DESARROLLO E IMPLEMENTACIÓN DE UNA APLICACIÓN PARA DISPOSITIVOS MÓVILES QUE PERMITA CONECTAR DIRECTAMENTE A PRODUCTORES AGRÍCOLAS CON CONSUMIDORES FINALES MEJORANDO EN MARGEN DE BENEFICIOS PARA AMBOS

Ramiro Antonio Giraldo Escobar
Anderson Arenas Monsalve
Andrés Felipe Uribe Sánchez

Trabajo presentado como requisito al título de: Especialista en ingeniería de software

Área: Sistemas e Informática

Asesor:

Alicia Osorio Builes

Línea de Investigación: Ciencias Computacionales

Grupo de Investigación: Automática Electrónica y Ciencias Computacionales -GAEYCC-

Instituto Tecnológico Metropolitano -ITM-
Facultad de Ingenierías
Medellín, Colombia
Año 2023

Dedicatoria

A nuestras familias, amigos, compañeros y maestros que a lo largo de nuestras vidas han sido los motores que impulsan nuestro crecimiento.

Anderson A. M.

Andres U. S.

Antonio G. E

Agradecimientos

Al Instituto Tecnológico Metropolitano y cada uno de los funcionarios que conforma el talento humano de la institución desde los porteros, personal de oficios generales hasta la rectoría, pasando por los docentes, que aportaron a nuestro proceso de formación, cada uno desde sus saberes y experiencias.

A nuestras familias por el acompañamiento y la comprensión en los momentos que no pudimos compartir por estar dedicados al proceso de formación.

Resumen

MECAD es el acrónimo de Mercados Campesinos Digitales, cuyo propósito es el desarrollo e implementación de una aplicación móvil tipo e-commerce de fácil acceso que conecte directamente a los productores agrícolas con los consumidores finales, mejorando la eficiencia en la comercialización y aumentando el margen de beneficios para ambos. El proyecto MECAD aborda el planteamiento del problema existente en la comercialización de productos agrícolas y establece hipótesis para su solución. Se definen los objetivos generales y específicos del proyecto, así como la metodología propuesta y el plan de trabajo a seguir.

En el marco teórico y estado del arte, se realiza una revisión exhaustiva de los conceptos relevantes y el contexto actual en el desarrollo de aplicaciones móviles para la comercialización agrícola. Por otro lado, se describe el modelo de negocio para la aplicación móvil, destacando sus características y funcionalidades.

De la misma manera se identifican los procesos que se sistematizarán en la aplicación. Se describen los Requisitos técnicos, funcionales y no funcionales, así como la estructura de desglose de trabajo, la arquitectura y la infraestructura tecnológica necesaria. Se detalla la selección y adquisición de herramientas y tecnología, así como del entorno de desarrollo. Se define la arquitectura y se realiza el diseño de bases de datos, interfaces y aplicaciones. Finalmente se integran todas las partes del desarrollo y se realiza la construcción final.

Palabras Claves: App, Aplicaciones Móviles, e-commerce, e-agricultura

Abstract

MECAD is the acronym for Digital Farmer Markets, whose purpose is the development and implementation of an easy-access mobile application, similar to e-commerce, that directly connects agricultural producers with end consumers, improving efficiency in marketing and increasing profit margins for both parties. The MECAD project addresses the problem statement in agricultural product marketing and establishes hypotheses for its solution. The general and specific objectives of

the project are defined, as well as the proposed methodology and work plan to be followed.

In the theoretical framework and state of the art, a comprehensive review of relevant concepts and the current context in the development of mobile applications for agricultural marketing is conducted. On the other hand, the business model for the mobile application is described, highlighting its characteristics and functionalities.

Similarly, the processes to be systematized in the application are identified. Technical, functional, and non-functional requirements are described, as well as the work breakdown structure, architecture, and necessary technological infrastructure. The selection and acquisition of tools and technology, as well as the development environment, are detailed. The architecture is defined, and the design of databases, interfaces, and applications is carried out. Finally, all parts of the development are integrated, and the final construction is completed.

Keywords: App, Mobile Applications, e-commerce, e-agriculture

Tabla de contenido

Lista de figuras

Lista de tablas Pág.

Lista de abreviaciones

MECAD	Mercados Campesinos Digitales
SOLID	Conjunto de cinco principios de diseño de software que promueven la creación de código limpio, modular y fácil de mantener.
API	Application Programming Interface. Conjunto de reglas y protocolos que permiten la comunicación y la interacción entre diferentes aplicaciones de software
HTTP	Hypertext Transfer Protocol. Protocolo de comunicación utilizado en la World Wide Web para el intercambio de datos y la transferencia de información entre servidores y clientes
SDK	Software Development Kit". Kit de Desarrollo de Software en español.
TIC	Tecnologías de Información y la Comunicación
FAO	Organización de las Naciones Unidas para la Alimentación y la Agricultura, por sus siglas en inglés
CTi	Ciencia Tecnología e Innovación
I+D	Investigación y Desarrollo
App	Aplicación. Se refiere a un programa informático diseñado para funcionar en dispositivos electrónicos como teléfonos inteligentes y/o tabletas

1. Introducción

En el ámbito de la agricultura y la comercialización de productos agrícolas, la necesidad de una conexión directa entre los productores y los consumidores finales se ha vuelto cada vez más evidente. La tradicional cadena de intermediarios ha generado ineficiencias y costos adicionales que afectan tanto a los agricultores como a los consumidores. En respuesta a esta problemática, surge el proyecto MECAD, una innovadora aplicación móvil que busca revolucionar la forma en que se comercializan los productos agrícolas.

El objetivo principal de MECAD es mejorar la eficiencia en la comercialización de productos agrícolas al conectar directamente a los productores con los consumidores finales a través de una plataforma digital fácil de usar. Al eliminar intermediarios innecesarios, se busca reducir los costos asociados y aumentar el margen de beneficios tanto para los productores como para los consumidores.

MECAD ofrecerá a los agricultores la oportunidad de crear perfiles y listar sus productos en la aplicación, donde podrán proporcionar detalles sobre sus productos, métodos de producción, precios y cantidades disponibles. Los consumidores, a su vez, podrán acceder a la aplicación móvil y explorar una amplia variedad de productos agrícolas frescos y de calidad, provenientes directamente de los productores.

Con el objetivo de garantizar la calidad y la confiabilidad de los productos, MECAD implementará un sistema de evaluación y retroalimentación, permitiendo a los consumidores compartir sus experiencias y calificar la calidad y el servicio de los

productores. Esta retroalimentación contribuirá a crear un ambiente de confianza y a facilitar la toma de decisiones de compra por parte de los consumidores.

El proyecto MECAD se propone transformar la forma en que los productos agrícolas llegan a los consumidores finales, eliminando intermediarios innecesarios y creando una conexión directa entre los productores y los consumidores a través de una aplicación móvil intuitiva y confiable. Este proyecto busca mejorar la eficiencia en la comercialización agrícola, fomentar la transparencia y la confianza en las transacciones, y contribuir al desarrollo sostenible del sector agrícola en beneficio de todos los actores involucrados.

2. Capítulo I: Marco de Investigación y Planificación

2.1. Planteamiento del problema

En Colombia, a raíz del gran número de intermediarios que participan en la cadena de suministro de productos agrícolas, surge una gran problemática, pues en cada eslabón el producto aumenta considerablemente su valor, esto conlleva al detrimento de las condiciones laborales de los campesinos colombianos, pues, los agricultores, al ser los primeros en la cadena de suministro, se ven obligados a reducir considerablemente los precios de los productos al punto de no obtener ganancia alguna, sino por el contrario generar pérdidas y/o perder sus cosechas. EL país con sus tratados de libre comercio importa de países altamente industrializados, gran cantidad de productos que podrán ser producidos internamente y que a la larga poseen precios más bajos de cara al consumidor final.

Para la solución a dicha problemática, se plantea desarrollar una app tipo E-commerce de productos campesinos colombianos, orientada especifícateme a qué los productores puedan suscribirse como proveedores y así puedan vender de una forma más directa sus productos, bien sea a consumidores finales cerca de su zona de influencia o directamente a los distribuidores principales en las grandes cabeceras municipales. También está orientada a que los consumidores finales tengan un acceso rápido y confiable a productos agrícolas orgánicos frescos producidos en Colombia. Esta solución se plantea enmarcada en dos de los diecisiete objetivos de desarrollo sostenible que propone la Asamblea General de las Naciones Unidas. Por un lado y relacionado con el objetivo “2. Hambre cero”, se tiene como meta para 2030, duplicar la productividad agrícola y los ingresos de los productores de alimentos en pequeña escala, en particular las mujeres, los pueblos indígenas, los agricultores, los pastores y los pescadores, entre otras cosas mediante un acceso seguro y equitativo a las tierras, a otros recursos de producción e insumos, conocimientos, servicios financieros, mercados y oportunidades para la generación de valor añadido. Por otro lado, y relacionado con el objetivo “8. Trabajo decente y crecimiento económico”, se tiene como meta lograr niveles más elevados

de productividad económica mediante la diversificación, la modernización tecnológica y la innovación, entre otras cosas centrándose en los sectores con gran valor añadido y un uso intensivo de la mano de obra.

La presencia de un mercado digital en el sector agrícola ha dejado de ser una opción para muchas empresas y se ha convertido en una necesidad competitiva para mantenerse en el mercado. Este mercado digital promoverá la innovación, incluyendo la innovación en los negocios agrícolas, al presentar una plataforma intermediaria que conecta a inversores, propietarios de tierras, agricultores y clientes en una plataforma de mercado móvil única. Proporciona a los clientes un acceso directo a los principales procesadores utilizando sus teléfonos inteligentes, lo que les permitirá comparar precios, calidad, entrega y servicios de múltiples proveedores. Como resultado, los clientes tendrán un mayor poder de negociación en el mercado (Anshari et al., 2019)

2.2. Hipótesis

Con el desarrollo e implementación de una Aplicación para dispositivos móviles de fácil acceso se podría mejorar la eficiencia en la comercialización de los productos agrícolas, aumentando el margen de beneficios para productores y consumidores finales.

2.3. Objetivo General

Desarrollar una aplicación móvil de fácil acceso que conecte directamente a los productores agrícolas con los consumidores finales, mejorando la eficiencia en la comercialización y aumentando el margen de beneficios para ambos.

2.4. Objetivos Específicos:

1. Desarrollar un modelo de negocio para MECAD, que permita generar ingresos a través de diferentes fuentes, como comisiones por transacciones, servicios premium para los clientes y publicidad relevante, asegurando así la viabilidad financiera a largo plazo de la aplicación móvil.

2. Definir una arquitectura para la aplicación móvil, de acuerdo con las necesidades del usuario, que asegure una estructura sólida y modular, permita la disponibilidad y escalabilidad necesarias para su funcionamiento efectivo y sin interrupciones en momentos de alta demanda.

3. Desarrollar la aplicación móvil MECAD aplicando buenas prácticas de programación y utilizando herramientas tecnológicas que permitan una experiencia de usuario satisfactoria desde una interfaz atractiva, intuitiva y de fácil uso.

4. Diseñar un plan de aseguramiento de la calidad que garantice la calidad del producto de software a través de una serie de prácticas y procesos de aseguramiento de calidad que garanticen que la aplicación cumple con las expectativas y necesidades del usuario minimizando y/o mitigando los riesgos potenciales del proyecto.

2.5. Metodología propuesta

Para el desarrollo de este proyecto se ha utilizado el marco de trabajo SCRUM, siendo ésta una metodología ágil de gestión de proyectos que se utiliza comúnmente en proyectos de desarrollo de software, incluyendo el desarrollo de aplicaciones móviles como "MECAD". A continuación, se presentan algunos elementos y prácticas clave de SCRUM y cómo se aplican en este proyecto:

Product Backlog: En SCRUM, el Product Backlog es una lista priorizada de las funcionalidades y características a desarrollar en la aplicación móvil. En el caso de "MECAD", el Product Backlog incluye funcionalidades como registro de usuarios, búsqueda de productos, realización de pedidos, pagos en línea, entre otras.

Sprints: Los Sprints son ciclos de trabajo iterativos y repetitivos en SCRUM, que generalmente tienen una duración de dos a cuatro semanas. Durante cada Sprint, se selecciona un conjunto de funcionalidades del Product Backlog para desarrollar y se trabaja en ellas hasta que estén completas.

Reuniones diarias de SCRUM: denominadas en el marco SCRUM como “dailys”, son reuniones diarias breves, también se les conoce como reuniones de pie (stand-up meetings) en las que el equipo de desarrollo se actualiza sobre el progreso del Sprint y discute cualquier impedimento o problema que haya surgido. Esta reunión tiene como objetivo que el equipo de desarrollo se sincronice en cuanto al trabajo que se está realizando y el progreso hacia el objetivo del sprint. Durante la daily, cada miembro del equipo responde a tres preguntas básicas: ¿Qué hizo desde la última daily?

¿Qué piensa hacer hasta la siguiente daily?

¿Hay algún impedimento o problema que le impida avanzar en su trabajo?

La daily ayuda a mantener a todos los miembros del equipo de desarrollo informados sobre el progreso del proyecto, identificar y solucionar rápidamente cualquier impedimento que pueda surgir y fomentar la colaboración y el trabajo en equipo.

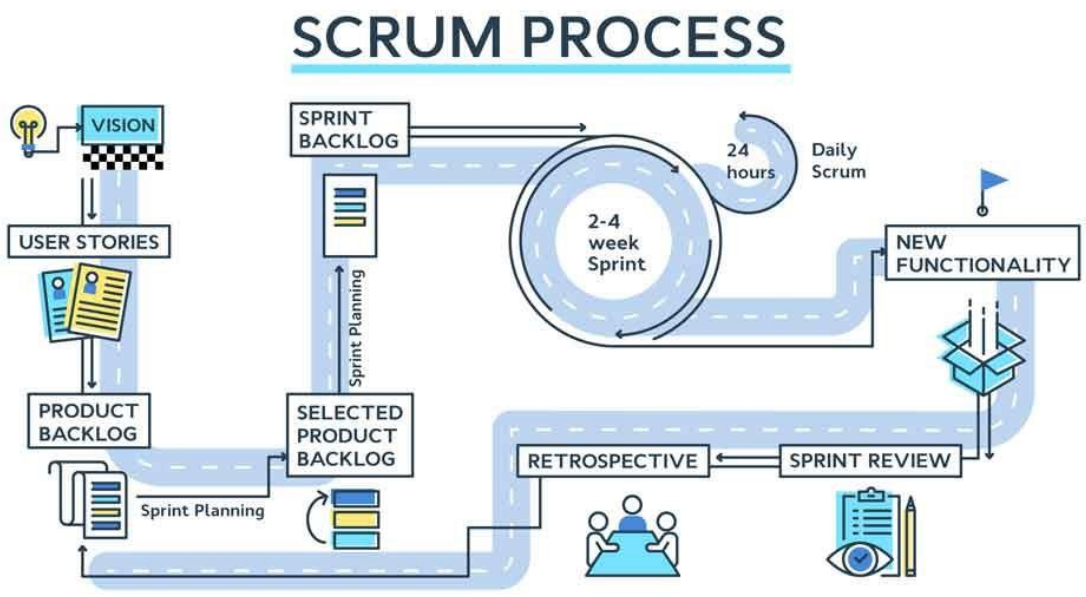
Revisión del Sprint: Al final de cada Sprint, se realiza una revisión del trabajo completado y se muestra a los interesados (stakeholders) lo que se ha desarrollado durante ese Sprint.

Retrospectiva del Sprint: Después de la revisión del Sprint, el equipo de desarrollo se reúne para discutir lo que funcionó bien y lo que se puede mejorar en el próximo Sprint.

En el proyecto "MECAD", la metodología o marco SCRUM se ha aplicado para gestionar el desarrollo de la aplicación móvil de manera iterativa e incremental, permitiendo que el equipo de desarrollo se adapte a los cambios y requisitos del proyecto de manera ágil. La metodología SCRUM también ha ayudado a asegurar una comunicación constante y efectiva entre el equipo de desarrollo y los interesados del proyecto, lo que puede mejorar la calidad y la eficiencia del desarrollo de la aplicación móvil.

Figura 1

Ciclo de vida de MECAD con el marco SCRUM



Nota: La gráfica representa las ceremonias de la metodología SCRUM. Fuente: https://www.camara.es/blog/innovacion-y-competitividad/que-es-scrum-y-como-puede-ayudar-tu-empresa.

2.6. Plan de Trabajo

En este proyecto se trabajó en dos frentes, uno de fundamentación o teórica y otro práctico. Así las cosas, el desarrollo metodológico se llevó cabo en las siguientes fases:

Fase de fundamentación o teórica.

En la primera fase del proyecto, se llevará a cabo una exhaustiva fundamentación teórica que servirá como base sólida para el desarrollo de la aplicación móvil MECAD. En esta etapa, se realizará una revisión exhaustiva de la literatura existente relacionada con el desarrollo de aplicaciones móviles y su aplicación en la comercialización agrícola. Se investigarán conceptos fundamentales, metodologías, tecnologías y buenas prácticas en el desarrollo de aplicaciones móviles, así como los aspectos específicos relacionados con la comercialización de productos agrícolas.

El objetivo principal de esta fase es adquirir un profundo conocimiento teórico sobre el tema y comprender las necesidades y requerimientos específicos de la aplicación MECAD. Se analizarán casos de estudio, investigaciones previas y proyectos similares para identificar las mejores prácticas y enfoques exitosos utilizados en el desarrollo de aplicaciones móviles para la comercialización agrícola. Esta fase sentará las bases para la toma de decisiones informadas en la etapa práctica del proyecto.

Fase Práctica.

En la segunda fase del proyecto, se lleva a cabo el desarrollo práctico de la aplicación móvil MECAD. Basándose en los conocimientos y la información recopilada en la fase de fundamentación teórica, se procede a diseñar, implementar y probar la aplicación móvil. Se utilizan herramientas y tecnologías adecuadas para el desarrollo de la interfaz de usuario, la gestión de datos, la conectividad, la seguridad y otras funcionalidades requeridas por MECAD.

Durante esta fase, se siguen las mejores prácticas de programación y se aplican metodologías ágiles, como Scrum, para garantizar un proceso de desarrollo eficiente y efectivo. Se realizan pruebas rigurosas para asegurar la calidad de la aplicación, tanto en términos de funcionalidad como de experiencia de usuario. Además, se lleva a cabo la integración con sistemas existentes, la configuración de servidores y la preparación de la infraestructura necesaria para el despliegue de la aplicación.

El objetivo final de esta fase es obtener una aplicación móvil completamente funcional, lista para ser lanzada y utilizada por los agricultores y consumidores finales. Se buscará optimizar la eficiencia en la comercialización agrícola, mejorar la conexión entre los actores involucrados y aumentar los beneficios para los productores agrícolas. El desarrollo práctico del proyecto se basará en los fundamentos teóricos adquiridos en la fase anterior, asegurando así la solidez y efectividad de la aplicación final.

La siguiente tabla presenta el plan de trabajo relacionando la metodología SCRUM con los objetivos específicos del proyecto.

Tabla 1

Plan de trabajo

Objetivo Específico	Fases	Actividades	¿Cómo se lograría la actividad?	Entregables
Desarrollar un modelo de negocio para MECAD, que permita generar ingresos a través de diferentes fuentes	Análisis y Diseño	- Investigar el mercado y los actores involucrados en la cadena de valor- Identificar las necesidades y demandas de los grupos de interés- Diseñar un modelo de negocio basado	- Realizar investigaciones de mercado y análisis de la competencia- Realizar encuestas y entrevistas a los campesinos y consumidores finales- Realizar estudios financieros para determinar la	- Modelo de negocio de MECAD- Estrategias de monetización- Planes de servicios premium- Acuerdos publicitarios

		en la eliminación de intermediarios y la generación de comisiones por transacciones- Establecer servicios premium para los clientes- Implementar estrategias de publicidad relevante	viabilidad del modelo de negocio- Diseñar planes de servicios premium y establecer acuerdos publicitarios- Crear un plan de monetización y generación de ingresos	
Definir una arquitectura para la aplicación móvil, de acuerdo a las necesidades del usuario	Diseño y Desarrollo	- Realizar un relevamiento de requisitos y necesidades de los usuarios- Diseñar una arquitectura sólida y escalable- Seleccionar las tecnologías y herramientas adecuadas- Establecer estándares de desarrollo y buenas prácticas- Realizar pruebas de rendimiento y optimización	- Realizar entrevistas y encuestas a los usuarios- Realizar análisis de requisitos y definir los casos de uso- Diseñar la arquitectura teniendo en cuenta la escalabilidad y disponibilidad- Investigar y seleccionar las tecnologías adecuadas- Establecer estándares de desarrollo y documentación- Realizar pruebas de rendimiento y optimizar el código	- Documento de requisitos- Diseño de la arquitectura- Tecnologías y herramientas seleccionadas- Estándares de desarrollo y buenas prácticas establecidas- Informes de pruebas de rendimiento
Desarrollar la aplicación móvil MECAD aplicando buenas prácticas de programación y utilizando herramientas tecnológicas que permitan una experiencia de	Desarrollo	- Implementar la lógica de negocio de la aplicación- Diseñar y desarrollar la interfaz de usuario- Aplicar buenas prácticas de programación- Utilizar frameworks y herramientas	- Desarrollar los módulos y funcionalidades de la aplicación móvil- Diseñar y prototipar la interfaz de usuario con un enfoque en la experiencia del usuario- Aplicar	- Aplicación móvil MECAD funcional- Interfaz de usuario diseñada y prototipada- Código de programación limpio y

usuario satisfactoria desde una interfaz atractiva, intuitiva y de fácil uso		modernas- Realizar pruebas de funcionalidad y usabilidad	principios de diseño y usabilidad- Utilizar frameworks y herramientas como Flutter y Firebase- Realizar pruebas unitarias y pruebas de usuario	documentado- Informes de pruebas de funcionalidad y usabilidad
Diseñar un plan de aseguramiento de la calidad que garantice la calidad del producto de software	Pruebas y Mejora Continua	- Definir los criterios de calidad y métricas a utilizar- Establecer procesos y estándares de pruebas- Realizar pruebas de funcionalidad, rendimiento, seguridad y usabilidad- Identificar y corregir errores y problemas- Realizar revisiones y auditorías del código	- Establecer un plan de aseguramiento de calidad detallado- Definir los criterios de aceptación y métricas de calidad- Realizar pruebas en diferentes etapas del desarrollo- Utilizar herramientas de pruebas y métricas de calidad- Realizar revisiones y auditorías del código- Documentar y corregir los errores identificados	- Plan de aseguramiento de calidad- Informes de pruebas realizadas- Errores y problemas identificados y corregidos- Revisiones y auditorías del código documentadas

Nota: Esta tabla contiene el plan de trabajo relacionando la metodología con los objetivos específicos del proyecto. Elaboración propia.

