.2. Antecedentes Internacionales

- Análisis, Diseño, Desarrollo e Implementación de un Sistema Integral Web-Móvil para el Control de Inventarios de Equipos Electrónicos de la Universidad Internacional SEK (Lemus, 2015). El sistema desarrollado contribuye al uso eficiente y eficaz de los recursos, además permite que el

Departamento de Recursos Tecnológicos de los campus Guápulo y Carcelén logra llevar un registro histórico básico de los movimientos de los equipos lo que permite tomar decisiones sobre ellos y presentar informes oportunos. Utilizando la metodología de desarrollo en cascada, con el lenguaje de programación C# y la base de datos en SQL-Server. Universidad Internacional SEK Ecuador.

- Desarrollo de una Aplicación Móvil híbrida e-commerce para la gestión de ventas de la empresa Calzado Anabel (Chicaiza, 2020). El proyecto esta basado en solventar en parte la gestion de ventas, existencias y costos de los productos, la aplicación se enfoca en promocionar los productos de la empresa para que los clientes mayoristas y los consumidores finales puedan adquirirlos a través de esta, desarrollado con el Framework de Ionic. A través de la metodología Extreme Programming para llevar tiempos cortos de entrega continua del producto y la satisfacción del cliente. Además, con el servicio de Backend como lo es Firebase. Universidad Técnica de Ambato Ecuador.

- Diseño e Implementación de una Aplicación Móvil para la Mejora del Proceso de Venta de Líneas Prepago de una Empresa de Telecomunicaciones, (Gaona, 2020). El proyecto de grado propone la implementación de mejorar el proceso de venta, mediante la App donde se tiene una reducción significativa en los tiempos de atención al cliente. Para lograr esto, se ha diseñado una arquitectura que permita utilizar los mismos servicios web de la empresa, donde se ha optado por utilizar el marco de trabajo SCRUM. Universidad Tecnológica del Perú.

1.2.3. Antecedentes Nacionales

- Aplicación Móvil de Control de Ventas e Inventarios con Alertas Tempranas caso: Empresa Importadora y Distribuidora de Alimentos e Insumos para mascotas San Gabriel Pet, (Villca, 2018). El Proyecto de Grado tiene como

propósito modelar, diseñar y desarrollar una aplicación móvil, donde tiene el alcance en función del tiempo disponible y necesario para implementar todas las funcionalidades, utilizando la metodología Mobile-D, se ha utilizado Android Studio como herramienta de desarrollo de la aplicación móvil, se creó una base de datos SQLite para el almacenamiento de la información de manera interna, Universidad Mayor de San Andrés.

- Aplicación para Dispositivos Sistema de Información Geográfico basado en Plataforma Web y Móvil para el apoyo a Empresas de distribución productos (Mamani, 2018). El proyecto de grado tiene como objetivo desarrollar un sistema de información geográfico en plataforma web y móvil orientado al apoyo a empresas que distribuyen productos, permitiéndoles monitorear la ubicación de los vehículos y obtener la ruta óptima para la distribución de productos. Universidad Católica Boliviana “San Pablo”.

- Aplicación para Pedidos de una Importadora de Vidrios, Utilizando Tecnología en Dispositivos Móviles (Navia, 2014). El propósito del proyecto es satisfacer la demanda de vidrios para edificios en construcción, mediante el manejo de pedidos que tiene la empresa almacenando toda la información de manera centralizada, permitiendo el acceso y disponibilidad de su información, notificando a los clientes del estado de pedidos, promocionando productos para una mejorar atención al cliente, con la metodología de desarrollo SCRUM. Universidad Mayor de San Simón.

1.2.4. Antecedentes Locales

- Aplicación Web con Android para la Creación de una Red Social basada en la Localización de Socios de la Empresa de transportes Templarios (Pinela, 2018). El proyecto hace énfasis al control del personal y las rutas del transporte con una comunicación en tiempo real con la administración de la empresa, con la metodología de desarrollo UWE utilizando las herramientas

o lenguajes de programación, PHP, Java, JavaScript, con manejo de la base de datos el entorno MySQL. Universidad Pública de El Alto.

- Sistema Web de Compra, Venta e Inventarios de Medicamentos y Servicios Complementarios (Nacho, 2020). El proyecto pretende mejorar la administración de bienes y servicios del “Hospital Capitán juan Uriona”. En el desarrollo del sistema se utilizaron la metodología ágil AUP y la metodología web UWE, y para la implementación del backend, en la base de datos MariaDB, con el lenguaje de programación PHP, con su Framework Laravel, con VueJS, y con el modelo Cocomo II, para hallar el costo del sistema. Universidad Pública de El Alto.

- Aplicación Móvil de Control de Inventarios mediante Código QR (Patty, 2020). El propósito del proyecto, es gestionar el control que se realiza cada vez que ingresa un activo nuevo, cuando se hace algún préstamo o cuando se realiza auditoria, la aplicación móvil le dará un control exacto del inventariado, mediante el escaneo del código QR que es asignado a cada activo, utilizando la Metodología de desarrollo XP, con las herramientas y lenguajes de programación utilizadas como, Android Studio, ADT Plugin, Web Services, con una base de datos en MariaDB. Universidad Pública de El Alto.

1.3. Planteamiento del Problema

El control sobre las ventas y distribuciones consiste principalmente en la información que se agrupa en una lista de actividades comerciales, esta información se registra manualmente, lo que en muchos casos conlleva a la perdida de información, y en consecuencia pérdidas económicas y como resultado reduce el nivel competitivo de la empresa frente a las demás empresas. Por otra parte, en la distribución de productos eventualmente se recibe vía teléfono, donde se anota la información de la comercialización del producto de manera rudimentaria y en

ocasiones existe confusión en el detalle de la orden del pedido y el control de los mismos.

En la Ciudad de La Paz, El Alto la empresa Importadora y Distribuidora Telaza, que da servicio de comercialización de productos textiles, y la distribución de los mismos, debido al interés de seguir expandiéndose y ganando mercado. Es por esta razón que se ha visto necesario la implementación de una aplicación móvil para realizar el control de las actividades comerciales. A través de esta implementación los clientes podrán ver una información detalla de cada producto, catálogos y si es el caso adquirirlos.

1.3.1. Problema Principal

La gestión de la información en la comercialización de la empresa Importadora y Distribuidora Telaza se realiza de manera manual lo cual dificulta los procesos de ventas y pedidos, lo que impacta negativamente en las condiciones actuales dificultando el desarrollo eficiente de las actividades comerciales.

1.3.2. Problemas Secundarios

Entre los problemas secundarios que se tiene en la empresa textil Telaza se pueden citar los siguientes:

- El encargado del área de la comercialización tiene apuntado de manera manual la lista de los productos destacados, promociones o en liquidación, donde la información provista a los clientes respecto a los productos es escasa, limitando su adquisición.
- La compra de los productos vía telefónica, casualmente es anotado toda su información de manera rudimentaria, por lo cual no se brinda alguna información relevante para el personal encargado del área de distribución.
- Los repartidores eventualmente, no tienen asignado el detalle de la orden y las rutas por donde realizar la entrega de los productos, por ello desconoce el área de distribución.

- Existen dificultades para monitorear las listas de distribución de productos, lo que resulta en que los consumidores no sean atendidos según lo programado y que ocasionalmente quedan perjudicados.
- El respaldo de los recibos, historial de las ventas se realiza manualmente de la comercialización de los productos, ocasionando pérdidas de información en las notas relacionadas con la comercialización de los productos.

1.3.3. Formulación del Problema

¿De qué forma se podrá mejorar la gestión en la información, comercialización y la distribución de los productos textiles de la empresa Importadora y Distribuidora Telaza?

1.4. Objetivos

1.4.1. Objetivo General

Desarrollar una aplicación móvil que permita mejorar la gestión de la información sobre la comercialización y distribución de productos y coadyuve a desarrollar eficientemente las actividades comerciales de la empresa Importadora y Distribuidora “Telaza”.

1.4.2. Objetivos Específicos

- Diseñar una interfaz enfocada en brindar una mejor experiencia a los usuarios y de fácil manejo, mediante la Framework de Flutter para una mejor administración y organización de la información de los productos.
- Centralizar la información de los productos y los clientes en una base de datos, garantizando el acceso autorizado, para una mejor administración de información en la empresa.
- Implementar un proceso de autenticación seguro que cumpla con los estándares de seguridad recomendados, garantizando que los datos del usuario estén protegidos contra amenazas como ataques de suplantación de identidad y fugas de información.

- Elaborar un módulo donde se pueda visualizar la descripción, precio de cada producto, para ofrecer un catálogo de productos actualizados.
- Crear módulos de acceso rápido a la información actualizada donde se detalle la orden de la compra del producto, agilizando el proceso de los mismos.
- Desarrollar un módulo para el repartidor, para notificar sobre los pedidos, donde tendrá la información del producto a entregar en tiempo real.
- Mejorar la experiencia del repartidor al proporcionar rutas eficientes y precisas en la aplicación móvil utilizando servicios de Google, garantizando la visualización rápida de instrucciones de navegación claras y oportunas.
- Generar datos de información sobre el historial de ventas según la fecha, para proporcionar la información del producto comprado por el cliente.

1.5. Justificación

1.5.1. Justificación Técnica

El desarrollo de aplicaciones móviles va cobrando mayor importancia en la actualidad debido a los costos accesibles de smartphones de media y alta gama que pueden reemplazar en muchos aspectos a los ordenadores de escritorio o portables. El ingreso de aplicaciones móviles al sector empresarial es indudablemente un espacio que muchas compañías de desarrollo de software encuentran una demanda.

Lo que plantea este trabajo es de diseñar una aplicación móvil, que brinde una mejor experiencia al usuario, con un manejo fácil, rápido y eficiente de los datos de la comercialización de los productos en la aplicación y de este modo poder optimizar la calidad de administración de los productos, ya que el principal propósito es de satisfacer los requerimientos de la empresa de manera más rápida gracias al diseño de la aplicación móvil. En el desarrollo de aplicación móvil, se hará el uso de las nuevas tecnologías que están en el mercado y el masivo uso que se hace de ellos, además con el auge de las aplicaciones móviles, es por ello que el proyecto se lo desarrollará principalmente en Android.

1.5.2. Justificación Económica

Con la aplicación móvil, la empresa Importadora y Distribuidora Telaza, tendrá una ayuda para mejorar el tiempo de acceso a la información de los productos, evitando la tardanza al momento de ofrecer los productos a los consumidores, para ofrecer una mejor atención al cliente, minimizando costos y generar más flujo en la comercialización de los productos. Al mejorar las actividades del comercio de la empresa, eventualmente no será necesario el material físico, un fin de papeles, notas y fichas de información, permitiendo ahorrar estos y otros recursos. Con la aplicación móvil tendrá un aporte en la expansión de la comercialización de los productos.

1.5.3. Justificación Social

El presente proyecto es desarrollado con la finalidad de facilitar las actividades del comercio de la empresa, por medio de la implementación de la aplicación móvil los clientes tendrán acceso a los productos, ver precios, existencias, realizar pedidos en tiempo real llenando formularios y así mantener a los clientes satisfechos y buscar la lealtad hacia la empresa a través de la App, el tiempo y la distancia donde se encuentren los clientes ya no es un obstáculo, de esta manera el servicio se puede dirigir a un grupo de personas masivamente, ya que abre una serie de oportunidades y de retos primordiales para la sociedad, la desaparición de barreras entre comprador y vendedor.

1.6. Metodología

El desarrollo de software es una de las áreas de investigación más influyentes en la actualidad, dado que constantemente surgen nuevas tendencias que buscan beneficiar a la producción de software con calidad. Realizando un conjunto de enfoques, prácticas, procedimientos y reglas organizadas que se utilizan para guiar el proceso de desarrollo de software. Estas metodologías proporcionan un marco de trabajo para planificar, diseñar, construir, probar y mantener aplicaciones de software de manera eficiente y efectiva. Las metodologías de desarrollo son responsables de lograr dicho objetivo por ende se observan cambios evidentes en su línea de tiempo.

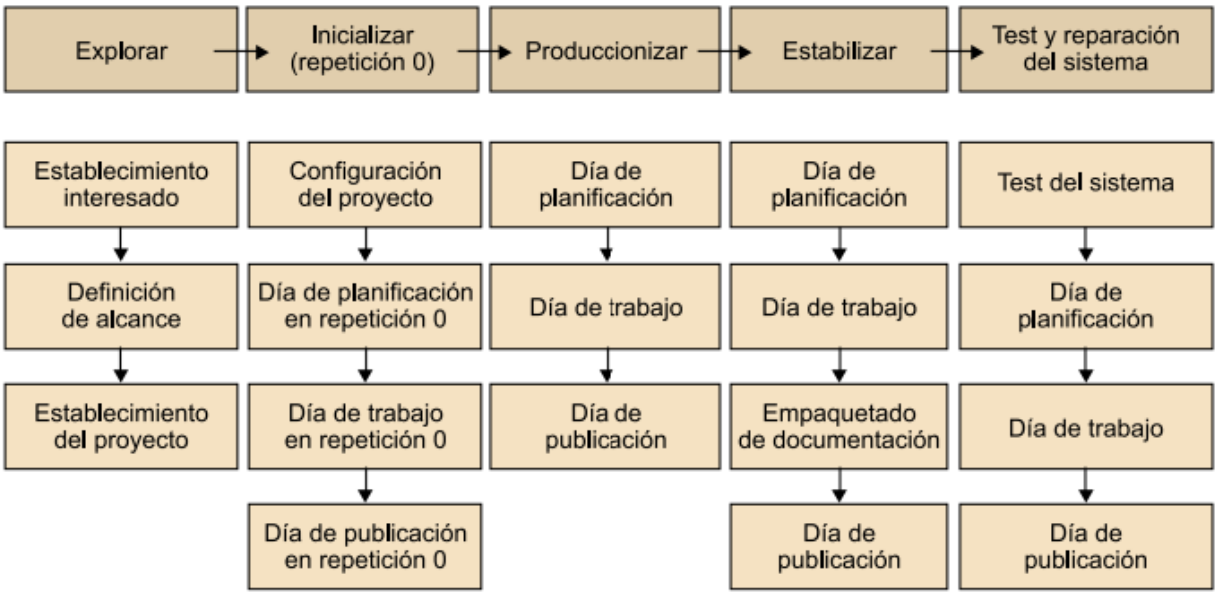
1.6.1. Mobile – D

El objetivo de esta metodología es conseguir ciclos de desarrollo muy rápidos en equipos muy pequeños, ya que su diseño consta de otras metodologías existentes como XP, RUP. Es una metodología ágil de desarrollo de aplicaciones móviles, basado en pruebas que es una de las mejores formas de asegurar calidad, tiene un enfoque centrado en la satisfacción del usuario final, fue creado con el objeto de ser una metodología de rápidos resultados, enfocada en grupos de trabajos pequeños, los cuales debería poseer confianza entre sus miembros, y un nivel de habilidad similar, además busca entregar resultados funcionales en periodos cortos de tiempo.

En la figura 2 se muestra las cinco fases de la metodología ágil Mobile-D.

Figura 2

Ciclo de desarrollo de Mobile-D



Nota. Metodología de desarrollo Mobile-D. Modificada de Metodología de desarrollo ágil para sistemas móviles. Introducción al desarrollo con Android y el iPhone (Blanco et al., 2009).

Las principales ventajas de esta metodología para el desarrollo de la aplicación móvil son:

- Mejora la calidad en la experiencia y las funcionalidades para el cliente.
- Acorta los ciclos de producción y minimiza los tiempos de reacción y toma de decisiones.
- Aumenta la productividad, al asignar mejor los recursos, y de forma más dinámica, mejora la producción según las prioridades que tenga la empresa.

Por lo cual se decide implementar la metodología de desarrollo ágil Mobile-D.

1.6.2. Métricas de Calidad

Las métricas de calidad de software son medidas cuantitativas o cualitativas que se utilizan para evaluar diversos aspectos de la calidad en el desarrollo y mantenimiento de software. Estas métricas proporcionan información objetiva sobre el rendimiento, la confiabilidad y la eficacia de un producto de software.

1.6.3. Costos

Los costos asociados con el desarrollo de software pueden variar significativamente según diversos factores, como la complejidad del proyecto, el tamaño del equipo, la tecnología utilizada y la ubicación geográfica.

1.6.4. Seguridad

La seguridad de software es un componente crítico en el desarrollo, implementación y mantenimiento de sistemas y aplicaciones informáticas. Implica la implementación de prácticas, controles y medidas para proteger el software contra amenazas y vulnerabilidades que podrían comprometer la confidencialidad, integridad y disponibilidad de los datos y sistemas.

Protección Contra Ataques Comunes: Implementar medidas para protegerse contra ataques comunes, como ataques de fuerza bruta, inyecciones SQL, Cross-site scripting (XSS) y otros.

1.6.5. Pruebas de Software

Las pruebas de software son un componente crucial en el desarrollo de aplicaciones y sistemas para garantizar su calidad, fiabilidad y rendimiento. Existen varios tipos de pruebas que se realizan a lo largo del ciclo de vida del desarrollo de software.

1.7. Herramientas

Para la elaboración y desarrollo del proyecto se utilizará las siguientes herramientas y lenguajes de programación, para realizar un seguimiento de las actividades que ayudará en el desarrollo de la aplicación.

Android Studio: Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de apps para Android y está basado en IntelliJ IDEA. Android Studio ofrece incluso más funciones que aumentan su productividad para desarrollar apps para Android, como las siguientes:

- Un sistema de compilación flexible basado en Gradle.
- Un emulador rápido y cargado de funciones.
- Un entorno unificado donde se podrá desarrollar para todos los dispositivos Android.
- Variedad de marcos de trabajo y herramientas de prueba.

Visual Studio Code: Es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS, y que tiene una terminal integrada, autocompletado, biblioteca de extensiones lo que da una flexibilidad al crecimiento del proyecto, se utilizará para programar la parte de NodeJS en la parte del Backend.

Flutter: Es un Framework para desarrollar aplicaciones para diferentes plataformas elaborado por Google. Este kit de desarrollo ofrece un gran número de bibliotecas para elementos estándar de la interfaz de usuario de Android, se desarrollará con Flutter porque es una herramienta que permite ahorrar tiempo y recursos y agiliza el desarrollo de una aplicación con un rendimiento nativo.

1.7.1. Lenguajes de Programación

Dart: Es un lenguaje de programación open source, relativamente nuevo. Se utilizará por que facilita la creación de animaciones y transiciones, tiene una programación estructura, flexible y con un lenguaje orientado a objetos. Además, permite reutilizar código obteniendo un resultado más organizado.

NodeJS: Es un entorno de tiempo de ejecución de JavaScript. Utiliza un modelo de entrada y salida sin bloqueo controlado por eventos, de esta manera lo hace un entorno ligero y eficiente, con el cual se trabajará para desarrollar la API de las aplicaciones en el lado del servidor.

1.7.2. Base de Datos

PostgreSQL: Es un sistema de base de datos relacional, que archiva datos en tablas separadas, lo que permite tener mayor velocidad y flexibilidad, las tablas están relacionada de formas definidas que va permitir realizar las operaciones en periodos cortos.

Firebase: Es una plataforma en la nube para el desarrollo de aplicaciones web y móvil. Está disponible para distintas plataformas (iOS, Android y web), con lo que es más rápido trabajar en el desarrollo. Se utilizará integración dinámica de los usuarios usando Firebase Authentication y enviar notificaciones a varias plataformas con Cloud Messaging, facilitando su uso. Es especialmente interesante para el desarrollo sin necesidad de invertir tanto tiempo al Backend, tanto en cuestiones de desarrollo como de mantenimiento.

1.7.3. Servidores

Se creará un servidor en NodeJS, porque permite escribir código Backend, aunque tradicionalmente se usaba en el navegador para escribir código Frontend. Al tener el Frontend y el Backend juntos de esta forma, se reduce el esfuerzo necesario para crear un servidor, que es el motivo principal por el que NodeJS es una opción popular para escribir código Backend. También se realiza el almacenamiento en las plataformas de la nube de Firebase Storage para las imágenes, donde ofrece un servidor para alojar las apps de manera rápida y sencilla**,** esto es un hosting estático y

seguro. Proporciona certificados de seguridad SSL y HTTP2 de forma automática y gratuita para cada dominio, reafirmando la seguridad en la navegación. Funciona situándolas en el CDN (Content Delivery Network) de Firebase, una red que recibe los archivos subidos y permite entregar el contenido.

1.8. Límites y Alcances

1.8.1. Límites

- Los dispositivos móviles para ejecutar la aplicación deben de contar al menos con la versión 5.0 Android Lollipop, o versiones superiores.
- La aplicación móvil requiere de una conexión a Internet para poder almacenar la información.
- Se optimizará el almacenamiento de las imágenes reduciendo los pixeles al momento de subir a la nube.
- Los usuarios no podrán acceder a la información actualizada de los nuevos productos, si no tienen conexión a Internet.

1.8.2. Alcances

La aplicación móvil está proyectada para el uso exclusivo de la empresa, y los clientes.

Como alcances son los siguientes:

- Con esta aplicación se desea dotar de una innovada forma de comunicación e información de los productos de la empresa.
- Se pretende con esta aplicación facilitar el crecimiento de la empresa, poniendo a disposición todos los productos disponibles en la App.
- Cuenta con un módulo para mostrar el repartidor asignado en el detalle de la orden.
- En el menú opción deslizable de cada cliente habrá un listado de estados de sus pedidos.
- Se incluirá una opción de para llamar al cliente para realizar la entrega.
- Se podrá actualizar el estado de la orden del pedido a entregado.

- Para mayor facilidad y encontrar un producto en específico, habrá una opción de filtrado o búsqueda de los mismos.
- El presente proyecto contribuirá a optimizar el proceso de distribución de los productos al consumidor.

1.9. Aportes

1.9.1. Aportes Académicos

Gracias a los conocimientos, habilidades y destrezas adquiridas durante el ciclo formativo de la carrera de Ingeniería de Sistemas, permite el desarrollo de una solución tecnológica factible de implementar con las herramientas de desarrollo y diseño de aplicaciones móviles, dando soluciones alternativas a sus problemas que presenta la empresa. Además de realizar una exploración que ayuda a cumplir con los requerimientos de la empresa Importadora y Distribuidora Telaza para resolver con todas las necesidades que proponen sus problemas. El desarrollo de la aplicación permitirá que todos los clientes de la empresa estén en constante actualización e información de todos los productos disponibles de la empresa. La empresa contará con esta aplicación móvil para la administración de sus actividades del comercio.

1.9.2. Aportes Institucionales

En la actualidad existen diferentes sistemas de seguimiento y gestión de las actividades del comercio, que realizan tareas que anteriormente tenían que hacerse manualmente y consumían más tiempo, el aporte más relevante del estudio es brindar una solución tecnología que contribuye a mejorar la gestión de información de la comercialización de productos que contara la empresa Importadora y Distribuidora Telaza, para su desarrollo tecnológico disfrutando así de operaciones sistematizadas y optimizadas. Gracias a ello, la empresa puede disfrutar de operaciones con mejores diseños de UI/UX, y servicios en la nube. Así misma mejora el proceso de toma de decisiones, lo que es vital para la institución que esperan perdurar en el tiempo y ser estables.

CAPÍTULO II

MARCO TEORICO

2.1. Introducción

En el presente capitulo que se desarrolla a continuación se describen todas las definiciones, términos y conceptos, así como también la metodología y herramientas a utilizar dentro del desarrollo del proyecto para conocer de cada uno de ellos.

2.2. Aplicación

Una aplicación o App es un programa informático diseñado para realizar tareas específicas o proporcionar servicios específicos según las necesidades de los usuarios. Duffy (2012) define una aplicación como un pequeño programa informático diseñado para funcionar en dispositivos móviles, como teléfonos inteligentes o tabletas. Estas aplicaciones pueden proporcionar una amplia gama de funciones y servicios, desde juegos y entretenimiento hasta productividad y utilidades. En esencia, las aplicaciones son software diseñado específicamente para ejecutarse en dispositivos móviles y ofrecer diversas funcionalidades para el usuario. Pueden ser descargadas e instaladas en el dispositivo a través de tiendas de aplicaciones como la App Store de Apple o Google Play Store, lo que permite a los usuarios personalizar y ampliar las capacidades de sus dispositivos móviles según sus necesidades y preferencias.

De la misma forma Carrasco (2020) describe las aplicaciones como programas que se utilizan en línea para llevar a cabo funciones específicas o proporcionar soluciones específicas a los usuarios en el entorno digital. Estas aplicaciones pueden abarcar una amplia variedad de propósitos, desde redes sociales y comunicación hasta productividad y entretenimiento. Por otro lado, Yoskovitz y Croll (2014) definen una aplicación como una entidad digital que genera datos sobre el comportamiento de los usuarios y el rendimiento de una startup o producto. Destacarían la importancia de medir y analizar estas aplicaciones para tomar decisiones informadas, una de las formas de conseguir una aplicación es por medio de las tiendas de aplicaciones.

2.3. Móvil

Es un dispositivo electrónico portátil que se utiliza principalmente para la comunicación y el acceso a la información a través de redes inalámbricas, como redes celulares. Según Ruelas (2014) el teléfono móvil es un dispositivo inalámbrico electrónico basado en la tecnología de ondas de radio, estos dispositivos utilizan redes celulares para conectarse a la red y proporcionar acceso a servicios como Internet, redes sociales, aplicaciones móviles y más. Del mismo modo el teléfono móvil es un dispositivo inalámbrico electrónico que permite tener acceso a la red de telefonía celular o móvil. Se denomina celular debido a las antenas repetidoras que conforman la red, cada una de las cuales es una célula, si bien existen redes telefónicas móviles satelitales. Su principal característica es su portabilidad, que permite comunicarse desde casi cualquier lugar. Aunque su principal función es la comunicación de voz, como el teléfono convencional, su rápido desarrollo ha incorporado otras funciones como son cámara fotográfica, agenda, acceso a Internet, reproducción de vídeo e incluso GPS y reproductor mp3 (Salazar, 2022). Así mismo es la tecnología que va adónde va el usuario. Consiste en dispositivos portátiles de comunicaciones bidireccionales, dispositivos de computación y la tecnología de red que los conecta. Actualmente, la tecnología móvil se caracteriza por dispositivos habilitados para Internet como smartphones, tablets y relojes. Estos son los últimos en una progresión que incluye buscapersonas bidireccionales, computadoras portátiles, teléfonos celulares (teléfonos plegables), dispositivos de navegación GPS y más (International Business Machines [IBM], s.f.).

2.4. Aplicación Móvil

Una aplicación móvil (también conocida como App) se refiere a un programa de software diseñado específicamente para funcionar en dispositivos móviles, como teléfonos inteligentes y tabletas. Serna (2016) plantea que una aplicación móvil es un pequeño paquete de software que sirve para resolver una o varias tareas en específico. Estas aplicaciones están diseñadas para ejecutarse en sistemas operativos móviles, como Android y iOS, y ofrecen una amplia gama de funcionalidades y servicios para los usuarios.

El desarrollo de aplicaciones móviles es el proceso de creación de software para smartphones, tabletas y asistentes digitales, más comúnmente para los sistemas operativos Android y iOS. El software se puede preinstalar en el dispositivo, descargar desde una tienda de aplicaciones móviles o acceder a través de un navegador web móvil. (IBM, s.f.). Una aplicación móvil es un software diseñado para ser utilizado en dispositivos móviles como teléfonos inteligentes o tabletas. Enriquez & Casas (2013) argumentan que, al desarrollar una aplicación móvil, se debe tener en cuenta las restricciones de software como su tamaño y velocidad de procesamiento. Estas aplicaciones están diseñadas para ofrecer una variedad de funcionalidades y pueden abarcar una amplia gama de propósitos y usos.

2.4.1. Sistemas Operativos Móviles

El sistema operativo móvil son versiones simplificadas de los sistemas conocidos tales como Mac OS, Windows y Linux, con la diferencia que cuenta con funciones simplificadas. Serna (2016) menciona que dichos sistemas son implementados en dispositivos como tabletas, teléfonos inteligentes o también llamados smartphones y otros dispositivos, los cuales tienen la característica en común, cuentan con una pantalla táctil, es un software diseñado específicamente para gestionar y controlar las operaciones de hardware y software en dispositivos móviles. Estos sistemas operativos desempeñan un papel fundamental en la funcionalidad y la experiencia del usuario en dispositivos móviles.

2.4.1.1. Android

Es un sistema operativo de código abierto y ecosistema de servicios. Su desarrollo inicio en el año 2003 en la empresa Android Inc. Fundada por Andy Rubín y adquirida por Google en 2005. Luego de esta adquisición, fue presentada en el año 2008 como apuesta principal de la Open Handset Alliance, agrupación de más de 80 empresas lideradas por Google, entre ellas se encuentran fabricantes de dispositivos, operadores de telefonía y empresas de desarrollo de software. (Serna, 2016, p.4).

Android es un sistema operativo móvil desarrollado por Google. Es uno de los sistemas operativos más utilizados en dispositivos móviles en todo el mundo, incluyendo smartphones, tabletas y otros dispositivos. Dicho sistema operativo fue creado a partir del núcleo de Linux, diseñada para instalarse en cualquier dispositivo móvil de distinta marca. El primer dispositivo en implementar el sistema operativo móvil Android fue el HTC Dream/G1 fabricado por HTC (High Tech Computer Corporation) en septiembre de 2008 y estaba disponible a través de la compañía de telecomunicaciones T-Mobile en los Estados Unidos. Este dispositivo marcó el inicio de la era de Android en dispositivos móviles, pero ahora en la actualidad es utilizada por los principales fabricantes como LG, Huawei, Sony, Samsung, Xiaomi, entre otras. Android en los últimos años ha lanzado versiones compatibles con varios dispositivos móviles. (Ramírez, 2023).

Tabla 1

Comparación de las versiones de Android

Versiones	**Año de lanzamiento**	**Descripción**
Android 1.0	2008	El primer lanzamiento oficial de Android.
Android 1.1	2009	Mejoras y correcciones de Android 1.0.
Android 1.5 Cupcake	2009	Introdujo widgets y la pantalla táctil.
Android 1.6 Donut	2009	Mejoras en la velocidad y la búsqueda.
Android 2.0/2.1 Éclair	2009	Nuevas funciones y soporte multitouch.
Android 2.2 Froyo	2010	Mejoras en el rendimiento y Flash Player.
Android 2.3 Gingerbread	2010 - 2011	Optimización y mejora de la interfaz.

Android 3.0/3.1/3.2 Honeycomb	2011	Diseñado específicamente para tabletas.
Android 4.0 Ice Cream Sándwich	2011	Unificación de la experiencia en dispositivos.
Android 4.1/4.2/4.3 Jelly Bean	2012	Mejoras en la velocidad y notificaciones.
Android 4.4 KitKat	2013	Optimización para dispositivos de gama baja.
Android 5.0/5.1 Lollipop	2014	Material Design y mejor administración de la batería.
Android 6.0 Marshmallow	2015	Permisos de aplicaciones y soporte nativo de huellas dactilares.
Android 7.0/7.1 Nougat	2016	Mejoras en la productividad y realidad virtual.
Android 8.0/8.1 Oreo	2017	Arranque más rápido y notificaciones mejoradas.
Android 9 Pie	2018	Inteligencia artificial y bienestar digital.
Android 10	2019	Navegación por gestos y modo oscuro.
Android 11	2020	Mayor control de permisos y conversaciones en burbujas.
Android 12	2021	Rediseño visual y mejoras en la privacidad.

Nota. Comparación de las versiones de Android. Tomada de versiones de sistemas operativos móviles (Ramírez, 2023).

2.4.1.2. iOS

Es el sistema operativo que se instala en el iPod Touch, iPhone, y el iPad. iOS ha sido desarrollado a partir del sistema operativo Mac OS disponible en los diferentes ordenadores de escritorio y portátiles de Apple. Fue presentado por

> primera vez en 2007 con el lanzamiento del iPhone, sin embargo, en los años siguientes, Apple lo promociono solo como iPhone OS y finalmente en 2010 presento iOS de forma oficial como su sistema operativo móvil, unificando su estrategia para todos los equipos de la compañía. (Serna, 2016, p.6).

iOS es el sistema operativo móvil desarrollado por Apple Inc. para sus dispositivos móviles, también es conocido por su diseño elegante, su facilidad de uso y su integración con otros productos y servicios de Apple. A lo largo de los años, iOS ha experimentado múltiples actualizaciones y evoluciones, añadiendo nuevas características y mejoras en cada versión.

2.4.2. Tipos de Aplicaciones Móviles

2.4.2.1. Aplicaciones Nativas

Los sistemas operativos nativos son aquellos que utilizan un lenguaje de programación propio del sistema operativo móvil, con el fin de aprovechar el hardware que dispone. Según Serna (2016), “es el tipo de aplicación más potente, pero suele ser compleja de desarrollar ya que requiere de personal especializado en estas tecnologías” (p. 26). Las aplicaciones nativas se desarrollan utilizando las herramientas y recursos proporcionados por el fabricante del sistema operativo, como Apple para iOS o Google para Android, pero existen aplicaciones nativas multiplataformas que se pueden crear mediante el uso de nuevas tecnologías.

La ventaja de contar con una aplicación nativa es que permiten alcanzar un desarrollo veloz, es decir, tiene un mejor tiempo de respuesta y acceso a casi todo su hardware. Pero su desventaja es que su desarrollo se encuentra limitado a un solo lenguaje de programación, por tanto, es necesario el conocimiento de uno o más lenguajes.

2.4.2.2. Aplicaciones Web

También denominadas WebApps, Luna (2014) menciona que son aplicaciones móviles que son accedidas mediante la web o una red de intranet, para lo cual

necesitan de un navegador web para poderlas ejecutar, las características que tienen las aplicaciones móviles web, es la capacidad de adaptarse a las pantallas de cualquier dispositivo ya que cuentan con técnicas móviles como el responsive y el adaptative design, evitando así la creación de varios sitios web. Serna (2016) comenta si bien las aplicaciones web ofrecen muchas ventajas, también tienen limitaciones, como la necesidad de una conexión a Internet constante y posibles limitaciones de rendimiento en comparación con las aplicaciones nativas.

2.4.2.3. Aplicaciones Hibridas

Las aplicaciones móviles hibridas son aquellas que cuentan con tecnologías nativas como web. Luna (2014) menciona que las aplicaciones híbridas son aplicaciones móviles que combinan elementos de desarrollo web y desarrollo nativo. Estas aplicaciones se crean utilizando tecnologías web estándar como HTML, CSS y JavaScript, pero se ejecutan dentro de un contenedor nativo que permite acceder a las características del dispositivo móvil, como la cámara, el GPS y las notificaciones.

Características clave de las aplicaciones híbridas:

- **Desarrollo basado en web**: Las aplicaciones híbridas se desarrollan principalmente utilizando tecnologías web estándar como HTML, CSS y JavaScript, lo que facilita a los desarrolladores crear una aplicación que funcione en múltiples plataformas.
- **Contenedor nativo**: Para acceder a las características del dispositivo móvil, las aplicaciones híbridas se ejecutan dentro de un contenedor nativo. Este contenedor actúa como un puente entre el código web y el hardware del dispositivo.
- **Multiplataforma**: Una de las ventajas clave de las aplicaciones híbridas es su capacidad para funcionar en múltiples plataformas, como iOS, Android y, a veces, Windows Phone, lo que ahorra tiempo y esfuerzo en el desarrollo.
- **Acceso a características nativas**: Las aplicaciones híbridas pueden acceder a funciones nativas del dispositivo, como la cámara, el GPS, los

contactos y las notificaciones push, utilizando complementos o APIs específicas de la plataforma.

- **Diseño personalizado**: A través del uso de marcos de trabajo y bibliotecas, es posible lograr un aspecto y una sensación personalizados que se asemejen más a aplicaciones nativas.
- **Actualización centralizada**: Las actualizaciones de la aplicación se pueden realizar en el servidor web sin necesidad de actualizar la aplicación en las tiendas de aplicaciones, lo que facilita la distribución de correcciones y nuevas características.

2.5. Comercialización

La comercialización es un conjunto de actividades y estrategias que una empresa o negocio lleva a cabo para promocionar y vender sus productos o servicios. El objetivo principal de la comercialización es identificar las necesidades y deseos de los clientes, desarrollar productos o servicios que satisfagan esas necesidades y luego promocionarlos y distribuirlos de manera efectiva para alcanzar el éxito en el mercado. Kotler (2014) enfatiza que la comercialización es una disciplina social y administrativa que involucra una serie de actividades, desde la investigación de mercado hasta la promoción y distribución, con el objetivo de lograr el éxito en el mercado. La comprensión de las necesidades del cliente y la entrega de valor son elementos esenciales en este proceso. Se define como la estrategia y la acción que emprende una organización, con el fin de colocar sus productos o servicios en el mercado, procurando una ventaja competitiva sostenible. Para conseguir tales metas es indispensable que considere el entorno y sus tendencias (Fuentes, 2010). Es una disciplina en constante evolución que se adapta a las cambiantes tendencias del mercado y a las necesidades cambiantes de los consumidores. La gestión de comercialización de una empresa es la que se dirige a satisfacer las necesidades del mercado (de los clientes) que la dirección o gerencia general ha seleccionado como objetivo. El mercado son todos los clientes potenciales que comparten una necesidad o deseo específico, y que podrían comprar para satisfacer esa necesidad o deseo (Hernández y Machado, 2010).