3. Capítulo II: Marco Teórico y Estado del Arte.

3.1. Marco teórico

El marco teórico es un componente fundamental en cualquier proyecto bien sea de investigación o desarrollo, ya que provee una base conceptual y contextual para la comprensión de los resultados. En el caso de la implementación de aplicaciones móviles como MECAD, contar con un marco teórico sólido es esencial para identificar las mejores prácticas y tecnologías apropiadas para su desarrollo y operación.

Por lo tanto, este marco teórico abarca conceptos clave relacionados con el desarrollo de aplicaciones móviles, las plataformas móviles más comunes y sus características, los modelos de negocio relacionados con la comercialización de productos agrícolas, y los desafíos tecnológicos y de usabilidad que enfrentan las aplicaciones móviles para brindar una experiencia de usuario satisfactoria.

Aplicaciones móviles

Las aplicaciones móviles, también conocidas como apps móviles, son programas de software diseñados específicamente para funcionar en dispositivos portables o móviles, como teléfonos inteligentes o tabletas. Estas aplicaciones se descargan e instalan en el dispositivo a través de tiendas de aplicaciones, como Google Play Store para dispositivos Android o App Store para dispositivos iOS.

Las aplicaciones móviles están diseñadas para realizar diversas tareas y ofrecer una amplia gama de funcionalidades, desde juegos y entretenimiento hasta productividad, comunicación, educación, salud, finanzas y mucho más. Pueden acceder a diversas características del dispositivo, como la cámara, el GPS, los sensores y la conectividad a Internet, lo que les permite proporcionar experiencias interactivas y personalizadas.

Las aplicaciones móviles pueden ser desarrolladas para plataformas específicas (como Android o iOS) o como aplicaciones multiplataforma, que pueden funcionar en diferentes sistemas operativos. Estas aplicaciones se han vuelto cada vez más populares debido a la creciente adopción de dispositivos móviles y ofrecen a los usuarios acceso rápido y conveniente a servicios y contenido mientras están en movimiento.

Tabla 2

Aplicaciones Móviles

TÍTULO	CONTENIDO	AÑO
Software assurance practices for mobile applications	"Los sistemas móviles llevan a cabo sus operaciones en una amplia gama de entornos, desde servicios tradicionales como comunicación de voz, mensajería de texto y juegos, hasta aplicaciones comerciales, servicios basados en la ubicación, realidad aumentada, herramientas de productividad y mucho más." (Corral *et al.*, 2015)	2015
Choosing between responsive-design websites versus mobile apps for your mobile behavioral intervention: presenting four case studies	"Las aplicaciones móviles son programas que pueden realizar determinadas tareas o proporcionar determinadas funciones para un usuario" (Turner-McGrievy *et al.*, 2017)	2017
Analyzing and automatically labelling the types of user issues that are raised in mobile app reviews	"El mercado de aplicaciones móviles sigue creciendo a un ritmo muy rápido con miles de desarrolladores, miles de aplicaciones y millones de dólares en ingresos." (McIlroy *et al.*, 2016)	2016

Nota: La tabla presenta la concepción de Aplicaciones Móviles de algunos autores. Fuente: (Molina Ríos et al., 2021)

Metodologías de desarrollo de software:

Las metodologías de desarrollo de software son importantes en cualquier proyecto de desarrollo de software, incluyendo proyectos de aplicaciones móviles como "MECAD", porque permiten planificar, organizar y controlar el proceso de desarrollo de software de manera más efectiva y eficiente.

Las metodologías establecen un marco de trabajo para la gestión del proyecto, definen las actividades y tareas necesarias para el desarrollo de la aplicación, así como los roles y responsabilidades de los miembros del equipo de desarrollo. También establecen un proceso para la toma de decisiones y la gestión de cambios y riesgos.

Además, las metodologías de desarrollo de software fomentan la colaboración entre los miembros del equipo de desarrollo y ayudan a mantener a todos los involucrados en el proyecto enfocados en el objetivo final. Las metodologías ágiles, por ejemplo, hacen énfasis en la comunicación y colaboración constante entre los miembros del equipo y los clientes, lo que ayuda a asegurar que la aplicación final cumpla con los requisitos del cliente y las expectativas del usuario.

Así las cosas, las metodologías de desarrollo de software son fundamentales en cualquier proyecto de desarrollo de software para asegurar el éxito del proyecto, mejorar la calidad del software y asegurar que se cumplan los requisitos y expectativas de los clientes y usuarios. A continuación de describen brevemente las más utilizadas.

Tabla 3

Metodologías enfocadas al desarrollo de aplicaciones móviles.

TÍTULO	CONTENIDO	AÑO
Job-work fit as a determinant of the acceptance of large-scale agile methodology.	"Una metodología de desarrollo de sistemas o software (SDM) se puede definir como una colección documentada de políticas, procesos y procedimientos para mejorar el proceso de desarrollo de software con respecto al aumento de la productividad del personal de tecnología de la información (TI) y una mayor calidad de las soluciones de TI finales." (Batra, 2020)	2020
A Comparative Analysis on Effort Estimation for Agile and Non-agile Software Projects Using DBN-ALO.	"Estas metodologías ágiles tienen ventajas como la flexibilidad, responden a los cambios incluso en etapas posteriores de desarrollo, son amigables para el cliente y entregan el software en funcionamiento rápidamente." (Kaushik *et al.*, 2020)	2020
Using agile methodologies for adopting COBIT	"Al elegir una metodología ágil frente a una tradicional, los autores buscan facilitar la implementación, apoyándose en prácticas ágiles como habilitadores para lograr los objetivos propuestos. El principio ágil de ofrecer soluciones a menudo aumentará el apoyo de la alta dirección y disminuirá la resistencia al cambio." (Amorim *et al.*, 2020)	2020

Nota: Conceptos sobre metodologías enfocadas al desarrollo de aplicaciones móviles. Fuente: (Molina Ríos et al., 2021)

RATIONAL UNIFIED PROCESS (RUP): tiene como objetivo ordenar y estructurar el desarrollo de software a través de un conjunto de actividades para transformar los requisitos del usuario en un sistema de software. RUP es un proceso basado en los modelos en cascada y por componentes, centrado en los casos de uso y la arquitectura, y es iterativo e incremental. Se explica que los casos de uso describen los servicios que el usuario requiere del sistema y que la arquitectura del software es importante para comprender el sistema en su totalidad y fomentar la reutilización de componentes. RUP fue lanzada en 1998 y sus creadores son Ivar Jacobson, Grady Booch y James Rumbaugh (Oiver Pérez, 2011).

MICROSOFT SOLUTIONS FRAMEWORK (MSF) es una guía de desarrollo de software flexible y escalable, que se basa en un conjunto de principios, modelos, disciplinas, conceptos, directrices y prácticas aprobadas por Microsoft. Esta metodología asegura resultados con menor riesgo y de mayor calidad, centrándose en el proceso y las personas. MSF fue introducido por primera vez en 1994 como un conjunto de las mejores prácticas en los desarrollos de software de Microsoft y

Microsoft Consulting Service y ha evolucionado y mejorado con la experiencia de grupos de trabajo reales. MSF también retoma algunas características de las metodologías tradicionales.(Oiver Pérez, 2011)

Programación Extrema (XP) la programación extrema es una disciplina de desarrollo de software basada en los métodos ágiles, que se enfoca en el desarrollo incremental, la participación del cliente y la simplicidad. Esta metodología tiene diez principios que involucran al equipo de trabajo, los procesos y el cliente, y es una buena práctica a considerar en el desarrollo de software. Los artefactos generados en XP incluyen tarjetas de historias de usuario (Story Card), tarjetas de tareas para la descarga de documentos, código, pruebas unitarias y de integración, y pruebas de aceptación. Estos artefactos son fundamentales para comprender el proceso de desarrollo del software, entender cómo está construido el sistema y planificar la ruta a seguir para agregar funcionalidad al mismo (Oiver Pérez, 2011)

SCRUM: Scrum es un marco de trabajo ágil para el desarrollo de software que se centra en el control continuo del estado actual del proyecto. El cliente establece las prioridades y el equipo Scrum se autoorganiza para determinar la mejor forma de entregar resultados. Scrum se caracteriza por dar prioridad a los individuos y las interacciones sobre los procesos y las tareas, lo cual significa que gran parte del éxito del proyecto radica en la forma en que el equipo se organice para trabajar.

A diferencia de otros marcos de trabajo, Scrum propone el software funcional sobre la excesiva documentación. Se presenta al cliente soluciones operables en lugar de reportes de progreso, lo que permite al cliente tomar decisiones rápidas y hacer ajustes en la marcha si es necesario. Además, Scrum promueve la colaboración con el cliente en lugar de una negociación rígida de contratos. Es importante tener capacidad de respuesta para los cambios en lugar de seguir estrictamente una planificación, ya que el proyecto de software es cambiante. El propósito es que el cliente vaya observando los resultados, pueda decidir cambios en la marcha o incluso darle un giro completo al proyecto. En resumen, Scrum es una metodología que permite una mayor flexibilidad en el proceso de desarrollo de software y una mejor adaptación a los cambios.(Oiver Pérez, 2011)

Comercio electrónico:

Comercio electrónico más conocido globalmente como e-commerce, se refiere a la compra y venta de bienes y servicios a través de Internet. Es un modelo de negocio que permite a las empresas y consumidores realizar transacciones comerciales de forma electrónica, sin necesidad de interacciones físicas directas.

En el comercio electrónico, las transacciones se llevan a cabo utilizando plataformas en línea, sitios web o aplicaciones móviles. Los compradores pueden explorar catálogos de productos, realizar pedidos, realizar pagos y recibir los productos o servicios directamente en su ubicación. Los vendedores, por otro lado, pueden mostrar sus productos o servicios en línea, gestionar inventarios, procesar pagos y administrar la entrega.

El comercio electrónico se refiere a la compra y venta de bienes y servicios a través de Internet. Puede involucrar transacciones directas entre una empresa y un consumidor (B2C), entre empresas (B2B) o entre consumidores (C2C). El comercio electrónico abarca las operaciones de una empresa en línea, desde su propia plataforma de ventas hasta la entrega de productos y servicios.

Mercados digitales:

Este es un concepto que ha evolucionado con las tecnologías de internet, también se conocen como marketplaces digitales. Son plataformas en línea donde los vendedores pueden ofrecer sus productos o servicios y los compradores pueden encontrar y adquirir lo que necesitan. Estos mercados se basan en la tecnología digital y permiten la interacción y transacciones comerciales entre múltiples participantes en un entorno virtual.

Estos mercados ofrecen ventajas tanto para vendedores como para compradores. Para los vendedores, los mercados digitales proporcionan una plataforma de visibilidad y alcance amplio, facilitando la llegada a una audiencia global, mientras que los compradores se benefician de la conveniencia de encontrar una amplia variedad de productos o servicios en un solo lugar, comparar precios y

características, acceder a reseñas y opiniones de otros usuarios, y realizar transacciones de forma segura.

Agricultura digital:

La agricultura desempeña un papel crucial en el desarrollo económico y social global. La demanda actual no solo se centra en aumentar la productividad de los cultivos, sino también en producir de manera más eficiente y sostenible, fomentar una alimentación saludable y proteger la biodiversidad. Para lograr estos objetivos a nivel global, es necesario democratizar el conocimiento y las buenas prácticas. La agricultura digital, a través de la aplicación de las nuevas tecnologías, es la herramienta más prometedora para un cambio cualitativo en los próximos años en comparación con otras alternativas (Bórnez, 2022)

La agricultura está siendo transformada por la tecnología digital, lo que ha dado lugar a nuevos modelos de negocio en un mercado en crecimiento. La adopción de la agricultura digital permitirá una gestión más eficiente de los recursos, tomar decisiones más precisas y automatizar tareas mediante el uso de sistemas de inteligencia artificial o robots y aplicaciones móviles que se adapten a los nuevos desafíos de la producción y el consumo. (Bórnez, 2022)

Arquitectura de software:

Es la representación de alto nivel de cómo los componentes del software interactúan entre sí, cómo se organizan y cómo cumplen con los requisitos funcionales y no funcionales del sistema.

La arquitectura de software abarca la estructura global del sistema, incluyendo los componentes del software, las relaciones entre ellos, los patrones de diseño utilizados, las decisiones de diseño clave y las restricciones arquitectónicas. También puede incluir aspectos como la distribución de componentes en diferentes entornos, la comunicación entre los componentes y los flujos de datos dentro del sistema.

Para Jimenez-Torres et al.(2014) la Arquitectura de Software se considera un enfoque novedoso para comprender un sistema de información de manera integral, teniendo en cuenta cómo cada componente afecta los requisitos esenciales, tanto funcionales como no funcionales. Esta disciplina aborda estos requisitos de manera rigurosa, asegurando un diseño óptimo de la aplicación final. Esto se traduce en una mayor calidad, un retorno de inversión mejorado en los proyectos y una mayor facilidad para el mantenimiento de los sistemas de información construidos.

La arquitectura de software se encarga de aspectos tales como: modularidad, escalabilidad, mantenibilidad, fiabilidad, entre otros, conocidos también como requisitos no funcionales o condiciones de calidad.

Experiencia del usuario:

La experiencia del usuario (UX) es un concepto referido a la forma en que los usuarios interactúan con un producto o servicio y cómo se sienten al hacerlo. Se centra en diseñar y mejorar la satisfacción, la usabilidad y la accesibilidad de la experiencia de los usuarios.

Para el caso de las aplicaciones de software el diseño de interfaces de usuario intuitivas es un aspecto fundamental de la UX. Se trata de crear ambientes que sean fáciles de entender y de usar para los usuarios, de modo que puedan interactuar de manera fluida con el producto o servicio. Esto implica utilizar un diseño limpio y claro, con una disposición lógica de elementos y una navegación intuitiva.

El concepto UX también implica garantizar que todas las personas, independientemente de sus habilidades o discapacidades, puedan utilizar el producto o servicio de manera efectiva. Esto puede incluir el uso de técnicas como la inclusión de texto alternativo en imágenes, el uso de colores y contrastes adecuados, y la consideración de las pautas de accesibilidad web.

Por otro lado, es importante considerar aspectos emocionales y psicológicos en el diseño de la experiencia del usuario. Esto implica comprender las emociones y las motivaciones de los usuarios y diseñar experiencias que les brinden satisfacción, deleite y una sensación de cumplimiento.

Para Ramírez-Olivera Gabriela et al., s/f(2021) Actualmente, no hay una definición única para la experiencia del usuario (UX). No obstante, su objetivo principal radica en maximizar los momentos positivos que los usuarios experimentan al utilizar un sistema, e idealmente, generar un sentimiento de amor hacia el producto, al menos en parte o la mayor parte del tiempo.

El concepto de Apps nativas, híbridas y web:

Las aplicaciones nativas son aquellas que han sido desarrolladas con el software que ofrece cada sistema operativo a los programadores, llamado genéricamente Software Development Kit o SDK, por lo que se adapta al 100% al dispositivo.

El desarrollo de aplicaciones nativas implica que los desarrolladores, después de escribir el código fuente, creen un archivo ejecutable en formato binario que se puede empaquetar junto con otros recursos como audio, imágenes, videos y diferentes tipos de archivos necesarios para el sistema operativo (OS). Todo esto requiere el uso de herramientas, archivos y suministros específicos proporcionados por el proveedor del OS. (Bautista Sánchez, 2020)

Una aplicación híbrida se crea con una combinación de HTML5, CSS y JavaScript. Es independiente de los sistemas operativos iOS y Android. Una vez escrita se compila o empaqueta, de tal forma que el resultado final es parecido a una aplicación nativa. Esto supone la gran ventaja de que con el mismo código obtenemos aplicaciones para, por ejemplo, Android y iOS.

Las aplicaciones híbridas ofrecen la ventaja de escribir código una vez y ejecutarlo en diferentes plataformas, lo que ahorra tiempo y recursos en comparación con el desarrollo de aplicaciones nativas separadas. Sin embargo, es importante tener en cuenta que las aplicaciones híbridas pueden presentar limitaciones en términos de rendimiento y acceso completo a las capacidades del dispositivo en comparación con las aplicaciones nativas.(Bautista Sánchez, 2020)

Las aplicaciones web, o desarrollo multiplataforma basado en HTML, también llamadas webapps, están basadas en HTML, CSS y JavaScript. No se emplea un

SDK para el desarrollo, lo cual permite programar con independencia de la plataforma.

Al tratarse de aplicaciones que funcionan sobre la web, no es necesario que el usuario reciba actualizaciones, ya que siempre va a estar viendo la última versión. Sin embargo, requieren de conexión a Internet.

Marcos de desarrollo

Un marco de desarrollo más conocidos como framework de desarrollo es una herramienta diseñada para simplificar el proceso de creación de aplicaciones web o móviles. Su objetivo es proporcionar una estructura base que los desarrolladores deben seguir. Esta estructura incluye archivos de configuración estándar y componentes predefinidos que han sido creados previamente por los desarrolladores del framework para ser reutilizados. En general se puede decir que un framework es una guía básica que define los elementos necesarios para un proyecto que además ofrece una variedad de componentes opcionales que los desarrolladores pueden utilizar según sus necesidades. (Simmonds, 2019)

A continuación, se describen algunos de los frameworks más destacados para el desarrollo de aplicaciones móviles.

Flutter:

Flutter es un framework revolucionario desarrollado por Google que ha introducido cambios significativos en el desarrollo de aplicaciones móviles. A diferencia de los frameworks anteriores, Flutter adopta un enfoque único en la forma en que se escribe una aplicación.

Flutter se basa en el concepto central de "Widgets", que son componentes gráficos predefinidos incluidos en el framework. Estos widgets se seleccionan en función de los elementos comunes que se encuentran típicamente en una aplicación móvil. Por ejemplo, si se necesita un botón flotante, Flutter proporciona un widget específico para ello. No es necesario escribir el código y posicionarlo manualmente como se haría normalmente. (Simmonds, 2019)

La arquitectura de Flutter también presenta innovaciones significativas. Introdujeron una arquitectura llamada BloC (Business Logic Component) que consta de dos capas: una capa dedicada a la interfaz de usuario de la aplicación y otra capa encargada de la lógica. Estas capas se comunican mediante streams, es decir, flujos de datos. Por ejemplo, si hay un botón que realiza una acción determinada, en lugar de ejecutar directamente dicha acción y modificar la interfaz, el botón envía un flujo de datos a la capa lógica, que devuelve una acción que la interfaz puede ejecutar si es necesario. (Simmonds, 2019)

React Native:

React-Native es un framework desarrollado por Facebook basado en React, el popular framework de desarrollo web. Los desarrolladores de React Native aprovecharon las ventajas y la popularidad de React para crear una experiencia similar en el desarrollo de aplicaciones móviles. Aunque el proceso de desarrollo en React-Native es similar al de React, hay dos cambios clave a tener en cuenta.

En primer lugar, en lugar de utilizar el lenguaje JSX específico de React, se utiliza una adaptación que se ajusta a las necesidades típicas de los dispositivos móviles. Además, se deben considerar las particularidades propias de los dispositivos, como la gestión de permisos que pueden ser solicitados al usuario.

En términos de arquitectura, a diferencia de Ionic, React-Native no impone una estructura predeterminada. Sin embargo, se recomienda seguir el patrón de diseño Modelo-Vista-Controlador (MVC) para organizar el código de manera efectiva. Esto implica separar la lógica de la aplicación, el diseño y cualquier aspecto relacionado con las conexiones externas. De esta manera, se puede lograr un desarrollo más estructurado y modular en React-Native. (Simmonds, 2019)

Ionic:

Ionic es un framework que utiliza Cordova para la parte móvil y Angular para la parte web. Aunque la mayor parte del código se escribe en TypeScript y HTML, que son

las bases de Angular, es Cordova quien se encarga de crear el entorno de ejecución para las aplicaciones en los dispositivos. En una aplicación típica de Ionic, cada componente tiene 4 archivos, o 5 si se crea una página independiente de la principal. Estos archivos incluyen el HTML para definir la estructura de la aplicación, un archivo SCSS para agregar estilos, un archivo spec que se genera automáticamente y generalmente no se modifica, y un archivo component donde se trabaja toda la lógica. Si en lugar de un componente se crea una página, se agrega un archivo adicional llamado módulo, donde se declaran todos los paquetes de Ionic que se utilizarán en esa página.(Simmonds, 2019)

Native Script:

NativeScript es un framework de desarrollo de aplicaciones móviles de código abierto que permite crear aplicaciones nativas para iOS y Android utilizando tecnologías web como JavaScript, TypeScript y Angular. Proporciona una capa de abstracción sobre las API nativas de cada plataforma, lo que permite acceder directamente a las características y funcionalidades específicas de cada sistema operativo. Con NativeScript, los desarrolladores pueden escribir código una vez y ejecutarlo en ambas plataformas, lo que ahorra tiempo y esfuerzo en el desarrollo de aplicaciones móviles nativas. Además, ofrece una amplia gama de plugins y extensiones para acceder a funcionalidades adicionales del dispositivo.