2.5.1. Mercadotecnia

Mercadotecnia, también conocida como marketing, es el conjunto de actividades y estrategias que las empresas y organizaciones utilizan para identificar, crear y mantener relaciones con los clientes, así como para satisfacer sus necesidades y deseos a través de productos o servicios. La mercadotecnia implica investigar el mercado, identificar a la audiencia objetivo, desarrollar productos o servicios, establecer estrategias de precios, promoción y distribución, y evaluar constantemente el rendimiento y la retroalimentación del mercado. Es el conjunto de actividades destinadas a lograr con beneficio, la satisfacción del cliente. El concepto de mercadotecnia sostiene que para alcanzar las metas la organización se deben definir las necesidades y anhelos de los mercados meta (Lopez y Ruiz, 2022). Es una disciplina empresarial fundamental que se enfoca en entender y satisfacer las necesidades y deseos de los consumidores mediante la creación, promoción y entrega de productos o servicios de valor. La mercadotecnia es el análisis, organización, planeación y control de los recursos, políticas y actividades de la empresa que afectan al cliente, con vistas a satisfacer las necesidades y deseos de los grupos escogidos de clientes, obteniendo con ello una utilidad (Kotler, 2007).

2.5.2. El Impacto de la Mercadotecnia en Línea (Marketing Online)

Meneses (2015) señala que la mercadotecnia en línea, también conocida como marketing en línea o marketing digital, ha tenido un impacto significativo en la forma en que las empresas y organizaciones promocionan sus productos y servicios. Algunos de los aspectos más destacados del impacto de la mercadotecnia en línea incluyen:

- **Alcance global**: A través de internet, las empresas pueden llegar a una audiencia global de manera efectiva y económica, lo que amplía sus oportunidades de mercado.
- **Segmentación precisa**: La mercadotecnia en línea permite una segmentación detallada del público objetivo, lo que permite a las empresas dirigirse a grupos específicos de consumidores con mensajes y ofertas personalizados.

- **Medición y análisis**: La analítica web proporciona datos detallados sobre el comportamiento de los usuarios en línea, lo que permite a las empresas evaluar y ajustar sus estrategias de marketing en tiempo real.
- **Interacción y compromiso**: Las redes sociales y otras plataformas en línea permiten una mayor interacción con los clientes, lo que facilita la construcción de relaciones sólidas y la generación de lealtad de marca.
- **Acceso a múltiples canales**: Las empresas pueden utilizar una variedad de canales en línea, como redes sociales, motores de búsqueda, publicidad en línea y correo electrónico, para llegar a su audiencia de diferentes maneras.
- **Reducción de costos**: En muchos casos, el marketing en línea puede ser más rentable que las estrategias de marketing tradicionales, como la publicidad impresa o la televisión.
- **Nuevas oportunidades de negocio**: La mercadotecnia en línea ha dado lugar a nuevas oportunidades de negocio, como el comercio electrónico y los modelos de negocio basados en suscripciones.
- **Competencia global**: La competencia en línea puede ser intensa, ya que las empresas de todo el mundo pueden competir por la atención de los consumidores en línea.

El impacto de la mercadotecnia en línea es un tema amplio y en constante evolución, y su influencia en la forma en que las empresas se promocionan y conectan con los consumidores seguirá creciendo a medida que la tecnología y las plataformas en línea continúen desarrollándose.

2.6. Productos

En mercadotecnia, un producto es cualquier cosa que una empresa ofrezca en el mercado para su atención, adquisición, uso o consumo. Monferrer (2013) señala los productos pueden ser tanto bienes físicos como servicios, y también pueden incluir elementos intangibles, como experiencias, ideas o conceptos. El proceso de distribución de productos es una parte fundamental de la cadena de suministro y

marketing de una empresa. Implica todas las actividades necesarias para llevar los productos desde el fabricante o productor hasta los consumidores finales. Santesmases et al. (2014) enfocan en la gestión de productos, un aspecto esencial de la mercadotecnia que implica la planificación, lanzamiento y supervisión de un producto o línea de productos a lo largo de su ciclo de vida en el mercado para satisfacer las necesidades del consumidor y lograr los objetivos comerciales. Kotler et al. (2021) destaca que la distribución de los productos debe estar diseñada para satisfacer las necesidades y deseos de los consumidores, y debe ser coherente con otros elementos del marketing mix, como el precio y la promoción. Aquí hay algunos principios clave que Kotler sugiere en relación con la distribución de productos:

- **Accesibilidad**: Los productos deben estar disponibles donde los consumidores los necesitan o desean. Esto implica seleccionar los canales de distribución adecuados y establecer ubicaciones de venta estratégicas.
- **Cobertura de mercado**: La distribución debe abarcar el mercado objetivo de manera efectiva. Esto puede requerir la utilización de múltiples canales de distribución para llegar a diferentes segmentos de mercado.
- **Conveniencia**: La distribución debe ser conveniente para los consumidores. Esto puede incluir la ubicación de tiendas físicas en áreas de alto tráfico, la disponibilidad de compras en línea y opciones de entrega flexibles.
- **Eficiencia logística**: La gestión eficiente de la cadena de suministro y la logística es esencial para garantizar que los productos lleguen a su destino de manera oportuna y a un costo razonable.
- **Integración con otros elementos del marketing mix**: La estrategia de distribución debe ser coherente con otros aspectos del marketing, como el producto en sí, su precio y las actividades de promoción. Todo debe trabajar en conjunto para satisfacer las necesidades del mercado.
- **Valor agregado**: La distribución también puede ser una fuente de valor agregado. Por ejemplo, ofrecer un excelente servicio al cliente, una entrega rápida o una experiencia de compra excepcional puede diferenciar a una empresa de la competencia.

- **Evaluación continua**: Kotler enfatiza la importancia de evaluar y ajustar continuamente la estrategia de distribución en función de la retroalimentación del mercado y los cambios en las necesidades de los consumidores.

2.7. Servicios

Un servicio se refiere a una actividad intangible que proporciona un beneficio o valor a un cliente y, a menudo, implica la interacción con personas, recursos o sistemas para satisfacer las necesidades o deseos del cliente. Kotler y Armstrong (2008) proponen que un servicio es como cualquier actividad o beneficio que una parte puede ofrecer a otra, que es esencialmente intangible y no tiene como resultado la propiedad de nada. Del mismo modo, Norton (2005) señala un servicio como cualquier función o utilidad proporcionada por un sistema o software para facilitar la gestión, el mantenimiento y la utilización eficiente de una computadora o red, por otra parte Tanenbaum (2009) plantean un servicio informático como cualquier función o capacidad que permita la comunicación y la transferencia de datos entre dispositivos en una red, como servicios de enrutamiento, conmutación, seguridad o utilidad proporcionada por sistemas informáticos, software o redes para satisfacer necesidades específicas de los usuarios, aplicaciones o la infraestructura de tecnología de la información.

2.8. Nube

La nube es un paradigma en la informática que permite acceder y utilizar recursos informáticos a través de Internet, lo que brinda ventajas como escalabilidad, flexibilidad y reducción de costos. La computación en la nube como un modelo para permitir un acceso ubicuo, conveniente, bajo demanda y desde cualquier red a un conjunto compartido de recursos informáticos configurables, por ejemplo, redes, servidores, almacenamiento, aplicaciones y servicios que se pueden aprovisionar y liberar rápidamente con un esfuerzo mínimo de gestión o interacción del proveedor de servicios (Bohn et al., 2020). La nube es la plataforma que permite a las empresas construir aplicaciones más rápido, de manera más escalable y a un costo más bajo al

eliminar gran parte de la carga de administrar infraestructura de TI, a través de la virtualización de recursos de hardware y la administración de recursos de manera eficiente (Phillips, 2022). La nube como concepto no es más que una abstracción del lugar real donde se almacenan y conectan entre sí la información y los datos del usuario. Internet, este término se refiere a los servidores de las empresas de Internet en los que se almacena y gestiona esta información. Esto significa lo mismo que cargar datos en Internet, nube significa alojarlos en un servidor. En la nube se almacena todo tipo de datos e información, fotos, imágenes, documentos, videos, audios, archivos de todo tipo, software y mucho más (Alapati, 2019).

2.8.1. *Servicios en la Nube*

Los servicios en la nube, también conocidos como Cloud Services en inglés, se refieren a una variedad de servicios y recursos informáticos que se ofrecen a través de Internet. Estos servicios son proporcionados por proveedores de nube y permiten a individuos, empresas y organizaciones acceder, almacenar y gestionar datos, aplicaciones y recursos de cómputo de manera remota y escalable. De acuerdo con Wittig y Wittig (2018) la computación en la nube es un grupo de sistemas de tecnología informática y empresarial que utilizan el modelo de la web de Internet para brindar servicios bajo demanda. Estos sistemas suelen ser parte de un grupo en red grande, como Internet, una intranet corporativa o la web. Por lo general, utilizan tecnologías de virtualización, a través de las cuales el usuario puede combinar recursos de cómputo, almacenamiento y red a medida que los necesita. Ainsley (2020) mencionan que la computación en la nube se refiere a la entrega de servicios de computación, como servidores, almacenamiento, bases de datos, redes, software, análisis y más, a través de Internet. En lugar de poseer y mantener servidores y recursos de TI localmente, las empresas y los individuos pueden acceder a estos servicios en la nube bajo demanda, pagando solo por lo que utilizan. Esto proporciona una serie de beneficios, como escalabilidad, flexibilidad y reducción de costos. La computación en la nube se ha convertido en una herramienta popular para empresas y organizaciones que buscan reducir costos y aumentar la eficiencia, es un modelo de entrega y consumo de servicios bajo demanda. Flores (2021) indica que el crecimiento de la computación en

la nube ha sido exponencial, por lo que se han desarrollado modelos de servicio e implementación para cubrir las necesidades de los usuarios, Algunos usuarios y empresas utilizan la nube para almacenar copias de seguridad de sus datos. En algunos casos se usa un híbrido entre la infraestructura local y el entorno Cloud, existen varias formas de implementar una aplicación porque no hay restricciones, básicamente la capacidad es ilimitada. Inteligencia artificial, aplicaciones web, aplicaciones móviles, Big Data, etc. Normalmente las nuevas empresas desde un inicio apuestan por implementar su infraestructura en la nube, y las que tienen sus centros de datos localmente, la mayoría están migrando a la nube. Estos servicios son proporcionados por proveedores de nube, como Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP) y otros.

2.8.2. Características de Cloud Services

Según Flores (2021) estas son las principales características de computación en la nube:

- **Autoservicio bajo demanda:** Es un servicio disponible de forma automática y a demanda, significa que puedes aprovisionar y gestionar los recursos cuando los necesites, sin la intervención del proveedor.
- **Accesible a través de la red:** Los recursos deben de estar disponible en la red, sin importar el tipo de nube. El servicio puede estar en la red privada o de forma pública, accesible a través de Internet.
- **Agrupación de recursos:** Los recursos del proveedor están agrupados para atender a varios clientes. Este modelo permite que muchos clientes puedan compartir el mismo hardware físico. No obstante, los recursos de cada cliente están seguros y aislados de los demás.
- **Elasticidad:** Va un poco más allá del concepto de escalabilidad que conocemos tradicionalmente. Se refiere a la capacidad de crecer la infraestructura y los recursos según las necesidades, y de igual manera reducirla, liberando los recursos que ya no se requieran.
- **Servicio medido y pago por uso:** El uso de recursos es monitoreado, controlado y reportado, proporcionando un nivel de trasparencia. Hay

diferentes tipos de medición, almacenamiento, ancho de banda, tiempo de actividad (horas, minutos, segundos), etc., en cualquier caso, solo se facturan los recursos utilizados, ni más, ni menos.

2.8.2.1. Ventajas de Cloud Services

- **Agilidad:** Ésta es una de las principales ventajas de la computación en la nube. Los recursos están literalmente a solo un clic de distancia. Los entornos que antes tardaban días en estar disponibles ahora pueden estar disponibles para los desarrolladores en cuestión de minutos. Por supuesto, esto supone un gran beneficio para la empresa, ya que puede centrarse en sus ideas de negocio e implementarlas más rápido.
- **Sin grandes inversiones iniciales:** Los gastos de capital (CapEx) son reemplazados por gastos operativos (OpEx). Ya no hay que pensar en hacer grandes inversiones para adquirir servidores y mantener centros de datos, en su lugar, puedes montar tu infraestructura en la nube y pagar solo lo que consumes, por eso, son gastos variables de operación.
- **Sin compromisos contractuales:** Cuando queremos desplegar una aplicación nos enfrentamos a una decisión, definir la capacidad de cómputo para dicha aplicación, donde podemos sobrepasarnos y no utilizar todos los recursos, o quedarnos cortos y tener que lidiar con capacidad limitada. Con la computación en la nube, nos olvidamos de estas adivinanzas. Podemos acceder a tanta o poca capacidad como sea necesario, y escalar hacia arriba y hacia debajo rápidamente según la demanda.
- **Beneficio por economías de escala:** Esto es simple, si al proveedor le va bien, los clientes se ven beneficiados. Cuando cientos de miles de clientes nuevos consumen el servicio, el proveedor logra mayores economías de escala, por lo que se puede conseguir un costo variable más bajo de lo normal, en otras palabras, menores precios de pago por uso.
- **Infraestructura global:** Puede implementar su aplicación en una o más regiones, intentando acercarse a sus clientes para garantizar una menor latencia y una mejor experiencia de usuario. Lo más sorprendente es que

todo esto sucede en apenas unos minutos. También proporciona alta disponibilidad, lo que permite una rápida recuperación ante fallas o desastres. Con una arquitectura bien diseñada y automatizada, el usuario final ni siquiera notará el cambio o interrupción del servicio.

2.8.2.2. Desventajas de Cloud Services

- **Necesitas de una conexión a Internet estable:** Es verdad que la facilidad para acceder desde cualquier parte es una ventaja, pero también es cierto que dependes de la conexión a Internet. De alguna manera existe un punto de falla, si hay un problema con el ISP o proveedor de Internet, no tienes salida, aunque el servicio en la nube esté disponible.
- **Dependes de la seguridad de un tercero:** En materia de seguridad existe un modelo de responsabilidad compartida, por lo que tenemos que confiar parte de ella al proveedor. Sin embargo, mi justificación aquí es que el proveedor obviamente tiene mejor tecnología que la que nosotros podemos adquirir. Esto lo veo como una desventaja en algunos casos a la hora de sugerir una migración a la nube; Es todo un desafío convencer a alguien que desconfía y no está dispuesto a cambiar.

2.8.3. Tipos de nubes de Cloud Services

Se ha desarrollado un modelo de implementación, que comúnmente se conoce como tipo nube o tipo computación en la nube. Inicialmente, este modelo se separó en dos dimensiones, nube privada y nube pública. Pero de aquí surge otro tipo de nube que no es más que una combinación de uno o dos modelos anteriores. A lo que me refiero es a nube híbrida y nube múltiple o multicloud (Flores, 2021).

- **Nube publica:** Es el tipo más común de computación en la nube. En el modelo de nube pública, los recursos potencia informática, almacenamiento, bases de datos, plataforma de desarrollo, aplicaciones, etc. Se ofrecen bajo demanda e inmediatamente a través de Internet. Los recursos subyacentes, hardware y software, así como todo lo que pertenece a la infraestructura

física, son propiedad del proveedor, quien es responsable de su mantenimiento y gestión. En la nube pública, puede implementar cualquier aplicación, experimentar, migrar todo su centro de datos; en resumen, las posibilidades son infinitas.

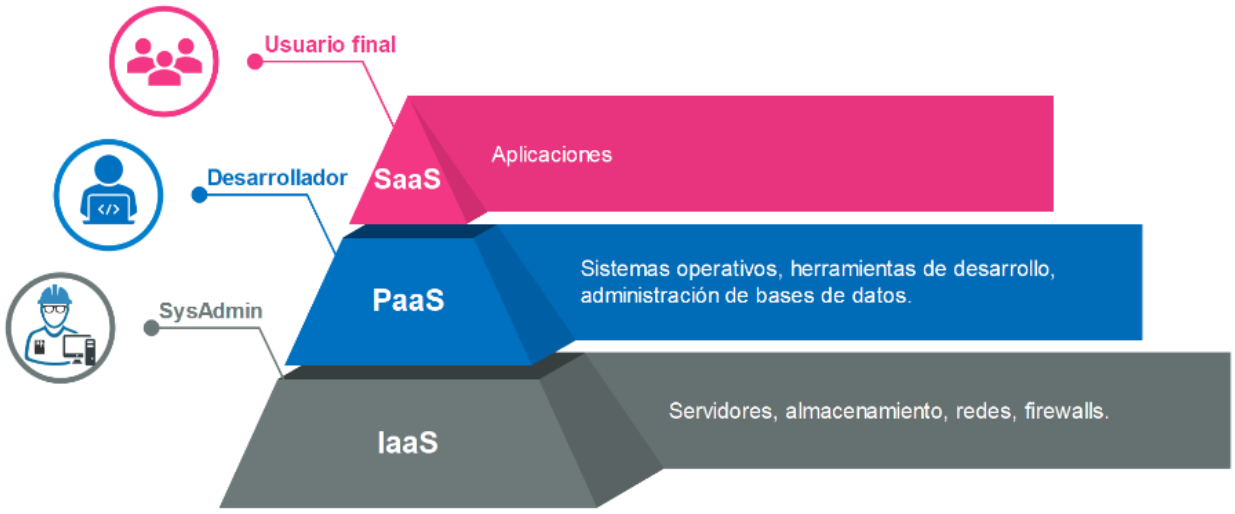
- **Nube privada:** Con el modelo de nube pública, el cliente debe confiar parte de la seguridad al proveedor. Hay empresas que no confían en la seguridad de terceros o creen que deben tener control total sobre su entorno con su equipo de trabajo y su propia infraestructura. Por ello surgió la necesidad de replicar el modelo en entornos privados, en los centros de datos de la empresa. De esta forma se consiguen los beneficios del cloud como agilidad, automatización, escalabilidad, etc., pero dentro de las instalaciones de la misma empresa. Por tanto, una nube privada consta de recursos informáticos utilizados sólo por una institución, la infraestructura siempre se gestiona en una red privada y el hardware y software están dedicados a la propia organización.
- **Nube hibrida:** Una nube híbrida es una composición de dos tipos diferentes (nube pública y nube privada), que se consideran entidades separadas, pero al mismo tiempo, unidas por tecnologías estándar, que facilitan la comunicación, la portabilidad de datos y las aplicaciones. El objetivo es combinar ambos modelos de implementación y obtener lo mejor de ambos.
- **Nube multicloud:** Este tipo de implementación se da con mayor frecuencia en las empresas. Es la combinación de dos o más implementaciones en la nube del mismo tipo pública o privada. Por tanto, podemos combinar servicios de diferentes proveedores de nube.

2.8.4. Tipos de Servicios en la Nube

En la siguiente figura se puede ver una representación gráfica para diferenciar las capas y un resumen de lo que nos proporciona cada modelo de servicio. Pero de igual forma, explicaremos uno a uno.

Figura 3

Tipos de servicios en la nube



Nota. Servicios en la Nube. Reproducida de Cloud Computing: Tipos de nubes, servicios y proveedores, de Flores 2021 (OpenWebinars S.L.).

2.8.4.1. Software as a Service (SaaS)

El más utilizado. El software está alojado en servidores de los proveedores y el cliente accede a ellos a través del navegador web. Todo lo relacionado con mantenimiento, soporte y disponibilidad es manejado por el proveedor.

2.8.4.2. Platform as a Service (PaaS)

En este tipo de servicios en la nube el proveedor ofrece acceso a un entorno basado en cloud en el cual los usuarios pueden crear y distribuir sus propias aplicaciones. El proveedor proporciona la infraestructura subyacente.

2.8.4.3. Infrastructure as a Service (IaaS)

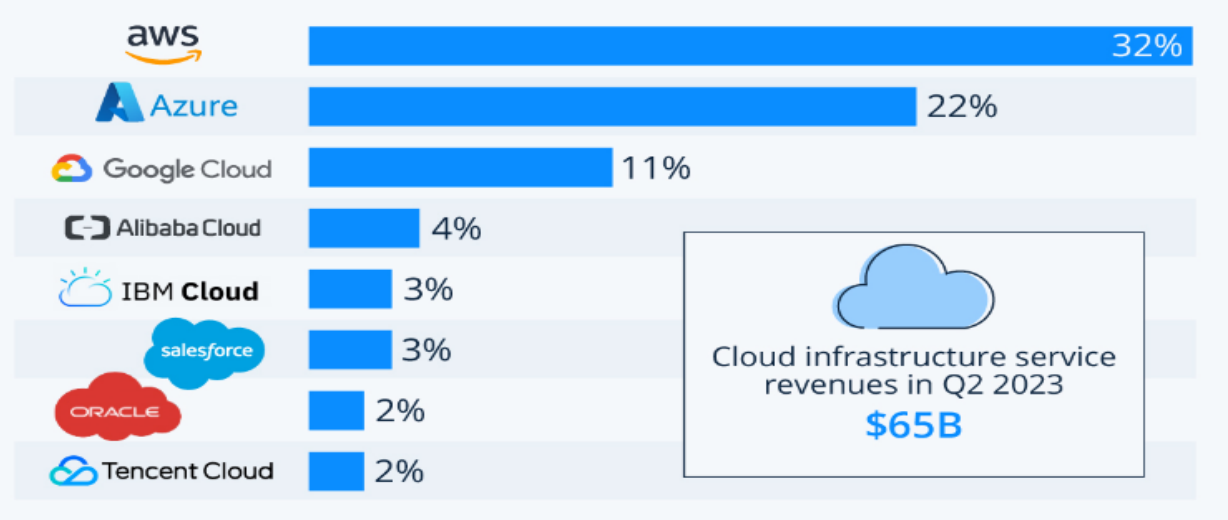
Un proveedor de servicios proporciona el software y las aplicaciones a través de Internet. Los usuarios se suscriben al software y acceden a él a través de la web o las APIs del proveedor.

2.8.5. Proveedores de Cloud Computing

La demanda de Cloud Computing es enorme. En la siguiente figura trataremos los principales proveedores de nube para conocer las alternativas disponibles.

Figura 4

Proveedores de infraestructura en la nube



Nota. Proveedores en la Nube. Reproducida de Mercado de Infraestructura en la Nube: Amazon mantiene el liderazgo en el mercado de la nube, de Richter 2023 (Statista.com).

2.9. Metodología de Desarrollo

La metodología de desarrollo de software se refiere a un sistema de trabajo que se utiliza para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información. A lo largo del tiempo, una gran cantidad de métodos han sido desarrollados diferenciándose por su fortaleza y debilidad. Esta metodología consiste en conocimientos del desarrollo de programas informáticos con énfasis en el proceso de desarrollo de software. Estos son a menudo vinculados a algún tipo de organización, que además desarrolla, apoya el uso y promueve la metodología.

Las metodologías de desarrollo de software son enfoques y procesos sistemáticos que guían el ciclo de vida del desarrollo de software, desde la concepción y la planificación hasta la implementación, las pruebas, la entrega y el mantenimiento. Estas metodologías proporcionan un marco de trabajo estructurado para el desarrollo de software y ayudan a los equipos de desarrollo a organizar sus tareas, administrar recursos y entregar de manera eficiente productos de software de alta calidad (Jimenez y Bolivar, 2016). La metodología ágil no se define por una serie de actividades o métodos de desarrollo específicos, más bien la metodología ágil consiste en un grupo de metodologías que se centran en circuitos de retroalimentación exhaustivos y la mejora continua. Debido a la popularidad de la gestión ágil de proyectos, cada vez más organizaciones están pasando de equipos y proyectos individuales a programas completos. La metodología ágil se ha extendido incluso más allá de los equipos de desarrollo y ahora es utilizada por equipos de TI, marketing y desarrollo empresarial, entre otros (Atlassian, 2023).

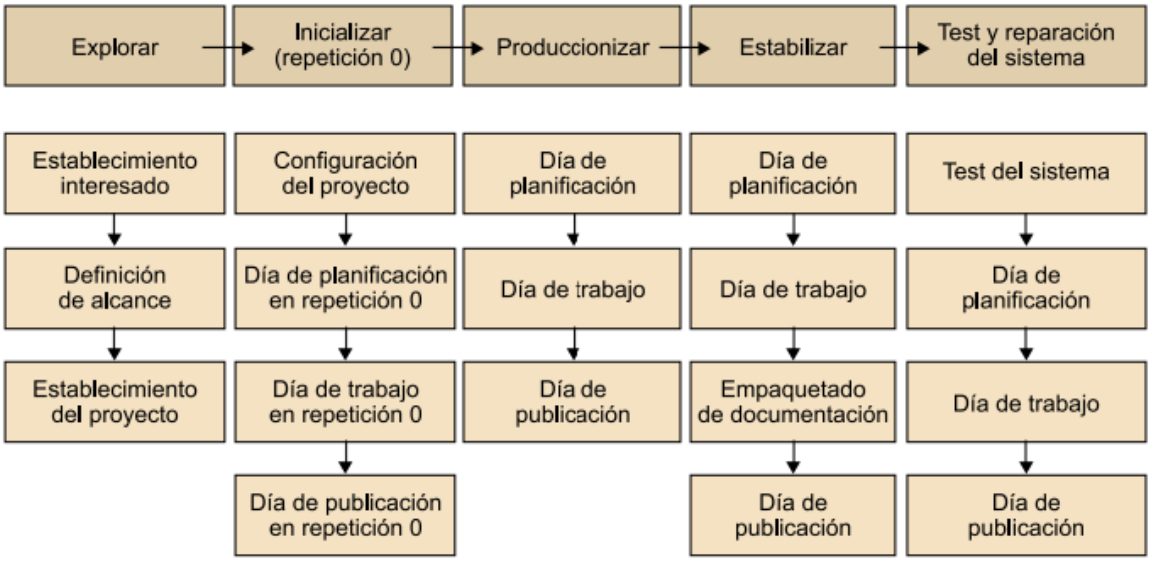
2.9.1. Metodología Mobile-D

El objetivo de este método es conseguir ciclos de desarrollo muy rápidos en equipos muy pequeños. Fue creado en un proyecto finlandés en 2005. Basado en metodologías conocidas pero aplicadas de forma estricta como: extreme programming, Crystal Methodologies y Rational Unified Process (Blanco et al., 2009). Fue creado con el objeto de ser una metodología de rápidos resultados, enfocada en grupos de trabajos pequeños, los cuales debería poseer confianza entre sus miembros, y un nivel de habilidad similar, además busca entregar resultados funcionales en periodos cortos de tiempo.

Se compone de cinco fases: exploración, inicialización, fase de producto, fase de estabilización y la fase de pruebas, cada una posee un tiempo de planificación y un tiempo de entrega.

Figura 5

Ciclo de desarrollo de Mobile-D



Nota. Metodología de desarrollo Mobile-D. Modificada de Metodología de desarrollo ágil para sistemas móviles. Introducción al desarrollo con Android y el iPhone (Blanco et al., 2009).

Cada fase excepto la inicial tiene siempre un día de planificación y otro de entrega. Las fases son:

2.9.1.1. Fase de Exploración

Se centra en el interés de la planificación y a los conceptos básicos del proyecto. Se realizan los alcances del proyecto y su establecimiento con las funcionalidades donde se va a llegar. Los autores de la metodología ponen además especial atención a la participación de los clientes en esta fase. El propósito de esta fase es la planificación y establecimiento lo cual es muy importante para establecer las bases para una implementación bien controlada de software, la arquitectura del producto y el proceso de desarrollo (Blanco et al., 2009).

Los objetivos de la fase de exploración son:

- Establecer los grupos de actores necesarios en la planificación y el seguimiento del proyecto de desarrollo de software.
- Planificar el proyecto respecto al entorno, el personal y los problemas del proceso.

Las entradas de la fase de exploración son:

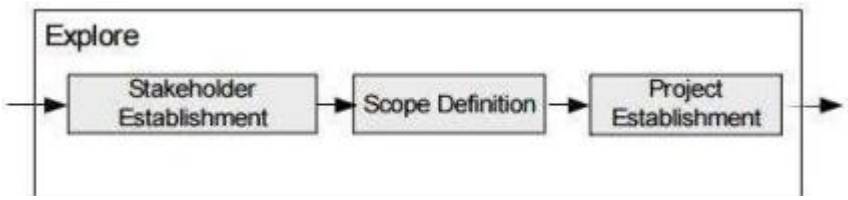
- La propuesta del producto.
- Biblioteca de procesos de Mobile D.
- Documento de requisitos iniciales.

Las salidas de esta fase son:

- El documento de requisitos iniciales donde se ha definido los requerimientos iniciales del desarrollo del producto.
- Plan de proyecto incluyendo línea de tiempo, el ritmo, las terminaciones, los recursos del proyecto, los actores y sus responsabilidades.
- Plan de Medición y plan de Formación., descripción de la línea de la arquitectura.

Figura 6

Fase 1 Exploración



Nota. Fases de la Metodología de desarrollo Mobile-D. Tomada de Metodologías ágiles en el desarrollo de aplicaciones para dispositivos móviles (Amaya, 2013).

2.9.1.2. Fase de Inicialización

Durante esta fase, se preparan e identifican todos los recursos necesarios. Se preparan los planes para las siguientes fases y se establece el entorno técnico. Los autores de Mobile-D afirman que su contribución al desarrollo ágil se centra fundamentalmente en esta fase, en la investigación de la línea arquitectónica. Esta acción se lleva a cabo durante el día de planificación. En esta fase se analizan el conocimiento y los patrones arquitectónicos utilizados en la empresa y los relacionan con el proyecto actual. Se agregan las observaciones, se identifican similitudes y se extraen soluciones viables para su aplicación en el proyecto. Finalmente, la metodología también contempla algunas funcionalidades nucleares que se desarrollan en esta fase, durante el día de trabajo.

Los objetivos de esta fase son:

- Obtener una buena compresión global del producto para el equipo de desarrollo del proyecto, sobre los requisitos iniciales y la línea de la arquitectura.
- Preparar los requisitos físicos, técnicos y humanos, así como la comunicación con el cliente, los planes del proyecto y todas las cuestiones fundamentales de desarrollo a fin de que todo esté en plena disposición para la implementación.

Las entradas de esta fase son:

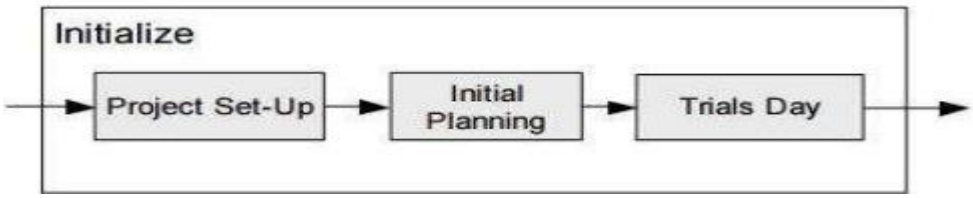
- Documento de requisitos Iniciales.
- Plan de proyecto y descripción del proceso base.
- Plan de medición y plan de formación.
- Descripción de la línea de arquitectura

Las salidas de la fase son:

- Documento con descripción del diseño.
- Documento de requisitos iniciales actualizados.
- Desarrollo de notas y la interfaz de usuario.

Figura 7

Fase 2 Inicialización



Nota. Fases de la Metodología de desarrollo Mobile-D. Tomada de Metodologías ágiles en el desarrollo de aplicaciones para dispositivos móviles (Amaya, 2013).

2.9.1.3. Fase de Producto

Antes de iniciar el desarrollo de una funcionalidad debe existir una prueba que verifique su funcionamiento, en esta fase se lleva a cabo toda la implementación de los módulos. Primero se planifica la iteración de trabajo en términos de requisitos y tareas a realizar. Se preparan las pruebas de la iteración de antemano (de ahí el nombre de esta técnica de Test Driven Development, TDD). Las tareas se llevarán a cabo durante el día de trabajo, Durante el último día se lleva a cabo la integración del sistema seguida de las pruebas de aceptación.

Los objetivos de esta fase son:

- Implementar la funcionalidad del producto priorizando los requerimientos del cliente.
- Centrarse en la funcionalidad básica fundamental para permitir múltiples ciclos de mejora.

Las entradas de esta fase son:

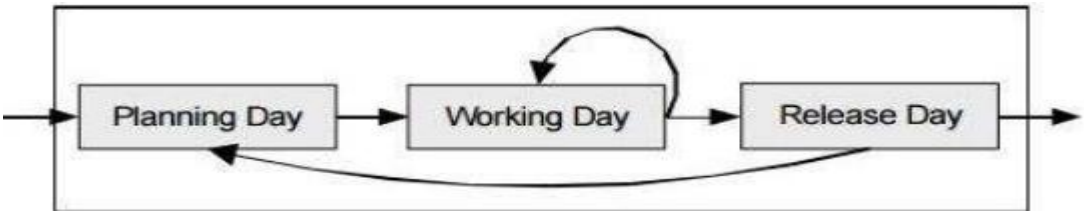
- Implementar plan de proyecto y plan de la línea de la arquitectura.
- La 1ra versión de la arquitectura de software y descripción del diseño.
- Planes para la comprobación de los elementos críticos del desarrollo.
- Funcionalidad implementada.

Los elementos de salida de esta fase son:

- Funcionalidad Implementada.
- Notas de desarrollo.
- Ilustraciones de Interfaz de Usuario.
- Lista de puntos de acción.
- Actualizado plan del proyecto.

Figura 8

Fase 3 Productización



Nota. Fases de la Metodología de desarrollo Mobile-D. Tomada de Metodologías ágiles en el desarrollo de aplicaciones para dispositivos móviles (Amaya, 2013).

2.9.1.4. Fase de Estabilización

Se llevan a cabo las últimas acciones de integración para asegurar que el sistema completo funciona correctamente. Esta será la fase más importante en los proyectos. En esta fase se realizarán tareas similares realizadas en la fase de producto, aunque en este caso todo el esfuerzo se dirige a la integración del sistema. En esta fase se llega la integración para vincular los módulos separados en una única aplicación. Adicionalmente se puede considerar en esta fase la producción de documentación.

Los objetivos de la fase de estabilización son:

- Finalizar la implementación del producto.
- Mejorar y garantizar la calidad del producto.
- Finalizar la documentación del proyecto.

Las entradas de la fase de estabilización son:

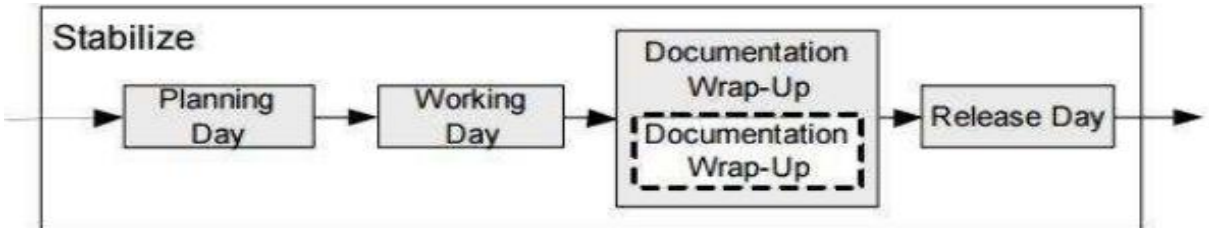
- La funcionalidad implementada del producto.
- Los artefactos de desarrollo relacionado.

Las salidas de esta fase son:

- La funcionalidad implementada de todo el proyecto de todo el software.
- La documentación del producto finalizado.

Figura 9

Fase 4 Estabilización



Nota. Fases de la Metodología de desarrollo Mobile-D. Tomada de Metodologías ágiles en el desarrollo de aplicaciones para dispositivos móviles (Amaya, 2013).

2.9.1.5. Fase de Pruebas y Reparación

La última fase tiene como meta la disponibilidad de una versión estable y plenamente funcional. El producto terminado e integrado se prueba con los requisitos de cliente y se eliminan todos los defectos encontrados.

Los objetivos de la fase de pruebas son:

- Probar el sistema basado en la documentación producida en el proyecto.
- Planificar la solución a los defectos encontrados.
- Producir una aplicación libre de errores como sea posible.