NativeScript es un framework que permite desarrollar aplicaciones móviles nativas para iOS y Android utilizando tecnologías web. Con su capa de abstracción sobre las API nativas, los desarrolladores pueden aprovechar las características específicas de cada plataforma. NativeScript ofrece un enfoque de "escribir una vez, ejecutar en todas partes" y cuenta con una amplia gama de plugins para acceder a funcionalidades del dispositivo. (Anaya & Salmon, s/f)

Xamarin:

Xamarin es una plataforma de desarrollo de aplicaciones móviles que permite a los desarrolladores crear aplicaciones nativas para Android e iOS utilizando el lenguaje de programación C#. Con Xamarin, es posible compartir gran parte del código entre

las plataformas, lo que agiliza el proceso de desarrollo y permite alcanzar una mayor eficiencia. Además, Xamarin proporciona acceso completo a las API y funcionalidades nativas de cada plataforma, lo que permite aprovechar al máximo las características específicas de Android e iOS. Con una amplia biblioteca de componentes reutilizables y una integración estrecha con las herramientas de desarrollo de Microsoft, Xamarin se ha convertido en una opción popular para desarrolladores que buscan crear aplicaciones móviles multiplataforma de alta calidad.

3.2. El estado del arte

A continuación, se presenta un análisis de la información rastreada sobre la temática elegida “Desarrollo de aplicaciones móviles tipo e-commerce para productos agrícolas”. Este rastreo se hizo mediante búsqueda sistematizada de la literatura científica disponible en Internet en bases de datos indexadas como Scopus, IEEE, Redalyc, Ebsco, entre otras.

La búsqueda se realizó en artículos científicos, trabajos de grado, así como en la normativa y legislación colombiana, además en reportes de entidades gubernamentales internacionales. Para la búsqueda se tuvieron en cuenta algunos criterios de inclusión y exclusión que se muestran en la tabla 2.

Los criterios de búsqueda incluyeron palabras como: App Movil, e-commerce, además se construye un string de búsqueda con operadores boléanos y con dichas palabras en inglés y español para aumentar la eficiencia en la búsqueda de archivos relacionados (e-commerce OR e-agricultura OR e-agriculture) AND (mercados digitales OR digital marketplaces).

El crecimiento económico de los países en desarrollo y el bienestar de las poblaciones rurales están estrechamente ligados al potencial de la agricultura. Este sector ofrece beneficios en términos de nutrición, empleo, ingresos y calidad de vida. A nivel mundial, el mercado de productos agropecuarios ha experimentado un crecimiento significativo, con tasas anuales superiores a los bienes no agrícolas. Las exportaciones agrícolas representan una parte importante de este comercio,

principalmente de productos elaborados. En Colombia, se destaca su potencial agrícola, con millones de hectáreas disponibles para su desarrollo y un clima favorable. Sin embargo, persisten desafíos como la baja competitividad, limitaciones en infraestructura y comercialización, dificultades para expandir y diversificar los mercados, así como desequilibrios regionales y limitaciones en la capacidad de respuesta y estabilidad de la inversión en áreas rurales.(Ministerio de Agricultura y desarrollo Rural et al., s/f)

La agricultura tiene un papel fundamental en el desarrollo económico y social de los países en desarrollo, ofreciendo oportunidades para mejorar las condiciones de vida de las poblaciones rurales. Aunque existen amplias posibilidades de expansión y crecimiento agrícola en Colombia, se requiere abordar obstáculos como la competitividad, la infraestructura y la comercialización, la diversificación de mercados y la capacidad de respuesta a los desafíos externos. Superar estos cuellos de botella permitirá aprovechar plenamente el potencial agrícola del país. (Ministerio de Agricultura y desarrollo Rural et al., s/f)

Los informes más recientes de la FAO (2013) sobre la seguridad alimentaria mundial señalan que el crecimiento económico por sí solo no es suficiente para combatir el hambre y la malnutrición de manera efectiva. En cambio, el enfoque en el crecimiento agrícola sostenible se ha demostrado como una estrategia eficaz para llegar a los sectores más pobres de la población, especialmente en países de bajos ingresos. En estos contextos, aumentar la productividad de los pequeños agricultores es fundamental, ya que la agricultura desempeña un papel crucial en la reducción de la pobreza, en la satisfacción de la demanda futura de alimentos y en la generación de oportunidades para adquirir bienes y servicios producidos localmente.(Ministerio de Agricultura y desarrollo Rural et al., s/f)

Ahora bien, en el contexto global se ha ido posicionando el término e-agricultura, también conocida como agricultura electrónica o agricultura digital, se refiere al uso de tecnologías de la información y comunicación (TIC) en el sector agrícola. Consiste en la aplicación de soluciones digitales, como Internet, dispositivos

móviles, sensores, sistemas de información geográfica, entre otros, para mejorar la eficiencia, la productividad, la comercialización y la sostenibilidad en la agricultura.

La e-agricultura abarca una amplia gama de aplicaciones, como la gestión de cultivos, la monitorización del clima y del suelo, la trazabilidad de productos, la comercialización en línea, la capacitación agrícola en línea, entre otros. Estas herramientas y servicios digitales permiten a los agricultores acceder a información en tiempo real, tomar decisiones informadas, optimizar el uso de recursos, reducir costos y mejorar la calidad de los productos agrícolas.

La e-agricultura tiene el potencial de transformar la forma en que se practica la agricultura, especialmente en áreas rurales y en países en desarrollo. Al promover la inclusión digital en el sector agrícola, se pueden superar barreras como la falta de acceso a información, la limitada conectividad y la escasez de recursos técnicos. Esto puede impulsar la innovación, aumentar la productividad agrícola y contribuir al desarrollo sostenible de las comunidades rurales.

Tras analizar los planes y documentos gubernamentales de CTi, así como informes técnicos y discusiones de validación a nivel nacional y territorial, se han identificado cinco mega tendencias que tendrán un impacto significativo en el cambio técnico del sector agropecuario. Estas mega tendencias fueron examinadas en detalle junto con expertos para definir estrategias y acciones que deben ser consideradas en el Plan Estratégico de Ciencia, Tecnología e Innovación del Sector Agropecuario Colombiano (Pectia). Las cinco mega tendencias son las siguientes: biodiversidad y biotecnología, seguridad alimentaria, sostenibilidad ambiental, variabilidad y cambio climático, tecnologías de la información y comunicación, y agro energías (Ministerio de Agricultura y desarrollo Rural et al., s/f)

Según el Ministerio de Agricultura y desarrollo Rural et al.(2017) Las áreas de investigación y desarrollo (I+D) relacionadas con el sector agropecuario se han identificado con base a los siguientes criterios: energías renovables, salud y alimentos. Además, se han destacado las tecnologías transversales que son relevantes en este contexto, como la biotecnología, materiales y nanotecnología, y

tecnologías de la información y comunicación (TIC). En cuanto a las TIC, se han establecido cinco enfoques específicos: (1) liderar el desarrollo de aplicaciones sociales dirigidas a las poblaciones más pobres, (2) desarrollar contenidos digitales, (3) impulsar el comercio electrónico, (4) fomentar el emprendimiento en el ámbito de las TIC y (5) promover el uso de aplicaciones gubernamentales (e-government).

En Colombia, el Ministerio de Tecnologías de la Información y las Comunicaciones (MTIC) es la máxima autoridad encargada de regular el sector de las tecnologías de la información y las comunicaciones. A través de este ministerio se establecen las políticas y se respalda la normativa relacionada. En el contexto de la interacción entre el sector de las TIC y el sector agropecuario, se ha establecido la iniciativa TIC + Agro como marco de referencia a nivel ministerial, en colaboración con el Ministerio de Agricultura y Desarrollo Rural (MADR). Esta iniciativa tiene como objetivo establecer un marco normativo para el desarrollo de actividades en el ámbito de la ciencia, tecnología e innovación (CTi), con un enfoque en nueve áreas temáticas específicas.(Ministerio de Agricultura y desarrollo Rural et al., s/f). En el contexto de aplicación tipo e-commerce destacamos el eje número 4. Desarrollo e implementación de soluciones de TI enfocadas en la gestión de información de mercados agrícolas, agroindustriales, pecuarios, avícolas, forestales, acuícolas y pesqueros, tales como manejo de insumos, procesos de logística, trazabilidad de productos, e-marketing, e-commerce y e-business.

Ahora bien, en cuanto a aplicaciones móviles tipo e-commerce cuyo objeto principal sea conectar directamente a los productores agrícolas que quieren vender sus cosechas con los consumidores finales, disminuyendo la intermediación, no se encuentran tantos desarrollos, no obstante, se encuentran aplicaciones móviles y páginas web como: AGROMARKET S.A.S que comercializa insumos agropecuarios, medicamentos veterinarios, presta servicios de consulta médica veterinaria y SPA para mascotas. AGRAPP, que facilita el acceso financiero a pequeños y medianos agricultores con la estructuración, asistencia y comercialización fomentando practicas sostenibles y tecnificación, Tuagro, es una aplicación que brinda información sobre precios de productos e insumos agrícolas

en diferentes mercados de Colombia. Los agricultores pueden conocer los precios actuales y tendencias del mercado, AGROSMART, brinda soluciones de innovación basadas en tecnología para el campo, entre ellas desarrollo de software.

En el contexto colombiano se han desplegado otras aplicaciones que buscan objetivos similares como: Frubana, Waruwa, Plaz, Koshcampo y Fruvii.

Existen otras estrategias impulsadas por los gobiernos locales, generalmente en las grandes ciudades, como mercados campesinos, esta estrategia presenta 3 modalidades como: Mercados Permanentes, estos se llevan a cabo en plazas de mercado y otros espacios designados, donde se establece un lugar físico para la realización continua del mercado y la atención al público, sin necesidad de mover el mobiliario. Mercados Itinerantes se llevan a cabo en varias ubicaciones de la ciudad, como parques y plazoletas, y no tienen estructuras fijas. Por lo tanto, cada vez que se programa el mercado, se requiere un despliegue logístico para su instalación y desarrollo. Mercados Campesinos alternativos se centran en la movilidad, la incorporación de nuevas tecnologías y la entrega a domicilio. Su principal característica es realizar el mercado de forma móvil y ofrecer servicios de entrega directamente a los hogares, prescindiendo de un lugar físico para atender al público.

Tabla 4

Criterios de inclusión de documentos

Criterios de inclusión	Descripción
Relevancia	Los documentos deben proporcionar información y conocimientos pertinentes en cuanto a la conceptualización de aplicaciones móviles y/o aplicaciones móviles en el contexto de la comercialización agrícola.
Actualidad	Los documentos deben ser recientes y actualizados, preferiblemente publicados en los 5 últimos años. Esto asegurará que la información contenida esté alineada con las tecnologías, tendencias y desafíos actuales en el campo del desarrollo de aplicaciones móviles para la agricultura.
Credibilidad	Los documentos deben provenir de fuentes confiables, como revistas científicas, conferencias reconocidas, instituciones académicas o expertos en el campo.
Enfoque técnico	Los documentos deben abordar aspectos técnicos relevantes para el desarrollo de aplicaciones móviles, como el uso de plataformas móviles,

	tecnologías de programación, diseño de interfaces de usuario, gestión de datos y seguridad.
Enfoque en la comercialización agrícola	Los documentos deben contener información con respecto a la aplicación de tecnologías móviles en la comercialización de productos agrícolas, considerando aspectos como la conexión entre productores y consumidores.
Estudios de caso o investigaciones	Los documentos que presenten estudios de caso o investigaciones empíricas serán valiosos, ya que proporcionarán ejemplos prácticos y evidencia sobre la efectividad y los resultados obtenidos en proyectos similares.

Nota: esta tabla contiene los criterios de inclusión de documentos para la búsqueda de la bibliografía. Elaboración propia.

A partir del análisis de la información y una vez aplicados los criterios de inclusión y exclusión, se hace una depuración de aquellos documentos que tienen algún objetivo similar al del presente trabajo, es decir, desarrollo e implementación de aplicaciones para dispositivos móviles tipo e-commerce que permitan la comercialización de productos agrícolas aumentando los beneficios para productores y consumidores al eliminar la intermediación.

3.3. Conclusiones del capítulo

En la última década el desarrollo de aplicaciones móviles ha experimentado un crecimiento exponencial lo que ha generado la necesidad de impulsar nuevas investigaciones en el campo de la ingeniería de software con miras a desarrollar aplicaciones de alta calidad que satisfagan los requisitos de los usuarios y los avances de las tecnologías asociadas. Es así como se ha dedicado un esfuerzo considerable al estudio y formulación de metodologías que aborden los aspectos relacionados con la calidad y el tiempo de producción. Por ende, en la actualidad el desarrollo de aplicaciones móviles tiene una gran aceptación gracias al avance de la tecnología y producción de toda clase de dispositivos que permiten a los usuarios realizar tareas cotidianas ya sean de entretenimiento o laboral. (Molina Ríos et al., 2021)

Las aplicaciones móviles, al igual que las webs o de escritorio, requieren de metodologías de desarrollo para garantizar un producto de alta calidad, según

(Batarseh y Gonzalez, 2018) citado por (Molina Ríos et al., 2021) actualmente, el desarrollo de aplicaciones móviles se realiza en gran parte mediante metodologías ágiles, que se centran en el desarrollo iterativo, la flexibilidad y las pruebas.

Se puede colegir que las buenas prácticas ágiles son las más utilizadas en la actualidad por los beneficios para la construcción de software. En el desarrollo de aplicaciones móviles, no todas las metodologías ágiles son aplicables debido a su estructura enfocada en software o sistemas web, sin embargo, Scrum se destaca por sus beneficios en la realización de este tipo de proyectos.

4. Capítulo III: Modelo de Negocio y Flujo de Proceso

4.1. Descripción del Modelo de Negocio para la Aplicación Móvil -MECAD-

Un modelo de negocio es una descripción detallada de cómo una empresa o proyecto crea, entrega y captura valor. En esencia, es la forma en que una organización estructura su actividad comercial para generar ingresos y obtener beneficios. Un modelo de negocio describe cómo se relaciona una empresa con sus clientes, cómo se produce y ofrece el producto o servicio, cómo se establecen las fuentes de ingresos y cómo se administra y se asegura la rentabilidad del negocio.

Un modelo de negocio abarca varios aspectos clave, como la propuesta de valor, los segmentos de clientes, los canales de distribución, las relaciones con los clientes, las actividades clave, los recursos necesarios, las asociaciones clave, la estructura de costos y las fuentes de ingresos. Estos elementos se interconectan para crear un sistema que permite que una empresa funcione y genere valor tanto para sí misma como para sus clientes. (Ramón Toniut, 2021)

Un modelo de negocio puede ser innovador y disruptivo, introduciendo nuevas formas de generar ingresos y desafiar las prácticas tradicionales de la industria. También puede ser adaptado de modelos existentes, pero ajustado a las necesidades y características específicas de la organización. En última instancia, un modelo de negocio efectivo define cómo una empresa crea una oferta única, la entrega a los clientes de manera eficiente y rentable, y logra una ventaja competitiva en el mercado.

Por otro lado, La tecnoemprendeduría representa una perspectiva innovadora del emprendimiento tradicional, en la cual el conocimiento técnico, las habilidades de individuos talentosos y la capacidad para comercializar soluciones innovadoras se convierten en factores determinantes para obtener una ventaja competitiva. Actualmente, tanto el modelo de negocio como la tecnoemprendeduría son impulsos clave para incrementar el valor empresarial. Entre las industrias que

dependen en gran medida de las nuevas tecnologías, destaca la industria de servicios creativos en constante crecimiento. El conocimiento y el talento, considerados como la capacidad de acción, desempeñan un papel crucial en la estimulación de nuevos modelos de negocio. Para el crecimiento de las empresas, resulta fundamental contar con conocimiento, innovación, investigación y la comercialización efectiva de estas habilidades, además de métodos para generar ideas y conceptos novedosos. (Jablonski, s/f)

El modelo de negocio propuesto para MECAD se basa en la creación de una plataforma digital que conecta directamente a los productores agrícolas con los consumidores finales, eliminando intermediarios y facilitando la comercialización de productos agrícolas frescos y de calidad. A continuación, se describe el modelo.

MECAD ofrece a los productores agrícolas la posibilidad de crear perfiles en la plataforma, donde podrán listar y describir sus productos, establecer precios y cantidades disponibles, y proporcionar información sobre su ubicación y métodos de producción.

Los consumidores finales, a su vez, podrán acceder a la aplicación móvil de MECAD y buscar los productos agrícolas que deseen adquirir. Podrán explorar diferentes categorías, realizar búsquedas específicas y obtener detalles sobre los productos y los productores.

Una vez que los consumidores encuentren los productos deseados, podrán realizar la compra directamente a través de la aplicación. MECAD proporciona un sistema de pago seguro y opciones de entrega, ya sea por envío a domicilio o la posibilidad de recoger directamente en las instalaciones del productor.

Para asegurar la calidad y la confiabilidad de los productos, MECAD implementa un sistema de evaluación y retroalimentación de los consumidores, lo que permite a los usuarios compartir sus experiencias y calificar la calidad y el servicio de los productores.

Para garantizar la transparencia en las transacciones, MECAD actua como intermediario financiero, reteniendo el pago del consumidor hasta que se haya realizado la entrega exitosa de los productos. Posteriormente, transferirá el pago al productor, deduciendo una comisión por el servicio.

Además de la venta directa de productos agrícolas, MECAD ofrece servicios complementarios, como la posibilidad de establecer suscripciones de entrega recurrente de productos específicos, la venta de kits de productos agrícolas temáticos y la opción de realizar pedidos personalizados.

Para generar ingresos adicionales, MECAD explora oportunidades de publicidad y patrocinio dentro de la aplicación, permitiendo a las marcas relevantes promocionar sus productos o servicios a los usuarios de la plataforma.

Como parte de su estrategia de crecimiento, MECAD buscará establecer alianzas estratégicas con asociaciones de productores agrícolas, cooperativas y otros actores clave en la cadena de valor agrícola, con el fin de ampliar su oferta de productos y fortalecer su posición en el mercado.

Para atraer y retener a los usuarios, MECAD se enfoca en proporcionar una experiencia de usuario intuitiva y atractiva, con una interfaz amigable y funcionalidades que faciliten la búsqueda, la compra y la interacción entre los usuarios y los productores.

Finalmente, MECAD se esfuerza por desarrollar una sólida estrategia de marketing y promoción, utilizando diversas herramientas y canales para aumentar su visibilidad y atraer a una amplia base de usuarios, tanto productores como consumidores, en diferentes regiones geográficas.

4.2. Flujo de Caja

Tabla 5

Modelo de negocio proyectado MECAD

PRECIO					
Producto	Año 1	Año 2	Año 3	Año 4	Año 5
Comisiones por transacción	3.000	$3.102	$3.206	$3.311	$3.427
Publicidad	1.000.000	$1.034.000	$1.068.536	$1.103.797	$1.142.430
Venta de datos	2.000.000	$2.068.000	$2.137.071	$2.207.595	$2.284.860
Suscripciones Premium	600.000	$620.400	$641.121	$662.278	$685.458
Servicios de asesoramiento	1.000.000	$1.034.000	$1.068.536	$1.103.797	$1.142.430
		$0	$0	$0	$0
		$0	$0	$0	$0
		$0	$0	$0	$0

CANTIDAD					
Producto	Año 1	Año 2	Año 3	Año 4	Año 5
Comisiones por transacción	240	22	25	31	42
Publicidad	12	21	25	38	50
Venta de datos	12	25	33	39	51
Suscripciones Premium	36	20	29	46	52
Servicios de asesoramiento	24	20	27	30	36
0	0	0	0	0	0
0	0	0	0	0	0
SERVICIO DE RIFAS	0	0	0	0	0
Total CANTIDADES	324	108	139	184	231

INGRESOS					
Producto	Año 1	Año 2	Año 3	Año 4	Año 5
Comisiones por transacción	$720.000	$68.244	$80.140	$102.653	$143.946
Publicidad	$12.000.000	$21.714.000	$26.713.390	$41.944.296	$57.121.509
Venta de datos	$24.000.000	$51.700.000	$70.523.350	$86.096.187	$116.527.878
Suscripciones Premium	$21.600.000	$12.408.000	$18.592.519	$30.464.805	$35.643.822
Servicios de asesoramiento	$24.000.000	$20.680.000	$28.850.461	$33.113.918	$41.127.486
0	$0	$0	$0	$0	$0
0	$0	$0	$0	$0	$0
SERVICIO DE RIFAS	$0	$0	$0	$0	$0
Total Ingresos	$82.320.000	$106.570.244	$144.759.860	$191.721.860	$250.564.642

Nota: En esta tabla representa el flujo de caja proyectado del modelo de Negocio.

Fuente, elaboración propia.