Las entradas de esta fase son las siguientes:

- La funcionalidad implementada.

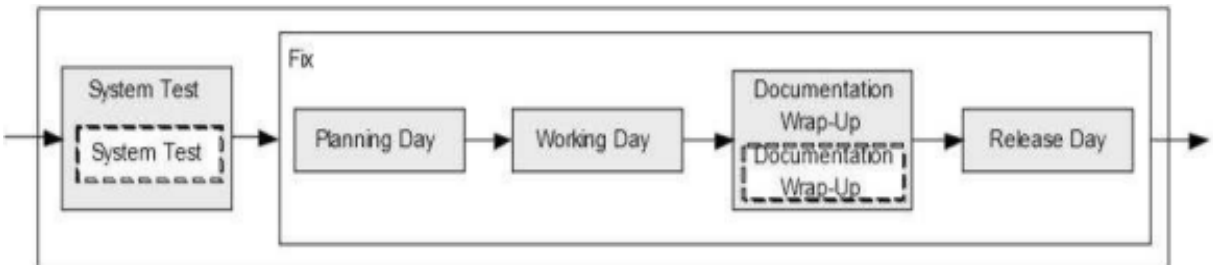
- Funcionalidad del usuario definida completamente.
- Descripción de la interfaz de usuario que se utiliza para crear casos de pruebas.

Las salidas de la fase de pruebas son:

- Un sistema testeado y corregido (versión final).
- Informe de pruebas del sistema descripción del proceso de pruebas y los errores y defectos encontrados en el software.
- Registro de pruebas realizados en el sistema y los resultados obtenidos al momento de ejecutar el testeo.

Figura 10

Fase 5 Pruebas



Nota. Fases de la Metodología de desarrollo Mobile-D. Tomada de Metodologías ágiles en el desarrollo de aplicaciones para dispositivos móviles (Amaya, 2013).

Las ventajas de esta metodología son las siguientes:

- Un costo bajo al realizar un cambio en el proyecto.
- Entrega resultados de manera rápida.
- Asegura el software adecuado en el momento adecuado.

La metodología también cuenta con las siguientes desventajas:

- No sirve para grupos de desarrollos grandes y segmentados.
- Depende de buena comunicación entre los miembros del equipo.

2.10. Ingeniería de Software

La ingeniería de software se puede definir desde la perspectiva de Sommerville (2011) como una disciplina de ingeniería que se interesa por todos los aspectos de la producción de software, desde las primeras etapas de la especificación del sistema hasta el mantenimiento del sistema después de que se pone en operación. Estos procesos incluyen actividades como la definición de requisitos, el diseño de la arquitectura, la implementación del código, la prueba y la entrega. De la misma forma Pressman (2010) describe que la ingeniería de software como una aplicación del conocimiento, diseño y construcción de programas, los requerimientos de la tecnología de la información que demandan los individuos, negocios y gobiernos. Las instituciones, negocios y gobiernos cada vez dependen de un software para tomar decisiones estratégicas y acertadas para su control cotidiano, la ingeniería de software trata de reducir inconvenientes con el software. Conforme va creciendo los usuarios y el tiempo de uso la también crecerán las demandas para mejorarlas. El fundamento para la ingeniería de software es la capa proceso. El proceso de ingeniería de software es el aglutinante que une las capas de la tecnología y permite el desarrollo racional y oportuno del software de cómputo. La ingeniería de software es una tecnología con varias capas como se muestra en la siguiente figura.

Figura 11

Capas de la Ingeniería de Software

Nota. El Software y la Ingeniería de Software. Tomada de Ingeniería de Software un Enfoque Practico (Pressman, 2010).

2.10.1. Herramientas

La ingeniería de software utiliza métodos, técnicas y herramientas específicas para facilitar el diseño, la implementación y el mantenimiento del software. Esto incluye lenguajes de programación, entornos de desarrollo, sistemas de control de versiones, entre otros. Las herramientas en la ingeniería de software son programas o aplicaciones que se utilizan para facilitar y mejorar el proceso de desarrollo de software. Estas herramientas pueden abarcar diversas etapas del ciclo de vida del desarrollo de software, desde la planificación y diseño hasta la implementación, pruebas y mantenimiento (Pressman, 2010).

2.10.1.1. Android Studio

Android Studio es un entorno de desarrollo integrado (IDE) ampliamente utilizado para crear aplicaciones Android. Aunque no existe una definición específica de Android Studio de un autor en particular, puedo proporcionarte una descripción general del IDE y su importancia según expertos en desarrollo de aplicaciones móviles. Lujan (2017) afirma Android Studio es desarrollado y mantenido por Google y se ha convertido en la herramienta preferida para desarrolladores de Android en todo el mundo. Proporciona un conjunto de herramientas poderosas para la creación de aplicaciones Android, que incluye un editor de código, emulador de dispositivos, herramientas de depuración y más. Android Studio se basa en el popular entorno de desarrollo IntelliJ IDEA.

- Algunas de las características y capacidades de Android Studio incluyen:
- Edición de código eficiente con resaltado de sintaxis y autocompletado.
- Integración con el sistema de compilación Gradle para administrar dependencias y construir proyectos.
- Un emulador de dispositivos Android para probar aplicaciones en diferentes configuraciones de hardware y versiones de Android.
- Herramientas de depuración avanzadas, incluyendo la posibilidad de depurar aplicaciones en tiempo real en dispositivos físicos.
- Soporte para desarrollo de aplicaciones en lenguajes como Java y Kotlin.

- Integración con Android Jetpack, un conjunto de bibliotecas y herramientas que aceleran el desarrollo de aplicaciones.

2.10.1.2. Visual Studio Code

Visual Studio Code (VS Code) es un entorno de desarrollo de código abierto ampliamente utilizado que ha sido elogiado por su flexibilidad, extensibilidad y eficiencia. Según Johnson (2019) Visual Studio Code es un entorno de desarrollo de código abierto altamente versátil y potente que se ha convertido en una herramienta esencial para desarrolladores de software en todo el mundo. Ofrece una experiencia de desarrollo moderna y eficiente con una interfaz de usuario intuitiva y una amplia gama de extensiones que permiten personalizar y adaptar el entorno a las necesidades de cada desarrollador. Visual Studio Code es conocido por su capacidad para admitir una variedad de lenguajes de programación y tecnologías, lo que lo convierte en una opción versátil para proyectos de desarrollo de software. Además, su comunidad activa y su constante evolución hacen que sea una opción atractiva tanto para principiantes como para desarrolladores experimentados.

2.10.1.3. Dart

De acuerdo con Buckett (2013) Dart es un lenguaje de programación de código abierto desarrollado por Google. Está diseñado para ser utilizado en una variedad de contextos, incluyendo desarrollo web y desarrollo de aplicaciones móviles. Dart es conocido por su sintaxis limpia y su enfoque en la eficiencia y el rendimiento.

Algunas de las características clave de Dart incluyen:

- **Tipado estático opcional**: Dart permite la inferencia de tipos, lo que significa que puedes optar por especificar los tipos de variables o permitir que el sistema de tipos los infiera automáticamente.
- **Rápido rendimiento**: Dart se compila a código nativo o JavaScript, lo que puede dar lugar a un rendimiento rápido en aplicaciones web y móviles.
- **Facilidad de uso**: Dart está diseñado para ser fácil de aprender y usar, lo que lo hace accesible tanto para principiantes como para programadores experimentados.

- **Colecciones y bibliotecas sólidas**: Dart ofrece una variedad de colecciones y bibliotecas estándar para ayudar en el desarrollo de aplicaciones.

2.10.1.4. Flutter

Como indica Napoli (2019) Flutter es un marco de desarrollo de código abierto desarrollado por Google que se utiliza para crear aplicaciones móviles nativas de alta calidad para iOS, Android y otras plataformas desde una única base de código. Algunas de las características y conceptos clave de Flutter incluyen:

- **Widget-Based**: Flutter se basa en un modelo de programación de widgets, donde todo en la interfaz de usuario se trata como un widget. Los widgets son componentes reutilizables que se pueden combinar para construir interfaces de usuario complejas.
- **Lenguaje Dart**: Flutter utiliza el lenguaje de programación Dart como su lenguaje principal. Dart es conocido por su velocidad de ejecución y su capacidad de compilar código nativo eficiente.
- **Interfaz de Usuario Personalizable**: Flutter permite una personalización extensa de la interfaz de usuario, lo que permite a los desarrolladores crear experiencias de usuario únicas y atractivas.
- **Rápido Desarrollo de Prototipos**: Flutter facilita el desarrollo rápido de prototipos y la iteración en el diseño de la aplicación.
- **Soporte para Plataformas Múltiples**: Con Flutter, puedes desarrollar una aplicación que se ejecute en múltiples plataformas sin necesidad de escribir código específico de plataforma.
- **Comunidad Activa y Extensiones**: Flutter cuenta con una comunidad activa de desarrolladores y una amplia gama de extensiones y paquetes que pueden ayudarte a agregar funcionalidad adicional a tus aplicaciones.

2.10.1.5. PostgreSQL

PostgreSQL es uno de los servidores de bases de datos más avanzados distribuidos como software libre que se destaca por su robustez, escalabilidad y

flexibilidad. El término avanzado puede describir tanto el conjunto de funciones disponibles como la capacidad de grabar código fuente e incluso la estabilidad y calidad de ejecución software. Además del código fuente, es el proyecto completo, la forma en que se desarrolla, se toman decisiones, así como un grupo de desarrollo que puede describirse como abierto. Este es uno de los puntos importantes para juzgar la calidad del software, todas las decisiones que afectan a la adición de una funcionalidad o el rechazo de una modificación están disponibles para todos (Lardiere, 2019).

PostgreSQL es una herramienta diseñada para almacenar y procesar datos. Una de las cuestiones importantes a la hora de elegir un sistema de gestión de bases de datos es el tamaño del volumen aceptado por el software. En general, PostgreSQL se basa en gran medida en capacidades del sistema operativo y por lo tanto no impone restricciones de volumen. Sin embargo, hay algunas excepciones:

- El tamaño de un campo está limitado a 1 gigabyte. En la práctica, este tamaño queda limitado por la cantidad de memoria RAM disponible para leer este campo.
- El tamaño de un registro está limitado a 1,6 terabytes.
- El tamaño de una tabla está limitado a 32 terabytes.
- El número máximo de columnas en una tabla puede ir desde 250 hasta 1600, según el tamaño de un bloque de datos y el tamaño de los tipos de datos utilizados para las columnas.

Características principales

- **Modelo de Datos Relacional:** PostgreSQL se basa en el modelo de datos relacional, lo que significa que organiza los datos en tablas con filas y columnas. Esto permite una estructura de datos clara y coherente.
- **Transacciones ACID:** PostgreSQL garantiza la integridad de los datos mediante el cumplimiento de propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) en las transacciones. Esto asegura que las operaciones de la base de datos sean fiables y consistentes.

- **Extensibilidad:** PostgreSQL es altamente extensible. Los usuarios pueden crear tipos de datos personalizados, funciones almacenadas, operadores y agregados para satisfacer las necesidades específicas de la aplicación.
- **Lenguaje de Programación Incorporado:** PostgreSQL admite varios lenguajes de programación para crear funciones almacenadas, incluyendo PL/pgSQL, PL/Python, PL/Java y más. Esto permite la ejecución de lógica de negocio directamente en la base de datos.
- **Índices y Optimización de Consultas:** PostgreSQL ofrece una variedad de técnicas de optimización de consultas, incluyendo el uso de índices, para acelerar el acceso a los datos y mejorar el rendimiento de las consultas.
- **Herencia de Tablas:** PostgreSQL permite la herencia de tablas, lo que significa que una tabla puede heredar propiedades y columnas de otra. Esto es útil para crear estructuras de datos jerárquicas o implementar particiones de datos.
- **Replicación y Alta Disponibilidad:** PostgreSQL admite la replicación, lo que permite tener copias de la base de datos en servidores secundarios para garantizar la alta disponibilidad y la recuperación ante desastres.
- **Soporte de JSON y NoSQL:** PostgreSQL ofrece soporte nativo para datos JSON y documentos semiestructurados. Esto permite trabajar con datos NoSQL en un entorno relacional.
- **Licencia de Código Abierto:** PostgreSQL se distribuye bajo la Licencia de PostgreSQL, una licencia de código abierto que permite su uso, modificación y distribución de forma gratuita.

2.10.1.6. Firebase

Es una plataforma de desarrollo de aplicaciones móviles y web desarrollada por Google. Ofrece una amplia variedad de herramientas y servicios que ayudan a los desarrolladores a crear aplicaciones de alta calidad de manera más rápida y eficiente. Firebase se utiliza comúnmente para desarrollar aplicaciones móviles y web en tiempo real, aplicaciones para dispositivos móviles, aplicaciones web progresivas (PWAs) y más. Como menciona Carbonell et al. (2019) Firebase proporciona una variedad de

herramientas y servicios que permiten a los desarrolladores crear aplicaciones de alta calidad de manera más eficiente.

- **Base de Datos en Tiempo Real:** Firebase Realtime Database es una base de datos en tiempo real que permite a las aplicaciones almacenar y sincronizar datos en tiempo real entre clientes y servidores. Es útil para aplicaciones colaborativas y en tiempo real, como aplicaciones de chat.
- **Autenticación de Usuarios:** Firebase Authentication facilita la autenticación de usuarios a través de métodos como correo electrónico/contraseña, autenticación con Google, Facebook, Twitter y más.
- **Almacenamiento en la Nube:** Firebase Storage permite a los desarrolladores almacenar y servir contenido multimedia, como imágenes y videos, en la nube de manera segura y escalable.
- **Hosting Web:** Firebase Hosting es un servicio de alojamiento web que permite a los desarrolladores implementar fácilmente sitios web estáticos y aplicaciones web en una infraestructura de alta velocidad.
- **Mensajería en la Nube:** Firebase Cloud Messaging (FCM) permite el envío de notificaciones push a dispositivos móviles y navegadores web, lo que permite la interacción con usuarios en tiempo real.
- **Analytics y Monitoreo de Aplicaciones:** Firebase Analytics proporciona información detallada sobre el comportamiento del usuario en la aplicación, lo que ayuda a los desarrolladores a tomar decisiones informadas. Firebase Performance Monitoring permite el seguimiento del rendimiento de la aplicación.
- **Funciones en la Nube:** Firebase Cloud Functions permite a los desarrolladores ejecutar código personalizado en la nube en respuesta a eventos en la aplicación.
- **Pruebas A/B y Experimentos:** Firebase ofrece herramientas para realizar pruebas A/B y experimentos en la aplicación para optimizar la experiencia del usuario.
- **Machine Learning:** Firebase ML Kit proporciona capacidades de aprendizaje automático (machine learning) para agregar funcionalidad de

visión, reconocimiento de texto y otras características de inteligencia artificial a las aplicaciones.

2.10.1.7. Servidor NodeJS

Node.js es un servidor web que se crea utilizando Node.js, que es un entorno de tiempo de ejecución de JavaScript en el lado del servidor. Node.js permite a los desarrolladores construir aplicaciones de servidor altamente escalables y eficientes utilizando JavaScript, un lenguaje de programación ampliamente conocido y utilizado en el lado del cliente. NodeJS proporciona un entorno de ejecución para un lenguaje específico programación y un conjunto de bibliotecas básicas, o módulos nativos, de ellas para crear aplicaciones orientadas principalmente a las comunicaciones en red, aunque algunas de estas bibliotecas permiten la interacción con componentes del sistema operativo (Morales, 2023).

2.10.2. Arquitectura MVC

El patrón de arquitectura MVC es una propuesta de arquitectura de software utilizada para separar código para sus diferentes responsabilidades, manteniendo diferentes niveles de responsabilidad que realiza una tarea muy específica, ofreciendo una amplia gama de beneficios, inicialmente utilizado en sistemas que requieren el uso de una interfaz de usuario, aunque en la práctica el mismo patrón arquitectónico se puede utilizar para diferentes tipos de aplicaciones. Esto surgió de la necesidad de crear software más robusto y con ciclos de vida más largos. apropiado, y la facilidad de mantenimiento, la reutilización del código y separación de conceptos (Hernandez, 2021).

Modelo (Model)

- El Modelo representa los datos y la lógica de negocio de la aplicación.
- Se encarga de acceder y gestionar los datos, realizar cálculos, validaciones y actualizaciones.
- No tiene conocimiento de la interfaz de usuario ni de cómo se presentan los datos al usuario.

Vista (View)

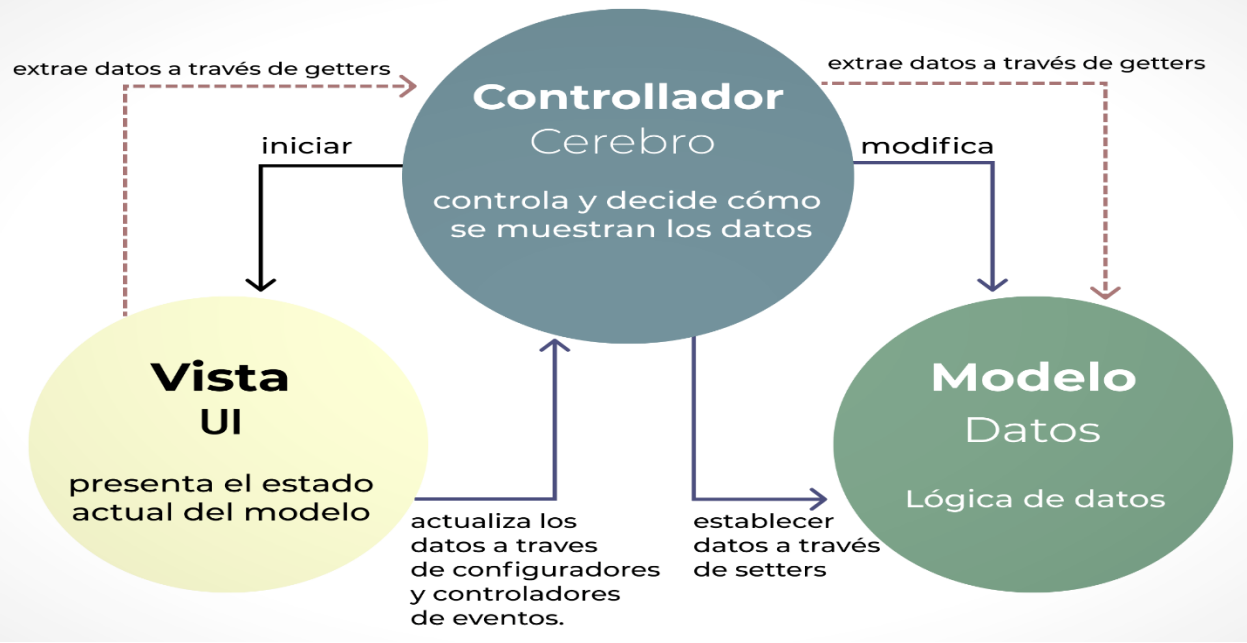
- La Vista es la parte de la aplicación que se encarga de la presentación de datos al usuario.
- Muestra la información al usuario de una manera comprensible y atractiva.
- No contiene lógica de negocio y se actualiza cuando el Modelo cambia.

Controlador (Controller)

- El Controlador actúa como intermediario entre el Modelo y la Vista.
- Responde a las interacciones del usuario y gestiona las solicitudes entrantes.
- Se encarga de enviar comandos al Modelo para actualizar los datos y a su vez, actualiza la Vista.

Figura 12

Patrones de arquitectura MVC



Nota. Patrones de arquitectura MVC. Tomada de El patrón modelo-vista-controlador: Arquitectura y Frameworks explicados (Hernandez, 2021).

2.11. Métricas de Calidad

Estas métricas son fundamentales para comprender y mejorar la calidad en diversos contextos, como el desarrollo de software, la fabricación de productos, la gestión de proyectos, la atención al cliente y más. Las métricas de calidad ayudan a identificar problemas, tomar decisiones informadas y realizar mejoras continuas. De acuerdo a la definición del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE, 2014), “La métrica de calidad del software es una función que asigna una unidad de software en un valor numérico. Este valor calculado se puede interpretar como el grado de cumplimiento de una propiedad de calidad de la unidad de software”. Las métricas de producto estáticas son utilizadas para medir el diseño del software, el programa real o la documentación que describe la función. Los criterios esenciales de la medición incluyen factores como la complejidad, pero también el mantenimiento y las métricas de producto dinámicas como expresión de la confiabilidad y el rendimiento de un producto de software. Los criterios métricos típicos son la eficiencia en la ejecución y el alcance de los errores que ocurrieron en el curso de la ejecución. Las métricas de proceso iluminan las características del proceso de desarrollo en la creación de software (Garcia, 2021).

2.11.1. La Norma ISO/IEC 9126

Según la Organización Internacional de Normalización y la Comisión Electrónica Internacional, (ISO/IEC 9126-1, 2001) esta norma permite especificar y evaluar la calidad de software desde diferentes criterios, establece un modelo de calidad. Este estándar propone un modelo de calidad que se divide en tres vistas: interior, exterior y en uso. Antes de la creación de esta norma existían varias normas ISO de calidad de software, incluidas ISO/IEC 9001:2000 y la ISO/IEC 12207:1995, que buscaban introducir un enfoque basado en procesos para mejorar la eficacia de un sistema de gestión de calidad y definir los procesos del ciclo de vida del software respectivamente (Verity, 2022).

El modelo de calidad de software mencionado Pressman (2010) establece seis atributos importantes de la calidad de software:

- **Funcionalidad.** Grado en el que el software satisface las necesidades planteadas según las establecen los atributos siguientes: adaptabilidad, exactitud, interoperabilidad, cumplimiento y seguridad.
- **Confiabilidad.** Cantidad de tiempo que el software se encuentra disponible para su uso, según lo indican los siguientes atributos: madurez, tolerancia a fallas y recuperación.
- **Usabilidad.** Grado en el que el software es fácil de usar, según lo indican los siguientes suba tributos: entendible, aprensible y operable.
- **Eficiencia**. Grado en el que el software emplea óptimamente los recursos del sistema, según lo indican los suba tributos siguientes: comportamiento del tiempo y de los recursos.
- **Facilidad de recibir mantenimiento**. Facilidad con la que pueden efectuarse reparaciones al software, según lo indican los atributos que siguen: analizable, cambiable, estable, susceptible de someterse a pruebas.
- **Portabilidad.** Facilidad con la que el software puede llevarse de un ambiente a otro según lo indican los siguientes atributos: adaptable, instalable, conformidad y sustituible.

2.12. Ingeniería de Costos

2.12.1. COCOMO II

Como menciona Sommerville (2011) para la estimación de costos, se debe tomar en cuenta el tamaño del sistema y los elementos que necesita para el desarrollo del sistema, cantidad de integrantes en el equipo, esfuerzo para desarrollar el sistema. El modelo mencionado tiene en cuenta las características actuales para el desarrollo de software, como el desarrollo temprano utilizando lenguajes dinámicos, el desarrollo a través de la composición de componentes y el uso de programación de bases de datos.

De acuerdo con Sommerville (2011) menciona los submodelos y sus características:

- **Un modelo de composición de aplicación:** Se crean a partir de componentes reutilizables y programación de base de datos, para determinar el esfuerzo requerido. Para calcular el esfuerzo requerido se usa una simple formula, tamaño sobre productividad. La estimación del tamaño de un sistema se mide de acuerdo al número de pantallas que se muestran, numero de informes generadas, cantidad de módulos y líneas de código.
- **Un modelo de diseño temprano:** Este modelo se usa después de establecer los requerimientos. Las estimaciones se basan en puntos de función, luego se convierten a número de líneas de código fuente. La cantidad de función en un programa se calcula al medir la cantidad de ingreso y salidas externas, las interacciones de usuario, las interfaces externas y la base de datos que usa el sistema.
- **Un modelo de reutilización:** Este modelo se emplea al integrar los componentes de código generado automáticamente.
- **Un modelo postarquitectónico:** Con este modelo se realiza una estimación exacta del tamaño del software, dicho modelo usa la formula estándar para estimación de costo, que incluye 17 multiplicadores que reflejan características de capacidad de personal, software y del proyecto.

Teniendo en cuenta a Pressman (2010) el modelo para estimación de costos COCOMO II requieren información sobre el tamaño del software. Como parte de la jerarquía del modelo, existen tres diferentes opciones de dimensionamiento: puntos objeto, puntos de función y líneas de código fuente. El punto de objeto es una medida de software indirecta que se calcula usando conteos de cantidad de ventanas en la interfaz de usuario, reportes y componentes en caso que requieran para la aplicación. Cada instancia se clasifica por complejidad como simple, medio y difícil. Una vez determinada el número de pantallas, reportes y componentes se pondera la complejidad. Se determina el conteo de puntos de objeto multiplicando el número original de instancias del objeto por el factor de ponderación.

$$NOP = OP \text{ x } [(100 - \%reuso) / 100]$$

donde NOP se define como nuevo punto de objeto. Para la tasa de productividad se calcula con el valor

$$\text{NOP } PROD = NOP/persona - mes$$

Una vez teniendo el valor de la tasa de productividad se calcula el esfuerzo del proyecto usando

$$esfuerzo\ estimado = NOP/PRO$$

2.13. Seguridad ISO/IEC 27000

Según la Organización Internacional de Normalización y la Comisión Electrónica Internacional. La norma 27000, están orientadas al establecimiento de buenas prácticas en relación con la implantación, mantenimiento del Sistemas de Gestión de Seguridad de la Información (SGSI). El modelo es el resultado del consenso entre especialistas, considerado el estado del arte en lo que se refiere a la estandarización para el segmento de seguridad de la información. El pilar de la serie ISO 27000 es ISO 27001:2013, que especifica todos los requisitos según los cuales se puede auditar y certificar el sistema de gestión de seguridad de la información. Todos los demás estándares de la familia ISO 27000 son códigos de conducta no interactivos que brindan pautas de mejores prácticas que las organizaciones pueden seguir en su totalidad o en parte y que respaldan la ISO 27001 (ESGInnova, 2023).

2.13.1. Sistema de Gestión de la Seguridad de la Información (SGSI)

Con la ISO 27001 se ha demostrado que la implementación de medidas y procedimientos de seguridad, muchas veces realizados sin un criterio general establecido, en torno a la compra de productos técnicos y sin tener en cuenta toda la información importante que debe protegerse, es insuficiente.

La Organización Internacional de Estandarización (ISO, 2018) a través de los estándares contenidos en ISO/IEC 27000, garantiza la implementación efectiva de la seguridad de la información empresarial desarrollada en los estándares ISO 27001/ISO 27002. Los requisitos de la norma ISO 27001 nos proporcionan un sistema de gestión de seguridad de la información (SGSI) compuesto por medidas

encaminadas a proteger la información, independientemente de su formato, de cualquier amenaza, permitiéndonos asegurar en todo momento la continuidad de las actividades de la empresa.

Los Objetivos del SGSI son preservar la:

- Confidencialidad
- Integridad
- y Disponibilidad de la Información

2.14. Pruebas de Software

Según Pressman (2010) define que la prueba de software es un conjunto de actividades que pueden planearse por adelantado y realizarse de manera sistemática, para identificar defectos, errores o problemas en el software antes de que llegue a los usuarios finales. Estas pruebas son esenciales para garantizar la calidad y confiabilidad del software. Por otro lado, define las pruebas de software como: El proceso que consiste en todas las actividades del ciclo de vida, tanto estáticas como dinámicas relacionadas con la planificación, preparación y evaluación de productos de software y productos relacionados con el trabajo para determinar que cumplen los requisitos especificados, para demostrar que son aptos para el propósito y detectar defectos.

De acuerdo con el Comité Internacional de Certificaciones de Pruebas de Software (ISTQB, 2022) el proceso de pruebas tiene los siguientes objetivos:

- Identificar defectos
- Aumentar la confianza en el nivel de calidad
- Facilitar la información para la toma de decisiones
- Evitar la aparición de defectos

2.14.1. Pruebas de Caja Blanca

Las pruebas de caja blanca implican analizar la codificación, la estructura y la arquitectura del programa de software para garantizar que los datos fluyan de la

entrada a la salida. El diseño la usabilidad y la seguridad de las aplicaciones se pueden mejorar mediante el uso de pruebas de caja blanca. Estos métodos también se denominan pruebas basadas en código, pruebas de caja abierta, pruebas de caja transparente y también se conocen como pruebas de caja de vidrio y pruebas de caja abierta. Las pruebas de caja blanca son integrales, a diferencia de las pruebas de caja negra, que enfatizan una experiencia de usuario perfecta. Las pruebas de caja blanca se utilizan para encontrar fallas internas ocultas y mejorar el código. Sin embargo, las pruebas de caja blanca están reservadas sólo para las partes más vitales de una aplicación (Sandeep, 2023).

2.14.2. Pruebas de Caja Negra

En las pruebas de caja negra se examina el funcionamiento de la aplicación sin tener primero un conocimiento profundo de su arquitectura y diseño internos. Durante la prueba se comparan el valor de entrada y el valor de salida. Debido a su naturaleza, las pruebas de caja negra también se conocen como pruebas basadas en especificaciones, pruebas de caja cerrada o pruebas de caja opaca. Incluye una amplia gama de casos de prueba que le permiten encontrar la mayoría de los errores. Esta técnica de prueba se utiliza durante todo el proceso de desarrollo de software (Sandeep, 2023).

Las pruebas de caja negra se centran principalmente en un análisis integral del desempeño del programa. Estas pruebas intentan encontrar errores en las siguientes categorías:

- Funciones incorrectas o faltantes
- Errores de interfaz
- Fallas en las estructuras de datos o en el acceso a bases de datos externos
- Omisión de comportamiento o rendimiento
- Errores de inicialización y terminación

2.14.3. Tipos de Pruebas

ISTQB (2022) define tipos de pruebas y sus características:

2.14.3.1. Pruebas Funcionales

Las pruebas funcionales ayudan a verificar el resultado final de una aplicación e identificar cualquier omisión en los requisitos de la aplicación, comportamiento no conforme o comportamiento inferior al esperado. ISTQB (2022) incluye en su clasificación las pruebas funcionales, las cuales se consideran importantes por su función para que la empresa alcance sus objetivos.

La funcionalidad se divide en las siguientes características:

- Completitud funcional: el grado en el que las funcionalidades cubren todas las tareas y objetivos del usuario especificados
- Corrección funcional: capacidad de la aplicación para proveer resultados correctos con el nivel de precisión requerido.
- Pertenencia funcional: capacidad del producto de software para proporcionar un conjunto apropiado de funciones para tareas y objetivos de usuario especificado.

2.14.3.2. Pruebas no Funcionales

Según Sandeep (2023) Evalúan las características de los sistemas y el software, como la usabilidad, la eficiencia operativa o la seguridad. Estas pruebas se utilizan para determinar la estabilidad de un sistema de software bajo varios niveles de carga y estrés para determinar si puede soportar escenarios desafiantes. Se han de tener en cuenta las siguientes características no funcionales en las pruebas:

- Pruebas de rendimiento: consiste en desarrollar estrategias eficaces para mejorar el rendimiento del sistema. En dicha prueba se recogen y analiza información mediante un proceso de medición, luego predecir cuándo los niveles de carga agotarán los recursos del sistema.

CAPITULO III

MARCO APLICATIVO

3.1. Introducción

Este capítulo es la puesta en marcha de los sustentos teóricos del capítulo anterior y de esta manera llevar a cabo el proyecto con las herramientas necesarias para su crecimiento, se realizará el diseño y desarrollo de la App usando la metodología de desarrollo de software Mobile-D, siendo que es una metodología ágil e implementando el ciclo de crecimiento de sus cinco fases, Fase de Exploración, Fase de Inicialización, Fase de Producción, Fase de Estabilización y Fase de Pruebas para el crecimiento de la dinámico de la aplicación móvil, usando un conjunto de bibliotecas y pautas que facilitan el proceso de desarrollo de la App.

3.2. Fase de Exploración

Esta fase se dedica a establecer el proyecto y marcar la planificación con una base sólida para el desarrollo de la aplicación móvil y garantizar que la solución propuesta cumpla con las expectativas y necesidades de los usuarios. Durante esta etapa, se establecen los requerimientos, se investiga el mercado y se crea un plan detallado para el desarrollo y la implementación de la aplicación móvil.

3.2.1. Análisis de Requerimientos

El análisis de requerimientos es fundamental para el éxito del proyecto de desarrollo de software, ya que sienta las bases para el análisis, diseño, implementación y las pruebas del software, y el comportamiento actual de la empresa, donde se identificó a los actores que interactúan con los requerimientos de la App lo primero es enfocarse en modelo de negocio y luego en los casos de uso de negocio.

3.2.2. Descripción de la Situación Actual

La empresa de comercialización de productos textiles Telaza se dedica a la venta de productos al por mayor y al por menor. Opera en un entorno altamente competitivo y tiene un catálogo diversos productos textiles que incluye una variedad de colores y diseños. Se ha identificado la necesidad de mejorar su proceso de distribución y entrega de productos textiles. Actualmente, la empresa enfrenta desafíos

en la gestión de pedidos, el seguimiento de entregas y la satisfacción del cliente en cuanto a los tiempos de entrega.

3.2.3. Requerimientos de la App

La ingeniería de requerimientos es el proceso que posibilita la identificación de los servicios esenciales y las limitaciones que componen un Aplicación o App. De igual manera, se dedica a llevar a cabo actividades con el propósito de comprender a fondo las necesidades precisas de los usuarios y definir las funcionalidades y operaciones que serán beneficiosas para el desarrollo de la App.

A continuación, en la recolección de datos de las entrevistas realizada a los clientes de la empresa Telaza, detallamos algunas de las especificaciones del requerimiento del usuario:

- Facilidad de Uso UI/UX
- Catálogo de productos
- Autenticación y Seguridad
- Compatibilidad
- Rendimiento y Velocidad
- Personalización
- Disponibilidad en tiempo real
- Actualizaciones y Mantenimiento
- Interacción Social
- Historial de pedidos

3.2.4. Actores Interesados

Luego de realizar un estudio preliminar sobre la situación actual de la empresa, se pudo observar e identificar a los actores. Los actores son usuarios de la aplicación que interactúan, proporcionan información y la reciben información de la App.

Figura 13

Mapa de Stakeholders

3.2.5. Requerimientos Funcionales

Estos requerimientos se detallan a continuación:

Tabla 2

Requerimientos funcionales

Ref.	Función	Categoría
Ref. 1	La App debe tener la autenticación de usuarios mediante contraseñas, inicio de sesión con Google, los usuarios deben poder recuperar contraseñas olvidadas.	Evidente

Ref. 2	La App debe mostrar un catálogo de productos textiles con imágenes, descripciones y precios, debe permitir a los usuarios hacer un filtrado por categoría.	Evidente
Ref. 3	Debe permitir a los administradores agregar nuevos productos y actualizar información de productos existentes.	Evidente
Ref.4	La aplicación móvil debe permitir a los usuarios guardar y gestionar sus perfiles. Debe ser posible agregar direcciones de entrega en los perfiles de cliente.	Evidente
Ref.5	La aplicación móvil debe permitir a los usuarios agregar productos textiles al carrito de compras. Los usuarios deben poder ver un resumen de sus pedidos antes de confirmarlos.	Evidente
Ref. 6	Los usuarios deben poder ver el estado de sus pedidos en tiempo real. La App debe proporcionar notificaciones sobre el estado de los pedidos, como confirmación de pedido, envío y entrega.	Evidente
Ref. 7	Los usuarios deben poder acceder a un historial completo de sus pedidos anteriores. Debe ser posible ver detalles de los pedidos anteriores, incluidos productos y precios.	Evidente
Ref. 8	Cierre de la App, la aplicación móvil deberá ser capaz de cerrar sesión.	Oculto

3.2.6. Requerimientos no Funcionales

Estos requerimientos se detallan a continuación:

Tabla 3

Requerimientos no funcionales

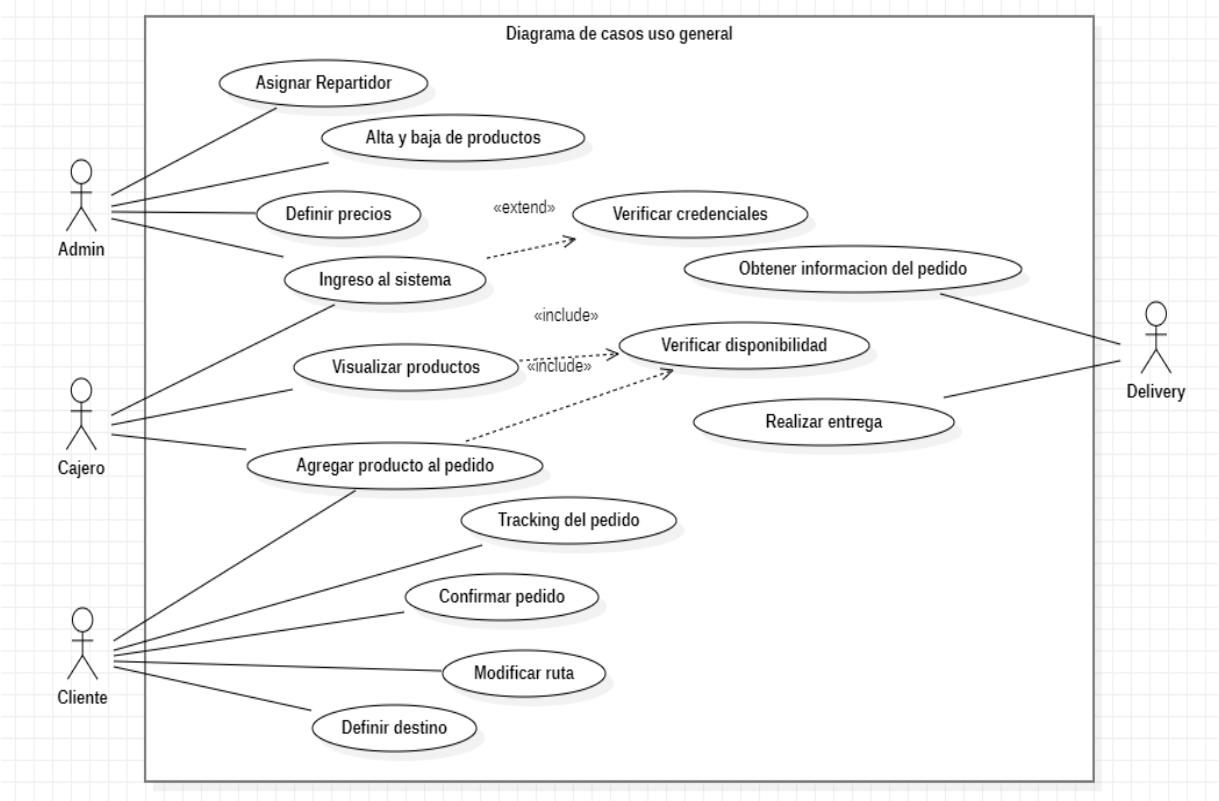
Ref.	Función	Categoría
Ref. 1	La aplicación debe ser rápida y eficiente, incluso durante períodos de alta demanda. Debe tener tiempos de carga rápidos y una experiencia de usuario fluida.	Evidente
Ref. 2	La App debe estar disponible las 24 horas del día, los 7 días de la semana, para que los usuarios puedan visualizar el catálogo de los productos.	Evidente
Ref.3	Debe ser escalable para manejar un aumento en la cantidad de usuarios y pedidos a medida que el negocio crece.	Evidente
Ref. 4	La aplicación debe ser compatible con una variedad de dispositivos móviles. Debe funcionar bien en diferentes tamaños de pantalla y resoluciones.	Evidente
Ref. 5	Debe ser capaz de actualizar al momento de sincronizar con la APIREST cuando se recupere la conexión.	Evidente

3.2.7. Diagrama de Casos de Uso

Un diagrama de casos de uso es una herramienta de modelado donde se utiliza para representar las interacciones entre la aplicación y sus actores. Estos diagramas son parte de la notación UML (Unified Modeling Language) y se utilizan para visualizar y documentar los requisitos funcionales de un sistema desde una perspectiva centrada en el usuario.

Figura 14

Diagrama de casos de uso general



Modulo principal: En este diagrama se muestra toda la interacción entre los usuarios y sus peticiones y las diferentes opciones que muestra la aplicación.

Tabla 4

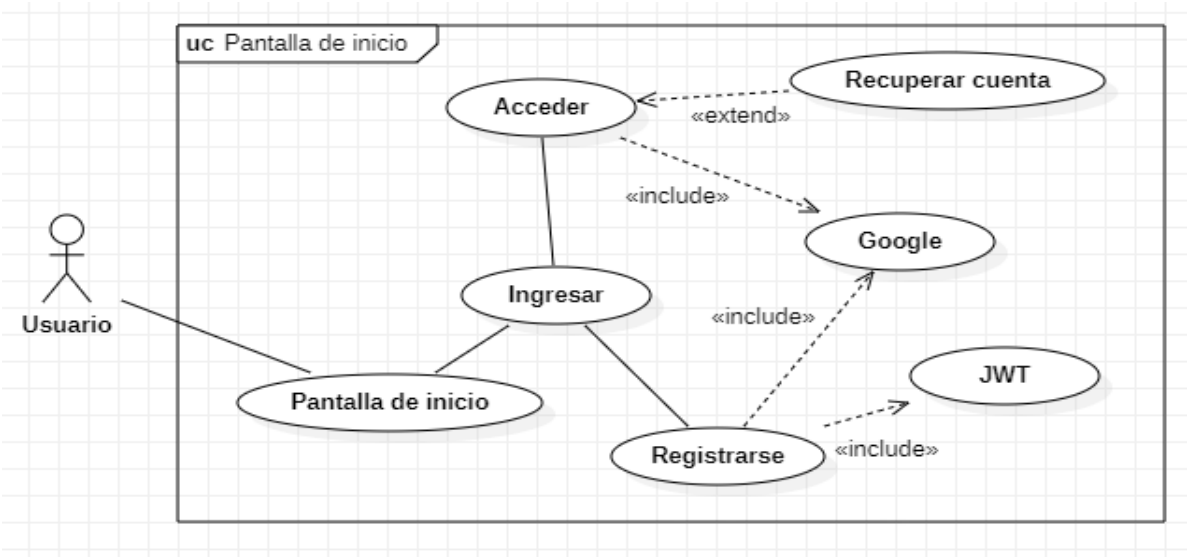
Caso de uso pantalla de inicio

Caso de uso	Pantalla de Inicio	
Actores	Usuarios (Cajero, Cliente y Delivery)	
Descripción	Este caso de uso representa la interacción del usuario con la pantalla de inicio o bienvenida de la aplicación.	
Flujo principal	**Eventos Actor**	**Eventos App**

	El usuario abre la aplicación en su dispositivo e interactúa con la pantalla de inicio.	Si el usuario no puede iniciar sesión debido a credenciales incorrectas, la aplicación muestra un mensaje de error. Si el usuario no tiene una cuenta y desea registrarse, se puede iniciar un flujo de registro.
Precondición	El usuario ha instalado la aplicación en su dispositivo. El usuario ha abierto la aplicación en su dispositivo.	
Postcondición	El usuario ha realizado una acción en la pantalla de inicio, donde selecciona iniciar sesión o registrarse. El usuario es redirigido a la pantalla o página correspondiente según la acción realizada.	

Figura 15

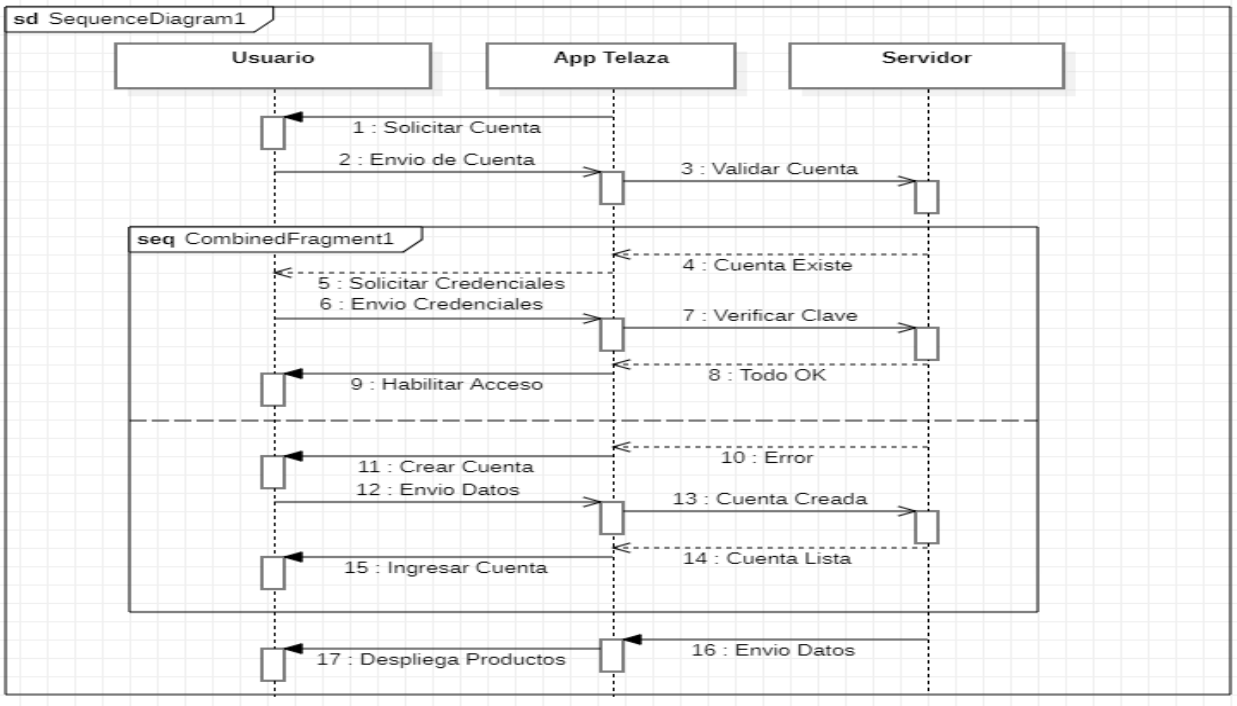
Diagrama de caso de uso pantalla de Inicio



3.2.8. Diagrama de Secuencia

Figura 16

Diagrama de secuencia ingreso a la aplicación



Modulo del administrador: Diseñado para ser utilizada por usuarios con privilegios elevados o especiales. Este módulo brinda acceso a funcionalidades y herramientas que permiten la gestión y supervisión de la aplicación en su conjunto, así como la administración de usuarios, contenido y configuración.

Tabla 5

Caso de uso administración de usuarios

Caso de uso	Administración de usuarios
Actores	Usuario (Administrador)
Descripción	Este caso de uso describe las acciones que un administrador de la aplicación puede realizar para gestionar las cuentas de usuario dentro de la App.

	Eventos Actor	Eventos App
Flujo principal	El administrador puede realizar las siguientes acciones en la cuenta de usuario seleccionada: Asignar roles o privilegios específicos al usuario.	Si el administrador intenta realizar una acción no permitida el sistema muestra un mensaje de error. Si el administrador decide no guardar los cambios realizados en la cuenta de usuario, el sistema mantiene el estado anterior de la cuenta.
Precondición	El administrador ha iniciado sesión en la aplicación con sus credenciales y la App muestra la interfaz de administración.	
Postcondición	Los cambios realizados por el administrador se aplican a las cuentas de usuario según las acciones específicas realizadas.	

Figura 17

Diagrama de caso de uso Administración de usuarios

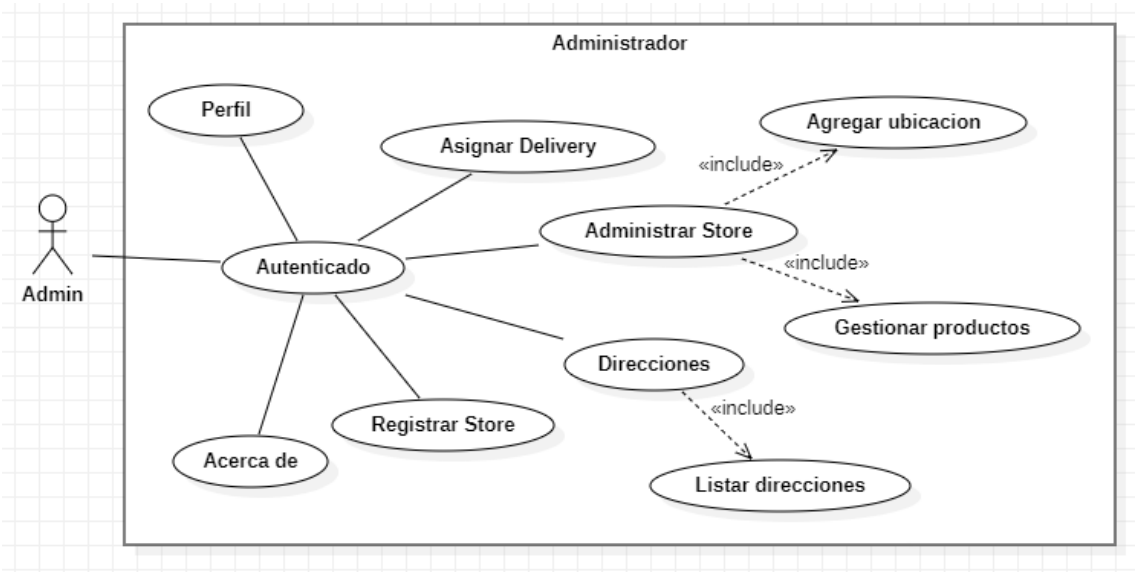
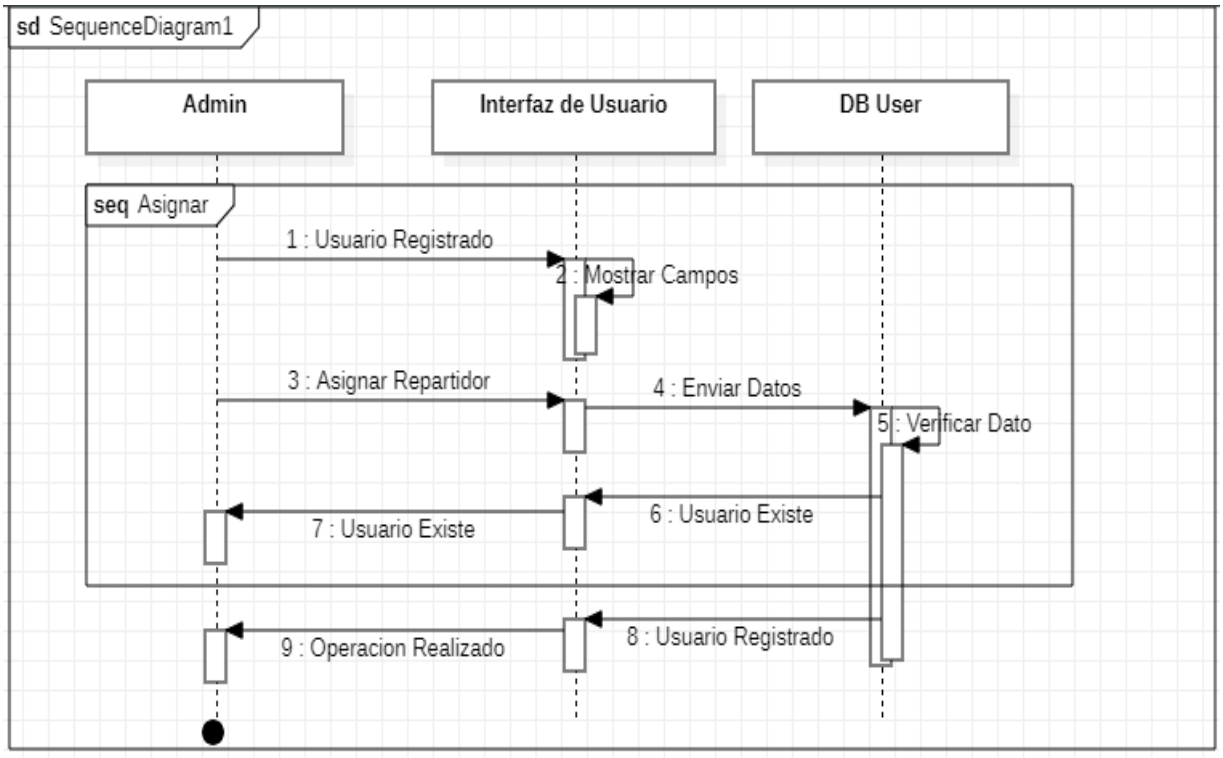


Figura 18

Diagrama de secuencia Asignar Repartidor



Modulo del cajero: El módulo de cajero desempeña un papel fundamental en la experiencia de los usuarios y en el funcionamiento general de una aplicación de entrega. Asegura que los productos vendidos reflejen un historial y que los pedidos se procesen de manera eficiente, lo que contribuye a la satisfacción del cliente y al éxito del negocio.

Tabla 6

Caso de uso cajero

Caso de uso	Registro de venta, pedido por cajero
Actores	Usuario (Cajero)
Descripción	Este caso de uso describe las acciones que un cajero realiza para registrar una venta, pedido en la aplicación móvil.

	Eventos Actor	Eventos App
Flujo principal	La aplicación móvil agrega los productos al catálogo ingresados por el usuario, mostrando el nombre del producto, el precio unitario y una descripción.	La aplicación móvil genera un historial que muestra los detalles de la compra y el monto pagado.
Precondición	El cajero ha iniciado sesión en la aplicación móvil. La aplicación móvil muestra el historial de las ventas, pedidos según su estado.	
Postcondición	La transacción se ha registrado correctamente en la aplicación móvil y en la base de datos. La aplicación móvil muestra el estado actualizado del pedido o venta.	

Figura 19

Diagrama de caso de uso cajero

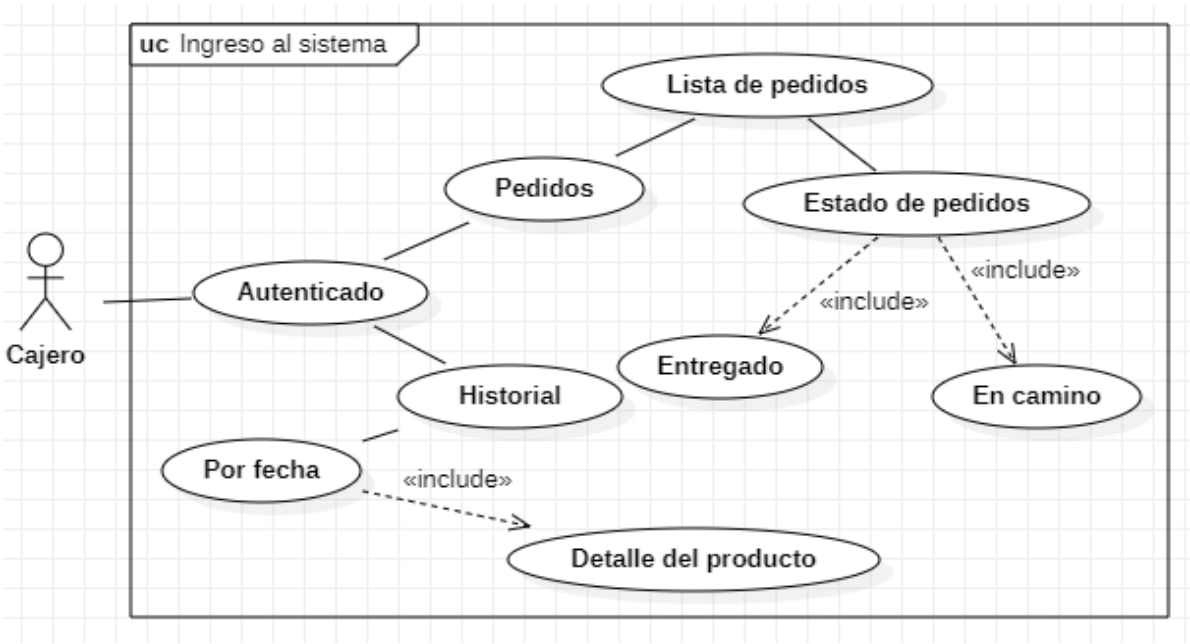
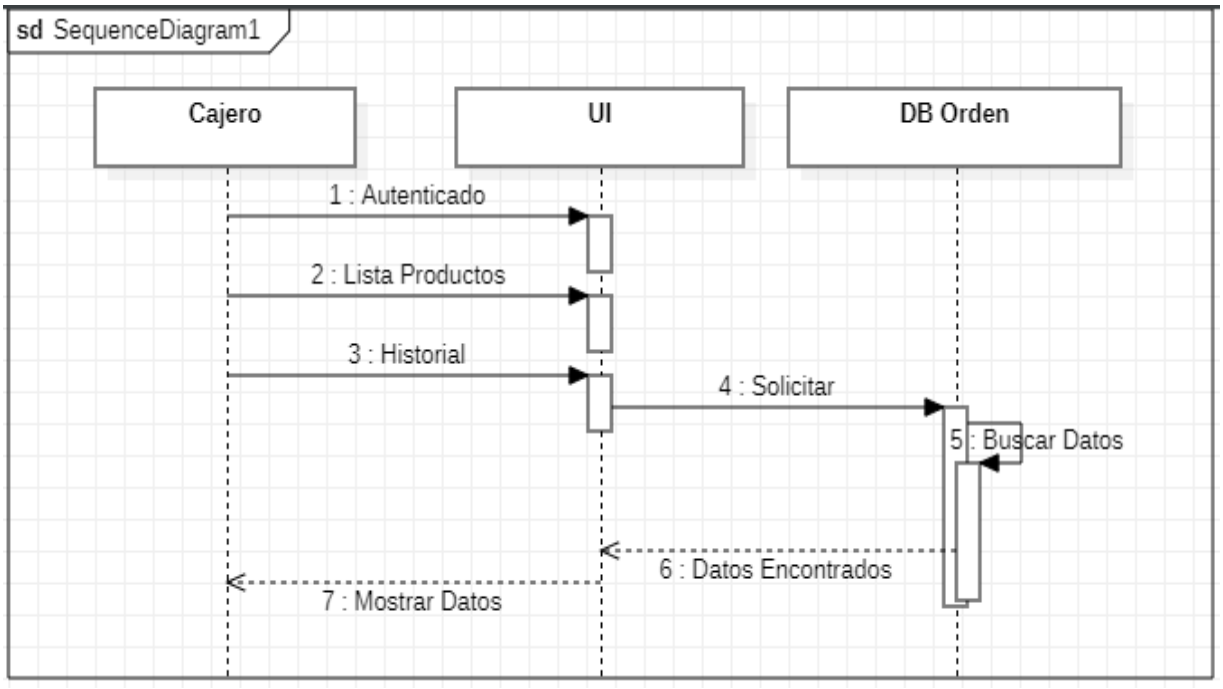


Figura 20

Diagrama de secuencia cajero



Modulo del cliente: Este módulo está diseñado para mejorar la experiencia del cliente y brindar herramientas que permiten la interacción con la App, la obtención de información y la gestión de sus cuentas.

Tabla 7

Caso de uso inicio de sesión

Caso de uso	Inicio de sesión	
Actores	Usuarios (Cajero, Cliente, Delivery)	
Descripción	Este caso de uso describe las acciones que un usuario realiza para acceder a la aplicación mediante el proceso de inicio de sesión.	
Flujo principal	**Eventos Actor**	**Eventos App**

	El usuario abre la aplicación en su dispositivo. La aplicación muestra la pantalla de inicio de sesión, que incluye campos para ingresar dirección de correo electrónico y la contraseña. El usuario ingresa su nombre de usuario y contraseña en los campos correspondientes.	Si el usuario ha olvidado su contraseña, la aplicación puede ofrecer una opción de "Recuperar Contraseña" que permite restablecer la contraseña a través de un proceso de recuperación. Si el usuario ingresa credenciales incorrectas, la aplicación muestra un mensaje de error.
Precondición	El usuario ha instalado la aplicación en su dispositivo. El usuario ha registrado una cuenta en la aplicación con credenciales de acceso.	
Postcondición	El usuario ha iniciado sesión en la aplicación y puede acceder a las funciones y características disponibles para usuarios autenticados. La aplicación mantiene una sesión activa hasta que el usuario elija cerrar sesión o hasta que expire por inactividad.	

Figura 21

Diagrama de caso de uso pantalla de Inicio de sesión

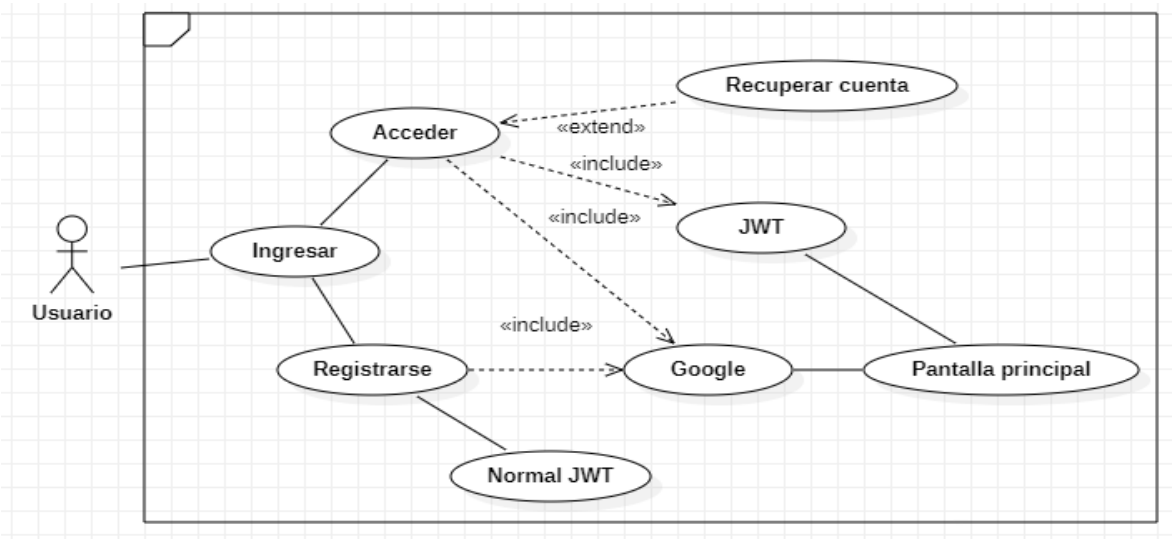


Tabla 8

Caso de uso pantalla principal

Caso de uso	Pantalla principal	
Actores	Usuario (Cliente)	
Descripción	Este diagrama de secuencia describe cómo un usuario interactúa con la aplicación para desplegar la pantalla principal.	
Flujo principal	**Eventos Actor**	**Eventos App**
	El usuario abre la aplicación móvil desde el ícono de la pantalla de inicio de su dispositivo e ingresa con sus credenciales. La aplicación carga y muestra la pantalla principal, donde contiene: Listas de productos. Categorías de productos. Botones o elementos de navegación.	Si el usuario hace clic en un producto o categoría específica, puede navegar a la pantalla de detalles del producto o a la pantalla de categoría correspondiente.
Precondición	El usuario ha iniciado sesión en la aplicación o se encuentra en un estado autenticado. El dispositivo móvil está encendido y tiene conexión a Internet.	
Postcondición	El usuario visualiza la pantalla principal de la aplicación. El usuario puede realizar varias acciones, como: Explorar y buscar productos. Acceder a categorías específicas. Hacer clic en elementos de navegación para acceder a otras partes de la aplicación.	

Figura 22

Diagrama de caso de uso despliegue de pantalla principal

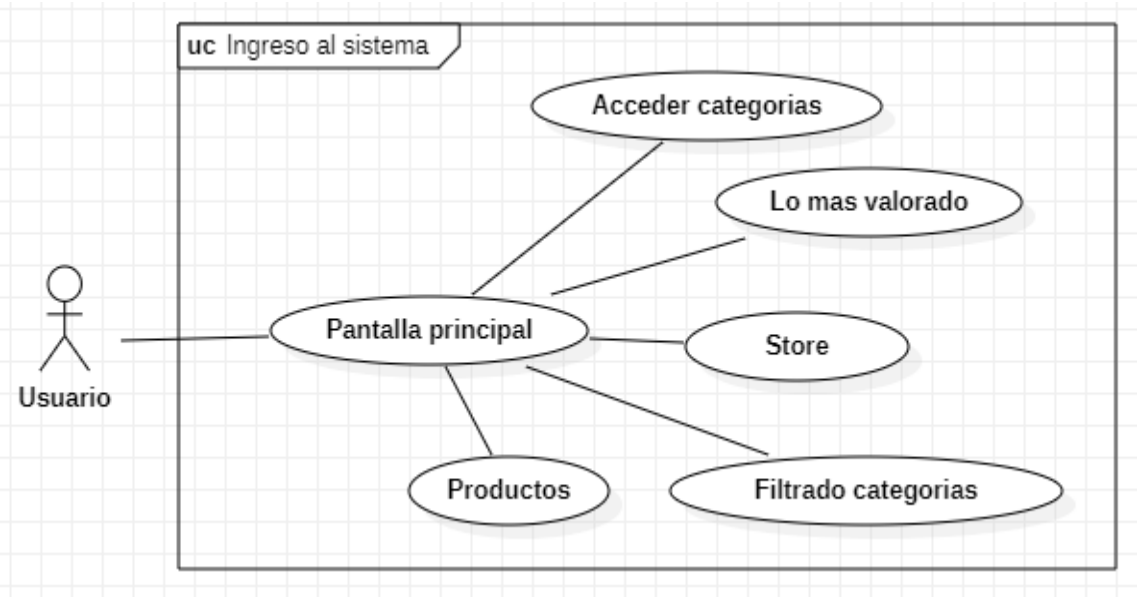


Tabla 9

Caso de uso Store

Caso de uso	Store	
Actores	Usuario (Cliente)	
Descripción	Este caso de uso describe cómo un usuario registrado puede utilizar la funcionalidad de buscar, explorar un store y visualizar sus productos dentro de la aplicación.	
Flujo principal	**Eventos Actor**	**Eventos App**
	La aplicación responde a la acción del usuario y muestra la pantalla de store, que generalmente incluye una lista de tiendas disponibles. Si el	Si el usuario realiza una acción que no es compatible con el acceso a la funcionalidad de store, la aplicación puede ignorar la

	usuario selecciona un store, la aplicación muestra información detallada sobre ese store, que puede incluir nombre de la store, descripción, ubicación, productos disponibles y reseñas de otros usuarios.	acción o mostrar un mensaje de error. Si el usuario selecciona un store y decide realizar una compra o pedido, la aplicación lo guía a través del proceso de compra y finaliza la transacción.
Precondición	El usuario ha iniciado sesión en la aplicación. La aplicación muestra la pantalla principal desde la cual se pueda acceder a la funcionalidad de explorar un store y sus productos.	
Postcondición	El usuario puede ver y explorar las tiendas disponibles. El usuario puede seleccionar una tienda específica para acceder a información detallada sobre esa tienda y sus productos y la gestión de las mismas.	

Figura 23

Diagrama de caso de uso Store

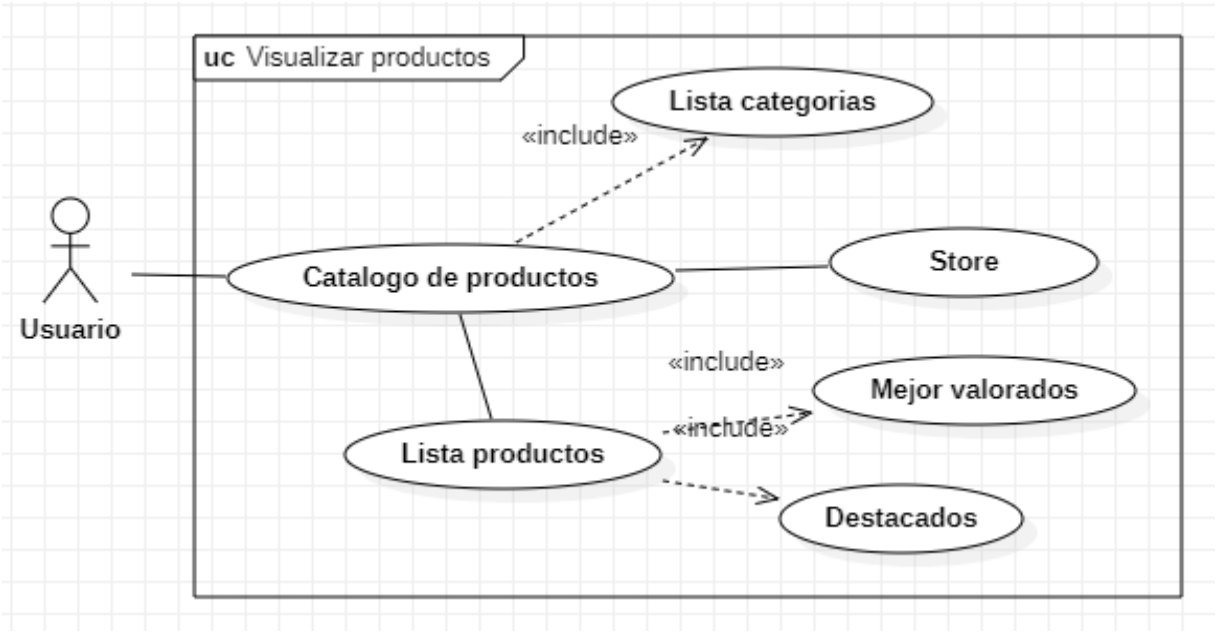


Figura 24

Diagrama de secuencia cliente

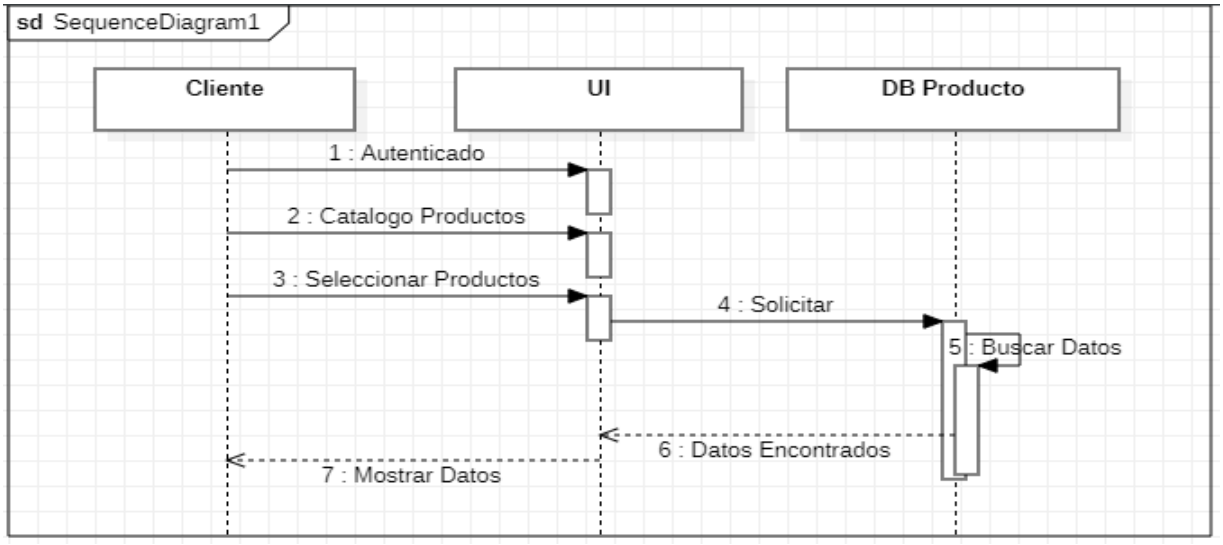


Tabla 10

Caso de uso carrito de compras

Caso de uso	Carrito de compras	
Actores	Usuario (Cliente)	
Descripción	Este caso de uso describe cómo un usuario registrado puede utilizar el carrito de compras en una aplicación para agregar, gestionar y finalizar sus compras o pedidos.	
Flujo principal	**Eventos Actor**	**Eventos App**
	El usuario realiza una acción que indica su intención de acceder al carrito de compras. Esto generalmente se hace tocando un ícono de "Carrito" en la barra de navegación superior de la	Si el usuario realiza una acción que no es compatible con el acceso al carrito de compras, la aplicación puede ignorar la acción y mostrar un mensaje de error. Si el usuario decide no finalizar la compra o pedido

	pantalla. El usuario puede ver y gestionar los productos en el carrito, lo que incluye acciones como aumentar o disminuir la cantidad de un producto, eliminar productos individuales. La aplicación guía al usuario a través del proceso de adquisición, que generalmente incluye ingresar información de envío, información de pago y revisar el pedido.	y abandona el carrito, la aplicación guarda el estado del carrito para futuras sesiones del usuario.
Precondición	El usuario ha iniciado sesión en la aplicación. La aplicación muestra la pantalla principal desde la cual se pueda acceder al carrito de compras. El usuario ha explorado productos y ha seleccionado al menos un producto para comprar.	
Postcondición	La pantalla del carrito de compras se ha desplegado en la pantalla de la aplicación. El usuario puede ver y gestionar los productos en el carrito. Los productos comprados se registran en el historial de pedidos del usuario.	