Tabla 6

*Indicadores **financieros del proyecto**.*

INDICADORES DEL PROYECTO						
	Año 0	Año 1	Año 2	Año 3	Año 4	Año 5
Indicadores Financieros Proyectados						
Liquidez - Razón Corriente		N.A.	N.A.	N.A.	8,89	5,82
Prueba Acida		0	0	47	9	6
Rotación cartera (días),		0,00	0,00	0,00	0,00	0,00
Rotación Inventarios (días)		0,0	0,0	0,0	0,0	0,0
Rotación Proveedores (días)		0,0	0,0	0,0	0,0	0,0
Nivel de Endeudamiento Total		78,1%	82,7%	73,0%	45,7%	15,7%
Concentración Corto Plazo		0	0	0	0	1
Ebitda / Gastos Financieros		61,8%	124,2%	265,7%	584,1%	1674,8%
Ebitda / Servicio de Deuda		27,5%	49,0%	88,0%	145,9%	235,3%
Ebitda		35.674.750	58.252.512	94.799.326	140.062.667	197.149.036
Rentabilidad Operacional		-13,8%	10,6%	33,0%	48,5%	75,9%
Rentabilidad Neta		-83,8%	-33,5%	6,3%	27,0%	53,4%
Rentabilidad Patrimonio		-85,2%	-78,7%	16,7%	48,8%	55,7%
Rentabilidad del Activo		-18,7%	-13,6%	4,5%	26,5%	47,0%
Flujo de Caja y Rentabilidad						
Flujo de Operación		5.674.750	28.252.512	64.799.326	107.033.139	179.878.212
Flujo de Inversión	-510.000.000	150.000.000	0	0	0	0
Flujo de Financiación	510.000.000	-129.686.608	-118.913.333	-107.681.213	-95.979.370	-83.771.821
Flujo de caja para evaluación	**-510.000.000**	**155.674.750**	**28.252.512**	**64.799.326**	**107.033.139**	**179.878.212**

CRITERIOS DE DECISIÓN	
Tasa de oportunidad	20%
TIR (Tasa Interna de Retorno)	1,54%
VAN (Valor actual neto)	-199.245.551
PRI (Periodo de recuperación de la inversión)	4,76
Nivel de endeudamiento inicial del negocio.	70,59%

Nota: En esta tabla se relacionan los indicadores financieros del proyecto Mecad.

Fuente, elaboración propia.

4.3. Flujos del Negocio

Se han desarrollado dos flujos de negocio clave: el flujo de inscripción de proveedores y manejo de inventarios, y el flujo de pedidos.

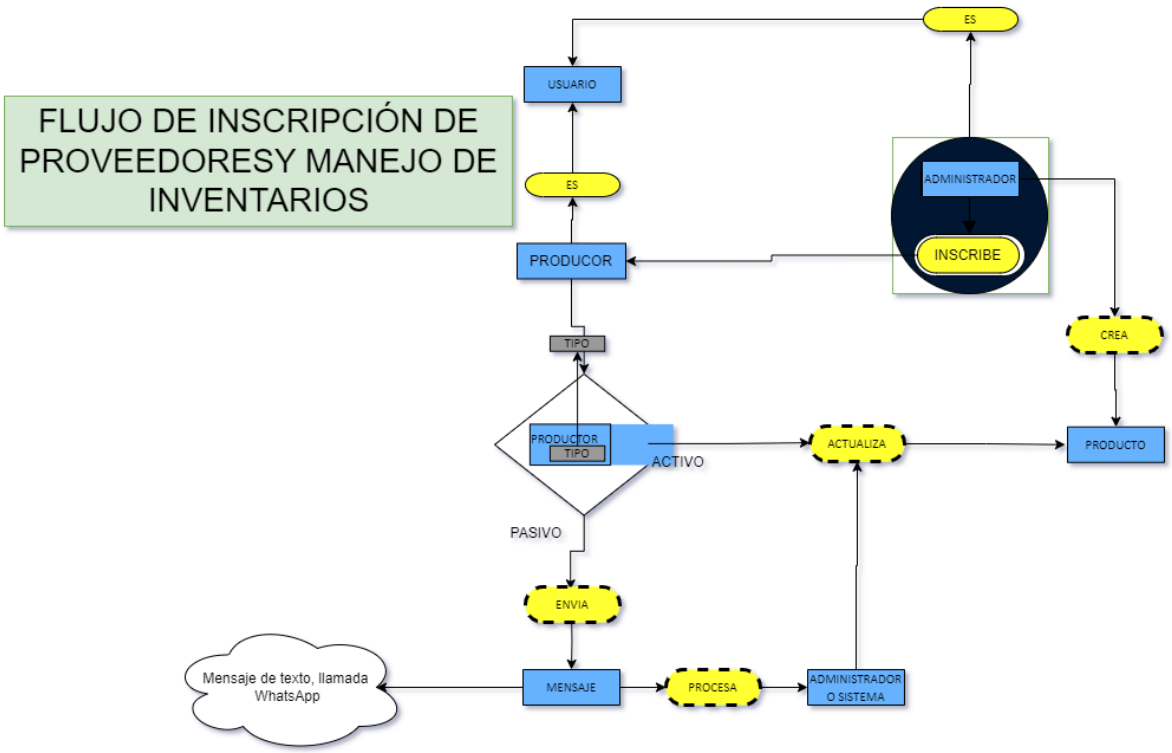
Ambos flujos de negocio están estrechamente relacionados con la descripción del modelo de negocio para la aplicación móvil de MECAD. La inscripción de proveedores y el manejo de inventarios son componentes clave para asegurar una amplia oferta de productos en la plataforma, lo cual es fundamental para atraer a los clientes. Por su parte, el flujo de pedidos es el proceso central de la aplicación móvil, ya que permite a los clientes realizar compras de manera conveniente y eficiente.

4.3.1. Flujo de Inscripción de Proveedores y Manejo de Inventarios

El flujo de inscripción de proveedores y manejo de inventarios se refiere al proceso mediante el cual los proveedores se registran en la plataforma MECAD por medio de un administrador para ofrecer sus productos. Este flujo involucra la captura de información del proveedor, como datos personales, información de contacto y detalles sobre los productos que ofrecen. Además, se implementa la funcionalidad para gestionar el inventario de los proveedores, lo cual implica realizar actualizaciones en tiempo real sobre la disponibilidad de productos, controlar las existencias y realizar ajustes cuando sea necesario.

Figura 2

Flujo de Inscripción de Proveedores y Manejo de Inventarios



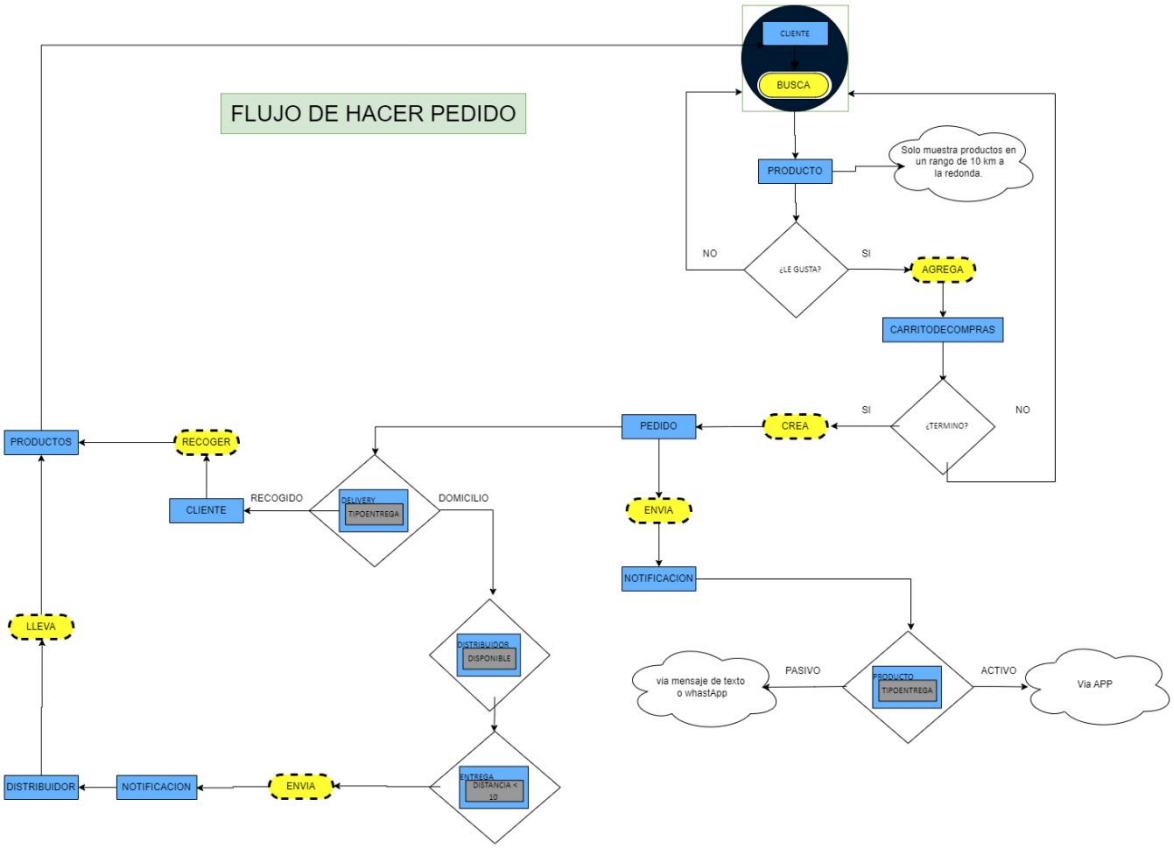
Nota: proceso mediante el cual los proveedores se registran en la plataforma MECAD. Elaboración propia.

4.3.2. Flujo de Pedido

Por otro lado, el flujo de pedidos se centra en la experiencia del usuario al realizar una compra en MECAD. Los clientes pueden explorar los productos disponibles, agregarlos al carrito de compras, especificar detalles como la cantidad deseada y realizar el pago correspondiente. Este flujo implica integraciones con el módulo de autenticación y seguridad para verificar la identidad del cliente, así como con el módulo de gestión de inventarios para garantizar que los productos seleccionados estén disponibles. Se debe aclarar que el sistema cuenta con una restricción de listar productos con un rango de 10KM a la redonda y de una compra mínima de 30 mil pesos colombianos para poder tener una logística de entrega viable económicamente.

Figura 3

Flujo de Pedido



Nota: proceso mediante el cual los clientes o consumidores finales hacen una compra a través de la plataforma MECAD.

4.4. Conclusiones del capítulo

El modelo de negocio de MECAD se ha estructurado como una estrategia efectiva para impulsar el crecimiento y el éxito de la empresa. Al enfocarse en la prestación de servicios de alta calidad y la satisfacción del cliente, MECAD busca establecer una sólida reputación en el mercado. Su enfoque en la innovación tecnológica y la mejora continua permite a MECAD mantenerse a la vanguardia de la industria, adaptándose a las cambiantes demandas y necesidades de los clientes.

El modelo de negocio de MECAD también se destaca por su enfoque en la colaboración y asociaciones estratégicas. El modelo permite establecer relaciones sólidas con proveedores y socios clave, lo que permite ofrecer soluciones integrales

y personalizadas a sus clientes. Además, su enfoque en la gestión eficiente de recursos y la optimización de costos contribuye a la rentabilidad y sostenibilidad de la empresa.

En general, el modelo de negocio de MECAD permite ser exitoso al ofrecer valor agregado a los clientes, mantenerse a la vanguardia de la tecnología y establecer relaciones sólidas en la industria. Con su enfoque en la calidad, la innovación y la colaboración, MECAD se proyecta como un referente en su sector y está preparada para enfrentar los desafíos futuros y aprovechar las oportunidades emergentes.

5. Capítulo IV: Metodología y Diseño de desarrollo de la aplicación MECAD

5.1. Identificación de Procesos por Sistematizar

Se han identificado los siguientes procesos como los más importantes para ser sistematizados en la plataforma de e-commerce de eliminación de intermediación entre campesinos y consumidores finales:

Registro de Productores: Este proceso permitirá a un usuario administrador registrar n cantidad de productores en la plataforma y crear un perfil que incluya información sobre sus productos, precios y ubicación.

Registro de Consumidores: Los consumidores finales también podrán registrarse en la plataforma y crear un perfil que les permita realizar pedidos, pagar y recibir los productos.

Búsqueda y Compra de Productos: Este proceso incluye la búsqueda de productos, la selección de un productor, la colocación de un pedido y el pago.

Gestión de Pedidos: Este proceso incluye la recepción de pedidos, la verificación de los productos y el envío a los consumidores finales.

Gestión de Pagos: Este proceso se encarga de procesar los pagos realizados por los consumidores y transferir los fondos a los productores.

Gestión de Inventarios: Este proceso permite a los productores gestionar sus inventarios y actualizar su disponibilidad de productos en la plataforma.

Servicio notificaciones: Este proceso incluye la notificación bien sea por medio de la APP, de un mensaje de texto o de WhatsApp al productor de la realización de una compra por parte del consumidor final, tipo de entrega, tiempo aproximado de llegada.

5.2. Proceso funcional de la APP

Según la ingeniería de requisitos, se pueden emplear diferentes métodos para el proceso de elicitación de requisitos, teniendo en cuenta el contexto y las necesidades del proyecto. En nuestro caso, al no contar con un único usuario final, se optó por realizar un par de encuestas dirigidas a los dos tipos de usuarios involucrados.

La realización de estas encuestas tiene varias ventajas. En primer lugar, nos permite recopilar datos cuantitativos de una muestra representativa de ambos tipos de usuarios (proveedor agricultor y cliente final), lo que nos brinda una visión generalizada de sus preferencias y necesidades. Esto nos ayuda a comprender las demandas y expectativas de cada grupo, lo cual es fundamental para el diseño y desarrollo del sistema.

Además, las encuestas nos permiten identificar patrones y tendencias en las respuestas de los usuarios, lo que nos ayuda a priorizar adecuadamente los requisitos del sistema. Al analizar los resultados, podemos identificar las características o funcionalidades más solicitadas o consideradas importantes por la mayoría de los usuarios, lo cual nos permite enfocar nuestros esfuerzos en áreas clave.

Asimismo, las encuestas nos brindan la oportunidad de obtener retroalimentación directa de los usuarios. A través de preguntas abiertas y comentarios adicionales, los usuarios pueden expresar sus opiniones, sugerencias y preocupaciones sobre el sistema. Esta retroalimentación es invaluable para comprender mejor las expectativas de ambos grupos de usuarios y ajustar los requisitos del sistema en consecuencia.

Al realizar estas encuestas, buscamos obtener una visión amplia y equilibrada de los requisitos del sistema, teniendo en cuenta las perspectivas de los dos tipos de usuarios. Esto nos permitirá diseñar un sistema que satisfaga las necesidades de ambos grupos y proporcione una experiencia óptima para todos los usuarios.

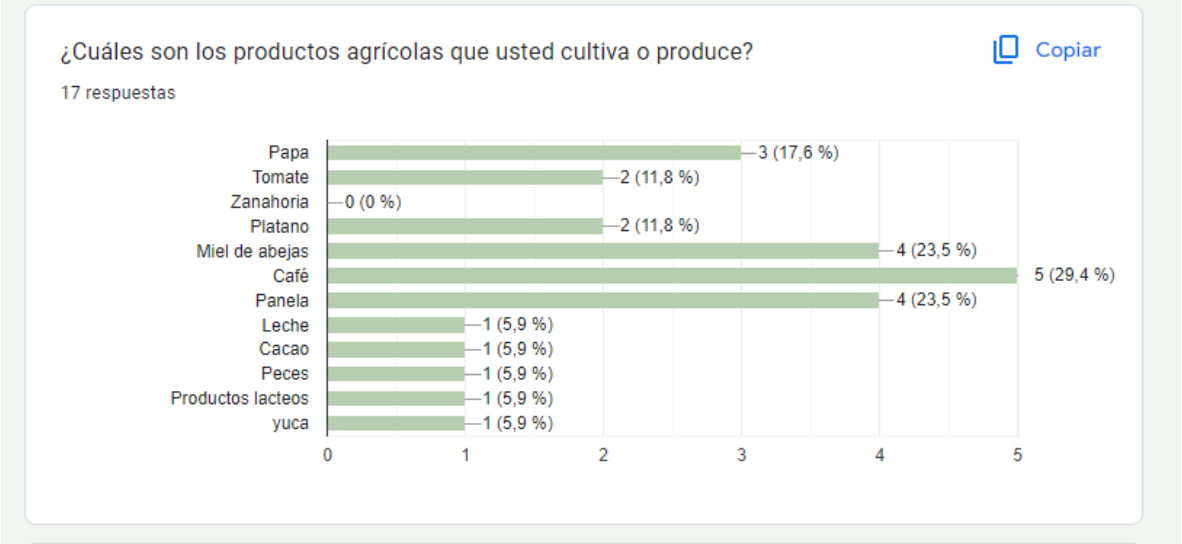
5.2.1. Encuesta a productores campesinos

La encuentra a productores campesinos, fue importante para tener una muestra de la cobertura de internet y la adopción de tecnología en el proceso de comercialización de productos agrícolas.

Así mismo proporcionó información valiosa sobre las necesidades y desafíos específicos que enfrentan los proveedores campesinos en el uso de tecnología para la comercialización de productos agrícolas. Esto permitió adaptar la plataforma MECAD a sus requerimientos y garantizar que se brinden soluciones efectivas para su participación exitosa en el mercado digital. Al comprender las realidades y limitaciones de los proveedores campesinos, se pueden implementar estrategias adecuadas para superar barreras tecnológicas y mejorar la inclusión digital en el sector agrícola.

Figura 4

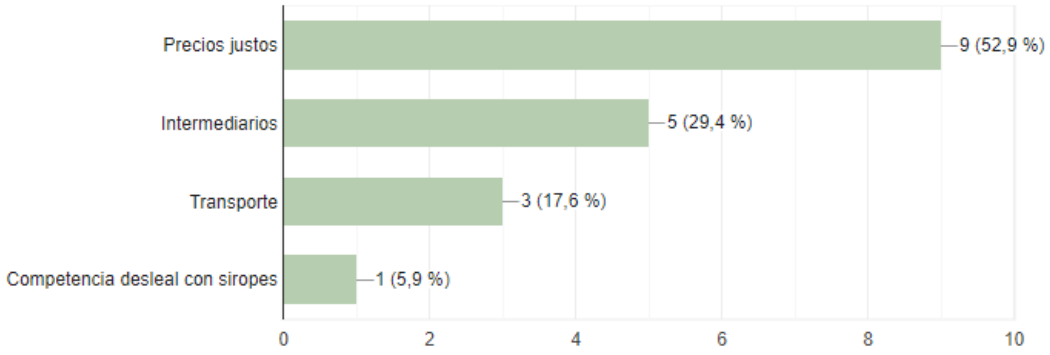
Resultados gráficos de la encuesta a productores campesinos.



¿Cuál o cuáles son las principales dificultades que usted encuentra al momento de comercializar los productos?

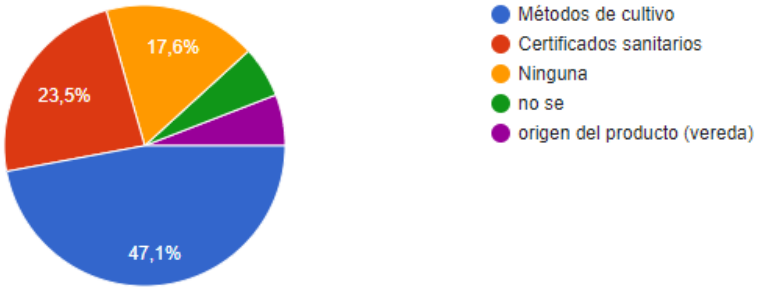
Copiar

17 respuestas



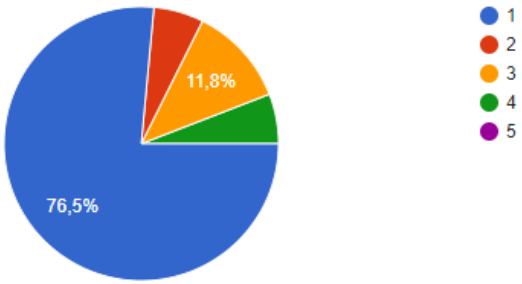
¿Qué información considera importante mostrarle a quién compre los productos?

17 respuestas



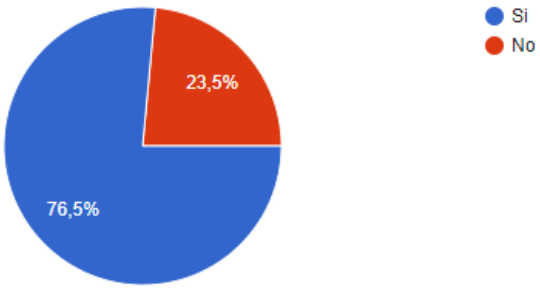
¿En qué medida utiliza el internet actualmente en el proceso de comercialización y ventas de lo producido? Siendo 1 muy poco y 5 mucho.

17 respuestas



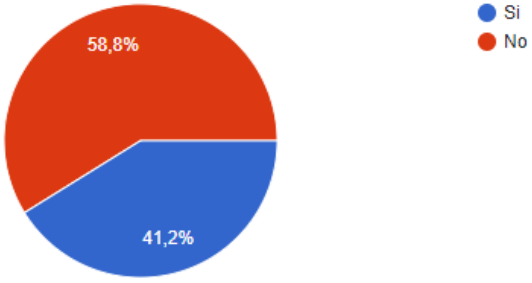
¿Cuenta con un celular inteligente?

17 respuestas



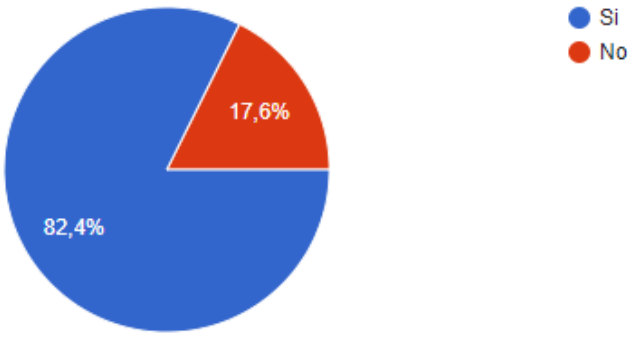
¿Tiene plan de datos o internet?