Figura 25

Diagrama de caso de uso carrito de compras

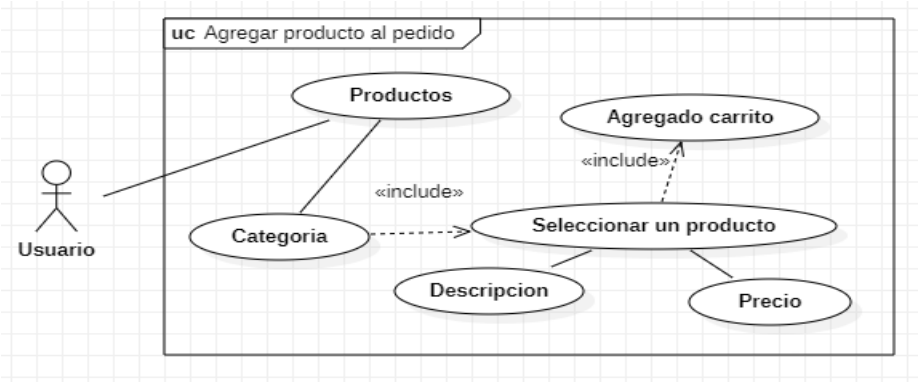


Figura 26

Diagrama de secuencia carrito de compras

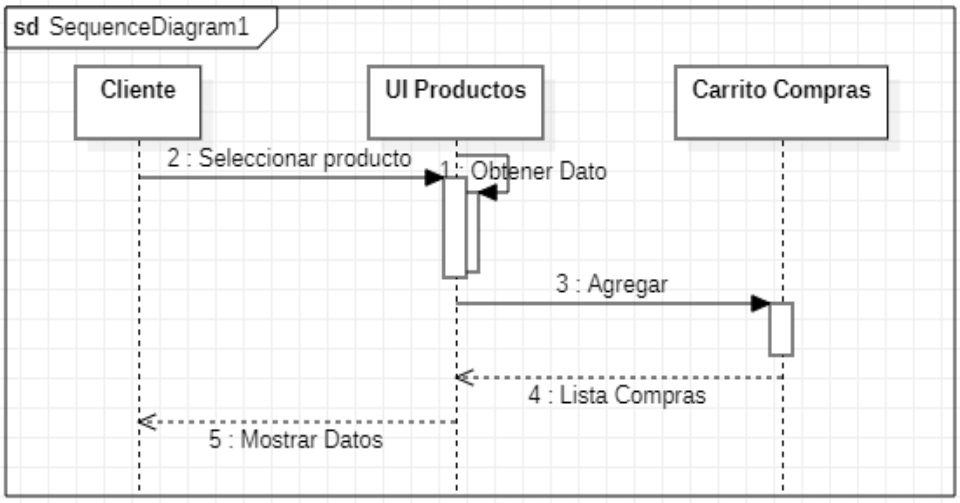


Tabla 11

Caso de uso barra de navegación de pedidos

Caso de uso	Barra de navegación de pedidos	
Actores	Usuario (Cliente)	
Descripción	Este caso de uso describe cómo un usuario registrado puede utilizar la barra de navegación de pedidos en una aplicación de compras para acceder a información y gestionar sus pedidos anteriores.	
Flujo principal	**Eventos Actor**	**Eventos App**
	El usuario realiza una acción que indica su intención de acceder a la barra de navegación de pedidos. Esto generalmente se hace tocando un ícono	Si el usuario realiza una acción que no es compatible con el acceso a la barra de navegación de pedidos, la aplicación puede ignorar la acción y mostrar un mensaje

	correspondiente. El usuario puede seleccionar una de las opciones de la barra de navegación de pedidos para acceder a información específica sobre sus pedidos anteriores.	de error. Si el usuario selecciona un pedido específico y realiza una acción, como solicitar ver la información, la aplicación puede guiar al usuario a través del proceso del estado del pedido.
Precondición	El usuario ha iniciado sesión en la aplicación. La aplicación muestra la pantalla principal desde la cual se pueda acceder a la barra de navegación de pedidos.	
Postcondición	El usuario puede ver y acceder a las opciones disponibles en la barra de navegación de pedidos. El usuario puede acceder a información detallada sobre sus pedidos anteriores y tomar acciones relacionadas con ellos.	

Figura 27

Diagrama de caso de uso listar pedido

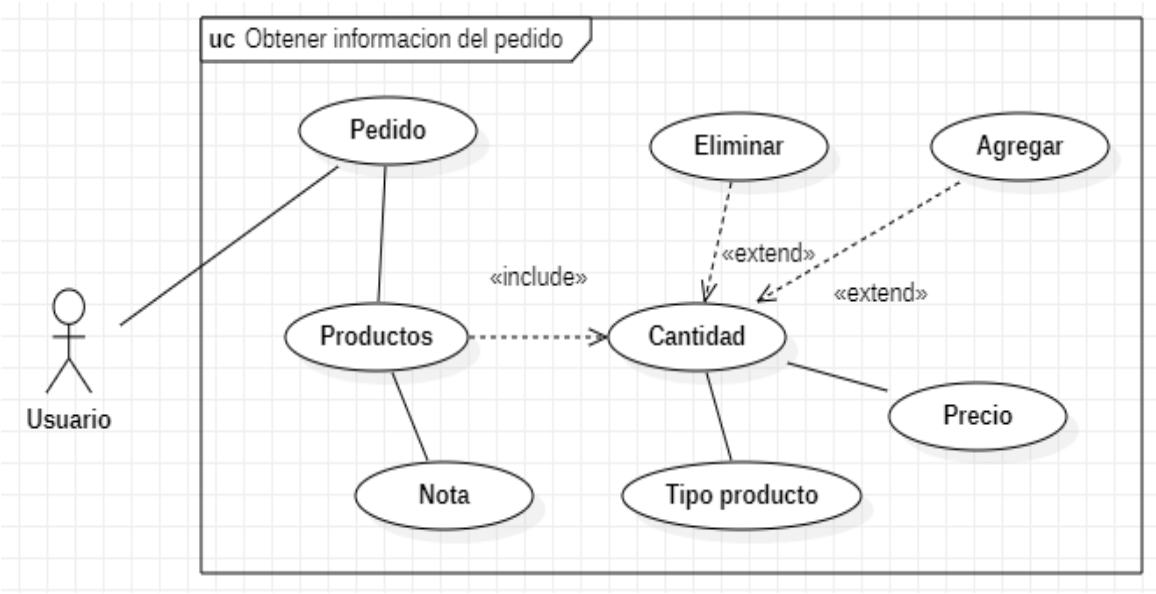


Tabla 12

Caso de uso Side menú

Caso de uso	Side menú (Menú lateral)	
Actores	Usuario (Cliente)	
Descripción	Este caso de uso describe cómo un usuario registrado puede desplegar el menú de su perfil en la aplicación.	
Flujo principal	**Eventos Actor**	**Eventos App**
	El usuario realiza una acción que indica su intención de desplegar el menú de perfil. Esto puede hacerse tocando un ícono de perfil en la parte superior izquierda de la pantalla, deslizando hacia abajo desde la parte superior de la pantalla y seleccionando una opción de menú específica.	Si el usuario hace clic en una opción específica dentro del menú de perfil, la aplicación realiza la acción correspondiente, lo que puede incluir la navegación a otras pantallas. Si el usuario despliega el menú de perfil y luego decide cerrarlo sin realizar ninguna acción adicional, la aplicación devuelve al usuario a la pantalla desde la cual accedió al menú de perfil.
Precondición	El usuario ha iniciado sesión en la aplicación. La aplicación muestra la pantalla principal o una pantalla desde la cual se pueda acceder al perfil del usuario.	
Postcondición	El menú de perfil del usuario se ha desplegado en la pantalla de la aplicación. El usuario puede ver y acceder a las opciones disponibles en el menú de perfil. El menú de perfil permanece visible hasta que el usuario lo cierre o realice una selección en el menú.	

Figura 28

Diagrama de caso de uso Side menú

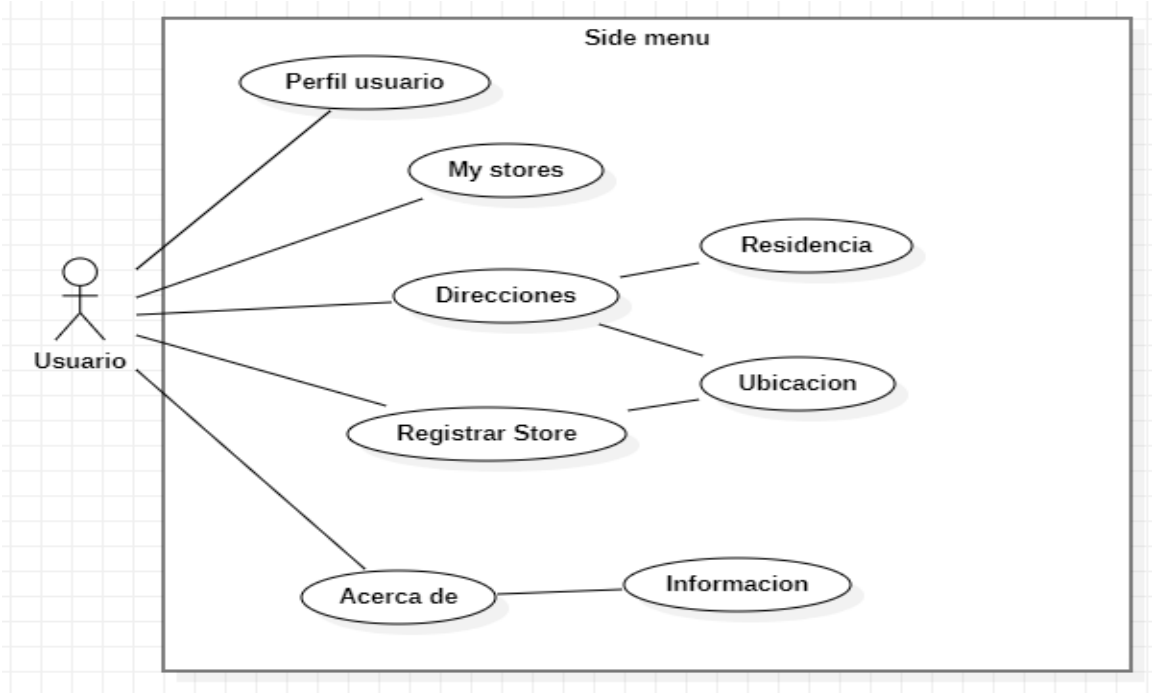


Tabla 13

Caso de uso perfil del usuario

Caso de uso	Perfil de usuario	
Actores	Usuario (Cliente)	
Descripción	Este caso de uso describe cómo un usuario registrado puede actualizar su perfil en la aplicación.	
Flujo principal	**Eventos Actor**	**Eventos App**
	El usuario selecciona la opción Perfil en la pantalla de correspondiente. La aplicación muestra un formulario de edición que contiene campos como	Si los datos ingresados no son válidos la aplicación muestra un mensaje de error y solicita que el usuario corrija los campos incorrectos. Si el usuario decide cancelar la

	nombre completo, dirección de correo electrónico, número de teléfono. El usuario modifica los campos del formulario según sea necesario.	edición en cualquier momento, puede seleccionar una opción de "Cancelar" en lugar de "Guardar Cambios", y la aplicación no realizará ninguna actualización en ese caso.
Precondición	El usuario ha iniciado sesión en la aplicación. El usuario ha accedido a la sección de Perfil de Usuario. La aplicación muestra los datos actuales del perfil del usuario.	
Postcondición	El perfil del usuario se ha actualizado con la nueva información proporcionada por el usuario. El usuario recibe una confirmación visual de que los cambios se han guardado con éxito. Los datos actualizados se almacenan en la base de datos de la App.	

Figura 29

Diagrama de caso de uso perfil del usuario

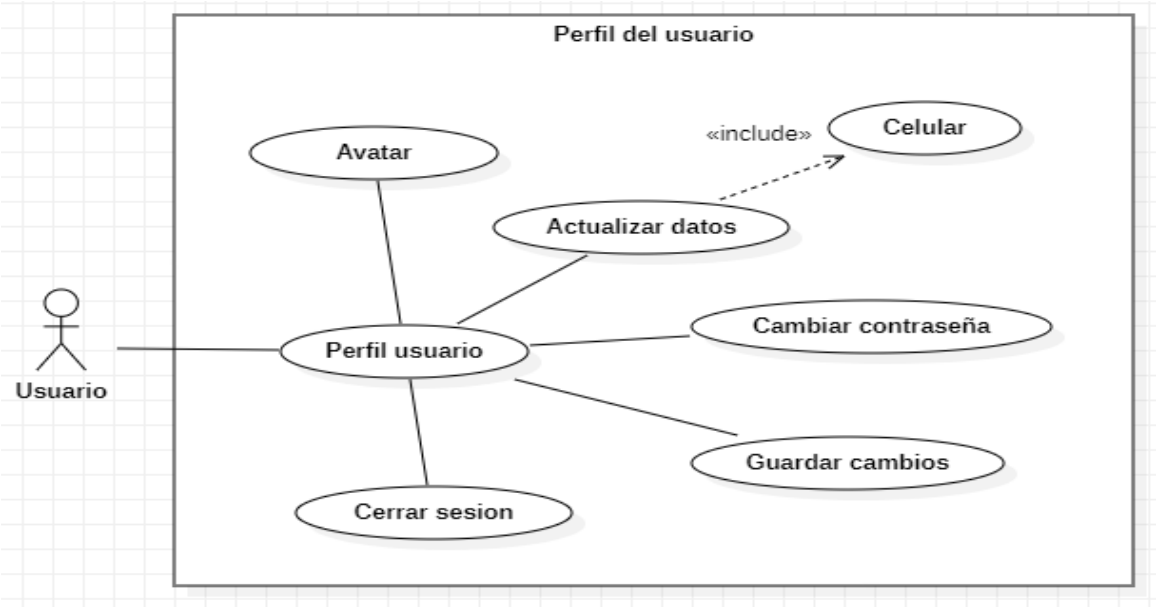
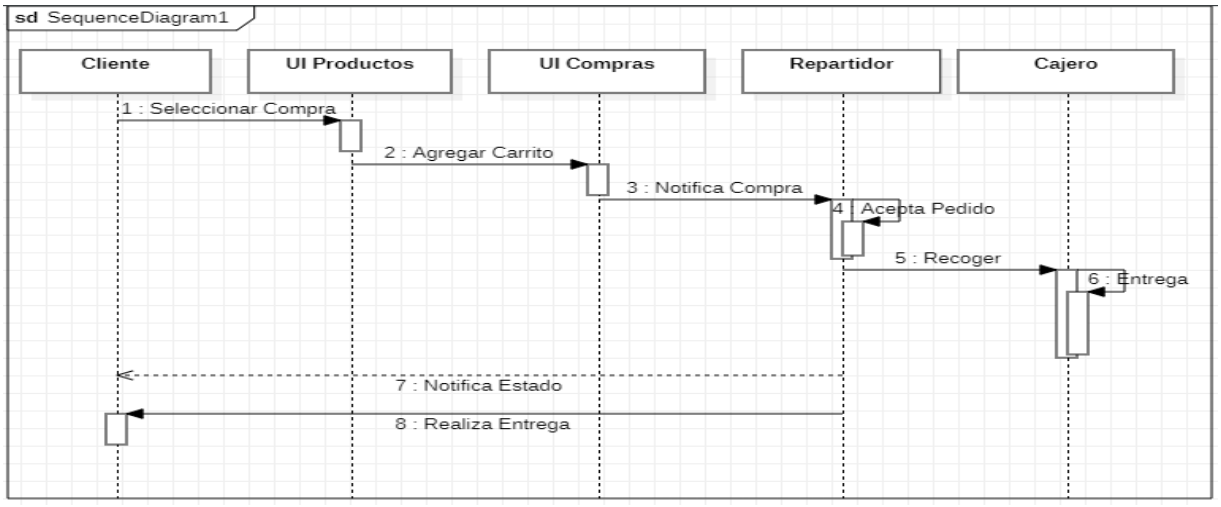


Figura 30

Diagrama de secuencia compra



Modulo del Delivery: En este módulo de entrega en la aplicación es la parte de la plataforma que se utiliza para gestionar y llevar a cabo la distribución de productos o servicios a los clientes.

Tabla 14

Caso de uso realizar entrega

Caso de uso	Realizar entrega	
Actores	Usuario (Delivery)	
Descripción	Este caso de uso describe las acciones que un repartidor realiza para entregar un pedido a un cliente a través de la aplicación de entrega.	
Flujo principal	**Eventos Actor**	**Eventos App**
	El repartidor recibe una notificación de la aplicación móvil de entrega con los detalles del pedido asignado, incluyendo la dirección de	Si el cliente no se encuentra en la dirección de entrega, el repartidor puede seguir un procedimiento para ponerse en contacto con el cliente

	entrega y los elementos del pedido. El delivery llega a la dirección de entrega y se comunica con el cliente por un mensaje para coordinar la entrega. El repartidor registra el estado de la entrega en la aplicación.	mediante el chat integrado y así dejar un aviso de entrega. Si el repartidor se encuentra con dificultades en la navegación o en la entrega, la aplicación puede proporcionar opciones para contactar al cliente.
Precondición	El usuario ha iniciado sesión en la aplicación. El pedido está listo para ser entregado y se ha asignado a un repartidor. El cliente ha proporcionado información de entrega, incluyendo la dirección y detalles de contacto.	
Postcondición	La entrega se ha completado y registrada en la aplicación móvil. El cliente ha recibido los productos y ha sido notificado de la entrega. El repartidor está disponible para futuras asignaciones de entrega.	

Figura 31

Diagrama de caso de uso realizar entrega

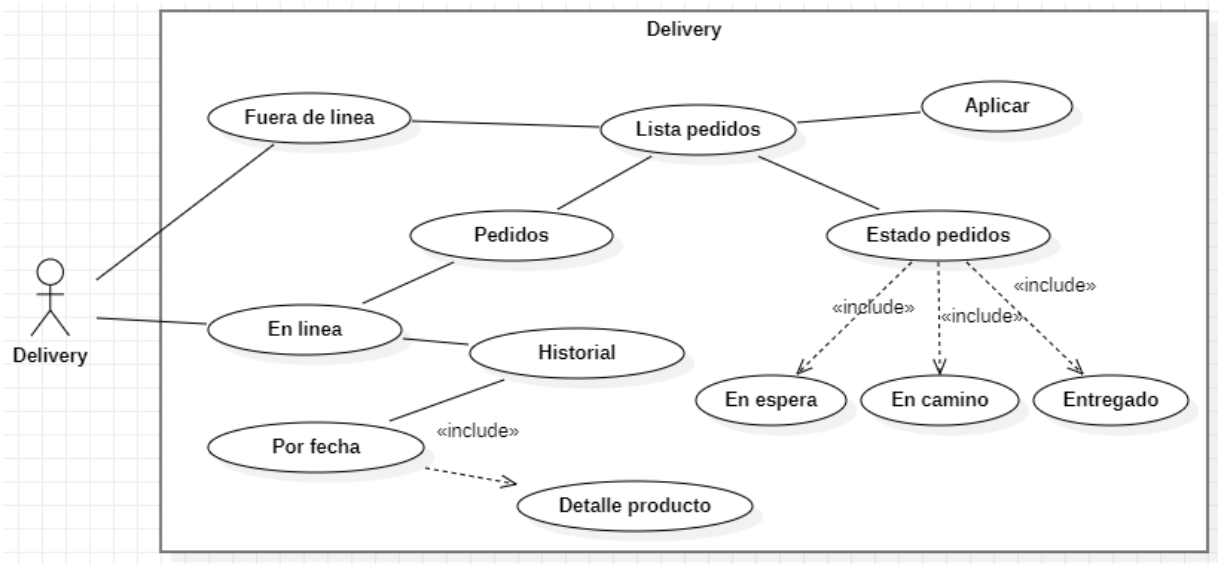


Figura 32

Diagrama de secuencia Delivery

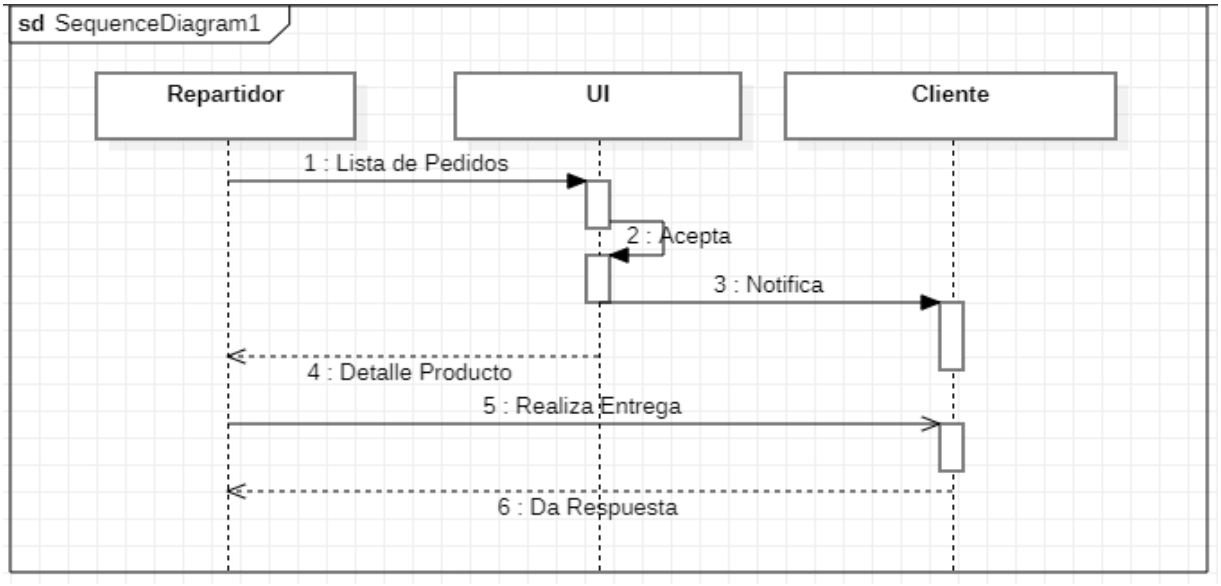


Tabla 15

Caso de uso ruta al cliente

Caso de uso	Ruta al cliente	
Actores	Usuario (Delivery)	
Descripción	Este caso de uso describe las acciones que un repartidor realiza para generar y seguir una ruta para llegar al destino del cliente y entregar el pedido.	
Flujo principal	**Eventos Actor**	**Eventos App**
	El repartidor selecciona el pedido asignado en la aplicación. La aplicación móvil de entrega utiliza la dirección de entrega del cliente y la ubicación actual del repartidor obtenida a	Si el cliente no se encuentra en la dirección de entrega o si la ubicación es incorrecta, el repartidor puede seguir un procedimiento para ponerse en contacto con el cliente mediante el chat o una llamada

	través del GPS para calcular una ruta óptima. El repartidor registra el estado de la entrega en la aplicación.	y dejar un aviso de entrega. La aplicación móvil actualiza el estado del pedido como entregado en la base de datos y notifica al cliente que la entrega se ha completado.
Precondición	El usuario ha iniciado sesión en la aplicación. Se ha seleccionado un pedido de un cliente con una dirección de entrega.	
Postcondición	La entrega se ha completado y se registra en la aplicación móvil bajo un indicador de estado. El cliente ha recibido los productos y ha sido notificado de la entrega.	

Figura 33

Diagrama de caso de uso ruta al cliente

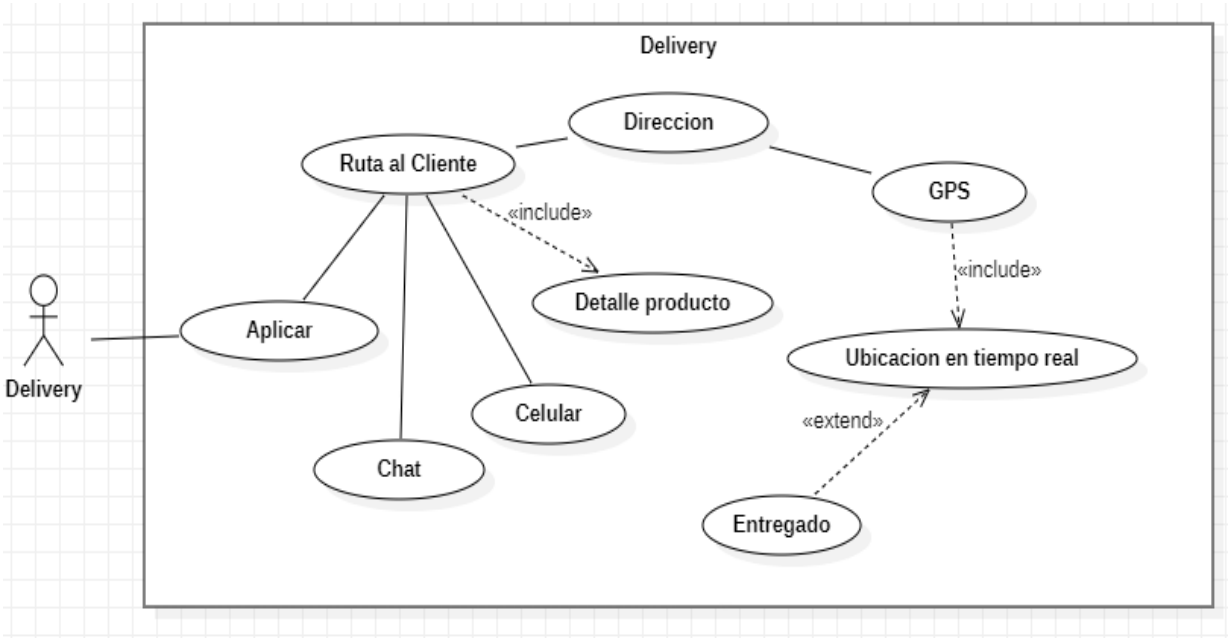


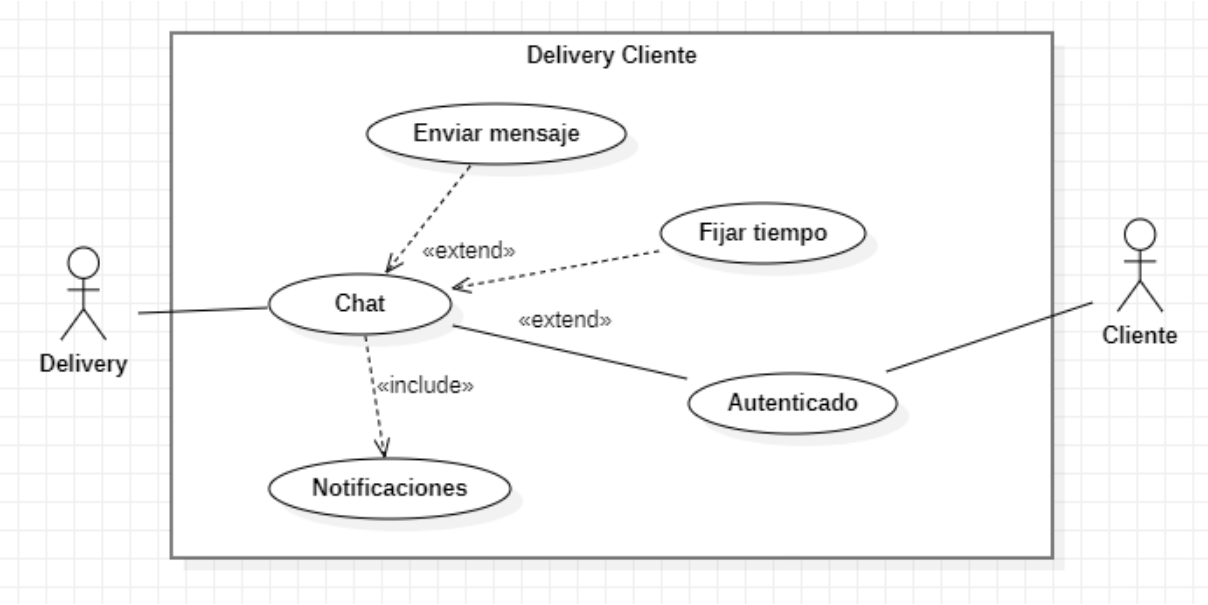
Tabla 16

Caso de uso chat de la App

Caso de uso	Chat de la App	
Actores	Usuarios (Delivery, Cliente)	
Descripción	Este caso de uso describe las acciones que los usuarios deliveryman y cliente realizan al utilizar la función de chat interno de la aplicación para intercambiar mensajes de texto.	
	Eventos Actor	**Eventos App**
Flujo principal	Los usuarios seleccionan la función de chat dentro de la aplicación. La aplicación móvil de Mensajería abre una ventana de chat donde los usuarios pueden escribir y enviar un mensaje. La conversación se actualiza en tiempo real para mostrar nuevos mensajes entrantes.	Cualquiera de los usuarios puede optar por abandonar la conversación o cerrar la ventana de chat en cualquier momento.
Precondición	Los usuarios deliveryman y cliente han iniciado sesión en la aplicación. Ambos usuarios tienen acceso a la función de chat y están conectados a la red.	
Postcondición	La conversación entre el Usuario deliveryman y el Usuario cliente se ha realizado satisfactoriamente. Los mensajes intercambiados se almacenan en la aplicación y están disponibles para futuras consultas.	

Figura 34

Diagrama de caso de uso chat de la App



3.3. Fase de Inicialización

En esta fase se muestra la identificación, análisis y determinación de los recursos primordiales necesarios para realizar el proyecto.

3.3.1. Soporte de Software y Hardware

Para el proceso de ejecución de la App en el hardware serán necesarias las siguientes características:

- Cámara de resolución mayor a 4 Megapíxeles
- Memoria RAM mínima 512 Mb.
- Memoria libre mínima de 1140 Mb.

El sistema operativo requerido será:

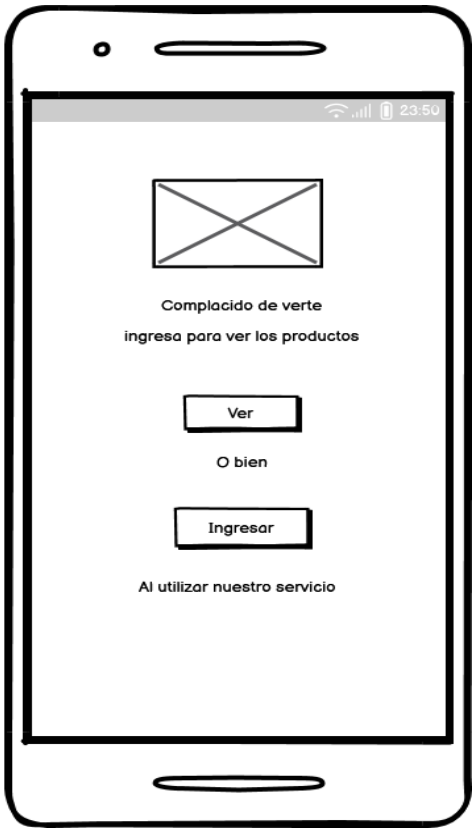
- Android v 5.0 (Lollipop) o superior.

3.3.2. Diseño de la Aplicación Móvil

Pantalla inicio: Esta es la primera pantalla que los usuarios ven cuando abren la aplicación y por lo tanto cuenta con los siguientes elementos.

Figura 35

Mockup Diseño de la pantalla de inicio



Pantalla de inicio de sesión: Esta es la interfaz donde los usuarios ingresan sus credenciales para acceder a sus cuentas personales o a las funciones de manera segura en la aplicación móvil. Su propósito principal es autenticar a los usuarios y brindarles acceso a sus perfiles y contenido personalizado, como se muestra a continuación.

Figura 36

Mockup Diseño de la pantalla de acceso

El Inicio de sesión con Google es una solución eficaz para mejorar la experiencia del usuario y aumentar la seguridad en la aplicación móvil. Esto ayuda a simplificar el proceso de registro y acceso, reduce los riesgos asociados con contraseñas débiles o reutilizadas y proporciona una manera segura y conveniente de autenticarse en la App.

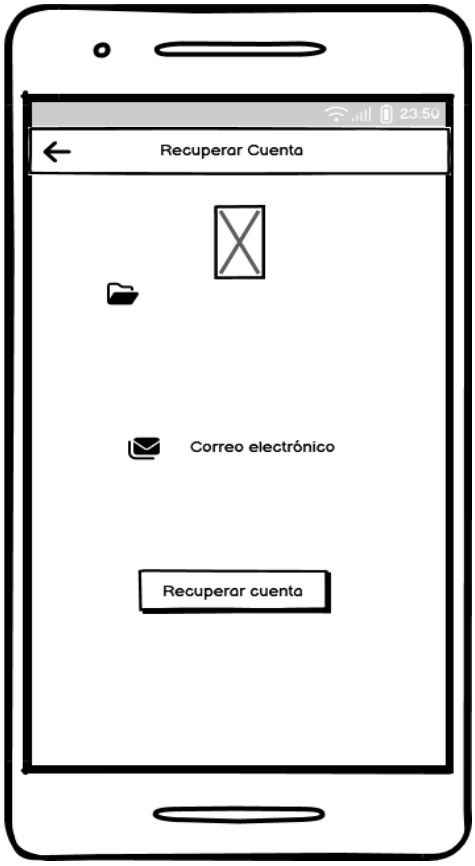
Figura 37

Mockup Diseño de la pantalla de registro

Pantalla de recuperar cuenta: Esta interfaz de usuario (UI) permite recuperar el acceso a sus cuentas en caso de que hayan olvidado sus contraseñas o tengan dificultades para iniciar sesión. Proporciona un mecanismo para restablecer la contraseña y asegurarse de que solo el propietario legítimo de la cuenta pueda recuperar el acceso.

Figura 38

Mockup Diseño de la pantalla recuperar



Pantalla panel principal: Esta es la interfaz intuitiva fácil de usar y ofrezca acceso rápido a las funciones que los usuarios ven después de autenticarse exitosamente en la aplicación. Esta pantalla está diseñada para proporcionar a los usuarios acceso a contenido personalizado, funciones y características específicas de sus cuentas, como se muestra a continuación.

Figura 39

Mockup Diseño de la pantalla principal de la aplicación móvil

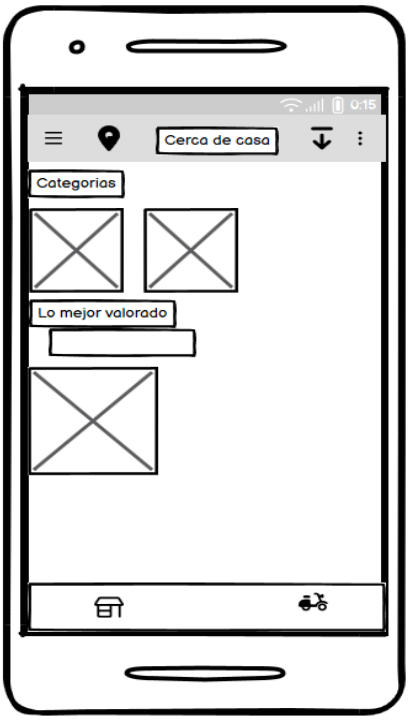


Figura 40

Mockup Diseño de la pantalla Side menú

Pantalla carrito de compras: Esta pantalla permite a los usuarios revisar y gestionar los productos que han seleccionado para comprar antes de proceder al pedido.