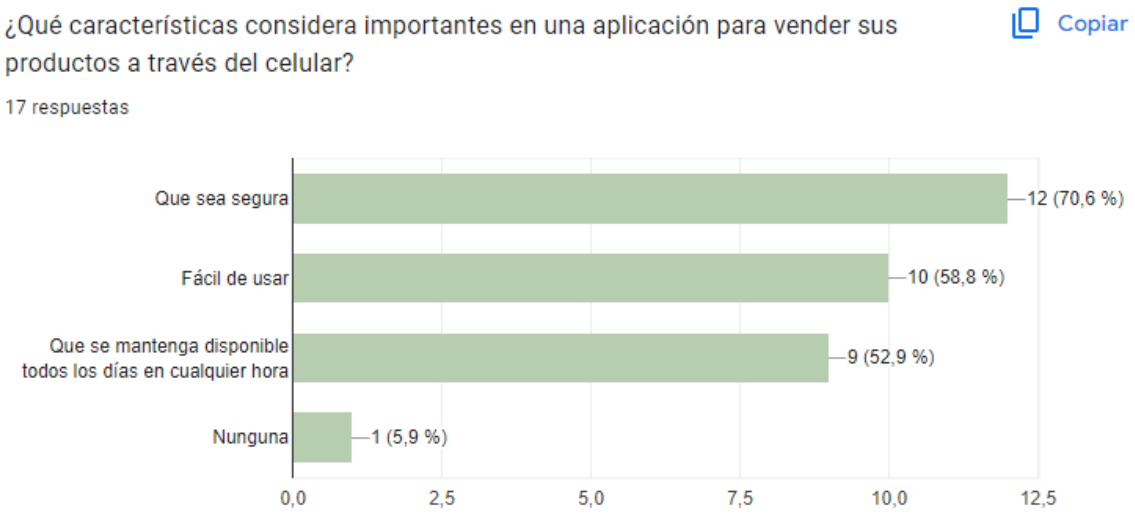
17 respuestas



¿Utilizaría usted una aplicación en el celular para mostrar y/o vender los productos que cultiva directamente al consumidor final?

17 respuestas





Nota: Las gráficas representan los resultados de la encuesta aplicada a los campesinos productores con respecto a la utilización de la aplicación MECAD para comercializar sus productos. Elaboración propia.

5.2.2. Encuesta al Consumidor Final

Por otro, con la encuesta a consumidores finales se identificaron las necesidades de los clientes potenciales obteniendo así algunos requisitos tanto funcionales como

no funcionales, además se tienen definieron algunas preguntas respecto a la viabilidad del proyecto mismo de cara a los consumidores finales.

5.3. Identificación y Descripción de Requisitos

5.3.1. Requisitos Funcionales

A continación se presenta un listado de los requisitos funcionales represntados historias de usuario según la metodología scrum que se detallan en el Backlog, cada uno de las historias de usuario está agrupada por epicas o módulos funcionales y serán gestionados a través de la herramienta de azure devops de Microsoft.

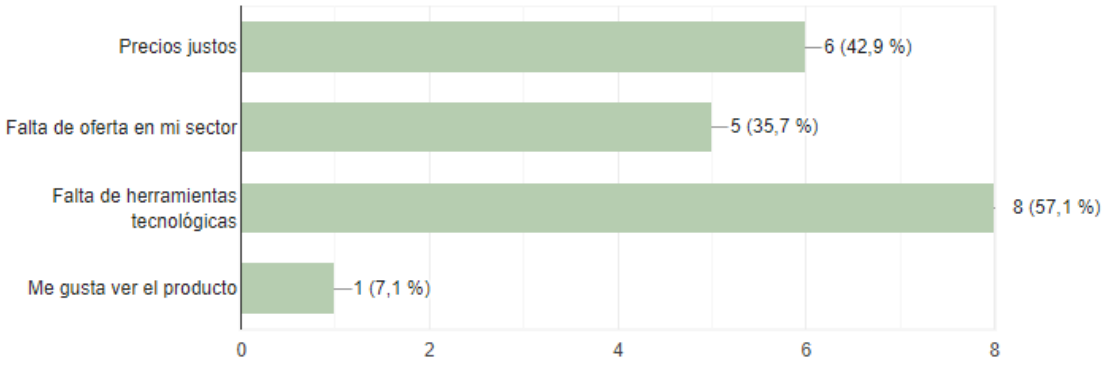
Figura 5

Resultados de las encuesta al consumidor final

¿Cuáles son los principales obstáculos que usted tiene al buscar productos agrícolas?

Copiar

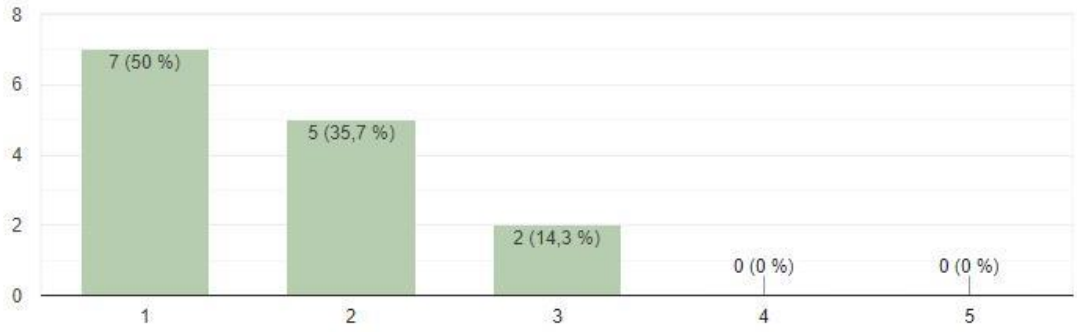
14 respuestas



En una escala del 1 al 5 ¿Qué tan satisfecho/a está con las soluciones actuales disponibles en el mercado para cubrir sus necesidades respecto a este tipo de productos? Siendo 1 muy poco satisfecho y 5 muy satisfecho.

Copiar

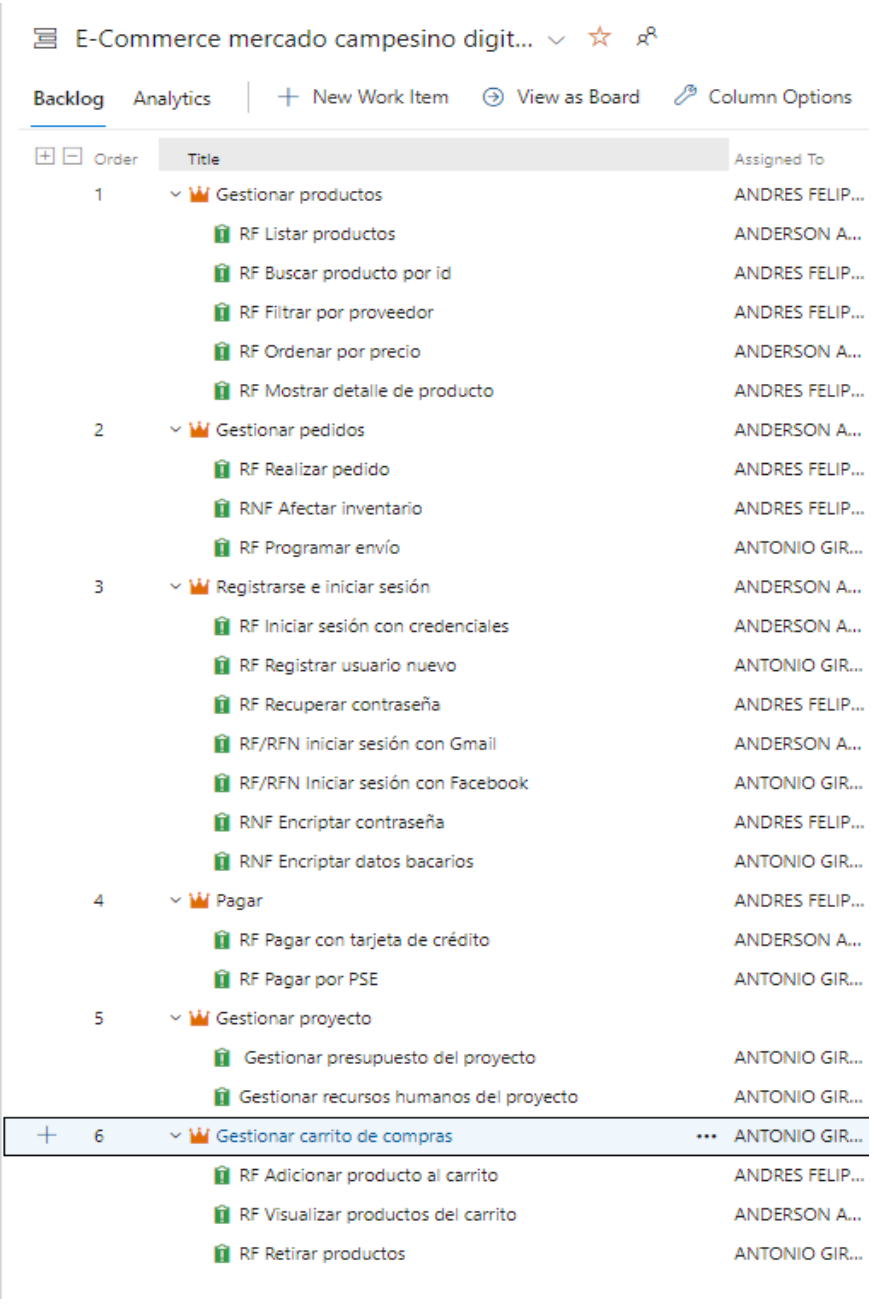
14 respuestas



Nota: Las gráficas representan los resultados de la encuesta aplicada a los consumidores o clientes finales con respecto a la utilización de la aplicación MECAD para comprar productos agrícolas. Elaboración propia.

Figura 6

Requisitos funcionales



Nota: esta imagen tomada de Azure DevOps representa los Requisitos funcionales de la aplicación. Elaboración propia.

5.3.2. Requisitos No Funcionales

Se establecieron los siguientes criterios de calidad según la norma ISO 25010 que deben acompañar la funcionalidad de la plataforma para satisfacer las necesidades de los diferentes usuarios de la aplicación.

Tabla 7

Requisitos no funcionales.

Código	Atributo	Descripción	Métrica
RFN1	Mantenibilidad (Capacidad para ser modificado)	Se debe garantizar que la aplicación pueda crecer en cantidad de usuarios, proveedores, cobertura y funcionalidades a futuro en un futuro próximo.	¿Qué tan grande es el impacto en la modificación en del producto del software? X=A/B, dónde A=Número de funciones afectadas en el software después de la modificación. B=Número de funcionalidades totales. Entonces: 0 <= X <= 1
RFN2	Seguridad	Se debe garantizar que la seguridad de la información sensible del usuario bien sea cliente o proveedor, se encuentre segura y no sea vista por nadie más, mediante tácticas de cifrado, para generar confianza en los diferentes usuarios de la APP.	¿Qué tan completa es la implementación del cifrado de datos? X=A/B, dónde A=Número de elementos de datos que requieren implementación de la funcionalidad de cifrado/descifrado y que fueron implementados según lo especificado en los requerimientos. B= Número de elementos de datos que requieren la función de cifrado/descifrado según las especificaciones. Entonces: 0 <= X <= 1.
RFN3	Usabilidad	Se debe garantizar que la aplicación sea fácil de manejar, pues una de las promesas de valor es que los agricultores colombianos puedan aprender a manejarla fácilmente si así lo desean y facilitar el proceso de gestión de pedidos y de inventarios.	¿Qué tan intuitiva y fácil de manejar es la app? Encuesta de satisfacción a los usuarios (Clientes y proveedores) en una escala de 1 a 5. Dónde X: Facilidad de manejo.

			Entonces: X>=4
RFN4	Interoperabilidad	Se debe garantizar interoperabilidad con sistemas de terceros (Para pagos)	¿Qué tan estable son las funcionalidades que se relacionan con la interoperabilidad durante el ciclo de vida del software? X=1-A/B, Dónde A=Número de funciones cambiadas durante fases del ciclo de vida del desarrollo B=Número de funciones descritas en los requisitos.

Nota: En esta tabla se describen los Requisitos no funcionales de la aplicación Medad. Elaboración propia

5.3.3. Escenarios de Calidad

En el presente trabajo, se ha considerado de vital importancia incluir los escenarios de calidad como una herramienta fundamental en el diseño y evaluación de la arquitectura del sistema "MECAD" (Mercados Campesinos Digitales). Los escenarios de calidad nos permiten comprender las necesidades y expectativas de los usuarios y partes interesadas en términos de calidad del sistema.

En el contexto de los mercados campesinos digitales, es crucial garantizar los atributos descritos en la tabla "*Requisitos no funcionales*" para lograr el éxito y la satisfacción de todos los actores involucrados.

Al utilizar escenarios de calidad, podemos identificar de manera temprana los requisitos no funcionales y otros aspectos de calidad que son clave para el buen funcionamiento del sistema MECAD. Estos escenarios nos ayudarán a definir y establecer métricas y objetivos claros que permitan medir y evaluar si el sistema cumple con los criterios de calidad establecidos.

Además, los escenarios de calidad nos brindarán una base sólida para tomar decisiones de diseño adecuadas. Nos permitirán evaluar diferentes opciones

arquitectónicas y seleccionar las soluciones que mejor se ajusten a los requisitos y expectativas de calidad identificados. De esta manera, aseguraremos que el sistema MECAD esté diseñado de manera óptima para cumplir con los estándares de calidad requeridos.

Los escenarios de calidad serán una herramienta invaluable para la mejora continua y la evolución del sistema MECAD. Nos permitirán evaluar el rendimiento del sistema en producción y durante su evolución, identificar áreas de mejora y establecer métricas y objetivos para lograr una satisfacción continua de las necesidades cambiantes de los usuarios finales.

A continuación, se muestra cada uno de los requisitos funcionales con al menos un escenario y su descripción, la respuesta esperada y su respectiva métrica.

Tabla 8

Requisitos funcionales de MECAD

ESCENARIO DE MANTENIBILIADAD	Fuente/origen	Usuarios
	Evento/Estímulo	Aumento de usuarios y cobertura
	Artefacto	Aplicación
	Entorno	Bajo condiciones normales de uso y expansión
	Respuesta	Mantener las funcionalidades y prestaciones a pesar del crecimiento de los usuarios

	Medida/Métrica	¿Qué tan grande es el impacto en la modificación en del producto del software? X=A/B, dónde A=Número de funciones afectadas en el software después de la modificación. B=Número de funcionalidades totales. Entonces: 0 <= X <= 1
ESCENARIO DE SEGURIDAD	Fuente/origen	Aplicación módulo logueo
	Evento/Estímulo	Autenticación
	Artefacto	Aplicación
	Entorno	Bajo condiciones de intento de ataque por fuerza bruta.
	Respuesta	Protección a la vulnerabilidad

	Medida/Métrica	¿Qué tan completa es la implementación del cifrado de datos? X=A/B, dónde A=Número de elementos de datos que requieren implementación de la funcionalidad de cifrado/descifrado de datos y que fueron implementados según lo especificado en los requerimientos. B= Número de elementos de datos que requieren la función de cifrado/descifrado de datos según las especificaciones. Entonces: 0 <= X <= 1.
ESCENARIO DE USABILIDAD	Fuente/origen	Usuario
	Evento/Estímulo	Cliente navegando en el entorno de la aplicación
	Artefacto	Aplicación - Front end
	Entorno	Momentos de alto tráfico por temporada.
	Respuesta	Usabilidad intuitiva y respuesta oportuna de las pantallas

	Medida/Métrica	¿Qué tan intuitiva y fácil de manejar es la app? Encuesta de satisfacción a los usuarios (Clientes y proveedores) en una escala de 1 a 5. Dónde X: Facilidad de manejo. Entonces: X>=4
ESCENARIO DE INTEROPERABILIDAD	Fuente/origen	Aplicación módulo de logueo
	Evento/Estímulo	Usuario inicia sesión con credenciales con proveedor de autenticación externo.
	Artefacto	Aplicación
	Entorno	Inicio de sesión
	Respuesta	La aplicación enruta hacia el proveedor de login externo y solicita el token.
	Medida/Métrica	¿Qué tan estable son las funcionalidades de interoperabilidad durante el ciclo de vida del software? X=1-A/B, Dónde A=Número de funciones cambiadas durante fases del ciclo de vida del desarrollo B=Número de funciones descritas en los requisitos.

ESCENARIO DE SEGURIDAD	Fuente/origen	Aplicación módulo pagos
	Evento/Estímulo	Pago por medio electrónico}
	Artefacto	Aplicación
	Entorno	Bajo condiciones de intento de ataque por fuerza bruta.
	Respuesta	Protección a la vulnerabilidad
	Medida/Métrica	¿Qué tan completa es la implementación del cifrado de datos? X=A/B, dónde A=Número de elementos de datos que requieren implementación de la funcionalidad de cifrado/descifrado de datos y que fueron implementados según lo especificado en los requerimientos. B= Número de elementos de datos que requieren la función de cifrado/descifrado de datos según las especificaciones. Entonces: 0 <= X <= 1.

Nota: Esta tabla muestra cada uno de los requisitos funcionales con al menos un escenario y su descripción, la respuesta esperada y su respectiva métrica. Elaboración propia.

5.3.4. Estructura de desglose de trabajo

Tabla 9

Estructura de desglose

Nivel 1	Nivel 2	Nivel 3
1 EDT	1.1 Planificación del proyecto	1.1.1 Identificación de los Requisitos del cliente 1.1.2 Definición de los objetivos del proyecto 1.1.3 Desarrollo del plan de proyecto 1.1.4 Aprobación del plan de proyecto
	1.2 Análisis y diseño	1.2.1 Investigación de mercado 1.2.2 Identificación de campesinos y productos 1.2.3 Diseño de la interfaz de usuario 1.2.4 Diseño de la arquitectura de la aplicación 1.2.5 Desarrollo de especificaciones técnicas 1.2.6 Aprobación del diseño y especificaciones
	1.3 Desarrollo	1.3.1 Configuración del ambiente de desarrollo 1.3.2 Desarrollo de la base de datos 1.3.3 Desarrollo del sistema de pago 1.3.4 Desarrollo de la gestión de pedidos 1.3.5 Desarrollo del sistema de calificación y comentarios 1.3.6 Integración de las funcionalidades desarrolladas 1.3.7 Pruebas unitarias y de integración
	1.4 Pruebas	1.4.1 Pruebas de aceptación del cliente 1.4.2 Pruebas de rendimiento 1.4.3 Pruebas de seguridad 1.4.4 Corrección de errores
	1.5 Implementación	1.5.1 Preparación del ambiente de producción 1.5.2 Instalación de la aplicación 1.5.3 Configuración del ambiente de producción 1.5.4 Pruebas en ambiente de producción 1.5.5 Aprobación del ambiente de producción
	1.6 Capacitación y lanzamiento	1.6.1 Capacitación a los campesinos y consumidores finales 1.6.2 Comunicación del lanzamiento de la aplicación 1.6.3 Lanzamiento de la aplicación
	1.7 Mantenimiento y soporte	1.7.1 Atención de solicitudes de soporte 1.7.2 Corrección de errores y fallas 1.7.3 Actualizaciones y mejoras

Nota: Esta tabla describe la descomposición por etapas del proyecto. Elaboración propia.

5.4. Diseño de base de datos

Modelar una aplicación mediante un diagrama de clases tiene varias ventajas y aporta una serie de beneficios en el proceso de desarrollo y mantenimiento del software. A continuación, se destacan algunas de las razones por las que es importante utilizar un diagrama de clases en el diseño de tu app:

- Representación visual clara: El diagrama de clases proporciona una representación visual clara de las entidades, atributos y relaciones del sistema. Esto facilita la comprensión y comunicación entre los miembros del equipo de desarrollo, permitiendo una visión global de la estructura y funcionamiento de la aplicación.

- Organización estructurada: El diagrama de clases ayuda a organizar la estructura del software de manera lógica y coherente. Permite identificar las entidades principales y sus atributos, así como las relaciones entre ellas. Esto facilita la identificación de los componentes clave de la aplicación y proporciona una base sólida para la implementación y el mantenimiento.

- Reutilización de código: Al modelar las clases y relaciones en el diagrama, es posible identificar oportunidades de reutilización de código. Esto significa que ciertos componentes, como las clases de productos o usuarios, pueden ser diseñados de forma modular y reutilizados en diferentes partes de la aplicación. Esto ahorra tiempo y esfuerzo en el desarrollo, ya que no es necesario escribir código nuevo para funcionalidades similares.