Figura 41

Mockup Diseño de la pantalla carrito de compras

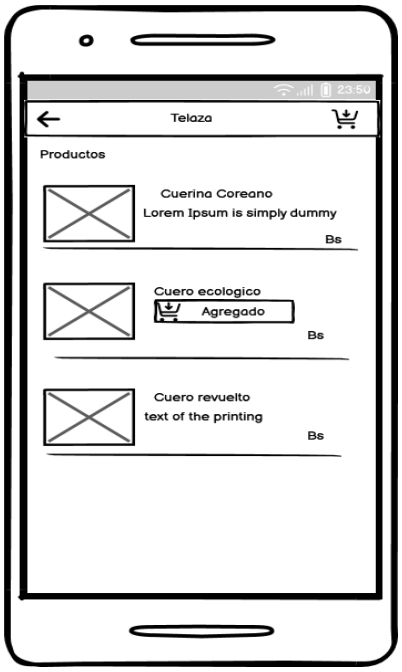
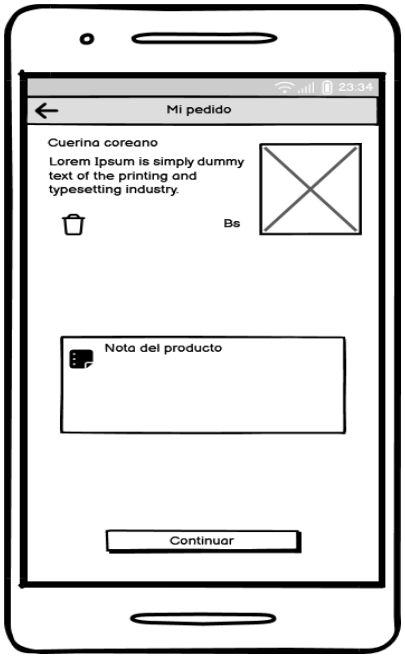


Figura 42

Mockup Diseño de la pantalla lista de pedidos



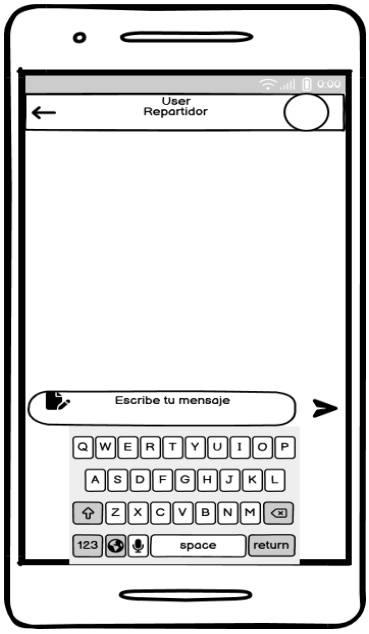
Pantalla estado del pedido: Esta UI permite a los usuarios realizar un seguimiento del progreso de sus pedidos desde el momento en que se realizan hasta su entrega. Esta pantalla proporciona información en tiempo real sobre el estado actual de un pedido.

Figura 43

Mockup Diseño de la pantalla estado del pedido

Figura 44

Mockup Diseño de la pantalla chat de la App



Pantalla repartidor o delivery: Esta interfaz proporciona a los usuarios una experiencia de entrega transparente y eficiente. Permite a los usuarios hacer un seguimiento en tiempo real de sus entregas.

Figura 45

Mockup Diseño de la pantalla delivery

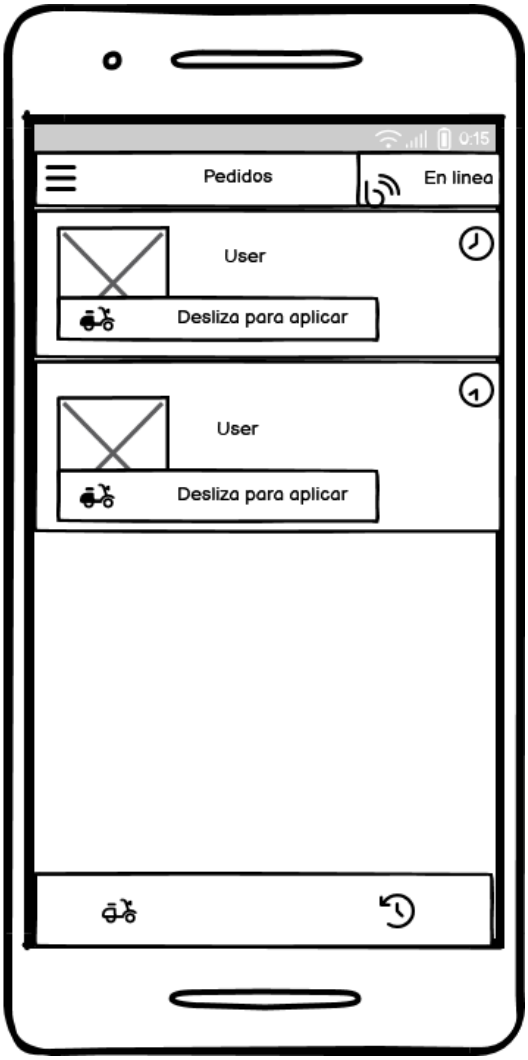
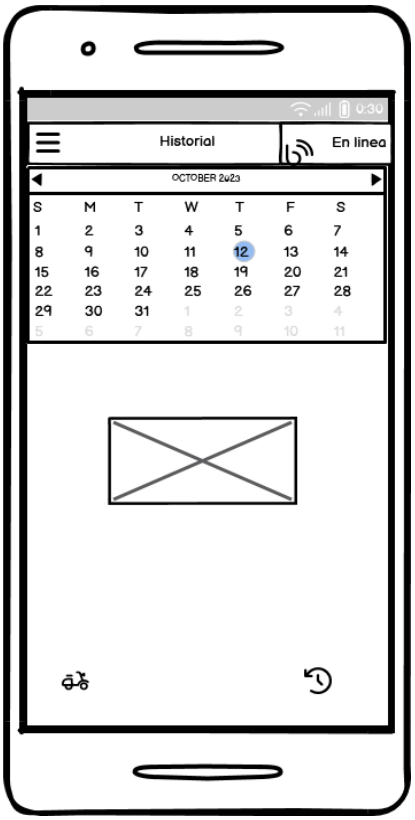


Figura 46

Mockup Diseño de la pantalla historial de entregas



3.4. Fase de Producción

Durante esta fase se especifica el proceso de desarrollo de la aplicación, en la implementación basándose en los diseños y especificaciones establecidos en etapas anteriores.

3.4.1. Backend Servidor NODEJS

Para la implementación de una instancia en Node.js usando la Framework de Express y Nest.js que se utiliza para ejecutar la aplicación en el lado del servidor, donde se proporciona peticiones HTTP, APIs para la aplicación móvil y realizar estas tareas relacionadas con el procesamiento de solicitudes y la gestión de recursos del servidor.

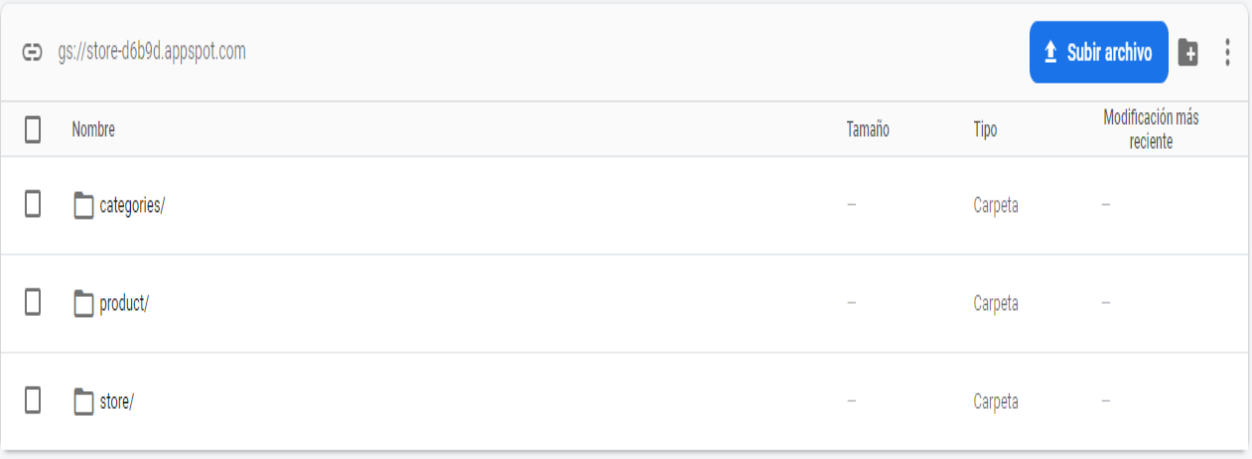
Figura 47

Inicio de la instancia Nest.js

```
import { AppModule } from './app.module';

async function bootstrap() {
  const app = await NestFactory.create(AppModule);
  app.setGlobalPrefix('api/');

  app.useGlobalPipes(
    new ValidationPipe({
      whitelist: true,
      forbidNonWhitelisted: true,
    })
  );

  await app.listen(3000);
}
bootstrap();
```

Figura 48

Firebase Storage

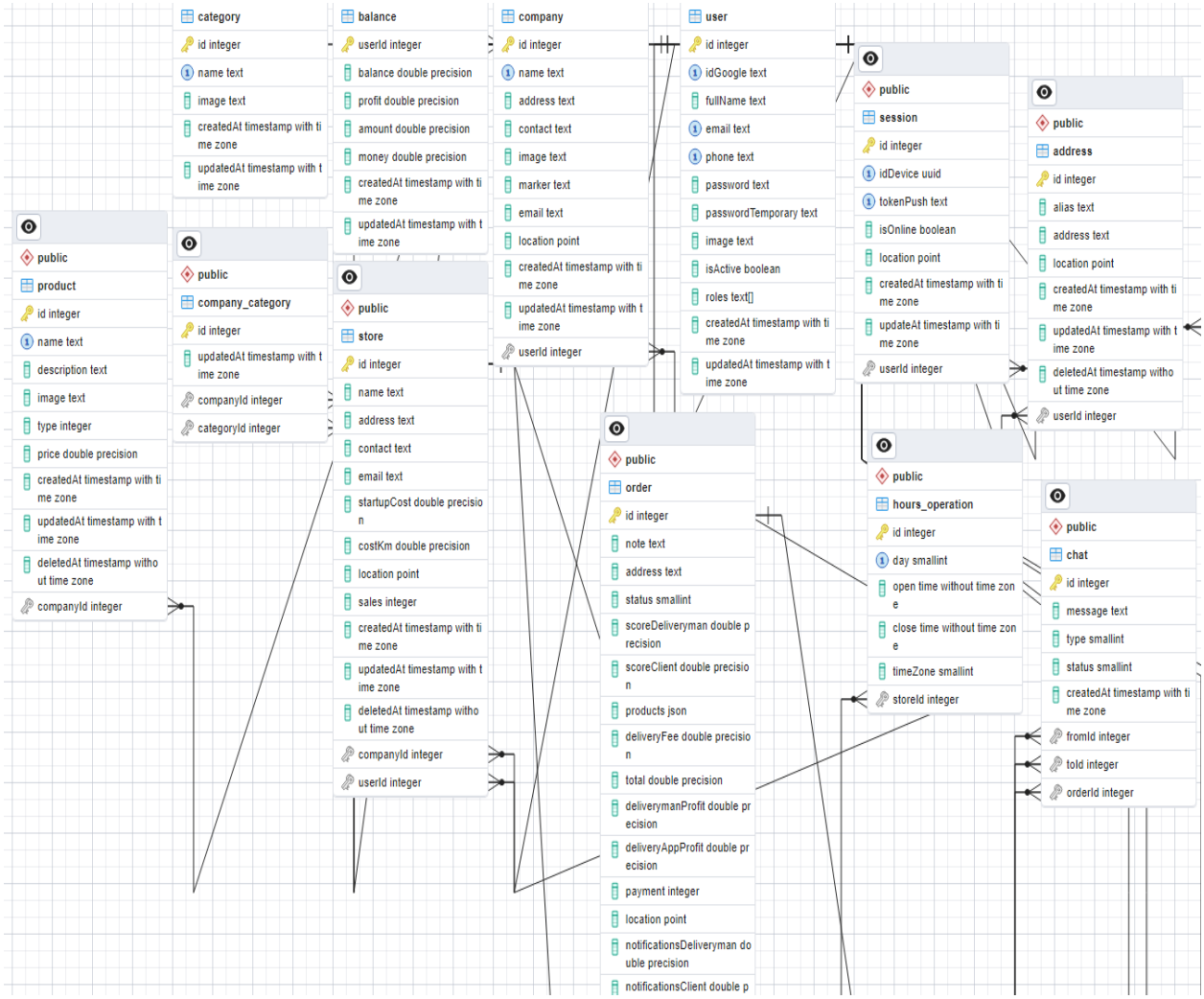


3.4.2. Modelo ER Base de Datos

En el modelo de entidad-relación (ER) se ha estructurado según a los requerimientos mencionados en la fase de exploración.

Figura 49

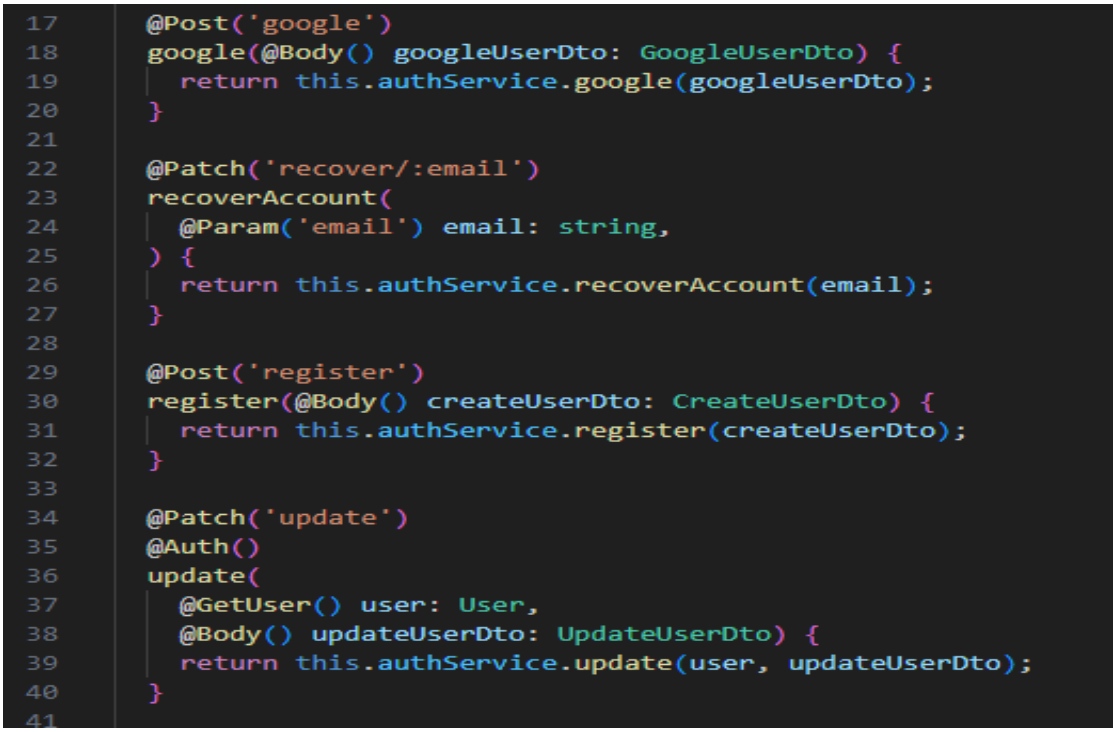
Base de Datos Modelo ER



Planeación y codificación: Tomando en cuenta de la fase de inicialización donde inicialmente se propuso la interfaz de pantalla de inicio de sesión. Para la autenticación y autorización de usuarios se está utilizando JWT (JSON Web Token). Bcrypt para almacenar y gestionar contraseñas de manera segura en aplicación móvil. A continuación, se muestra los Endpoints realizado en Visual Studio y la petición al servidor mediante Postman.

Figura 50

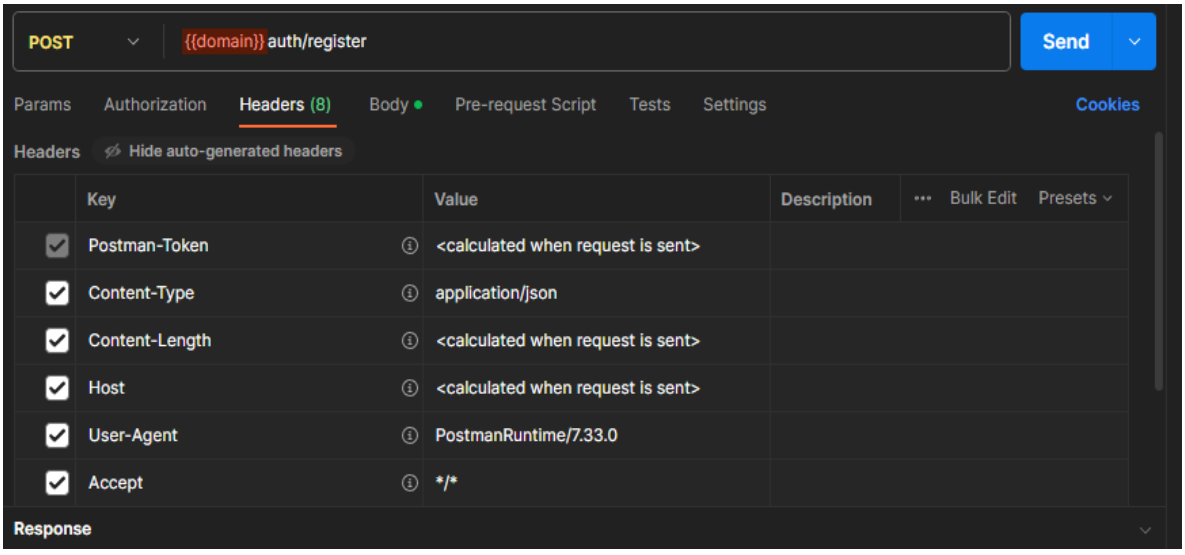
Enpoint inicio de sesión



```
17    @Post('google')
18    google(@Body() googleUserDto: GoogleUserDto) {
19      return this.authService.google(googleUserDto);
20    }
21
22    @Patch('recover/:email')
23    recoverAccount(
24      @Param('email') email: string,
25    ) {
26      return this.authService.recoverAccount(email);
27    }
28
29    @Post('register')
30    register(@Body() createUserDto: CreateUserDto) {
31      return this.authService.register(createUserDto);
32    }
33
34    @Patch('update')
35    @Auth()
36    update(
37      @GetUser() user: User,
38      @Body() updateUserDto: UpdateUserDto) {
39      return this.authService.update(user, updateUserDto);
40    }
41
```

Figura 51

Enpoint registrarse solicitud desde postman



Codificación: Creando rutas con el modelo arquitectónico REST (Representational State Transfer). Para utilizar las APIs (Interfaces de Programación de Aplicaciones) en la aplicación móvil. La API REST brinda la comunicación entre el servidor y la App a través de solicitudes HTTP estándar.

Figura 52

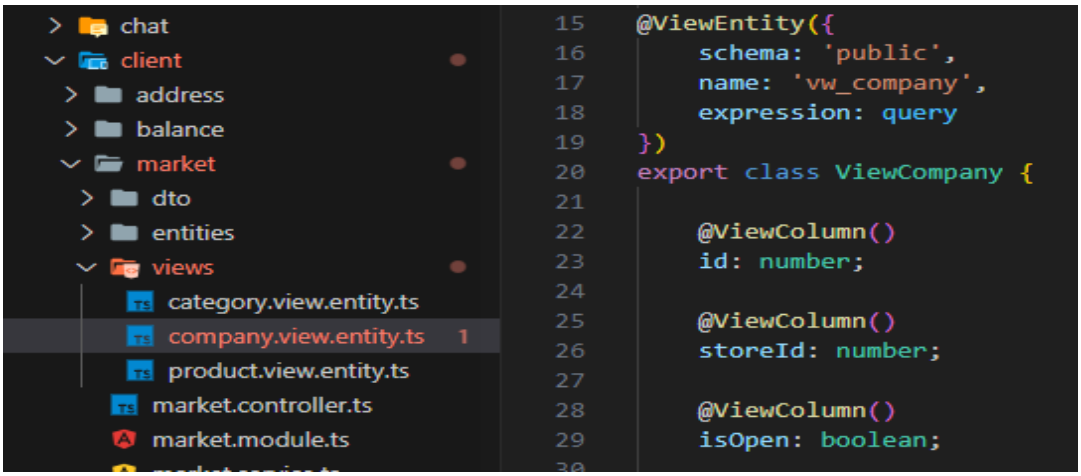
Rutas middleware REST

```
8
9   @Controller('client/address')
10  export class AddressController {
11    constructor(private readonly addressService: AddressService) { }
12
13    @Post()
14    @Auth()
15    create(
16      @GetUser() user: User,
17      @Body() createAddressDto: CreateAddressDto) {
18      return this.addressService.create(user, createAddressDto);
19    }
20
21    @Get()
22    @Auth()
23    findAll(
24      @GetUser() user: User,
25    ) {
26      return this.addressService.findAll(user);
27    }
28
```

Codificación: Mapeando consultas previas mediante la creación de Vistas de la Base de Datos para mostrar solo los datos relevantes para el usuario.

Figura 53

Tablas virtuales o vistas



3.4.3. Frontend UX (User Experience) y UI (User Interface)

Implementando la presentación visual de la aplicación móvil, tal como se ha diseñado en la fase de inicialización incluyendo elementos y componentes con los que los usuarios interactúan.

Para el módulo de inicio de sesión y guardar los datos de sesión con SharedPreferences, la ruta inicial tiene los tres perfiles de usuarios tomando en cuenta la autenticación.

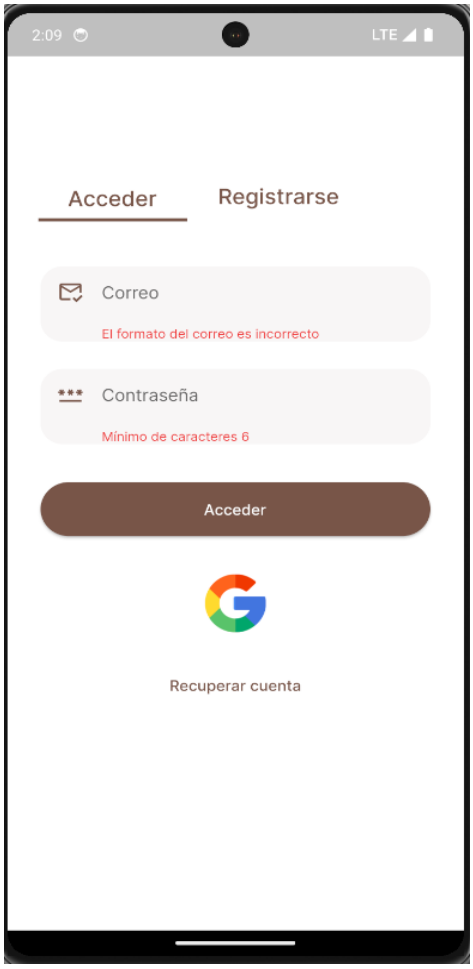
Figura 54

Modulo inicio de sesión

Obteniendo datos de los TextField y validando campos con ScaffoldMessenger para mostrar mensajes de error.

Figura 55

Manejo de códigos de error



Controlando con MultiProvider los estados de cada pantalla diseñados en la fase de inicialización y con TabController para que las pestañas se sincronicen.

Figura 56

Panel despegable con gestor de estados

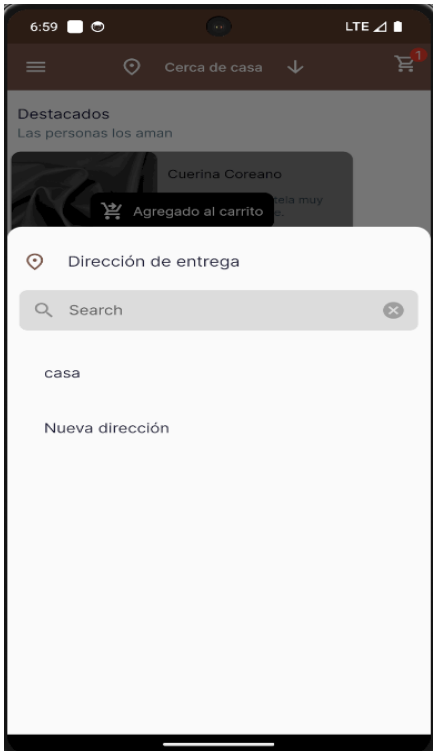


Figura 57

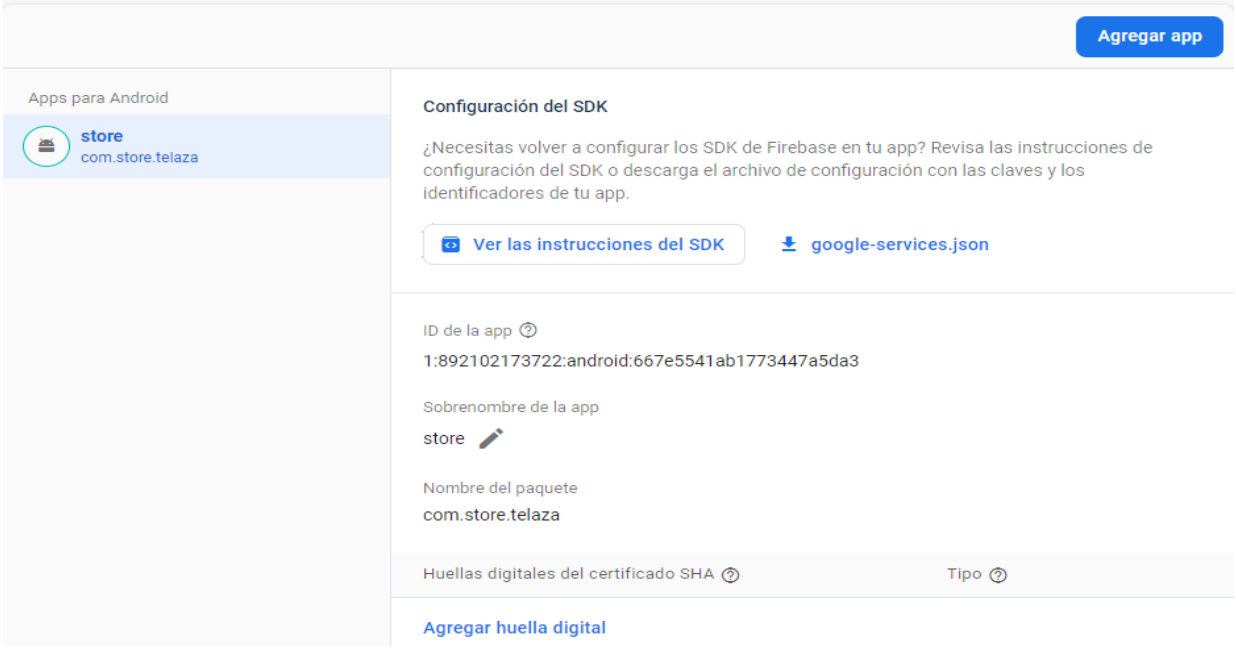
Rutas con MultiProvider

```
65      return MultiProvider(
66        providers: [
67          ChangeNotifierProvider(create: (_) => IconCartController()),
68          ChangeNotifierProvider(create: (_) => StoreController()),
69          ChangeNotifierProvider(create: (_) => TabManController()),
70          ChangeNotifierProvider(create: (_) => Tab1Controller()),
71          ChangeNotifierProvider(create: (_) => Tab2Controller()),
72          ChangeNotifierProvider(create: (_) => AddressDropdownController()),
73          ChangeNotifierProvider(create: (_) => PaymentDropdownController()),
74          ChangeNotifierProvider(create: (_) => CartSummaryController()),
75          ChangeNotifierProvider(create: (_) => LoginController()),
76          ChangeNotifierProvider(create: (_) => PetitionsController()),
77          ChangeNotifierProvider(create: (_) => RequestsController()),
78          ChangeNotifierProvider(create: (_) => CategoryDropdownController()),
79          ChangeNotifierProvider(create: (_) => EnrollmentController()),
80          ChangeNotifierProvider(create: (_) => ProductController()),
81        ],
82        child: MaterialApp(
83          title: kNameApp,
84          debugShowCheckedModeBanner: false,
85          initialRoute: initialRoute,
86          routes: {
87            'welcome': (BuildContext context) => const WelcomeScreen(),
88            'tabs': (BuildContext context) => const TabMainScreen(),
89            'petitions': (BuildContext context) => const PetitionsScreen(),
90            'requests': (BuildContext context) => const RequestsScreen(),
91          },
```

Implementando canales de notificación mediante Streams para la transmisión de información en tiempo real y usando Push Notifications de Firebase.

Figura 58

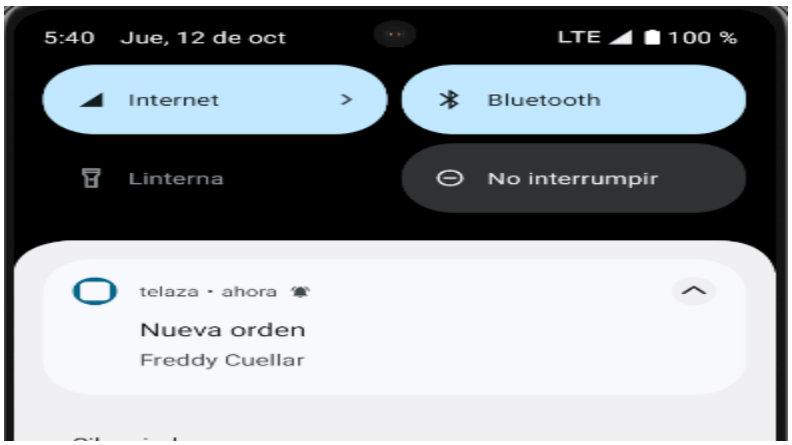
Registrando App en Firebase



Cambios de estado en la orden con FirebaseMessaging y FlutterLocalNotificationsPlugin como se muestra en el emulador de Android Studio.

Figura 59

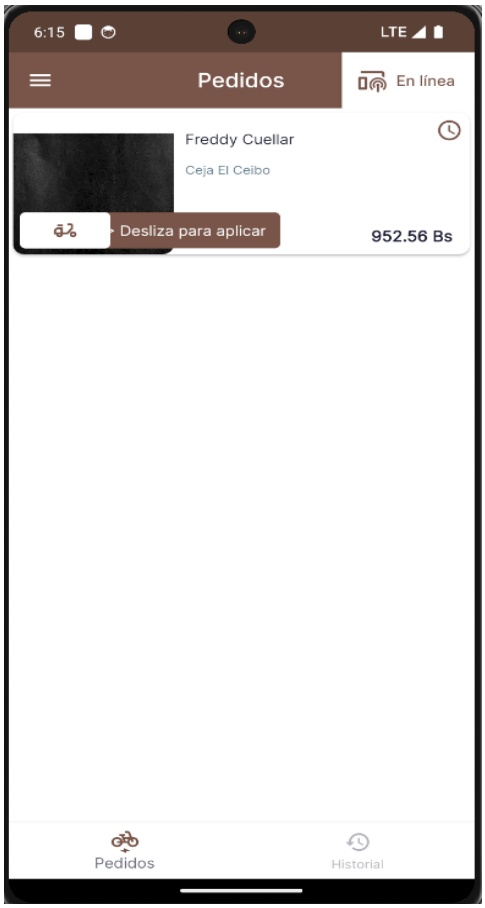
Recibiendo notificaciones



Pantalla de peticiones con estados es necesario activar GPS para poderse pasar a modo online.

Figura 60

Lista de peticiones con cambios de estado



3.5. Fase de Estabilización

En esta fase se realiza la unificación e integración de la implementación realizada en la fase de producción para usar la API REST de comunicación y transferencia de datos entre la aplicación móvil y el servidor.

Figura 61

Sincronizando Google OAuth

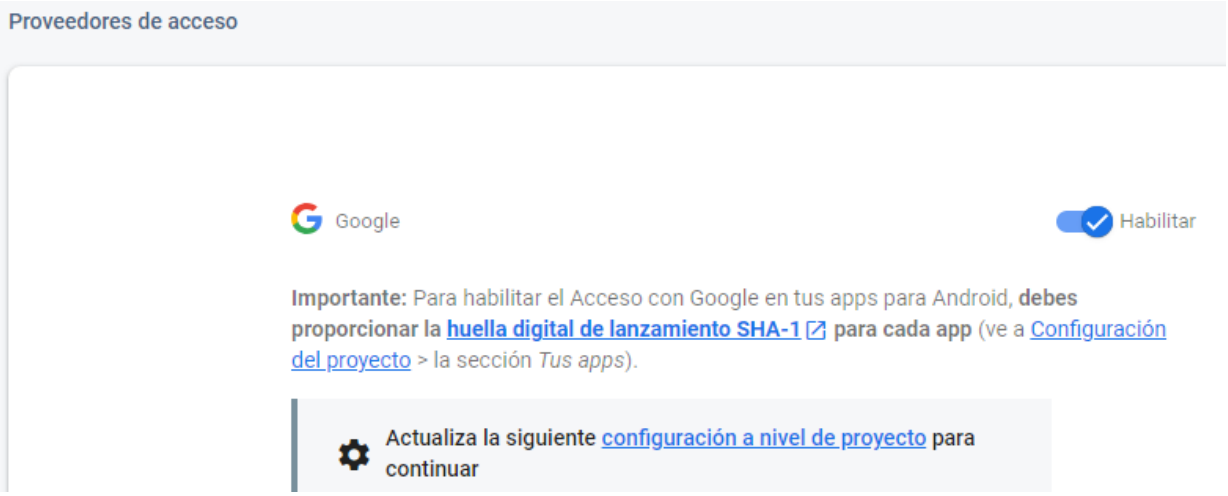


Figura 62

Habilitando APIs y servicios de Google

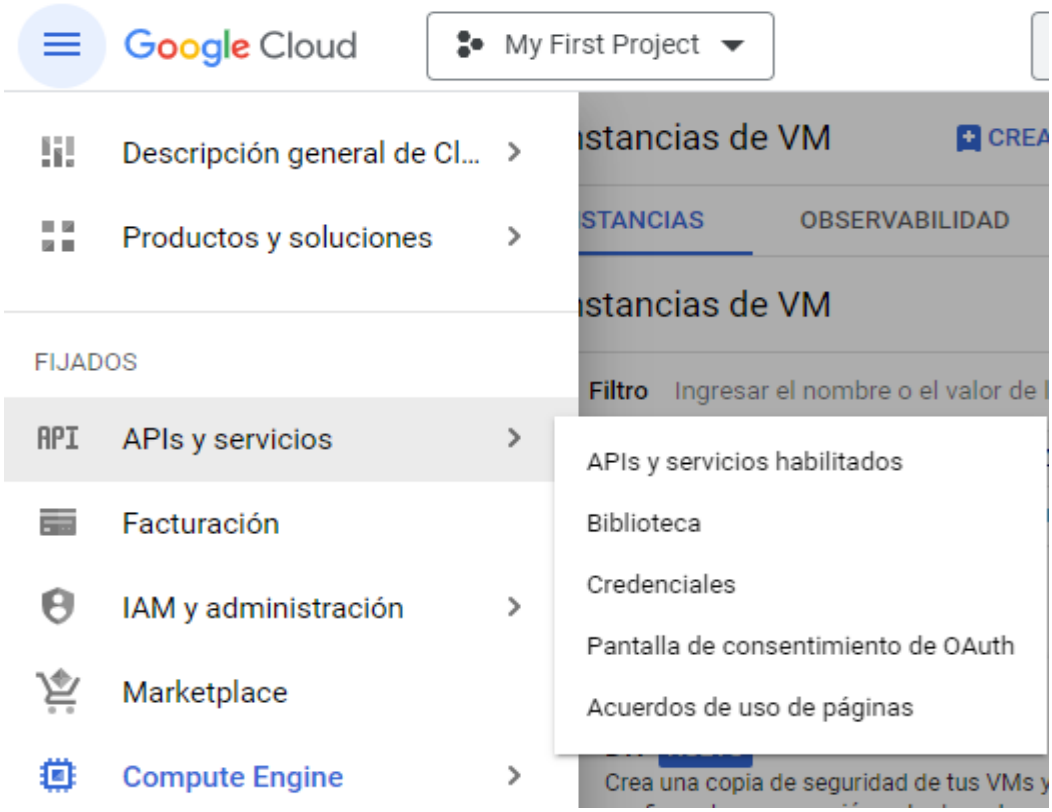
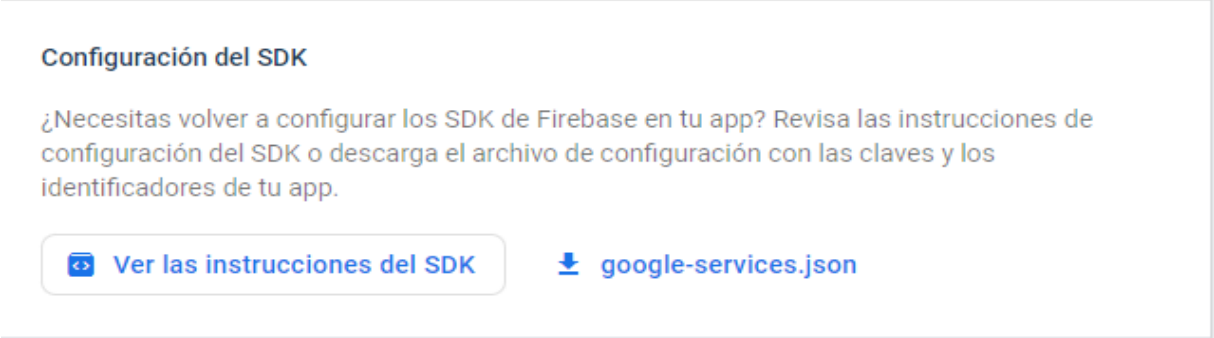


Figura 63

Habilitando Google Services en Firebase



3.6. Fase de Pruebas

En esta etapa se realiza un análisis de la disponibilidad de versiones estables. Con una aplicación totalmente funcional, probando en dispositivos móviles. Para realizar las pruebas, la aplicación se instaló en un dispositivo real con diferentes características como:

Tabla 17

Dispositivos móviles en las que se instaló la aplicación

Marca	Modelo	Pantalla	Resolución	RAM	Cámara MP
Samsung	Galaxy S10	6.1 pulgadas	1440 x 3040 píxeles	8 GB	16 megapíxeles
Huawei	P10 Lite	5.2 pulgadas	1080 x 1920 pixeles	4 GB	12 megapíxeles
Samsung	Galaxy A70	6.7 pulgadas	2400 x 1080 pixeles	6/8 GB	32 megapíxeles

A través de la verificación realizada en los distintos dispositivos móviles, se confirmó que la aplicación funciona perfectamente y se adapta a cualquier tipo de pantalla.

CAPITULO IV

CALIDAD, COSTO, SEGURIDAD Y PRUEBAS

4.1. Introducción

La calidad y la seguridad del software están estrechamente relacionadas, ya que el software de baja calidad puede tener más vulnerabilidades de seguridad. Estos dos aspectos son muy importantes hoy en día, porque los sistemas informáticos son muy importantes en casi todos los aspectos de la vida y las operaciones de la empresa. La garantía de calidad y la implementación de medidas de seguridad adecuadas son fundamentales para garantizar que el software funcione de manera eficiente y segura. La estimación de costos de software es esencial para la planificación y el control del proyecto, así como para la toma de decisiones presupuestarias. COCOMO II proporciona una estructura sólida para realizar estas estimaciones y permite tomar decisiones informadas sobre la inversión y los recursos necesarios para llevar a cabo el proyecto.

4.2. Métricas de calidad

En esta etapa se realizará la métrica de calidad de software mediante la norma ISO 9126 con las siguientes características: la funcionalidad, confiabilidad, usabilidad, mantenibilidad y portabilidad.

4.2.1. Funcionalidad

La funcionalidad no se puede medir directamente, por lo que utilizamos el factor de ponderación para encontrar el punto función que cuantifica el tamaño y complejidad de la aplicación móvil en cuanto a funciones de usuario. Los puntos de función se derivan utilizando una relación empírica basada en mediciones contables del área de información de software y en mediciones cualitativas de la complejidad del software. Se realiza la evaluación de la funcionalidad de la aplicación móvil de acuerdo a la formula.

$$\mathbf{PF = conteo\ total * [0.65 + 0.01 \sum (Fi)]}$$

Para poder medir la funcionalidad se mide primero las métricas de punto de función (PF) para luego obtener el conteo total. Los valores de dominio de información se definen en la siguiente manera:

Número de entradas de usuario. Cada entrada externa se origina de uno o más, las entradas deben distinguirse de las consultas que se cuentan por separado y se obtiene la siguiente tabla.

Tabla 18

Entrada de usuarios

N°	Entrada de usuarios	Cantidad
1	Ingreso a la aplicación	1
2	Ingreso a la interfaz de usuario	3
	Total	4

Tabla 19

Salida de usuarios

N°	Salida de usuarios	Cantidad
1	Salidas de la aplicación	2
2	SharedPreferences por usuario	3
	Total	5

Número de salidas Usuario. En la aplicación móvil las salidas por cada usuario son:

Tabla 20

Consulta de usuarios

N°	Consultas de usuarios	Cantidad
1	Consultas en la aplicación	3
2	Interacción con la aplicación por usuario	2
	Total	5

Número de consultas Usuario. Es una entrada interactiva que da como resultado la generación o la respuesta al usuario.

Tabla 21

Numero de archivos

N°	Archivos de usuario	Cantidad
1	Dependencias Externas	2
2	Archivos de imagen	10
	Total	12

Número de archivos. son los archivos que contiene la aplicación móvil.

Tabla 22

Numero de interfaz externo

N°	Entrada de usuario	Cantidad
1	Memoria de almacenamiento interno	1
2	Cloud Storage	2
	Total	3

Número de interfaz externo. Son los archivos de interfaz externas proporcionando información para usar la aplicación. Con los datos obtenidos se realiza el cálculo del conteo total del punto fusión.

Tabla 23

Factores de ponderación

Información	**Cuenta**	**Factores de ponderación**			**Cuenta total**
		Simple	Medio	Complejo	
Entrada de usuarios	4		4		16
Salida de usuarios	5		5		25
Consulta de usuarios	5		4		20
Numero de archivos	12		10		120
Interfaz externo	3		7		21
		Total			202

El siguiente paso es medir las preguntas de complejidad siendo que de acuerdo a las preguntas de complejidad se realiza la ponderación de las variables que serán evaluadas.

Tabla 24

Ponderación de valores

Categoría	Ponderación de valor
No importante	1
Menor importancia	2
Moderado	3
Medio	4
Significativo	5
Esencial	6

Se responde a las interrogantes con la escala de ponderación de valor:

Tabla 25

Preguntas de complejidad

N°	Preguntas de complejidad	Valor
1	¿La aplicación móvil requiere copias de seguridad?	3
2	¿Se requiere comunicación de datos?	5
3	¿Existen funciones de procesamiento distribuidas?	4
4	¿El desempeño es crucial?	4
5	¿Se ejecutará el sistema en un entorno operativo existente y Utilizado?	5
6	¿Se requiere entrada de datos?	3
7	¿La entrada de datos requiere que las transacciones de entrada se construyan sobre múltiples pantallas u operaciones?	5
8	¿Se ejecuta archivos de forma interactiva?	4
9	¿Las entradas, salidas o archivos son complejos?	2
10	¿El proceso interno es complejo?	4

11	¿Se ha diseñado el código para ser reutilizable?	5
12	¿La conversión y la instalación se incluyen en el diseño?	4
13	¿Se diseña para instalaciones múltiples en diferentes organizaciones (En diferentes paralelos)?	5
14	¿Se ha diseñado para ser fácilmente utilizable por el usuario?	5
	Total = ∑(Fi) =	58

Cálculo de punto fusión PF

$$PF = cuenta\ (grado\ de\ confiabilidad + Tasa\ de\ error * \sum (Fi))$$

Donde:

Cuenta total = Total de puntos función

Grado de Confiabilidad = Valor de 0.65

Tasa de Error = Valor 0.01 (Error de confiabilidad de la aplicación).

∑ (Fi) = Valor Total de la complejidad de la aplicación. Remplazando valores en la formula, se tiene lo siguiente:

PF = 202 *[0.65 + 0.01 * 58]

PF = 248.46

Calculando el punto fusión con el valor máximo de ∑ (Fi) = 70.

PF Max = 202 *[0.65 + 0.01 * 70]

PF Max = 272.5

Cálculo de la funcionalidad:

Funcionalidad = 478.47/525.15 = 0.911 ∗ 100% = 91%

Por lo tanto, El resultado obtenido muestra que la aplicación móvil tiene una funcionalidad o utilidad del 91% para la organización, lo que indica que la App cumple con los requisitos funcionales de forma satisfactoria.

4.2.2. Confiabilidad

Aquí se agrupan una serie de atributos que se relacionan con la capacidad del software de mantener su nivel de rendimiento durante un período prolongado de tiempo en condiciones normales.

Para calcular la confiabilidad de la aplicación se toma en cuenta el periodo de tiempo en el cual se ejecuta y se obtiene muestras, donde se encuentra:

La función siguiente muestra el nivel de confiabilidad del sistema:

$$\boldsymbol{F(t) = (Funcionalidad) * e^{-\lambda t}}$$

Se observa el trabajo hasta que se observa un fallo en un instante t, la función es la siguiente:

Probabilidad de hallar una falla: $P\,(T \leq t) = F(t)$

Probabilidad de hallar una falla: $P\,(T > t) = 1 - F(t)$

Valor de Funcionalidad previo = 91 %

λ = 0.01 (es decir 1 error en cada 6 ejecuciones).

t = 6 meses

Hallamos la confiabilidad de la aplicación:

$$F\,(12) = 91 \times e - (1/\,6 \times 12)$$

$$F\,(12) = 91 \times 0.14$$

$$F\,(12) = 12{,}74\%$$

La probabilidad de hallar una falla es de un 12,74% durante los próximos 12 meses. Por lo tanto, la probabilidad de no hallar un error es del 87.26%.

4.2.3. Usabilidad

La usabilidad consiste en evaluar el esfuerzo requerido que realizará el usuario al utilizar la aplicación móvil según su comprensión y la estructura lógica que tiene la

aplicación. Esta comprensión por parte de los usuarios con relación a la aplicación evalúa los siguientes casos:

- Comprensibilidad
- Fácil de aprender
- Operabilidad

Se realizan encuestas a los usuarios finales sobre manejo, compresión y facilidad de uso para medir la usabilidad según la siguiente tabla:

Tabla 26

Preguntas de usabilidad de la aplicación

Preguntas	Respuestas		Porcentaje
	SI	NO	%
¿El acceso a la aplicación es complicado?	0	10	100
¿Son compresibles las respuestas de la aplicación?	1	9	90
¿Son complicadas los procesos que realiza la aplicación?	0	10	100
¿La aplicación tiene interfaces entendibles?	9	1	90
¿La interfaz de la aplicación es agradable a la vista?	0	10	100
¿Son satisfactorias las respuestas que la aplicación devuelve?	9	1	90
¿La aplicación reduce su tiempo de trabajo?	9	1	90
¿Es difícil aprender a manejar la aplicación?	1	9	90
¿La aplicación satisface las necesidades que usted requiere?	9	1	90
¿Utiliza la aplicación con facilidad?	9	1	90
Promedio			93 %

4.2.4. Mantenibilidad

Para hallar la mantenibilidad de la aplicación se utiliza el índice de madurez de software (IMS), que proporciona una indicación de la estabilidad de un producto de software (basado en los cambios que ocurren con cada versión del producto). Se determina la siguiente fórmula para hallar el (IMS):

$$IMS = [Mt - (Fa + Fc + Fd)] / Mt$$

Donde:

Mt, número de módulos de la versión actual.

Fa, numero de módulos en la versión actual que se han añadido.

Fc, numero de módulos en la versión actual que se han cambiado.

Fd, numero de módulos de la versión anterior que se han borrado en la versión actual.

Reemplazando los valores:

$$IMS = [7 - (1 + 0 + 0)] /7 = 0.85$$

Por lo tanto, se puede decir que la aplicación tiene una estabilidad de 85% que indica la facilidad de mantenimiento, el 15% restante es el margen de error correspondiente a los cambios y modificaciones.

4.2.5. Portabilidad

La portabilidad de refiere a la habilidad del software de ser transferido de un ambiente a otro, se considera los siguientes aspectos.

- **Adaptabilidad.** Aquí se evalúa la capacidad de adaptar el software a diferentes ambientes sin la necesidad de aplicarle modificaciones.
- **Facilidad de instalación**. Es el esfuerzo necesario para instalar el software en un ambiente determinado.
- **Conformidad.** Permite evaluar si el sistema se adhiere a estándares o convenciones relativas a portabilidad.

- **Capacidad de reemplazo.** Hace referencia a la oportunidad y el esfuerzo usado en sustituir el software por otro producto con funciones similares.

Se puede medir en los siguientes niveles:

Nivel de Hardware

- Tamaño pantalla. Smartphone >= 5'' (por comodidad del usuario).
- Velocidad CPU Quad Core 1.2GHz.
- Conexión Wi-Fi y 4G (servicio de datos) para poder trabajar con servicio de datos.

Nivel de Software

- Versión Android 5.0, preferiblemente con Android 6 como mínimo.

4.2.6. Resultados de las Métricas de Calidad

El factor de calidad total está directamente relacionado con el grado de satisfacción con el usuario que ingresa a la aplicación móvil.

Tabla 27

Resultados de Evaluación de Calidad

Características	Resultado
Funcionalidad	91 %
Confiabilidad	87.26 %.
Usabilidad	93 %
Mantenibilidad	85 %
Evaluación Total de Calidad	89.07%

4.3. Estimación del Costo

Para realizar la estimación de costos del software desarrollado se utiliza el método algorítmico de aproximación COCOMO II, orientado a los puntos de función ajustados. Para este fin, en primer lugar, se calcula el punto función.

Datos presentes por puntos de función ya obtenidos:

- Puntos de función sin ajustar: $PFSA$ = 202
- Punto de función ajustado PFA = 248.46

Cálculo de esfuerzo

Se obtiene mediante la formula:

$$LDC = PFA \times (\text{factor } LDF/\ PF)$$

Donde:

- LDC = Líneas de Código
- PFA = Puntos de función ajustados
- Factor LCD/PF = Factor Líneas de código por punto función.

Tabla 28

Tabla de conversión factor LDC/PF

Lenguaje	Factor LDC/PF
C	128
Java	53
Lenguaje OO	19
C++	53
JavaScript	47

Utilizamos el factor para el lenguaje orientado a objetos y reemplazamos los valores.

$$LDC = 248.46 \times 19$$

$$LDC = 4720.74$$

Posterior a ello se convierte LDC a KLDC dividiendo el resultado entre 1000.

$$LDC = 4720.74/1000$$

$$KLDC = 4.72$$

Por esta razón, en nuestro caso el tipo orgánico será el más apropiado ya que el número de líneas de código es menor a los 50 KLDC, y además el proyecto no es muy complicado, por consiguiente, los coeficientes que usaremos serán las siguientes:

Tabla 29

Modelo Cocomo II

Proyecto Software	***a***	***b***	***c***	***d***
Orgánico	3.2	1,05	2.5	0,38

Tabla 30

Ajuste del factor de complejidad

Conductores de coste	**Valoración**					
	Muy bajo	**Bajo**	**Normal**	**Alto**	**Muy Alto**	**Extra Alto**
Fiabilidad requerida del software	0,75	0,88	1,00	**1,15**	1,40	
Tamaño de la base de datos		0,94	1,00	**1,08**	1,16	
Complejidad del producto	0,70	0,85	1,00	**1,15**	1,30	1,65
Restricciones del tiempo de ejecución			1,00	**1,11**	1,30	1,66
Restricciones del almacenamiento principal			**1,00**	1,06	1,21	1,56

Volatilidad de la máquina virtual		0,87	**1,00**	1,15	1,30
Tiempo de respuesta del ordenador		0,87	1,00	**1,07**	1,15
Capacidad del analista	1,46	1,19	1,00	**0,86**	0,71
Experiencia en la aplicación	1,29	1,13	1,00	0,91	**0,82**
Capacidad de los programadores	1,42	1,17	1,00	0,86	**0,70**
Experiencia en S.O. utilizado	1,21	1,10	1,00	**0,90**	
Experiencia en el lenguaje de programación	1,14	1,07	1,00	**0,95**	
Prácticas de programación modernas	1,24	1,10	1,00	**0,91**	0,82
Utilización de herramientas software	1,24	1,10	**1,00**	0,91	0,83
Limitaciones de planificación del proyecto	1,23	1,08	**1,00**	1,04	1,10

$\mathbf{FAE}$ = 1,55 * 1,08 * 1,15 * 1,11 * 1 * 1 * 1,07 * 0,86 * 0,82 * 0,70 * 0,90 * 0,95 * 0,91 * 1 * 1 $\boldsymbol{FAE}$ = 0,6

4.3.1. Cálculo del Esfuerzo de Desarrollo

$$E = a \times (\mathrm{KLDC})^{e} \times FAE = 3{,}2 \times 4.72^{1{,}05} \times 0{,}6$$

$$E = 9{,}7\ personas\ /mes$$

4.3.2. *Cálculo del Tiempo de Desarrollo*

$$T = c(\mathrm{Esfuerzo})^{d} = 2{,}5 \times 9.7^{0.38} = 5.9$$

$$T = 6\ meses\ Es\ el\ tiempo\ que\ tomará\ desarrollar\ el\ proyecto$$

4.3.3. Cálculo de Productividad

$$PR = LDC/\ \mathrm{Esfuerzo}$$

$$PR = 4720.74/9.7$$

$$PR = 486.6\ (LDC/personas\ mes)$$

4.3.4. Cálculo de Personal Promedio

$$P = E\ /T$$

$$P = 9.7/6$$

$$P = 1.6$$

Según estas cifras será necesario un equipo de 2 personas trabajando alrededor de 6 meses.

4.3.5. Costo de desarrollo

Para el presente proyecto se ha establecido el salario de un desarrollador el cuál es 3000 Bs. El costo total de desarrollo del proyecto, se calcula mediante:

$$\boldsymbol{Costo\ de\ desarrollo = nro.\ de\ programadores * Tiempo\ estimado * Salario}$$

$$Costo\ de\ desarrollo = 1 * 6 * 3000$$

$$Costo\ de\ desarrollo = 18000\ Bolivianos,$$

Por tanto, el costo total será de 18000 bolivianos.

4.4. Sistema de Gestión de la Seguridad de la Información

La ISO 27002, forma parte de la serie de normas 27000 verifica y evalúa su implementación, en cuanto a la seguridad se detalla los siguientes aspectos.

4.4.1. Seguridad de Base de Datos

NestJS es un Framework para Node.js en el que se utiliza Typescript para escribir código. Una de las mejores cualidades que tiene este Framework es que utiliza la inyección de dependencias lo cual nos permite tener un código bien estructurado y fácil de escalar. Bcrypt es una librería en Node.js que proporciona funciones para el cifrado seguro de contraseñas. En términos simples, ayuda a proteger las contraseñas de los usuarios en la aplicación móvil. La principal ventaja de utilizar bcrypt radica en su capacidad para hashear contraseñas de una manera segura y eficiente.

4.4.2. Seguridad lógica

La seguridad lógica es un conjunto de técnicas y medidas para proteger los datos confidenciales de la aplicación móvil contra posibles amenazas externas e internas.

NestJS provee dos librerías para validar los requests. La combinación de las bibliotecas class-validator y class-transformer es muy útil para validar y transformar datos en la aplicación móvil. JSON Web Tokens (JWT) permite que los usuarios se registren en la aplicación móvil y accedan a sus cuentas de manera segura. El proceso de ingreso a la App es fundamental mediante la autenticación de usuarios. Con JWT, podremos generar tokens de acceso que protegen las rutas y garantizarán que solo los usuarios autenticados puedan acceder a determinados recursos.

4.4.3. Seguridad física

El control al acceso físico de los dispositivos móviles es esencial. Los usuarios deben proteger sus dispositivos mediante el uso de contraseñas, PIN o autenticación biométrica, como la huella digital o el reconocimiento facial. Se recomienda realizar, Back-up de la base de datos y estas deberán estar protegidas y tener acceso al personal autorizado.

4.4.4. Seguridad de la App

Para la seguridad de la aplicación se consideran las siguientes precauciones, los cuales son:

- Autentificación de los usuarios.
- Manejo de privilegios y tipos de usuarios en la aplicación.
- Manejo de vistas.
- Integridad y control de datos por APIREST.
- Encriptación con Bcrypt de Node.js en las contraseñas.

4.5. Pruebas de Software

Se tratará de encontrar todos los errores posibles durante un proceso antes de que se ingrese en la aplicación, mediante las siguientes pruebas realizadas.

4.5.1. *Pruebas de Caja Blanca*

Esta prueba se aplica en los procesos más relevantes del modelo de pronóstico de demanda y el sistema en cual se realiza su aplicación siendo técnica de prueba del camino básico mediante grafos, complejidad ciclo matica y derivación de casos de prueba.

De acuerdo a ello la complejidad del flujo es V (G) y puede calcularse en tres alternativas:

Identificar el número de regiones del grafo de flujo, aplicando un segmento de código fuente.

V(G) = A – N + 2, donde:

A: Es el número de aristas

N: es el número de nodos.

V (G) = P + 1, donde P es números de nodos predicado

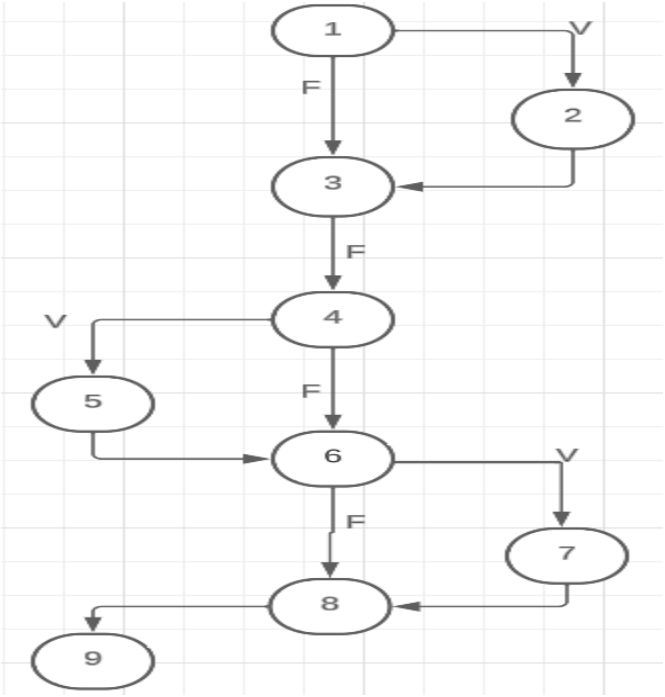
Figura 64

Proceso de Autenticación

De la figura 64 se genera los siguientes nodos.

Figura 65

Grafo de prueba de caja blanca



Para las pruebas de caja blanca se proporcionará una medición cuantitativa de la complejidad lógica del programa, se tomarán las siguientes variables:

A = 11

N = 9

$V(G)$ = 11 – 9 + 2 = 4

$V(G)$ = 4

Por lo tanto, existe una complejidad ciclo matica de 4 (11 aristas y 9 nodos).

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

Habiendo cumplido los requerimientos establecidos por la empresa, se ha logrado alcanzar el objetivo planteado por haber terminado el desarrollo de una aplicación Android que resuelve los problemas de gestión de la comercialización de productos de la empresa Importadora y Distribuidora Telaza Textil Bolivia.

4.5.2. Respecto al Objetivo General

Se cumple al determinar el proyecto con el objetivo general desarrollar e implementar una aplicación móvil de información para la gestión comercial de ventas y pedidos para la empresa Importadora y Distribuidora Telaza que le permita administrar la información de manera oportuna, agilizando y mejorando los procesos para la comercialización de productos la cual fue detallada en el capítulo III.

4.5.3. Respecto a los Objetivos Específicos

- Se logro diseñar una interfaz de usuario enfocada en brindar una mejor experiencia UX/UI y de fácil manejo, mediante la Framework de Flutter para una mejor administración y organización de la información de los productos de la empresa.
- Se implemento la base de datos según los requerimientos garantizando el acceso autorizado, para el almacenamiento de la información que maneja la empresa.
- Se implemento un proceso de autenticación seguro que cumpla con los estándares de seguridad recomendados, garantizando que los datos del usuario estén protegidos contra amenazas como ataques de suplantación de identidad y fugas de información.
- Se elaboro el módulo de categorías donde se pueda visualizar la descripción, precio de cada producto, para ofrecer un catálogo de productos conforme a la necesidad del cliente.
- Se crearon módulos de acceso rápido a la información actualizada donde se detalle la orden de la compra del producto, agilizando el proceso de los mismos.

- Se desarrollar un módulo para el repartidor, para notificar sobre los pedidos, donde tendrá la información del producto a entregar en tiempo real.
- Se realizo el módulo del repartidor o delivery al proporcionar rutas eficientes y precisas en la aplicación móvil utilizando servicios de Google, garantizando la visualización rápida de instrucciones de navegación claras y oportunas.
- Se implemento un módulo de historial donde se muestra los datos sobre información de las ventas según la fecha, para proporcionar la información del producto comprado por el cliente.

5.2. Recomendaciones

A partir del presente trabajo y en base a las observaciones realizadas, se propone la implementación de las siguientes recomendaciones para actualizar constantemente la aplicación móvil.

- Se recomienda a los usuarios cambiar la contraseña de acceso a la App. periódicamente, para mejorar la seguridad y evitar el acceso no autorizado.
- Para preservar los datos internos, se recomienda crear una copia de seguridad de la base de datos para evitar pérdida de información.
- En relación con el análisis y diseño de la aplicación y creación de nuevos módulos, se recomienda revisarlos primero documentación para poder tomar buenas decisiones gracias a la aplicación móvil que presenta elementos reutilizables para implementar nuevos módulos y otros.
- Se recomienda hacer pruebas de estrés a los servicios en caso ser implementado.
- Se recomienda crear un canal o área de soporte para realizar el seguimiento de casos o para atención al cliente.

REFERENCIAS

Amaya Balaguera, Y. D. (2013). Metodologías ágiles en el desarrollo de aplicaciones. *Investigación científica y tecnológica terminada*, 111-124.

Atlassian. (2023). *Desarrollo de software*. Obtenido de atlassian.com: https://www.atlassian.com/es/agile

Blanco, P., Camarero, J., Fumero, A., Werterski, A., & Rodríguez, P. (2009). *Metodología de desarrollo ágil para sistemas móviles Introducción al desarrollo con Android y el iPhone.* dit.

Bohn, R., Lee, C., & Michel, M. (12 de Mayo de 2020). *NIST Cloud Computing Program - NCCP*. Obtenido de nist.gov: https://www.nist.gov/programs-projects/nist-cloud-computing-program-nccp

Buckett, C. (2013). *Dart in Action.* Manning.

Carbonell, V., Bataller, J., & Tomás, J. (2019). *Firebase Trabajar En La Nube.* ALFAOMEGA MARCOMBO.

Carrasco Navarro, J. (2020). *Desarrollo de aplicaciones móviles en Kotlin: Introducción a la programación móvil.*

Comité Internacional de Certificaciones de Pruebas de Software (ISTQB). (2022). *Advanced Level Test Analyst*. Obtenido de istqb.org: https://www.istqb.org/help/test-analyst

Croll, A., & Yoskovitz, B. (2014). *Lean Analytics: Cómo utilizar los datos para crear más rápido una startup mejor* (1ra ed.). UNIR.

Duffy, T. (2012). *Programación con aplicaciones móviles: Android™, iOS y Windows Phone 7* (1ra ed.).

Enriquez, J., & Casas, S. (2013). USABILIDAD EN APLICACIONES MÓVILES. *dialnet.unirioja, 5*, 25-47.

ESGInnova. (2023). *La norma ISO/IEC 27000 va a ser revisada*. Obtenido de isotools.us: https://www.isotools.us/2018/03/05/la-norma-iso-iec-27000-va-a-ser-revisada/

Flores, F. (22 de Marzo de 2021). *Cloud Computing: Tipos de nubes, servicios y proveedores*. Obtenido de openwebinars.net: https://openwebinars.net/blog/tipos-de-cloud-computing/

Fuentes Jiménez, P. (2010). LA ORIENTACIÓN AL MERCADO: EVOLUCIÓN Y MEDICIÓN DE UN ENFOQUE DE. *PERSPECTIVAS*(25), 25-83. Obtenido de https://www.redalyc.org/articulo.oa?id=425942454004

Garcia, J. (15 de Julio de 2021). *¿Qué son las métricas de software? Definición | Programación*. Obtenido de tecno-simple.com: https://tecno-simple.com/que-son-las-metricas-de-software/#El_estandar_IEEE_1061_define_metricas_de_calidad_del_software

Hernández Aro, Y., & Machado Chaviano, E. (2010). Estrategia de posicionamiento del destino Villa Clara en el mercado chino. *Teoría y Praxis*(7), 55-77. Obtenido de https://www.redalyc.org/articulo.oa?id=456145285004

Hernandez, R. (28 de Junio de 2021). *El patrón modelo-vista-controlador: Arquitectura y frameworks explicados*. Obtenido de freecodecamp.org: https://www.freecodecamp.org/espanol/news/el-modelo-de-arquitectura-view-controller-pattern/

IBM. (s.f.). *¿Qué es el desarrollo de aplicaciones móviles?* Recuperado el 12 de septiembre de 2023, de ibm.com: https://www.ibm.com/mx-es/topics/mobile-application-development

Instituto de Ingenieros Eléctricos y Electrónicos (IEEE, 2014). (13 de Junio de 2014). *730-2014 - IEEE Standard for Software Quality Assurance Processes*. Obtenido de IEEE: https://ieeexplore.ieee.org/servlet/opac?punumber=6835309

International Business Machines [IBM]. (s.f.). *Tecnología móvil*. Recuperado el 12 de septiembre de 2023, de ibm.com: https://www.ibm.com/es-es/topics/mobile-technology

Jimenez Canquiz, E. J., & Bolivar, G. (2016). *MADEX Metodología Ágil de Desarrollo eXtremo.*

Johnson, B. (2019). *Visual Studio Code End-to-End Editing and Debugging Tools for Web Developers.* Wiley.

Kotler, P. (2007). *MARKETING: VERSION PARA LATINOAMERICA.*

Kotler, P. (2014). *Kotler on Marketing: How to Create, Win, and Dominate Markets.*

Kotler, P., & Armstrong, G. (2008). *Fundamentos de marketing.* Mexico: Pearson Educación de México, S.A. de C.V.

Kotler, P., Kartajaya, H., & Setiawan, I. (2021). *Marketing 5.0 (ACCION EMPRESARIAL).* LID EDITORIAL.

Lardiere, S. (2019). *PostgreSQL Administración y explotación de sus bases de datos.* Eni.

Lopez Barra, S., & Ruiz Otero, E. (2022). *Operaciones administrativas de recursos humanos.* McGraw-Hill Interamericana de España S.L.

Lujan Castillo, J. D. (2017). *ANDROID STUDIO APRENDE A DESARROLLAR APLICACIONES.* MADRID: RC LIBROS.

Luna, F. (2014). *Desarrollo Web Para Dispositivos Móviles.* Buenos Aires: Fox Andina S.A.

Meneses, M. (2015). *7 Estrategias De Internet Marketing: Para Crear Tu Propio Imperio Online* (1ra ed. ed.).

Monferrer Tirado, D. (2013). Fundamentosde marketing. *Universitat Jaume I*, 175. Obtenido de http://hdl.handle.net/10234/49394

Morales, G. (2023). *API con Node.js, Express y Prisma.*

Napoli, M. (2019). *Beginning Flutter A Hands On Guide to App Development.* Wrox.

Norton, P. (2005). *Computing Fundamentals.* McGraw-Hill Education.

Organización Internacional de Estandarización (ISO). (Febrero de 2018). *ISO/IEC 27000:2018 Information technology Security techniques Information security management systems*. Obtenido de iso.org: https://www.iso.org/standard/73906.html

Organización Internacional de Normalización y la Comisión Electrónica internacional, (ISO/IEC 9126-1, 2001) . (01 de Marzo de 2011). *Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models*. Obtenido de webstore.iec.ch: https://webstore.iec.ch/publication/11245

Phillips, A. (2022). *Aws: La guía de estudio completa para profesionales certificados en la nube para aprender los principios de AWS.*

Pressman, R. (2010). *Ingeniería del software* (7ma ed.). México, D. F.: McGRAW-HILL INTERAMERICANA EDITORES, S.A. DE C.V.

Ramírez, I. (25 de Julio de 2023). *Todas las versiones de Android de la historia*. Recuperado el 13 de septiembre de 2023, de xatakandroid: https://www.xatakandroid.com/sistema-operativo/todas-versiones-android-historia

Ruelas, A. (2014). El teléfono celular y los jóvenes sinaloenses. Adopción, usos y adaptaciones. *Comunicación y Sociedad*(21), 101-131. Obtenido de https://www.redalyc.org/articulo.oa?id=34631113005

Salazar, R. (2022). *medium*. Recuperado el 12 de septiembre de 2023, de Dispositivos moviles: https://medium.com/@ricardo.salazarc/dispositivos-móviles-fd0f0bc48a60

Sánchez, E. (17 de Mayo de 2021). *Estudio sobre comportamiento digital destaca: 3 de 10 bolivianos confiesan perder la noción del tiempo al conectarse*. Obtenido de Ariadna Communications Group: https://www.inforse.com.bo/2021/05/17/estudio-sobre-comportamiento-digital-destaca-3-de-10-bolivianos-confiesan-perder-la-nocion-del-tiempo-al-conectarse/

Sandeep, B. (29 de Julio de 2023). *Pruebas de Caja Blanca vs Caja Negra Diferencia y Comparación*. Obtenido de askanydifference.com: https://askanydifference.com/es/difference-between-white-box-and-black-box-testing/

Santesmases Mestre, M., Valderrey Villar, F., & Sánchez Guzmán, A. (2014). *Fundamentos de MERCADOTECNIA.* Mexico: GRUPO EDITORIAL PATRIA, S.A. DE C.V.

Serna, S. (2016). *Diseño de interfaces en aplicaciones móviles.* RA-MA S.A.

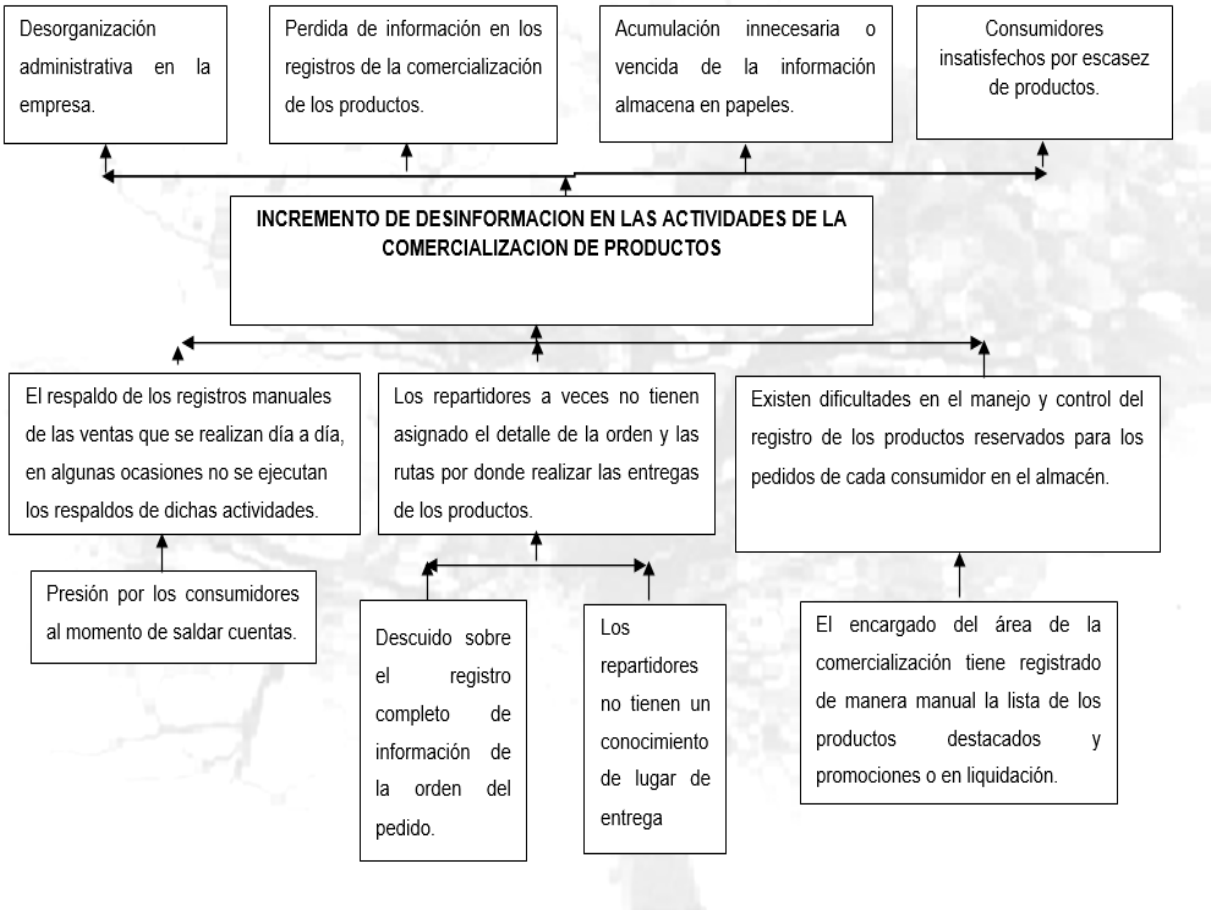
Sommerville, I. (2011). *INGENIERÍA DE SOFTWARE* (9na ed.). Mexico: PEARSON.

Tanenbaum, A. (2009). *SISTEMAS OPERATIVOS MODERNOS* (3ra ed.). Mexico: Pearson Educación de México, S.A. de C.V.