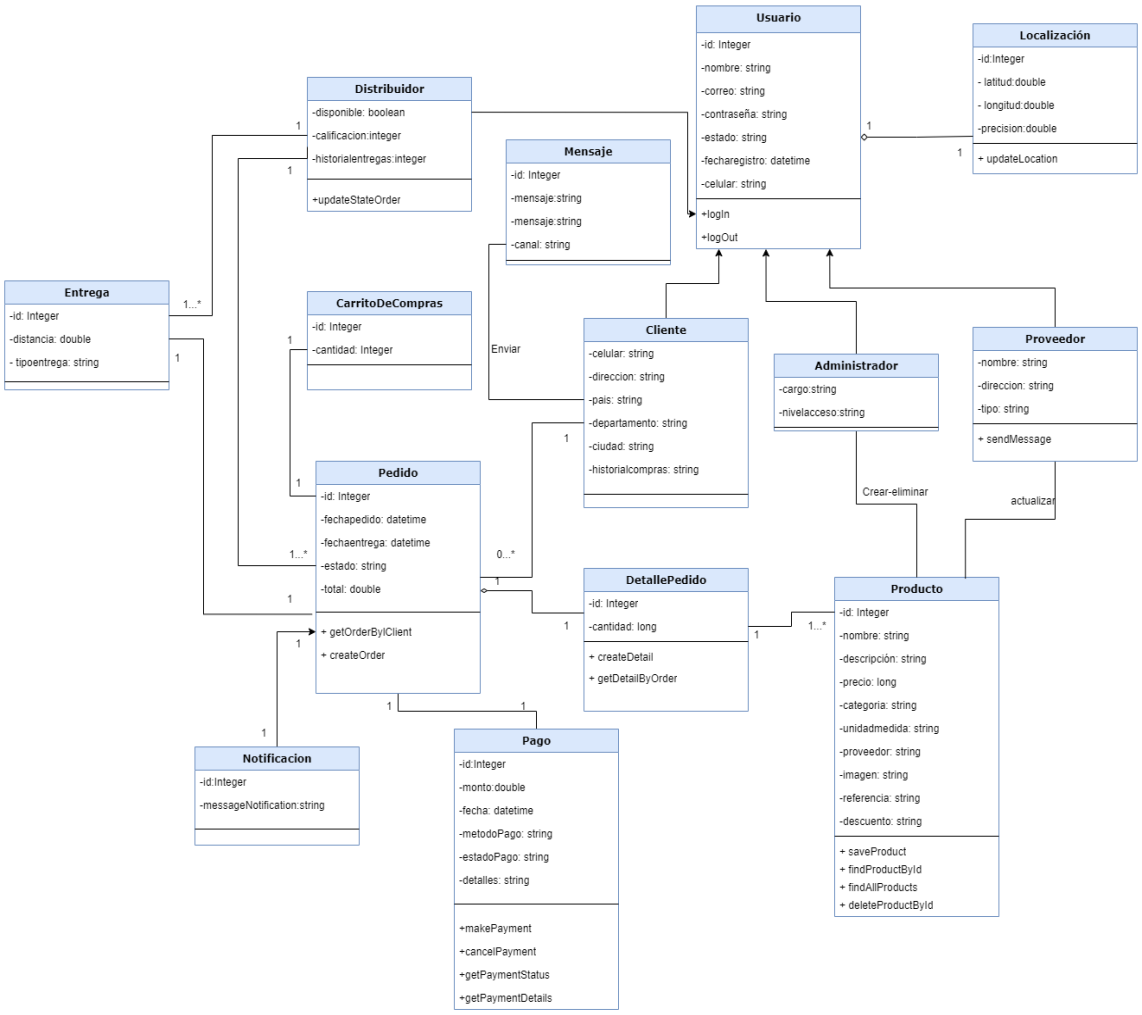
- Mantenibilidad y escalabilidad: Un diagrama de clases bien diseñado facilita el mantenimiento y la escalabilidad del software. Proporciona una visión clara de la estructura del sistema, lo que ayuda a identificar las áreas que necesitan modificaciones o mejoras. Además, al tener definidas las relaciones y responsabilidades de cada clase, se puede realizar un seguimiento más eficiente de los cambios y actualizaciones en el futuro.

- Documentación y comprensión del sistema: El diagrama de clases sirve como una forma de documentación del sistema, permitiendo a los desarrolladores y otros interesados comprender rápidamente la estructura y las funcionalidades del software. Además, ayuda a minimizar la dependencia de conocimiento en una sola persona, ya que el diseño está documentado y puede ser comprendido por otros miembros del equipo.

La siguiente es la representación de la vista lógica de los datos, representa una entidad y las relaciones entre ellas, mediante un diagrama de clases. El diseño se enfocó en garantizar la integridad y consistencia de los datos, así como en optimizar el rendimiento y la eficiencia en la gestión de la información, y mantener la usabilidad para los diferentes usuarios incluyendo proveedores que son agricultores de Colombia.

Figura 7

Vista lógica de los datos



Nota: Esta gráfica es la representación de la vista lógica de los datos, representa una entidad y las relaciones entre ellas, mediante un diagrama de clases

5.5. Diseño de arquitectura

La arquitectura es fundamental en cualquier proyecto de desarrollo de software, incluyendo "MECAD", ya que es la base sobre la cual se construye la aplicación móvil y es responsable de su rendimiento, seguridad, escalabilidad y mantenibilidad.

En el caso de "MECAD", la arquitectura es especialmente importante porque la aplicación debe ser capaz de manejar grandes cantidades de información sobre los productos y proveedores agrícolas, así como de permitir la realización de

transacciones comerciales en línea de manera segura y confiable. Además, la aplicación debe ser fácil de usar y estar disponible en múltiples plataformas, lo que puede representar un desafío adicional en términos de arquitectura.

Una buena arquitectura permite una mayor eficiencia en la implementación de funcionalidades nuevas, una mayor flexibilidad para responder a cambios en los requisitos y una mayor facilidad para la integración con otros sistemas y aplicaciones. Además, una arquitectura bien diseñada puede reducir el costo y el tiempo necesario para realizar pruebas y mantenimiento del software.

El proyecto se basa en una arquitectura de referencia conocida como microservicios. Esta arquitectura busca reducir la dependencia entre las distintas funcionalidades del sistema, lo que a su vez aumenta la resiliencia de la aplicación al permitir un mayor desacoplamiento entre los diferentes módulos. Esta separación en microservicios proporciona beneficios significativos, como una mayor facilidad de escalado y un mantenimiento más sencillo.

Al adoptar la arquitectura de microservicios, cada funcionalidad del sistema se implementa como un servicio independiente y autónomo. Estos microservicios se comunican entre sí a través de interfaces bien definidas, generalmente utilizando tecnologías como APIs o mensajería asincrónica. Esto permite que cada microservicio pueda ser desarrollado, desplegado y gestionado de forma independiente, lo que facilita la escalabilidad y el mantenimiento del sistema en su conjunto.

La principal ventaja de esta arquitectura es su capacidad para gestionar la complejidad al descomponer la aplicación en componentes más pequeños y especializados. Cada microservicio se centra en una funcionalidad específica, lo que facilita el desarrollo y la evolución de cada componente de forma aislada. Además, el desacoplamiento entre los microservicios permite que cada uno pueda escalar de manera independiente según sus necesidades, lo que mejora la capacidad de respuesta del sistema ante cambios en la carga de trabajo.

Otro beneficio clave de la arquitectura de microservicios es su facilidad de mantenimiento. Como los microservicios son unidades autónomas, los cambios en una funcionalidad específica no afectan al resto del sistema. Esto reduce el riesgo de errores y conflictos durante las actualizaciones y facilita la corrección de problemas o la introducción de nuevas características sin afectar a otras partes del sistema. Además, en el proyecto se utilizan herramientas de integración continua (CI) y entrega continua (CD) en el marco de desarrollo en la nube de Azure DevOps. Estas herramientas permiten agilizar y facilitar el trabajo colaborativo entre los miembros del equipo.

La integración continua (CI) implica la integración regular y automatizada del código fuente del proyecto. Con esta práctica, se realizan pruebas automáticas y se generan informes de calidad del código de manera constante, lo que permite detectar y corregir errores de forma temprana. Esto contribuye a mantener un código base estable y de alta calidad a medida que se desarrolla el proyecto.

Por otro lado, la entrega continua (CD) se refiere a la automatización del proceso de implementación del software en distintos entornos, como el de pruebas o producción. Esta automatización permite reducir el tiempo y los errores asociados con la implementación manual. Al utilizar Azure DevOps, se aprovecha la plataforma de nube de Microsoft para gestionar y orquestar el proceso de entrega continua de forma eficiente.

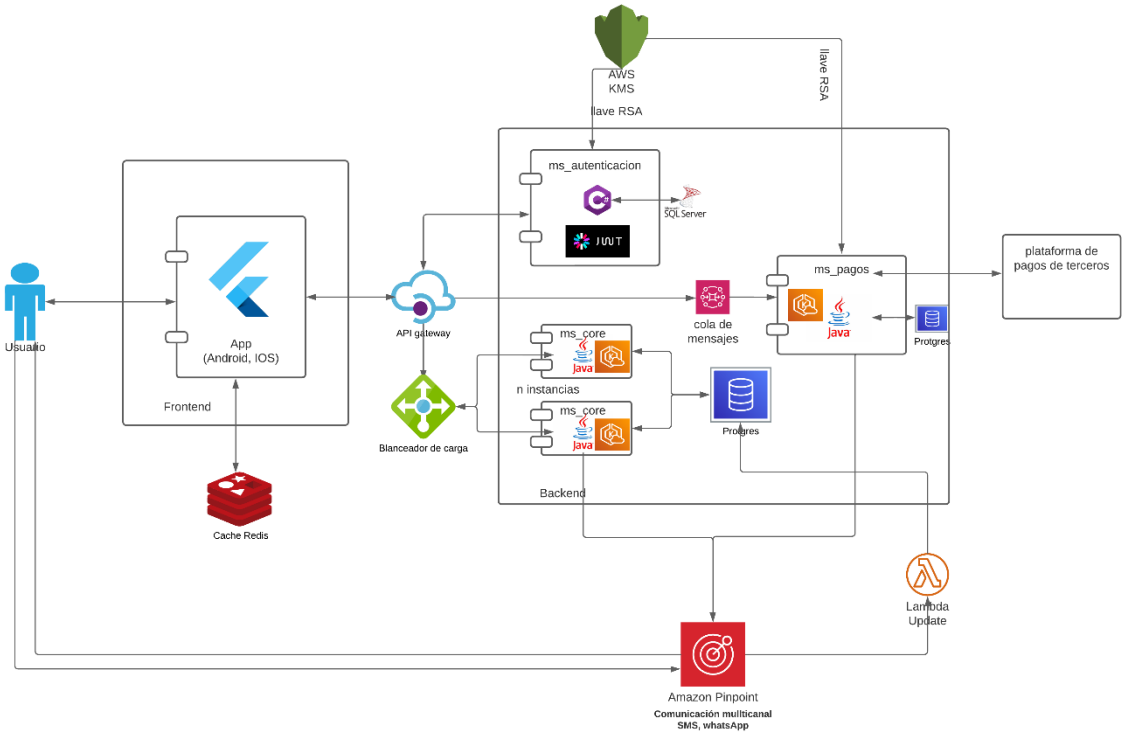
Azure DevOps también facilita la colaboración entre los miembros del equipo. Proporciona herramientas de seguimiento de tareas, gestión de proyectos y control de versiones, lo que permite una coordinación y comunicación efectiva. Los miembros del equipo pueden trabajar en paralelo en diferentes funcionalidades o módulos, y la plataforma de Azure DevOps se encarga de gestionar los cambios y garantizar la integridad del código y los artefactos del proyecto.

5.5.1. Arquitectura General

A continuación, se presenta una vista de componentes que describe de manera sencilla la arquitectura general del sistema y plasma el diseño de la aplicación:

Figura 8

Arquitectura general del Sistema MECAD



Nota: esta gráfica describe de manera sencilla la arquitectura general del sistema y plasma el diseño de la aplicación

La arquitectura del sistema se divide en varios componentes principalmente en dos grandes contenedores, (FronEnd del lado del cliente y backEnd del lado del servidor)

- FronEnd: El FronEnd se hace con flutter ya que como se mencionó en capítulos pasados, ofrece varias ventajas para el desarrollo de aplicaciones móviles. Se puede crear aplicaciones móviles para iOS y Android desde un solo código base, lo que ahorra tiempo y recursos. Además, proporciona un conjunto de widgets personalizables y herramientas que te permiten diseñar interfaces de usuario atractivas y receptivas. Permite ver los cambios en tiempo real sin reiniciar la aplicación y cuenta con una comunidad activa y

una amplia gama de herramientas de desarrollo, lo que facilita el proceso de desarrollo y brinda soporte adicional.

- BackenEnd: Para el desarrollo del backend, se ha decidido utilizar Java como lenguaje de programación debido a su robustez y amplia comunidad de desarrolladores. Como base de datos, se ha seleccionado PostgreSQL por su confiabilidad y capacidad para gestionar grandes volúmenes de datos. Para mejorar la modularidad y escalabilidad del sistema, se ha dividido el backend en tres microservicios principales: el servicio de autenticación y gestión de usuarios (ms_autenticacion), el servicio de gestión y procesamiento de productos y pedidos (ms_core) y el servicio de procesamiento de pagos. Cada microservicio se encarga de un dominio específico, lo que permite un desarrollo ágil y facilita la escalabilidad y el mantenimiento del sistema en el futuro. Además, se utilizan estándares de arquitectura y buenas prácticas de diseño para garantizar un código limpio y modular.

- Servicios adicionales:

Se utilizan servicios de AWS (Amazon Web Services) debido a sus numerosas ventajas y beneficios para el desarrollo y despliegue de aplicaciones. AWS ofrece una amplia gama de servicios en la nube que permiten a las empresas escalar sus aplicaciones de manera flexible, mejorar la seguridad, reducir costos y aumentar la eficiencia operativa. Se pago solo por lo que se use y disminuye tiempos de desarrollo de tareas complejas, a continuación se mencionan y explican cada uno de los servicios utilizados en la arquitectura y cómo es su interacción con el sistema

i) Amazon ElastiCache (Redis): La utilización de redis del lado del Front End, permite almacenar los recursos estáticos en el dispositivo del usuario, como imágenes, archivos CSS y JavaScript, evita tener que solicitar constantemente esos recursos al servidor. Esto reduce la carga en el servidor y mejora su rendimiento al liberarlo de la tarea de servir repetidamente esos recursos. Mejora la velocidad de carga al cargar los

recursos desde la caché del cliente, ahorra ancho de banda al no tener que solicitar los recursos estáticos al servidor en cada visita, lo que resultase especialmente beneficioso en conexiones de red limitadas como lo es el caso del campo colombiano y proporciona una mejor disponibilidad y experiencia del usuario en áreas con conectividad limitada o en situaciones donde la señal de red es intermitente.

ii) En la arquitectura de MECAD, se utiliza un API Gateway para proporcionar una capa de abstracción y gestión de los servicios y APIs expuestos por el sistema. El API Gateway actúa como punto de entrada único para las solicitudes de los usuarios, dirigiéndolas a los diferentes servicios correspondientes en función de la ruta y los parámetros de la solicitud. El API Gateway ofrece varias ventajas en la arquitectura de MECAD permite definir reglas de enrutamiento para dirigir las solicitudes de los usuarios a los servicios correspondientes. Esto simplifica la lógica de enrutamiento en los servicios individuales y centraliza la gestión de las solicitudes entrantes, también actúa como un punto de control centralizado para la seguridad y la autorización, brinda una verificación de permisos y protección contra ataques de seguridad y permite controlar y regular el tráfico entrante, evitando la sobrecarga en los servicios subyacentes.

iii) Elastic Load Balancer (ELB): Se puede observar en la arquitectura con el nombre “Balanceador de carga”. Se utilizó un balanceador de carga en la arquitectura de MECAD para el servicio de gestión y procesamiento de pedidos y productos (ms_core) porque proporciona varios beneficios clave. En primer lugar, el balanceador de carga permite distribuir de manera equitativa el tráfico entrante entre múltiples instancias del servicio ms_core. Esto asegura una distribución balanceada de la carga de trabajo, evitando que una sola instancia se sobrecargue y garantizando un rendimiento óptimo del sistema. Además, el balanceador de carga mejora la disponibilidad del servicio. En caso de que una instancia de

ms_core falle o se vuelva inaccesible, el balanceador de carga redirigirá automáticamente el tráfico a instancias saludables y operativas. Esto garantiza que el servicio de gestión y procesamiento de pedidos y productos siga estando disponible para los usuarios, incluso en situaciones de fallos o interrupciones. Otro aspecto importante es la escalabilidad del sistema, el balanceador de carga de MECAD permite escalar automáticamente el número de instancias de ms_core en función de la carga de trabajo. Si la demanda aumenta, el balanceador de carga puede agregar instancias adicionales para gestionar el aumento de tráfico y mantener un rendimiento óptimo. Del mismo modo, si la carga disminuye, el balanceador de carga puede reducir el número de instancias, lo que ahorra recursos y costos.

iv) KMS: En MECAD, se utiliza AWS Key Management Service (KMS) para la protección de datos sensibles, como contraseñas y datos bancarios, durante el proceso de inscripción. KMS es un servicio de administración de claves que facilita la creación y el control de claves de cifrado. Proporciona una capa adicional de seguridad al cifrar y descifrar los datos utilizando claves seguras y gestionadas de manera centralizada. Cuando un usuario ingresa sus datos personales durante el proceso de inscripción, MECAD utiliza KMS para cifrar esos datos antes de almacenarlos en la base de datos. Esto garantiza que los datos estén protegidos tanto en reposo como en tránsito. Además, KMS permite controlar y auditar el acceso a las claves de cifrado. Solo los usuarios autorizados tienen los permisos necesarios para utilizar las claves y descifrar los datos protegidos. Esto garantiza la confidencialidad y la integridad de los datos almacenados en MECAD. El uso de KMS en MECAD para la inscripción de datos personales ofrece una capa adicional de seguridad y cumple con los estándares y regulaciones de protección de datos. Los usuarios pueden tener la tranquilidad de que sus contraseñas y datos bancarios están protegidos de forma segura durante todo el proceso de inscripción.

v) Amazon MQ: En MECAD, se utiliza una cola Amazon MQ con RabbitMQ entre el API Gateway y el servicio de pagos con el propósito de garantizar que no se pierda ningún pago. La cola Amazon MQ actúa como un intermediario entre el API Gateway y el servicio de pagos, permitiendo una comunicación asíncrona y confiable. Cuando un usuario realiza un pago a través del API Gateway, en lugar de procesar el pago de inmediato, se encola en la cola Amazon MQ. Esta cola actúa como un buffer temporal, almacenando los pagos hasta que el servicio de pagos esté disponible para procesarlos. Esto es especialmente útil en situaciones en las que el servicio de pagos puede experimentar picos de carga o tiempos de inactividad planificados. Al utilizar RabbitMQ como el motor de mensajería en la cola Amazon MQ, se asegura una entrega confiable de los mensajes en el orden correcto. RabbitMQ sigue un enfoque de "primero en entrar, primero en salir", lo que significa que los pagos se procesan en el orden en que llegaron a la cola, evitando la pérdida de pagos o la desorganización en la secuencia de transacciones. Además, la cola Amazon MQ ofrece características de durabilidad y persistencia, lo que garantiza que los pagos almacenados en la cola no se pierdan en caso de fallos o reinicios del sistema. Esto brinda una capa adicional de seguridad y confiabilidad en el procesamiento de pagos.

vi) RDS: En MECAD, se utiliza Amazon RDS (Relational Database Service) para alojar las bases de datos de PostgreSQL y SQL Server lo que garantiza una solución administrada y facilidad de configuración, seguridad, escalabilidad y disponibilidad de la información.

vii) Pintpoint: En MECAD, se utiliza Amazon Pinpoint para el envío de notificaciones recepción de mensajes de actualización de pedidos. En MECAD, se aprovecha esta capacidad para mantener a los usuarios informados sobre actualizaciones importantes relacionadas con sus pedidos. Al utilizar Amazon Pinpoint, MECAD puede enviar notificaciones a los usuarios a través del canal preferido de cada uno, ya sea WhatsApp

o SMS en sus dispositivos móviles. Estas notificaciones pueden contener información sobre el estado del pedido, cambios en la entrega, confirmación de pago u otras actualizaciones relevantes. Además, Amazon Pinpoint permite recibir mensajes de actualización de pedidos de los proveedores. Esto significa que los usuarios pueden enviar consultas o solicitar información adicional sobre sus pedidos utilizando los canales de comunicación admitidos por Pinpoint. Permitiendo así la información en tiempo real sobre cualquier cambio en sus pedidos.

viii) Lambda: En MECAD, se utiliza una función Lambda para procesar los mensajes de actualización de pedidos que provienen de Amazon Pinpoint. Esta función está escrita en Python y se encarga de realizar las actualizaciones correspondientes en la base de datos PostgreSQL. Esta solución evita la necesidad de crear un microservicio específico para este propósito, ya que la función Lambda se encarga de ejecutar el código necesario de manera eficiente y escalable. Cuando se recibe un mensaje de actualización de pedido a través de Amazon Pinpoint, este mensaje se envía a la función Lambda asociada. La función Lambda, escrita en Python, procesa el mensaje y extrae la información relevante, como el identificador del pedido y los detalles de la actualización. A continuación, se establece una conexión con la base de datos PostgreSQL y se realiza la actualización correspondiente en la tabla de producto.

5.5.2. Estilos arquitectónicos, tácticas y otros aspectos técnicos

Tabla 10

Estilos arquitectónicos, tácticas y otros aspectos técnicos

Arquetipo	APP Móvil

Estilo/patrón arquitectónico y descripción	Puerta de enlace API (API Gateway): Se utiliza como el único punto de entrada para todos los clientes. Esta táctica se alinea bien con el patrón arquitectónico de microservicios. La puerta de enlace API maneja las solicitudes de dos maneras: enrutando algunas solicitudes al servicio correspondiente y desplegando otras solicitudes en múltiples servicios. Esto permite la flexibilidad de agregar nuevos servicios en el futuro sin afectar el funcionamiento del software, lo que mejora significativamente su mantenibilidad. Autenticación sin estado (Stateless authentication): Se utiliza JSON Web Token (JWT) para almacenar los datos de sesión del usuario en el lado del cliente (navegador). Los datos están firmados por la clave del proveedor de identidad (IdP) para garantizar la integridad y autoridad de los datos de sesión. Esta táctica proporciona seguridad tanto en la aplicación como en la integración con terceros, ya que no se depende del estado del servidor para autenticar y autorizar las solicitudes. Backend for Frontend (BFF): Se utiliza para mejorar la mantenibilidad y escalabilidad del sistema. Esta táctica implica tener un backend específico para el frontend, lo que permite un desarrollo más rápido y escalable al estar separado. Proporciona ventajas significativas en términos de mantenimiento, ya que se pueden realizar cambios y mejoras en el frontend sin afectar el backend y viceversa Material Design: Se utiliza este estilo de diseño que se enfoca en los aspectos visuales correspondientes al sistema operativo Android. El uso de Material Design mejora la usabilidad del sistema al proporcionar una interfaz consistente y familiar para los usuarios. Además, también se puede aplicar a diferentes páginas web y plataformas, lo que brinda coherencia en la experiencia del usuario.
Lenguajes programación	Los microservicios iniciales del lado del servidor (backend), se desarrollan con java y la autenticación con C#, lenguajes compilados de amplia comunidad y orientado a objetos con el que se puede hacer código robusto. Se utilizan bases de datos estructuradas y robustas como Postgres SQL y SQL server. Del lado del cliente, se desarrollará la interfaz móvil con Flutter y leguaje de programación DART y hojas de estilo en cascada.
Frameworks	Front-end: Flutter, dirigido específicamente a plataformas de desarrollo móvil, mayor facilidad para desarrollar aplicaciones móviles multiplataforma para iOS y Android. Backend: Sprint boot, un framework de java que maneja las configuraciones automáticamente con su función de configuración automática. Dado que el tiempo de

	desarrollo es menor, la productividad aumenta. Muy utilizado para servicios de backend

Nota: En esta tabla se relacionan los estilos arquitectónicos, tácticas y otros aspectos técnicos implementados en la aplicación MECAD. Elaboración propia.

La aplicación de estos patrones de arquitectura en MECAD, como la puerta de enlace API, la autenticación sin estado, el Backend for Frontend y el uso de Material Design, contribuyen a satisfacer los atributos de calidad requeridos, como la mantenibilidad, la seguridad, la escalabilidad y la usabilidad del sistema.

5.5.3. Tácticas de Despliegue

El diseño de despliegue de la aplicación en los mercados campesinos digitales (MECAD) se basó en la utilización de contenedores Docker y la orquestación de los mismos mediante Kubernetes, aprovechando la solidez de los servicios en la nube de AWS (Amazon Web Services). Esta combinación de tecnologías proporciona una arquitectura robusta y escalable para las aplicaciones en el contexto de los mercados campesinos digitales.

Los contenedores Docker permiten empacar la aplicación de manera consistente y portátil, lo cual resulta fundamental en un entorno dinámico como MECAD. Esto significa que la aplicación y todas sus dependencias se encapsulan en un contenedor, lo que garantiza su funcionamiento sin importar el entorno en el que se despliegue. Asimismo, la utilización de contenedores facilita la portabilidad y el despliegue consistente en diferentes entornos, ya sean locales o en la nube.

Por otro lado, la orquestación de los contenedores mediante Kubernetes proporciona una forma eficiente de gestionar y escalar la aplicación en un entorno distribuido. Kubernetes se encarga de distribuir y balancear la carga de trabajo entre los contenedores, garantizando una alta disponibilidad y escalabilidad automática según la demanda del sistema. Esto resulta esencial en MECAD, donde la capacidad de respuesta y el rendimiento del sistema son aspectos críticos.

En cuanto a los servicios de AWS, como ECS (Elastic Container Service) y EKS (Elastic Kubernetes Service), brindan una infraestructura confiable y escalable para

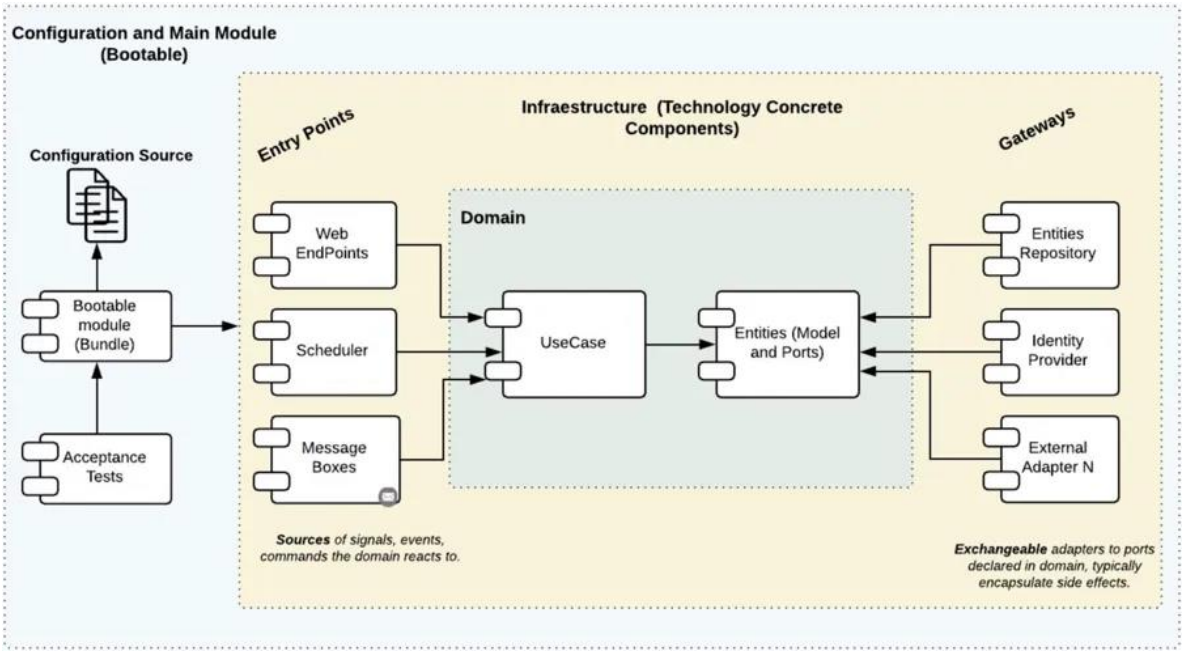
el despliegue, gestión y escalado de los contenedores. Estos servicios de AWS ofrecen características adicionales, como monitoreo, seguridad y automatización, que potencian la funcionalidad y escalabilidad del sistema en los mercados campesinos digitales. Al aprovechar los servicios de AWS, se obtiene una solución integral y de confianza para la implementación eficiente de las aplicaciones en la nube.

5.5.4. Arquitectura limpia

En esta aplicación, se sigue el principio de la Arquitectura Limpia (Clean Architecture) para el desarrollo del sistema. Se busca tener un código modular, fácil de mantener, y con una alta cohesión y bajo acoplamiento. Se utiliza la Arquitectura en Capas (Layered Architecture) como base para la organización del sistema. Esta arquitectura, también conocida como Arquitectura en Cebolla (Onion Architecture), se basa en la idea de organizar el código en capas concéntricas, donde cada capa tiene una responsabilidad clara y está protegida por las capas externas.

Figura 9

Diagrama de referencia de arquitectura limpia

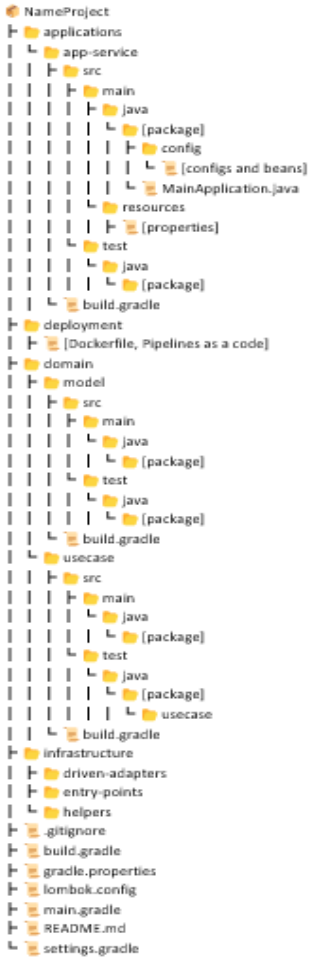


Nota: Esta gráfica el modelo de referencia de la arquitectura clean architecture. Fuente, Clean Architecture — Aislando los detalles | by Andrés Mauricio Gómez P | Bancolombia Tech | Medium

En el desarrollo de los microservicios en Java del lado del backend en MECAD, se utiliza un plugin de código abierto desarrollado por Bancolombia Tech llamado "Clean Architecture Plugin". Este plugin facilita el empaquetamiento y la adopción del principio de Arquitectura Limpia (Clean Architecture) en los microservicios, simplifica la implementación de la Arquitectura Limpia al proporcionar plantillas, configuraciones y herramientas que ayudan a los desarrolladores a seguir las mejores prácticas. Con este plugin, los microservicios pueden estructurarse fácilmente en capas como Entidades, Casos de Uso, Adaptadores de Controladores y Adaptadores de Infraestructura, de acuerdo con los principios de la Arquitectura Limpia. El uso de este plugin facilita el desarrollo y mantenimiento de los microservicios en Java, ya que promueve una estructura clara y modular, facilita la prueba unitaria y permite una evolución controlada del sistema. Además, al ser un proyecto de código abierto, brinda la oportunidad de contribuir y beneficiarse de las mejoras continuas realizadas por la comunidad.

Figura 10

Capas de la arquitectura



La imagen muestra la utilización de tres capas en la Arquitectura Limpia. A continuación, se describen brevemente cada una de ellas:

- Capa de Aplicación: En esta capa se encuentran los casos de uso o reglas de negocio de la aplicación. Aquí se definen las operaciones y lógica de negocio que el sistema realiza. La capa de aplicación se encarga de orquestar la interacción entre las capas de presentación y de infraestructura. Su objetivo principal es coordinar y dirigir el flujo de datos y las operaciones entre las diferentes capas del sistema.

- Capa de Dominio: Esta capa representa el núcleo de la Arquitectura Limpia y contiene las entidades y reglas de negocio principales del sistema. Aquí se definen los modelos de dominio y se implementa la lógica de negocio más crítica y específica de la aplicación. La capa de dominio es independiente de

las capas externas y no tiene dependencias hacia ellas, lo que la hace más flexible y fácil de mantener

- Capa de Infraestructura: En esta capa se maneja la comunicación con componentes externos, como bases de datos, servicios web, APIs, entre otros. También se encarga de la persistencia de datos y la integración con otros sistemas. Aquí se implementan adaptadores que permiten la comunicación entre la capa de dominio y los componentes externos. La capa de infraestructura proporciona los mecanismos necesarios para interactuar con el mundo exterior y gestionar los aspectos técnicos de la aplicación. Se divide en Entry Points, que son Gateway de entrada de la información y Driven Adapters que son Gateway de persistencia y salida de la información

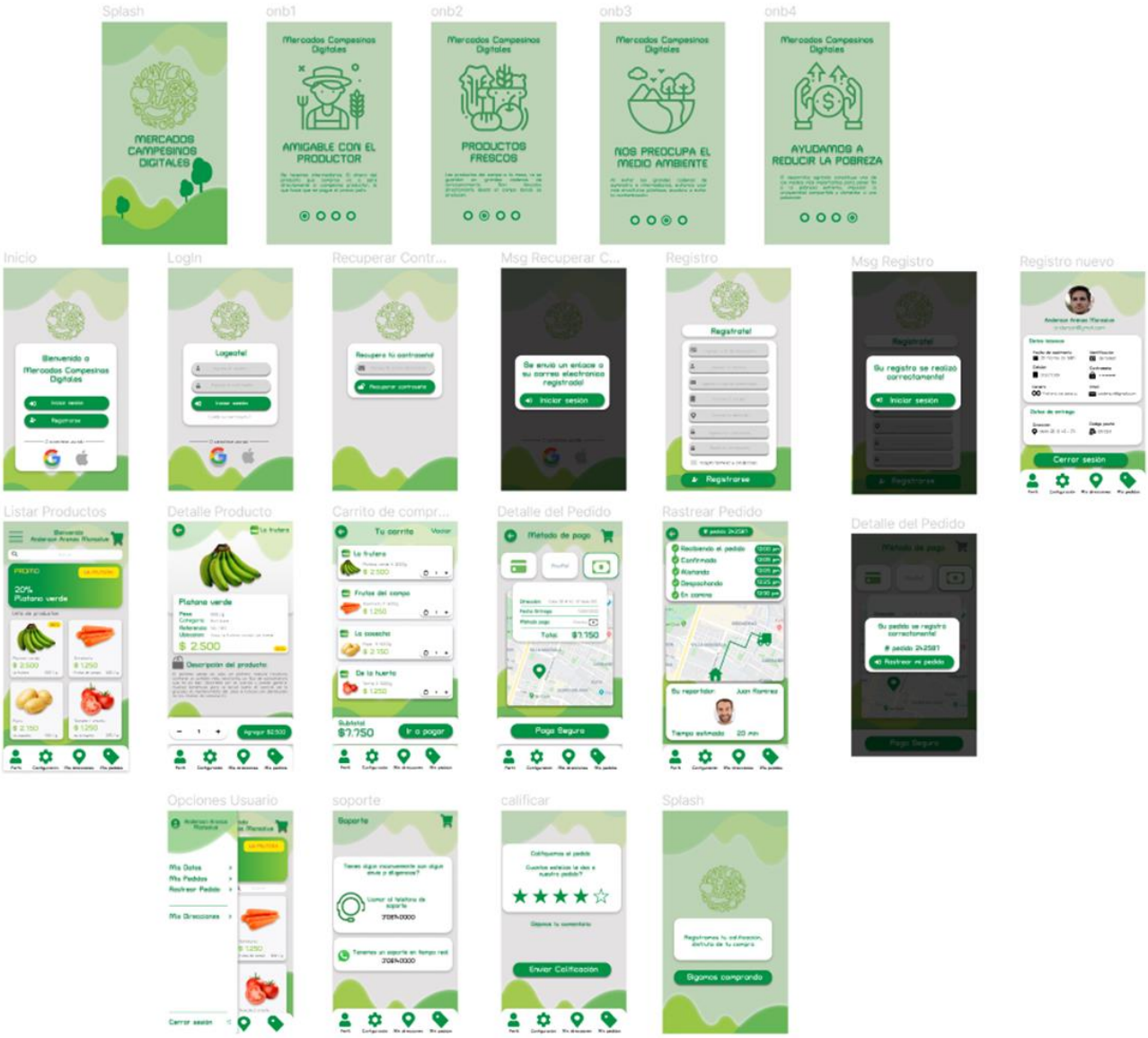
5.6. Diseño de interfaces

El diseño de interfaces de la aplicación es un aspecto fundamental para brindar una buena experiencia de usuario. En este caso, se ha optado por utilizar Material Design, un lenguaje de diseño creado por Google que se enfoca en la simplicidad, la claridad y la coherencia en la presentación de elementos visuales en la interfaz. Este enfoque permite que el usuario tenga una navegación intuitiva y agradable a través de la aplicación, lo que se traduce en una mayor satisfacción y fidelidad del usuario. Además, el uso de Material Design asegura una estética moderna y consistente en toda la aplicación.

A continuación, se muestra el diseño de pantallas de la aplicación:

Figura 11

Interfaz gráfica de la aplicación



Nota: La grafica son representaciones reales de la interfaz de la aplicación. Fuente, elaboración propia.

5.7. Conclusiones del capítulo

La metodología de desarrollo de la aplicación MECAD se ha enfocado en diversos aspectos clave para garantizar los atributos de calidad, como la mantenibilidad,

seguridad, usabilidad e interoperabilidad. A continuación, se presenta una conclusión sobre cada uno de estos aspectos abordados en el capítulo:

Elicitación de requisitos funcionales y no funcionales: La elicitación adecuada de requisitos funcionales y no funcionales es fundamental para comprender las necesidades y expectativas de los usuarios, así como los criterios de calidad del sistema. En el desarrollo de MECAD, se ha realizado un esfuerzo por identificar y documentar estos requisitos, asegurando que la aplicación cumpla con los objetivos y funcionalidades requeridas, y que también satisfaga aspectos como el rendimiento, seguridad y usabilidad.

Diseño de bases de datos: El diseño de bases de datos en MECAD ha sido cuidadosamente planificado para asegurar una estructura eficiente y escalable. Esto implica la definición de las entidades y relaciones necesarias para almacenar y acceder a los datos de manera óptima. Un diseño adecuado de la base de datos facilita la gestión de la información y contribuye a la mantenibilidad del sistema a largo plazo.

Arquitectura general y arquitectura limpia: La arquitectura general de MECAD se ha diseñado considerando principios como la modularidad, separación de responsabilidades y escalabilidad. Se ha adoptado la arquitectura limpia, que promueve la independencia de las capas y la modularidad, facilitando así la evolución y mantenibilidad del sistema. Esta arquitectura permite realizar cambios y mejoras en componentes específicos sin afectar al resto del sistema.

Utilización de servicios de la nube AWS: MECAD ha aprovechado los servicios de la nube de AWS para lograr una infraestructura escalable, segura y confiable. La utilización de servicios ha permitido desplegar y gestionar los recursos de manera eficiente, evitando la necesidad de administrar una infraestructura física. Esto mejora la escalabilidad y facilita la implementación de prácticas de seguridad recomendadas.

Tácticas de despliegue: Para garantizar la disponibilidad y continuidad del servicio, se han utilizado tácticas de despliegue como la implementación de balanceadores de carga, alta disponibilidad y autoscaling. Estas tácticas permiten gestionar el tráfico, mantener la estabilidad del sistema y responder a cambios en la demanda de manera automática.

Mockups para atributos de calidad: Los mockups se han utilizado para asegurar atributos de calidad como la usabilidad e interoperabilidad. Mediante los mockups, se ha podido visualizar y validar la interfaz de usuario antes de su implementación, garantizando una experiencia intuitiva y amigable para los usuarios.

En conclusión, la metodología de desarrollo de la aplicación MECAD ha abarcado diversos aspectos que garantizan atributos de calidad como la mantenibilidad, seguridad, usabilidad e interoperabilidad. Desde la elicitación de requisitos hasta el diseño de bases de datos, la arquitectura general, la utilización de servicios en la nube AWS, las tácticas de despliegue y el uso de mockups, se han considerado cuidadosamente para asegurar un sistema robusto y confiable.

6. Capítulo V: Implementación de MECAD, aplicación móvil potenciada por tecnologías de vanguardia

Utilizar herramientas tecnológicas que permitan una interfaz de usuario atractiva, intuitiva y fácil de usar, siguiendo buenas prácticas de programación para desarrollar una aplicación móvil que proporcione una experiencia de usuario satisfactoria.

6.1. Marco metodología de desarrollo

Para el desarrollo de la aplicación móvil MECAD, se seguirá una metodología ágil basada en el marco Scrum. Esto permitirá un enfoque colaborativo, iterativo e incremental para el desarrollo del proyecto. Se formará un equipo multidisciplinario que incluirá desarrolladores, diseñadores, especialistas en calidad y otros roles necesarios. El proyecto se dividirá en sprints, cada uno con una duración fija de dos semanas, en los cuales se planificarán y ejecutarán las tareas prioritarias. Se realizarán reuniones diarias de seguimiento (daily scrum) para compartir avances, identificar posibles obstáculos y establecer acciones correctivas. Al final de cada sprint, se realizará una revisión para demostrar las funcionalidades implementadas y recibir retroalimentación de los stakeholders. También se llevará a cabo una retrospectiva para identificar áreas de mejora y ajustar el proceso de desarrollo. La metodología Scrum permitirá una entrega temprana de valor, una mayor flexibilidad para adaptarse a los cambios y una mayor implicación de los interesados a lo largo del proyecto.

Además, se utilizarán herramientas de gestión de proyectos ágiles, como Jira o Trello, para realizar el seguimiento de las tareas, asignar responsabilidades y mantener un registro transparente del progreso del proyecto. Estas herramientas facilitarán la comunicación y colaboración entre los miembros del equipo, permitiendo una mejor coordinación y garantizando que todas las tareas se realicen dentro de los plazos establecidos. Asimismo, se establecerán reuniones periódicas de revisión y planificación, donde se revisarán y priorizarán las funcionalidades

pendientes, se asignarán recursos y se establecerán objetivos claros para el siguiente sprint. Con esta metodología ágil, se promoverá la eficiencia, la calidad y la entrega oportuna del proyecto de desarrollo de la aplicación móvil MECAD.

6.2. Tecnologías y Herramientas

En el desarrollo de la aplicación móvil MECAD, se utilizarán diversas tecnologías y herramientas de vanguardia para garantizar un resultado de alta calidad y una experiencia de usuario excepcional. En primer lugar, se empleará el framework de desarrollo multiplataforma Flutter, respaldado por Google, que permite crear aplicaciones nativas para iOS y Android con un solo código base. Flutter ofrece ventajas significativas, como un rendimiento rápido, una interfaz de usuario atractiva y una amplia gama de widgets personalizables.

Para el desarrollo del backend, se utilizará el framework Spring Boot, basado en Java, que facilita la creación de API RESTful y la gestión de la lógica de negocio. Spring Boot ofrece una arquitectura escalable y modular, así como una amplia gama de bibliotecas y herramientas para acelerar el desarrollo y garantizar la seguridad de la aplicación.

Además, se hará uso de una base de datos relacional como PostgreSQL para el almacenamiento eficiente y seguro de los datos. Estas bases de datos ofrecen una sólida integridad de los datos y soportan consultas complejas y transacciones robustas.

Para la autenticación y autorización de usuarios, se implementará un sistema de seguridad mediante JSON Web Tokens (JWT), que proporciona un mecanismo seguro para la autenticación basada en tokens. También se emplearán técnicas de encriptación para proteger los datos sensibles almacenados en la base de datos y durante las comunicaciones.

Además de estas tecnologías principales, se utilizarán otras herramientas y bibliotecas complementarias, como Postman para probar y documentar las API, Docker para la creación y gestión de contenedores, y Git como sistema de control de versiones para facilitar la colaboración y el seguimiento de los cambios en el código fuente.