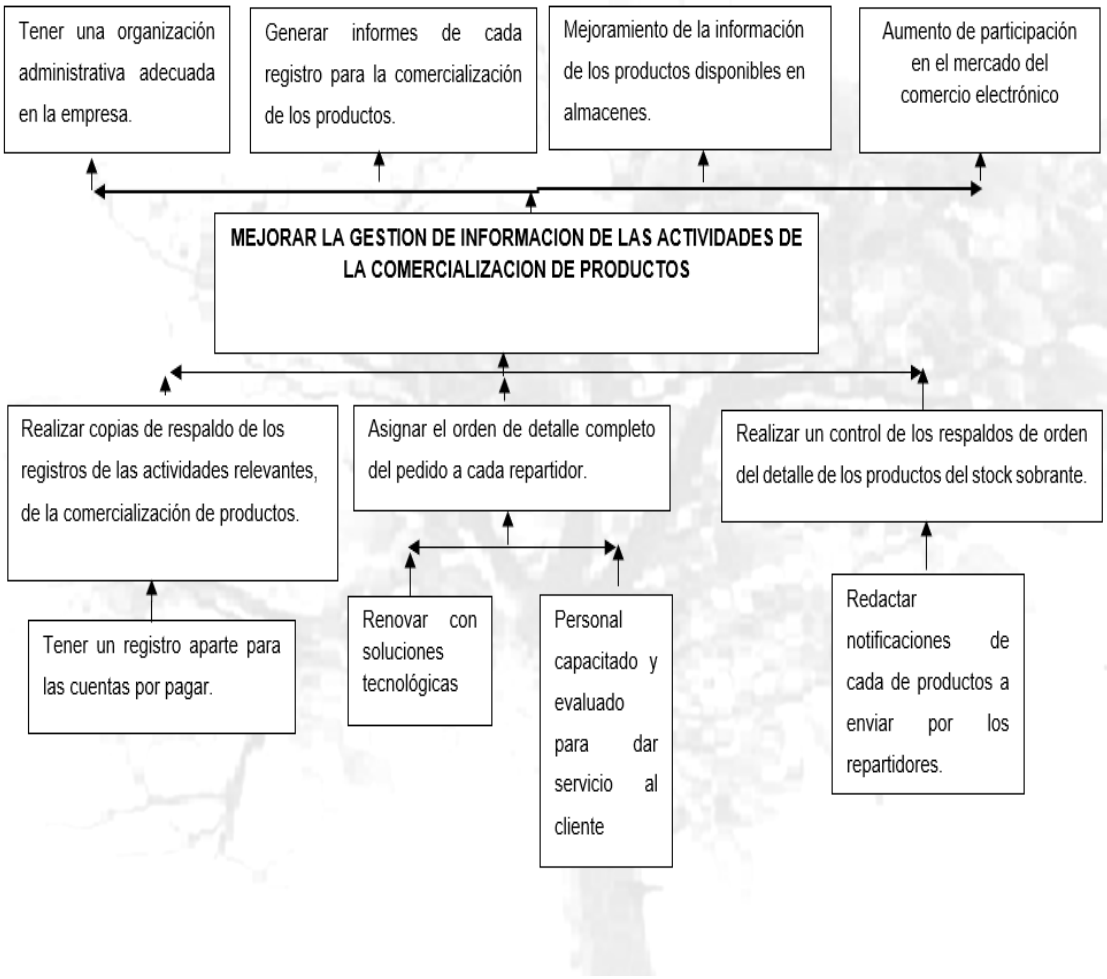
Verity. (28 de Julio de 2022). *La ISO/IEC 9126: 2001: Características de la calidad de software*. Obtenido de verity.cl: https://www.verity.cl/que-es-norma-iso-iec-9126-2001/

ANEXOS

ARBOL DE PROBLEMAS



ARBOL DE OBJETIVOS



AVALES

El Alto, 24 de noviembre de 2023

Señor:
LIC. ING. WILLIAM ROQUE ROQUE
DIRECTOR DE CARRERA
INGENIERÍA DE SISTEMAS
Presente. -

REF.: AVAL DE CONFORMIDAD

Distinguido director de carrera:
Mediante la presente tengo a bien comunicarle mi conformidad del Trabajo de Grado:

TITULO: APLICACIÓN MÓVIL PARA LA COMERCIALIZACIÓN DE PRODUCTOS MEDIANTE SERVICIOS EN LA NUBE
CASO: EMPRESA IMPORTADORA Y DISTRIBUIDORA TELAZA
MODALIDAD: PROYECTO DE GRADO
Univ. WILMER CRUZ QUISPE
Registro Universitario: 12000115
Cedula de Identidad: 8402978 LP.

Para su defensa pública, evaluación correspondiente a la materia de Taller de Grado II, de acuerdo al reglamento vigente de la Carrera Ingeniería de Sistemas de la Universidad Pública de El Alto.

Atentamente,

Lic. Ing. Dionicio Henry Pacheco Ríos
TUTOR METODOLÓGICO
TALLER DE GRADO II

El Alto, 24 de noviembre de 2023

Señor:
LIC. ING. DIONICIO HENRY PACHECO RÍOS
TUTOR METODOLÓGICO
TALLER DE GRADO II
Presente. -

REF.: AVAL DE CONFORMIDAD

Distinguido tutor metodológico:
Mediante la presente tengo a bien comunicarle mi conformidad del Trabajo de Grado:

TITULO: APLICACIÓN MÓVIL PARA LA COMERCIALIZACIÓN DE PRODUCTOS MEDIANTE SERVICIOS EN LA NUBE
CASO: EMPRESA IMPORTADORA Y DISTRIBUIDORA TELAZA
MODALIDAD: PROYECTO DE GRADO
Univ. WILMER CRUZ QUISPE
Registro Universitario: 12000115
Cedula de Identidad: 8402978 LP

Para su defensa pública, evaluación correspondiente a la materia de Taller de Grado II, de acuerdo al reglamento vigente de la Carrera Ingeniería de Sistemas de la Universidad Pública de El Alto.

Atentamente,

M. Sc. Lic. Ing. Neil Ramiro Gonzales Burgoa
TUTOR REVISOR

El Alto, 23 de noviembre de 2023

Señor:
LIC. ING. DIONICIO HENRY PACHECO RÍOS
TUTOR METODOLÓGICO
TALLER DE GRADO II
Presente. -

REF.: AVAL DE CONFORMIDAD

Distinguido tutor metodológico:
Mediante la presente tengo a bien comunicarle mi conformidad del Trabajo de Grado:

TITULO: APLICACIÓN MÓVIL PARA LA COMERCIALIZACIÓN DE PRODUCTOS MEDIANTE SERVICIOS EN LA NUBE
CASO: EMPRESA IMPORTADORA Y DISTRIBUIDORA TELAZA
MODALIDAD: PROYECTO DE GRADO
Univ. WILMER CRUZ QUISPE
Registro Universitario: 12000115
Cedula de Identidad: 8402978 LP.

Para su defensa pública, evaluación correspondiente a la materia de Taller de Grado II, de acuerdo al reglamente vigente de la Carrera de Ingeniería de Sistemas de la Universidad Pública de El Alto.
Atentamente.

M. Sc. Lic. Ing. Dulfredo Villca Lázaro
TUTOR ESPECIALISTA

TELAZA
EMPRESA DISTRIBUIDORA DEL SECTOR TEXTIL

El Alto, 24 de noviembre de 2023

Señor:
ING. DIONICIO HENRY PACHECO RÍOS
TUTOR METODOLÓGICO
TALLER DE GRADO II
Presente. -

REF.: AVAL DE CONFORMIDAD

Distinguido tutor metodológico:
Mediante la presente tengo a bien comunicarle la conformidad del Trabajo de Grado:

TITULO: APLICACIÓN MÓVIL PARA LA COMERCIALIZACIÓN DE PRODUCTOS MEDIANTE SERVICIOS EN LA NUBE
CASO: EMPRESA IMPORTADORA Y DISTRIBUIDORA TELAZA
MODALIDAD: PROYECTO DE GRADO
UNIVERSITARIO: WILMER CRUZ QUISPE
REGISTRO UNIVERSITARIO: 12000115
CEDULA DE IDENTIDAD: 8402978 LP.

De tal forma cabe recalcar que la APLICACIÓN MÓVIL satisface los requerimientos de la institución de esta forma se dio cumplimiento de los objetivos del presente.

La presente **APLICACIÓN MÓVIL** fue **IMPLEMENTADO** satisfactoriamente en la institución.

En cuanto certifico, en honor a la verdad, para fines consiguientes del interesado para su defensa pública y evaluación correspondiente a la materia de Taller de Grado II, de acuerdo al reglamento vigente de la Carrera Ingeniería de Sistemas de la Universidad Pública de El Alto.

Atentamente,

Ing. Nataly F. Laura Ventura
GERENTE GENERAL
TELAZA
Cel.: 73033321

TELAZA
EMPRESA DISTRIBUIDORA DEL SECTOR TEXTIL
Tel: 73033321

MANUALES

MANUAL DE USUARIO

Introducción

La presente Aplicación Móvil de control de la gestión comercial fue desarrollado con el objetivo de automatizar los procesos relacionados a las ventas. Esta guía ayudará a sacar el máximo provecho de la aplicación móvil, permitiendo realizar compras, administración de pedidos y visualizar catálogo de productos de manera eficiente y segura.

Descripción Inicio de la App Telaza

Botón para Ingresar como usuario invitado.

Botón para ingresar con las credenciales de autorización.

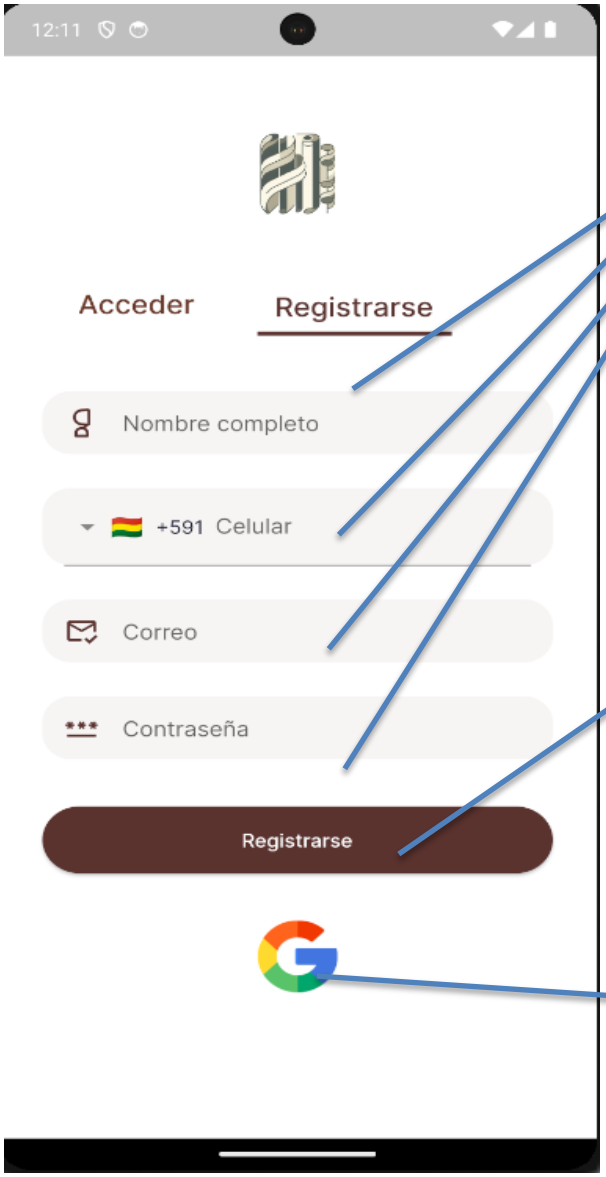
Pantalla de Acceso

Regístrate con tus datos personales.

Inicia sesión con Google.

Si olvidaste tu contraseña puedes recuperar tu cuenta.

Pantalla de Registro



Ingresar datos en todos los campos.

Una vez llenado el formulario presionar el botón de Registrarse para acceder a la pantalla principal.

También puede Registrarse con Google.

Pantalla de Principal

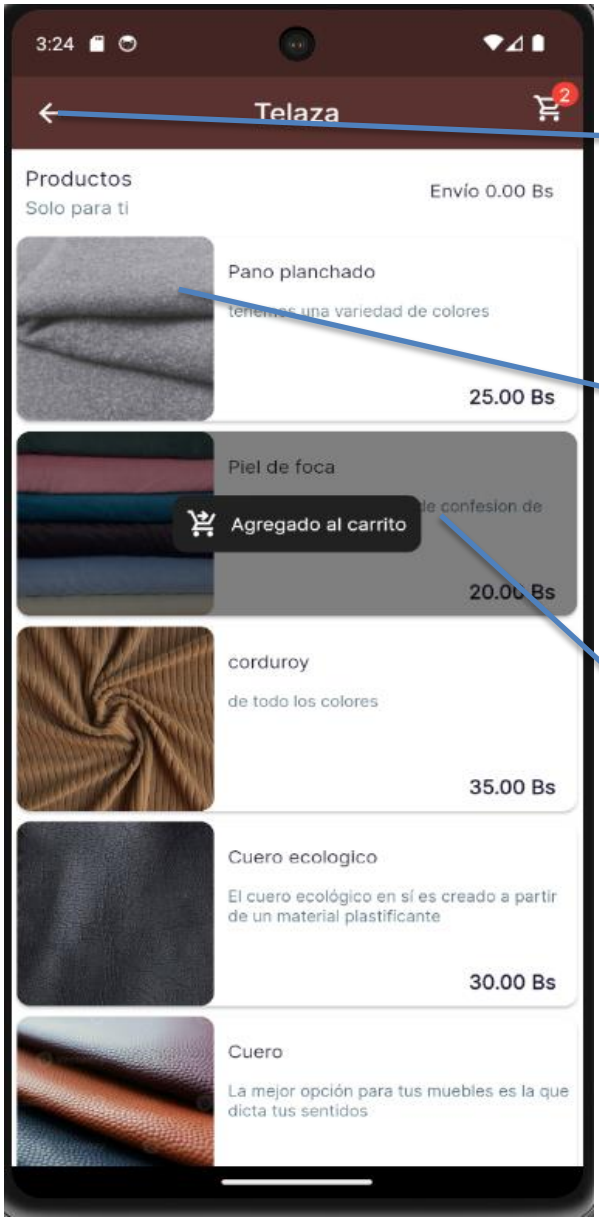
Una vez iniciado sesión muestra la pantalla principal.

Filtrado o búsqueda por categorías.

Click para ingresar a una tienda y ver el catálogo de los productos.

Navegación de pantallas.

Pantalla de Catálogo de productos

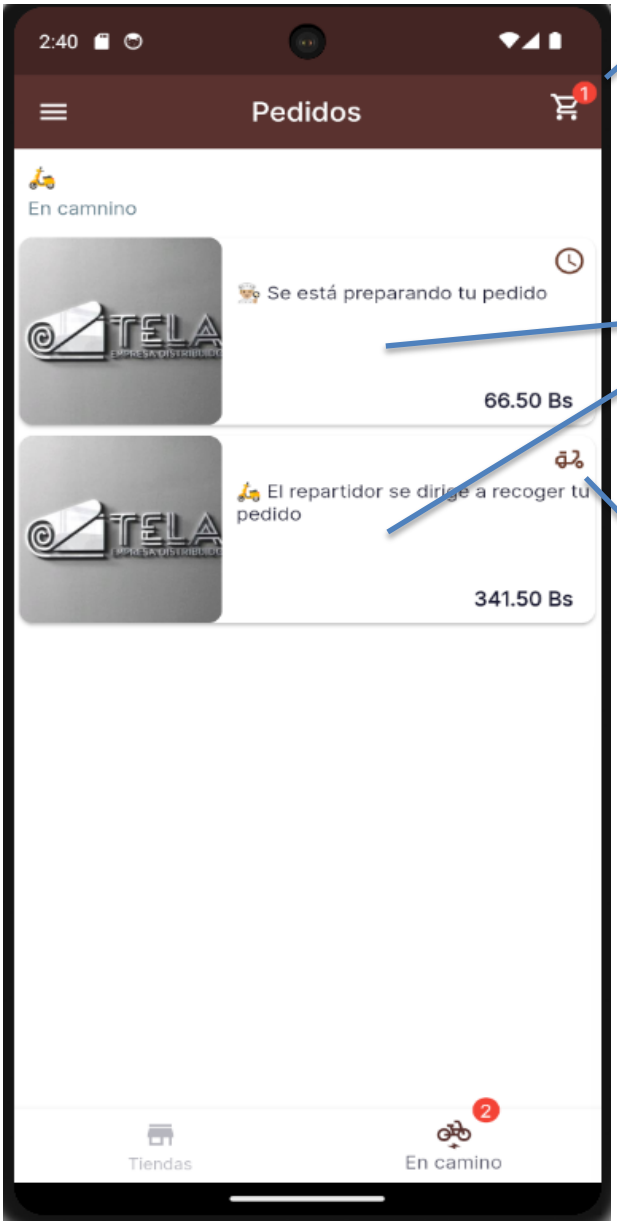


Botón para volver atrás o a la pantalla anterior.

Catálogo de productos.

Click sobre un producto para agregar o quitar del carrito de compras.

Pantalla de Pedidos

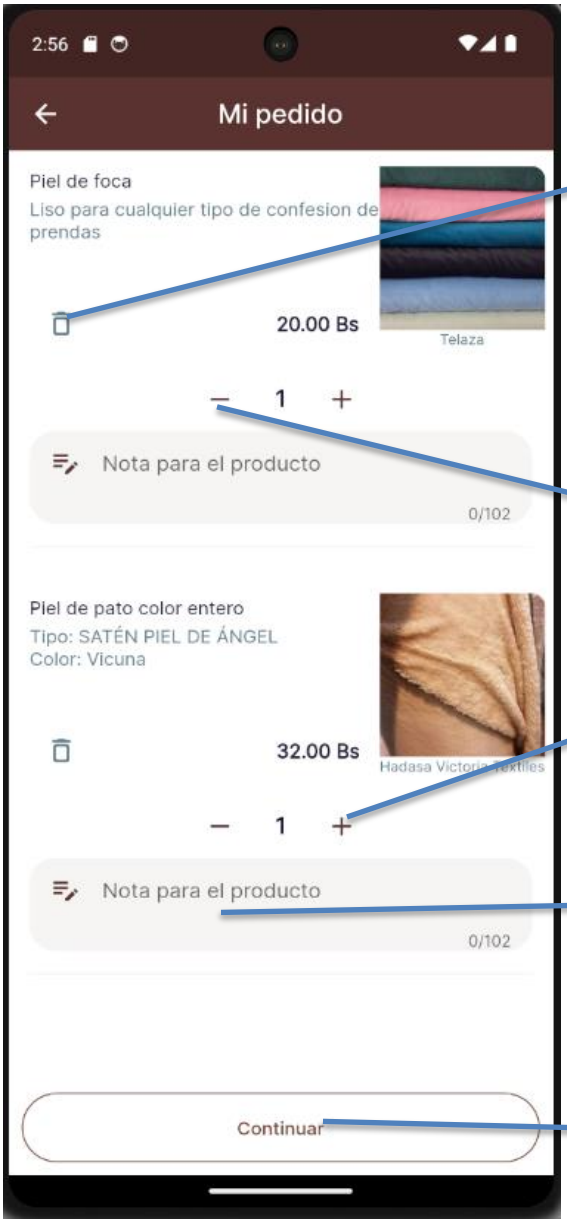


Cantidad de productos agregados al carrito de compras.

Lista de productos comprados.

Estado del pedido.

Pantalla carrito de compras



Eliminar producto del carrito de compras.

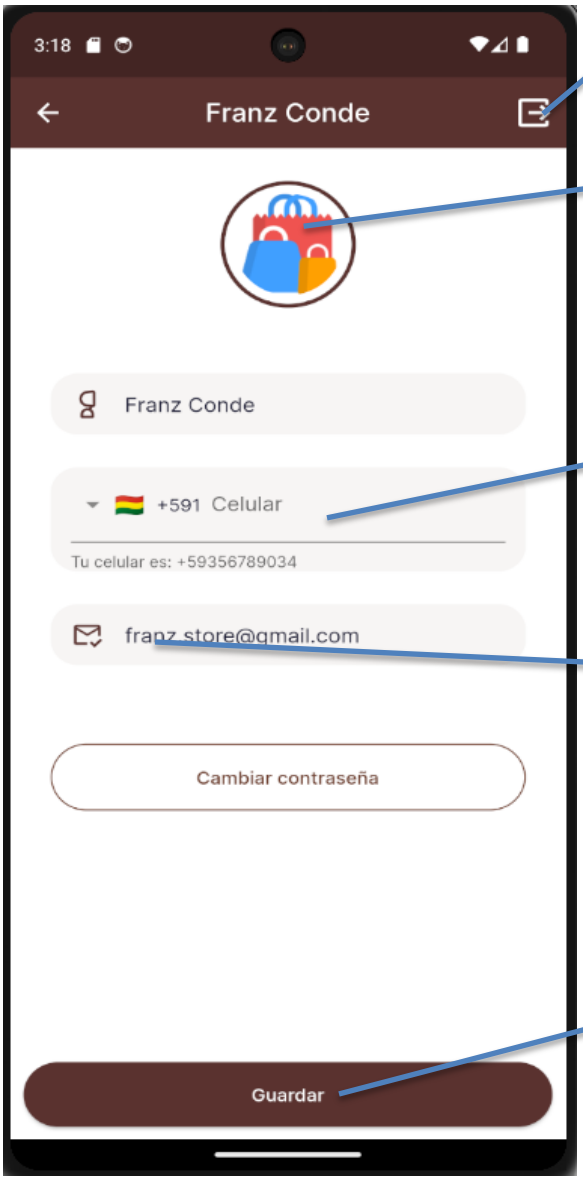
Disminuir cantidad del producto.

Agregar cantidad de metros del producto.

Nota para agregar texto.

Botón para confirmar pedido.

Pantalla perfil del usuario



Botón cerrar sesión.

Actualizar avatar o imagen de perfil.

Actualizar datos si es necesario.

Actualizar correo electrónico.

Botón para realizar o guardar cambios.

USUARIO CAJERO

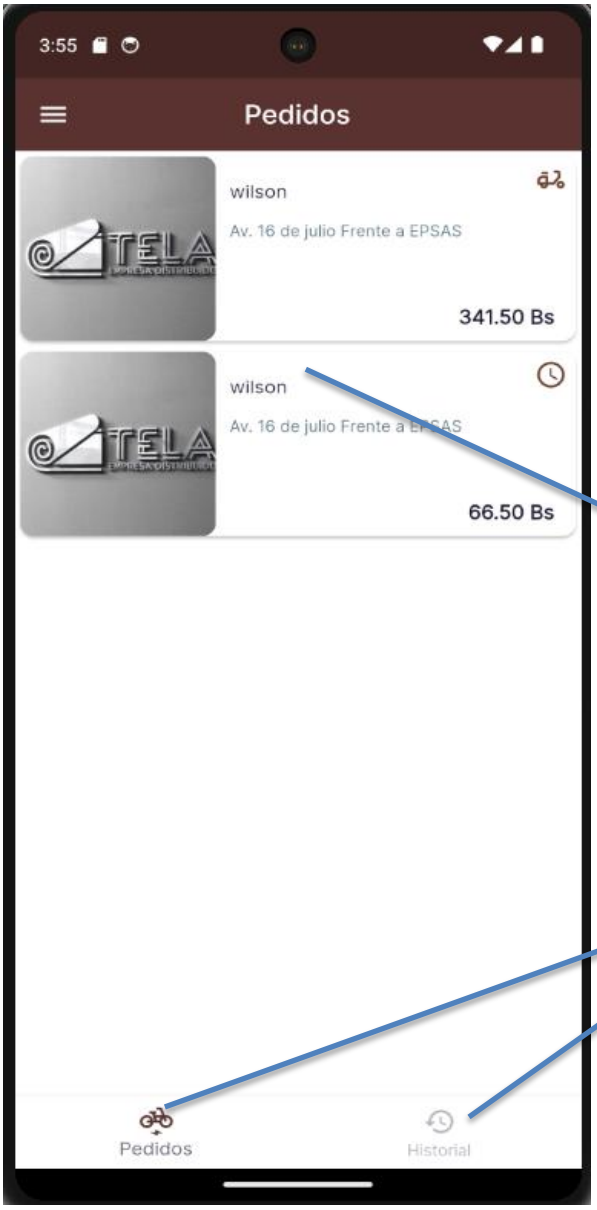
Pantalla del usuario cajero

Listar tiendas registradas por el usuario.

Agregar residencia del usuario.

Registrar una nueva tienda.

Pantalla de lista de pedidos

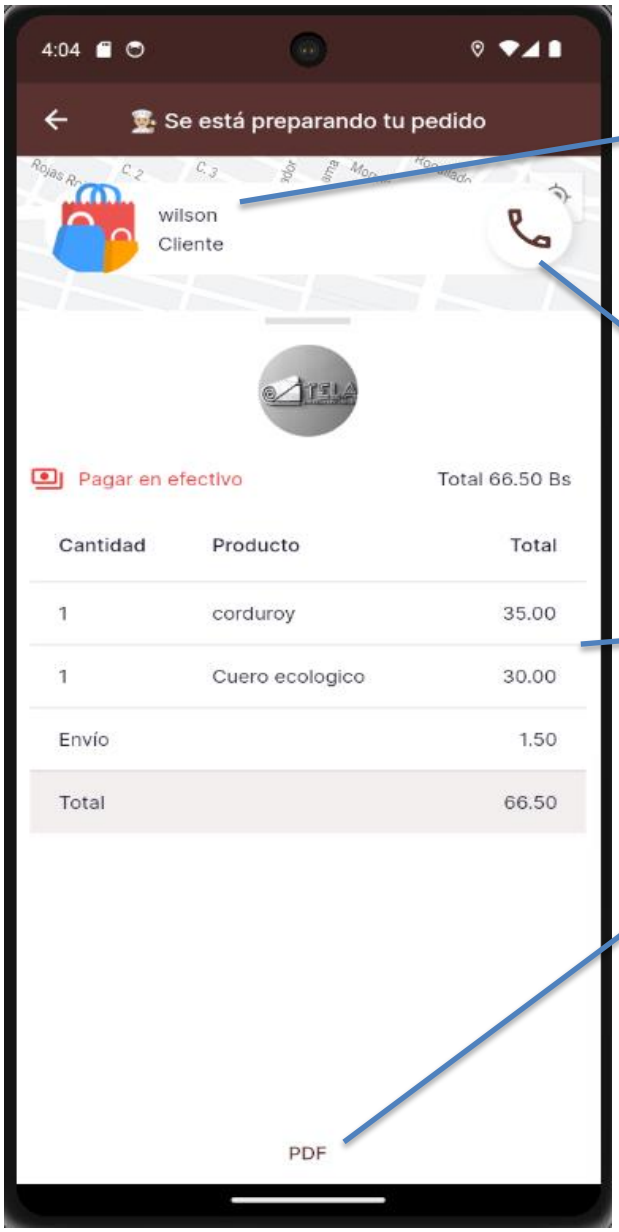


Lista de los pedidos.

Click sobre el contenedor para ver detalles del pedido.

Navegador de pantallas.

Pantalla detalles del pedido



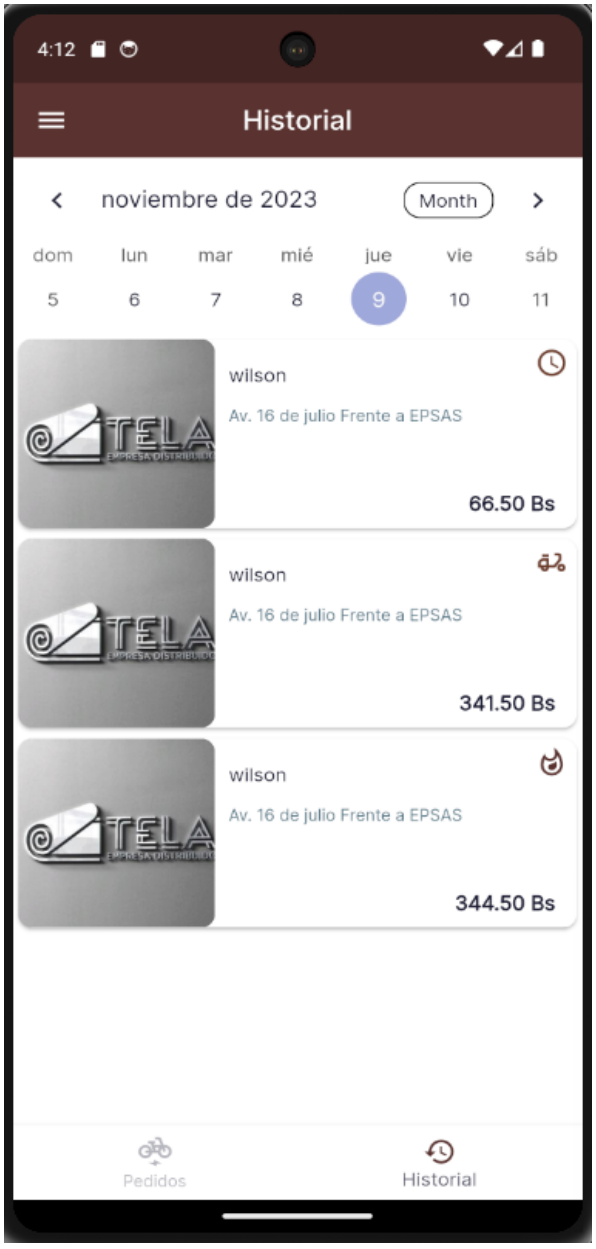
Información del cliente.

Botón para llamar al cliente.

Información del pedido.

Reporte en PDF del pedido.

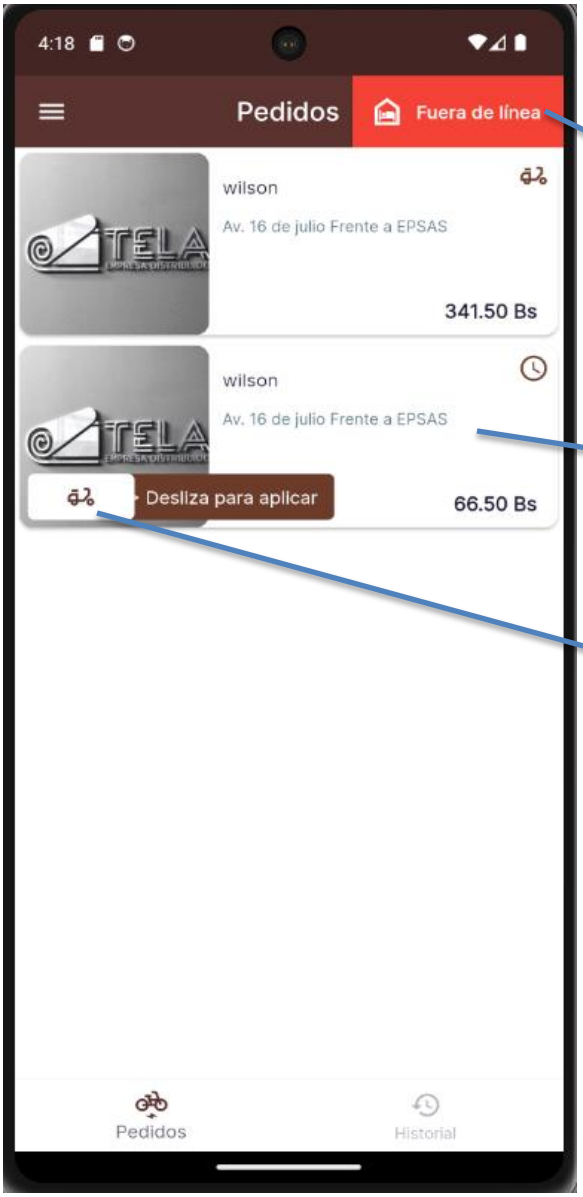
Pantalla de historial de los pedidos



Historial de pedidos por fecha.

USUARIO REPARTIDOR

Pantalla del usuario repartidor o delivery

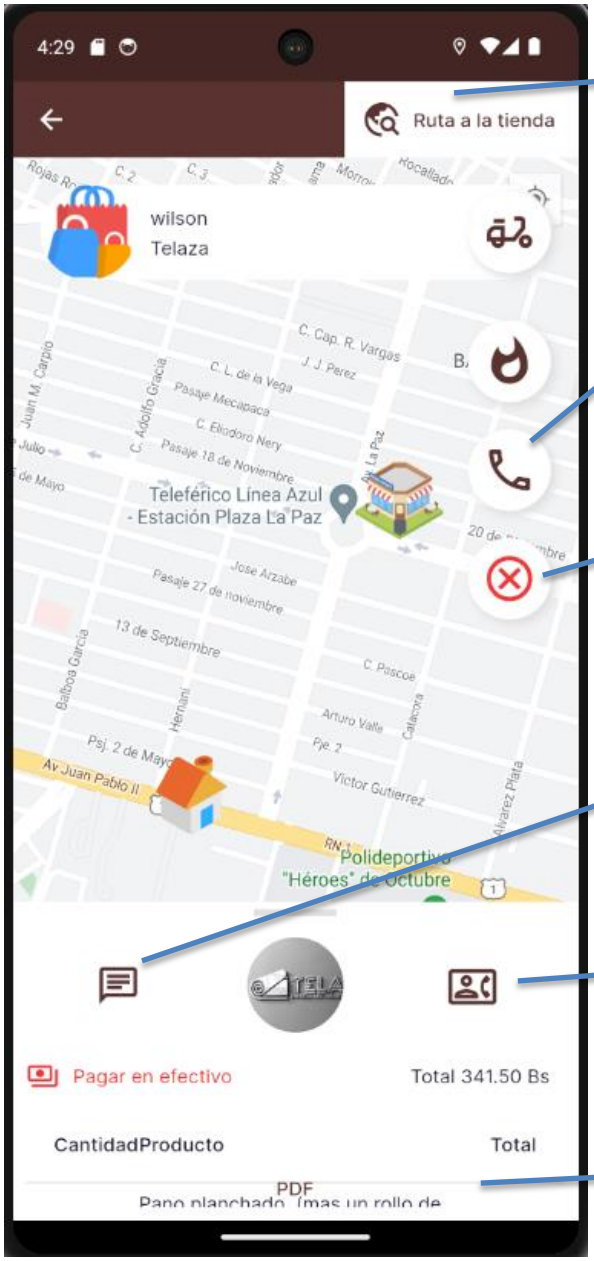


Botón para recibir notificaciones por SMS.

Click para ver detalles del producto y rutas en tiempo real para realizar la entrega.

Click para realizar el pedido.

Pantalla detalles del producto y rutas



Botón para navegar con Google Maps.

Botón para llamar a la tienda.

Botón para cancelar el pedido.

Botón para chatear con el cliente.

Botón para llamar al cliente.

Información del pedido.

Pantalla de historial de entregas

Historial de entregas por día.

MANUAL TÉCNICO

Introducción

Este manual describe los pasos para realizar la instalación de la aplicación móvil de la gestión comercial de la empresa Telaza y rastrear el estado de sus pedidos.

El objetivo de este manual es proporcionar información detallada sobre el desarrollo, implementación y mantenimiento de la aplicación.

Arquitectura de la Aplicación

La aplicación consta de un cliente móvil que se comunica con un servidor en la nube de Firebase que a su vez se conecta a una base de datos PostgreSQL para almacenar y recuperar datos.

Requisitos de la Aplicación

Requisitos del cliente móvil:
SO: Android 5.0 o superior.
Hardware: Cámara, conexión a Internet estable para el cliente móvil y el servidor en la nube.

Requisitos del servidor y de la base de datos en la nube:
Servidor: Node.js
Firebase Storage.
Google Services.

Configuración del entorno de desarrollo

Android Studio para desarrollo de aplicaciones móviles.
Node.js y Firebase SDK para el servidor.

IDE (Entorno de Desarrollo Integrado) como Visual Studio Code para el desarrollo del servidor.

Configuración de cuentas de desarrollador

Regístrate como desarrollador en Google Play Store (Android). Configura una cuenta de Firebase y Google Services.

Desarrollo del Cliente Móvil

Realizado con el lenguaje de programación Dart para un diseño intuitivo, navegación sencilla y capacidad de rastreo de pedidos. Y Android Studio para desarrollar la versión de Android de la aplicación.

Desarrollo del Servidor y Base de Datos

Desarrollado en Node.js para el lado del servidor y Firebase Storage para el almacenamiento de imágenes con una base de datos en PostgreSQL.

Integración de servicios en la Nube

Gestionado con Firebase Authentication para la autenticación de usuarios y Firebase Cloud Messaging para notificaciones en tiempo real.

Integración de Firebase Authentication para gestionar el registro y la autenticación de usuarios.