En resumen, la combinación de tecnologías y herramientas como Flutter, Spring Boot, bases de datos relacionales, JWT, Postman, Docker y Git permitirá desarrollar una aplicación móvil MECAD robusta, segura y altamente funcional. Estas tecnologías ofrecen un amplio soporte y una comunidad activa, lo que garantiza la disponibilidad de recursos y soluciones técnicas en caso de surgir desafíos o actualizaciones futuras.

6.3. Buenas prácticas de programación

En el desarrollo de la aplicación móvil MECAD, se seguirán rigurosamente las mejores prácticas de programación con el objetivo de garantizar un código limpio, eficiente y mantenible. A continuación, se detallan algunas de las prácticas que se aplicarán:

a) Utilización de patrones de diseño: Se implementarán patrones de diseño reconocidos, como el patrón Modelo-Vista-Controlador (MVC) o el patrón Singleton, para mejorar la estructura y modularidad del código.

b) Separación de responsabilidades: Se dividirá el código en módulos o componentes independientes, cada uno encargado de una funcionalidad específica. Esto permitirá una mejor organización del código y facilitará su comprensión y mantenimiento.

c) Documentación del código: Se incluirán comentarios claros y concisos en el código fuente para explicar su funcionamiento, facilitar su comprensión y agilizar futuras modificaciones o actualizaciones.

d) Uso de convenciones de nomenclatura: Se seguirán convenciones de nomenclatura adecuadas para nombrar variables, funciones y clases de manera coherente y comprensible.

e) Pruebas unitarias: Se implementarán pruebas unitarias utilizando frameworks como JUnit para verificar el correcto funcionamiento de los diferentes componentes y minimizar la aparición de errores.

f) Control de versiones: Se utilizará un sistema de control de versiones, como Git, para gestionar el código fuente y permitir la colaboración eficiente entre los miembros del equipo de desarrollo.

g) Optimización de rendimiento: Se optimizará el rendimiento de la aplicación a través de técnicas como el uso eficiente de recursos, la minimización de llamadas a bases de datos y la optimización de algoritmos.

h) Seguridad: Se implementarán medidas de seguridad, como la validación de datos de entrada, el manejo adecuado de contraseñas y la protección contra ataques comunes, para garantizar la integridad y confidencialidad de la información.

i) Mantenimiento y escalabilidad: Se diseñará el código pensando en su mantenibilidad a largo plazo y en la posibilidad de escalabilidad futura. Se buscará evitar la duplicación de código, utilizar estructuras modulares y aplicar principios como el principio SOLID.

En resumen, el desarrollo de la aplicación móvil MECAD se regirá por las mejores prácticas de programación, asegurando un código limpio, estructurado y fácilmente mantenible. Estas prácticas permitirán un desarrollo más eficiente, minimizarán la aparición de errores y facilitarán futuras actualizaciones y mejoras en la aplicación.

6.4. Experiencia de Usuario

En el desarrollo de la aplicación móvil MECAD, se priorizará brindar una experiencia de usuario excepcional, garantizando que cada interacción sea fluida, intuitiva y satisfactoria. A continuación, se detallan las acciones que se tomarán para lograrlo:

a) Diseño centrado en el usuario: Se realizará una investigación exhaustiva de los usuarios objetivo de la aplicación, comprendiendo sus necesidades, preferencias y comportamientos. Con esta información, se diseñará la interfaz de usuario de manera que sea fácil de entender y utilizar, adaptada a las expectativas y habilidades de los usuarios.

b) Prototipado y pruebas de usabilidad: Se realizarán prototipos interactivos de la aplicación, permitiendo a los usuarios potenciales experimentar y brindar retroalimentación sobre la usabilidad, el flujo de navegación y la disposición de los elementos. Se realizarán pruebas de usabilidad iterativas para identificar posibles obstáculos y áreas de mejora, garantizando así una experiencia fluida y satisfactoria.

c) Diseño responsivo y multiplataforma: La aplicación móvil MECAD se desarrollará para funcionar de manera óptima en diferentes dispositivos y tamaños de pantalla, ya sean teléfonos móviles o tablets. Se asegurará que los elementos de la interfaz se adapten de forma adecuada a las distintas resoluciones y orientaciones, brindando una experiencia coherente y agradable en todas las plataformas.

d) Optimización de rendimiento: Se prestará especial atención al rendimiento de la aplicación, asegurándose de que las transiciones sean suaves y rápidas, y que los tiempos de carga sean mínimos. Se optimizará el uso de los recursos del dispositivo y se realizarán pruebas exhaustivas para garantizar que la aplicación sea altamente receptiva y eficiente.

e) Personalización y preferencias del usuario: Se brindará la opción de personalización dentro de la aplicación, permitiendo a los usuarios ajustar ciertos aspectos de la interfaz, como el tema de color, la configuración de notificaciones o la elección del idioma. Esto permitirá adaptar la aplicación a las preferencias individuales de cada usuario, aumentando su satisfacción y comodidad al utilizarla.

f) Retroalimentación y soporte: Se proporcionará un canal de retroalimentación y soporte al usuario, como un formulario de contacto o un sistema de chat en vivo. Esto permitirá a los usuarios plantear dudas, reportar problemas y recibir asistencia en tiempo real, mejorando su experiencia y asegurando una atención al cliente eficiente.

g) Accesibilidad: La aplicación se diseñará y desarrollará siguiendo pautas de accesibilidad, asegurando que todas las personas, incluyendo aquellas con discapacidades visuales o motoras, puedan utilizarla sin dificultades. Se implementarán características como compatibilidad con lectores de pantalla, opciones de alto contraste y ajustes de tamaño de fuente, entre otros.

En conclusión, la experiencia de usuario en la aplicación móvil MECAD es una prioridad durante todo el proceso de desarrollo. Se siguen buenas prácticas de diseño centrado en el usuario, se realizan pruebas de usabilidad y se optimiza el rendimiento para ofrecer una experiencia fluida y satisfactoria.

6.5. Diseño de la interfaz de usuario:

El diseño de la interfaz de usuario de la aplicación móvil MECAD será uno de los aspectos clave para brindar una experiencia de usuario satisfactoria y atractiva. Se seguirán los principios y pautas de Material Design, un lenguaje de diseño desarrollado por Google que ofrece una estética moderna, consistente y fácil de usar en diferentes dispositivos y plataformas.

El diseño se centrará en crear una interfaz intuitiva y de fácil navegación, con una disposición lógica de los elementos y una jerarquía visual clara. Se utilizarán colores y tipografías adecuados para transmitir la identidad visual de la marca y resaltar la información relevante. Además, se prestará especial atención a la usabilidad, asegurando que los elementos interactivos sean accesibles y se adapten correctamente a diferentes tamaños de pantalla y orientaciones.

Se incorporarán animaciones sutiles y transiciones fluidas para mejorar la sensación de fluidez y respuesta de la aplicación. Esto contribuirá a una experiencia de usuario más agradable y envolvente.

Además, se implementarán elementos de personalización, como la posibilidad de ajustar preferencias de idioma, temas de color o configuraciones de notificaciones, para adaptarse a las necesidades y preferencias individuales de cada usuario.

Para el diseño de la interfaz, se utilizarán herramientas de diseño gráfico y prototipado como Adobe XD o Sketch, que permiten crear y visualizar los diseños de manera efectiva. Esto facilitará la iteración y colaboración entre el equipo de diseño y desarrollo, permitiendo una implementación precisa y coherente del diseño final.

En conclusión, el diseño de la interfaz de usuario en la aplicación móvil MECAD se basará en los principios de Material Design, buscando lograr una experiencia visualmente atractiva, intuitiva y adaptable a diferentes dispositivos. La atención meticulosa a los detalles y la colaboración estrecha entre los equipos de diseño y desarrollo garantizarán que la interfaz cumpla con los estándares de calidad y brinde una experiencia de usuario excepcional.

6.6. Identificación de la infraestructura tecnológica

La infraestructura tecnológica necesaria para el proyecto incluye un servidor web y de base de datos para alojar la aplicación y su contenido. Además, se requiere un servicio de alojamiento en la nube para garantizar la escalabilidad y disponibilidad de la aplicación. Se necesita también un sistema de monitoreo de rendimiento y seguridad para garantizar la estabilidad y la protección de los datos. Por último, se necesita hardware y software para el desarrollo y la prueba de la aplicación.

6.7. Selección y Adquisición de Herramientas y Tecnología

Para este garantizar el desarrollo del proyecto, se realizó una evaluación de las herramientas y la tecnología más adecuadas a usar. En este proceso se realizó la evaluación de algunas herramientas disponibles en el mercado y se seleccionó las que se consideran que más se adaptan a las necesidades del proyecto. Los factores a considerar fueron los siguientes:

- Costos
- Facilidad de uso
- Capacidad de integración
- Disponibilidad de soporte

En ese sentido, se realizó un análisis detallado en el cual se define la adquisición de las siguientes herramientas:

- ✓ Microsoft Visual Studio Code
- ✓ Microsoft Visual Studio 2022 Community
- ✓ PgAdmin
- ✓ IntelliJ
- ✓ Microsoft Project
- ✓ Adobe Creative Cloud

6.8. Selección y Adquisición de Entorno de Desarrollo

Visual Studio: es un IDE creado por Microsoft y es ampliamente utilizado para el desarrollo de aplicaciones .NET. Soporta múltiples lenguajes de programación, incluyendo C#, F# y Visual Basic.

Android Studio: es un IDE utilizado para el desarrollo de aplicaciones para Android. Está basado en el popular IDE IntelliJ IDEA y ofrece soporte para múltiples lenguajes de programación.

InteliJ: es un IDE utilizado principalmente para la programación en Java, aunque también soporta otros lenguajes de programación como PHP, C++ y HTML5.

6.9. Código limpio y Principios SOLID

En la implementación de la aplicación MECAD se aplicaron varios principios de código limpio, entre ellos:

- Nombres descriptivos: Se utilizaron nombres claros y descriptivos para las variables, métodos y clases, de modo que su función y propósito sean fácilmente comprensibles.
- Funciones y métodos pequeños: Se dividió la lógica en funciones y métodos más pequeños y cohesivos, lo que facilita su comprensión, reutilización y prueba.
- Evitar la duplicación de código: Se evitó la repetición de código mediante la creación de funciones y componentes reutilizables. Esto mejora la eficiencia y facilita las actualizaciones y correcciones
- Mantener la simplicidad: Se priorizó la simplicidad en el diseño y la implementación del código, evitando la complejidad innecesaria. Esto facilita su comprensión y mantenimiento a largo plazo.

Al aplicar estos principios de código limpio, se logra un código más legible, mantenible y escalable. Esto facilita la colaboración entre desarrolladores, mejora la calidad del software y permite un proceso de desarrollo más eficiente.

También se aplicaron los principios SOLID, que son un conjunto de principios de diseño de software que promueven el modularidad, la flexibilidad y la mantenibilidad del código. Estos principios se centran en la creación de componentes de software que sean independientes, reutilizables y fáciles de mantener. Algunos de los principios SOLID que se aplicaron son:

Tabla 11

Principios SOLID

Principios SOLID y su aplicación a MECAD	
Single Responsibility	Cada microservicio está desarrollado en n capas y cada capa tiene una única responsabilidad dentro del dominio, como por ejemplo: la capa de acceso a datos, capa de negocio y capa de servicio. Se puede observar en el código este principio.
Open/Closed	Se aplica en la mayoría de las clases en el software, por ejemplo: se tiene una clase llamada cliente que hereda de usuario y una clase llamada productor que también hereda de usuario, así en un futuro se pueden crear nuevos perfiles de usuario sin necesidad de modificar la clase usuario.
Liskov Substitution	Para acceder a las funcionalidades implementadas en cada capa se hace a través de la interface principal y no accediendo directamente a la clase donde se implementó. Ejemplo para acceder a la interface de usuario tipo cliente se debe acceder a la de usuario como clase principal.
Interface Segregation	No existen métodos innecesarios en las clases, es decir, no se implementan métodos en la clase padre cuando éste solo lo requiere la clase hija.
Dependency Inversion	Toda la comunicación entre clases y funcionalidades se hace a través de interfaces. Por ejemplo en código se puede observar que ProductRepository es una interfaz que extiende de la clase CrudRepository, los cuál nos provee todos los métodos necesarios para hacer un CRUD.

Nota: En esta tabla se relacionan los principios SOLID utilizados en la aplicación.

6.10. Desarrollo BackEnd

En el alcance del proyecto, se estableció el desarrollo de dos módulos funcionales: ms_autenticación desarrollada en C# y ms_core desarrollado en java. Estos módulos desempeñan roles específicos en el sistema MECAD.

El módulo ms_autenticación se encarga de gestionar la autenticación y autorización de los usuarios en el sistema. Proporciona funcionalidades relacionadas con el inicio

de sesión, registro de usuarios, gestión de sesiones y control de acceso a los recursos del sistema. Este módulo garantiza la seguridad y privacidad de los usuarios al verificar sus credenciales y autorizar sus acciones dentro de la plataforma.

Por otro lado, el módulo ms_core se enfoca en la gestión y procesamiento de productos y pedidos dentro de MECAD. Aquí se implementan las funcionalidades centrales del sistema, como la creación y gestión de productos, la gestión de inventario, la administración de pedidos y la generación de facturas. Este módulo se encarga de toda la lógica de negocio relacionada con la oferta de productos y la gestión de transacciones.

El desarrollo de estos dos módulos funcionales permite cubrir aspectos fundamentales del sistema MECAD, asegurando la autenticación segura de los usuarios y brindando una sólida base para la gestión y procesamiento de productos y pedidos. Estos módulos pueden interactuar entre sí y con otros componentes del sistema para ofrecer una experiencia completa y funcional a los usuarios.

El backend de la aplicación móvil MECAD es una parte fundamental del sistema, encargada de gestionar y procesar los datos, así como de brindar los servicios necesarios para que la aplicación funcione correctamente. A continuación, se detalla el desarrollo y funcionamiento del backend.

1. Arquitectura: El backend se desarrollará siguiendo una arquitectura basada en microservicios, lo que permite una mayor modularidad, escalabilidad y flexibilidad. Los microservicios se encargarán de manejar diferentes funcionalidades del sistema, como la autenticación de usuarios, el procesamiento de órdenes de compra, la gestión de productos, entre otros.

2. Framework: Se utilizará el framework de desarrollo Spring Boot, que proporciona una estructura sólida y eficiente para la creación de aplicaciones Java. Spring Boot simplifica la configuración y el desarrollo, facilitando la creación de APIs RESTful y la integración con bases de datos y otras tecnologías.

3. Base de datos: Se empleará una base de datos relacional, como PostgreSQL, para almacenar y gestionar los datos del sistema. Se diseñará y optimizará la estructura de la base de datos de acuerdo con los requisitos del proyecto, asegurando un almacenamiento eficiente y seguro de la información.

4. API RESTful: El backend expondrá una API RESTful que permitirá la comunicación entre la aplicación móvil y el servidor. Se definirán los endpoints necesarios para realizar operaciones como registro de usuarios, consulta de productos, creación de órdenes de compra, entre otros. Se seguirán las mejores prácticas para el diseño de APIs, como el uso de verbos HTTP adecuados y la implementación de autenticación y autorización.

5. Autenticación y seguridad: Se implementará un sistema de autenticación seguro para proteger los datos y las acciones de los usuarios. Se utilizarán técnicas de cifrado y almacenamiento seguro de contraseñas, así como tokens de acceso para validar la identidad de los usuarios en cada solicitud a la API. Además, se aplicarán medidas de seguridad adicionales, como el control de acceso a los recursos y la protección contra ataques de seguridad conocidos.

6. Integración de servicios externos: En caso de ser necesario, se integrarán servicios externos, como pasarelas de pago, proveedores de servicios de envío o sistemas de notificación, para enriquecer la funcionalidad de la aplicación. Se establecerán las conexiones y se realizarán las configuraciones correspondientes para asegurar una comunicación adecuada y segura entre el backend y estos servicios externos.

7. Pruebas unitarias y de integración: Se realizarán pruebas exhaustivas en el backend, tanto a nivel unitario como de integración, para garantizar su correcto funcionamiento y validar la respuesta de la API ante diferentes escenarios. Se utilizarán herramientas como JUnit y Mockito para la creación y ejecución de pruebas, asegurando la calidad y confiabilidad del código desarrollado.

8. Despliegue y mantenimiento: El backend se desplegará en un entorno de producción, ya sea en servidores propios o en la nube, siguiendo las mejores

prácticas de despliegue y configuración. Se establecerá un plan de mantenimiento y monitoreo continuo, con el fin de asegurar la disponibilidad y el rendimiento del sistema, así como de aplicar actualizaciones y correcciones necesarias.

En resumen, el desarrollo del backend de la aplicación móvil MECAD se llevará a cabo siguiendo una arquitectura basada en microservicios, utilizando el framework Spring Boot y una base de datos relacional. Se impSlementará una API RESTful segura, se integrarán servicios externos y se realizarán pruebas exhaustivas para garantizar su correcto funcionamiento. Además, se llevará a cabo un despliegue y mantenimiento adecuado para asegurar la disponibilidad y calidad del sistema.

6.11. Desarrollo FrontEnd

El frontend de la aplicación móvil MECAD es la interfaz con la que los usuarios interactúan, por lo que es crucial ofrecer una experiencia de usuario fluida, atractiva y de fácil uso. A continuación, se detalla el desarrollo y funcionamiento del frontend:

1. Diseño y estructura: El frontend se diseñará siguiendo los principios de Material Design, que proporciona una apariencia moderna y coherente, con elementos visuales intuitivos y consistentes en toda la aplicación. Se creará una estructura de navegación clara y jerárquica, facilitando la exploración de las diferentes secciones y funcionalidades.

2. Framework: Para el desarrollo del frontend, se utilizará el framework Flutter. Flutter permite crear interfaces de usuario multiplataforma de alta calidad, con un rendimiento rápido y una apariencia nativa en dispositivos iOS y Android. Además, Flutter ofrece un conjunto de widgets personalizables y flexibles que permiten crear una interfaz visualmente atractiva.

3. Diseño responsivo: Se asegurará que la aplicación se adapte de forma adecuada a diferentes tamaños de pantalla y orientaciones, brindando una experiencia de usuario consistente en dispositivos móviles y tabletas. Se utilizarán técnicas de

diseño responsivo para ajustar la disposición de los elementos y garantizar una visualización óptima en diferentes dispositivos.

4. Interacción y flujo de la aplicación: Se prestará especial atención al flujo de la aplicación, asegurando una navegación intuitiva y coherente. Se implementarán transiciones y animaciones sutiles para mejorar la experiencia de usuario y proporcionar retroalimentación visual durante las interacciones.

5. Manejo de estados: Se utilizarán técnicas y patrones de manejo de estados eficientes, como el uso de Provider o Bloc, para mantener una gestión adecuada de los datos y proporcionar actualizaciones en tiempo real en la interfaz de usuario. Esto permitirá una respuesta rápida a las interacciones del usuario y una experiencia fluida.

6. Integración de APIs: El frontend se conectará con el backend a través de APIs para obtener y enviar datos. Se implementará la lógica necesaria para realizar las solicitudes HTTP correspondientes, manejar las respuestas y mostrar los datos en la interfaz de usuario de manera adecuada. Además, se aplicarán técnicas de caché y optimización de la carga de datos para mejorar el rendimiento de la aplicación.

7. Pruebas de interfaz de usuario: Se realizarán pruebas exhaustivas en el frontend para asegurar su correcto funcionamiento en diferentes escenarios y dispositivos. Se utilizarán herramientas como Flutter Test para realizar pruebas unitarias y de widget, así como pruebas de integración para verificar la interacción adecuada con el backend.

8. Optimización de rendimiento: Se optimizará el rendimiento del frontend mediante técnicas como el código de carga diferida (code splitting), el uso de caché local para almacenar datos frecuentemente accedidos y la optimización de imágenes y recursos para una carga más rápida. Esto permitirá una experiencia de usuario ágil y sin demoras.

9. Adaptabilidad a cambios: Se desarrollará el frontend con un enfoque modular y escalable, de modo que sea fácil de mantener y adaptar en el futuro. Se seguirán

las mejores prácticas de desarrollo, como la separación de responsabilidades y el uso de patrones de diseño adecuados, para facilitar la incorporación de nuevas funcionalidades y mejoras.

En resumen, el frontend de la aplicación móvil MECAD se desarrollará utilizando el framework Flutter, siguiendo los principios de Material Design y garantizando una experiencia de usuario intuitiva y atractiva. Se prestará atención a la estructura, diseño responsivo, flujo de la aplicación, manejo de estados, integración de APIs, pruebas de interfaz de usuario, rendimiento y adaptabilidad a cambios, con el objetivo de ofrecer una aplicación móvil de calidad y satisfactoria para los usuarios.

6.12. Pruebas y control de calidad

En esta sección, se llevó a cabo un riguroso proceso de pruebas y control de calidad para garantizar el correcto funcionamiento y rendimiento de la aplicación móvil MECAD. Se diseñó una estrategia de pruebas integral que abarcó diferentes niveles y aspectos de la aplicación.

Se realizaron pruebas unitarias exhaustivas en el backend para verificar la funcionalidad y la lógica de negocio de cada uno de los componentes. Esto incluyó la validación de la correcta interacción con la base de datos, la gestión adecuada de las solicitudes y respuestas, así como la validación de los resultados esperados.

Además, se llevaron a cabo pruebas de integración para evaluar la interacción entre los diferentes módulos del sistema y garantizar la correcta comunicación y flujo de datos. Se verificó que todos los componentes funcionaran de manera integrada y que los datos se transfirieran correctamente entre ellos.

Para evaluar el rendimiento de la aplicación, se realizaron pruebas de carga y estrés, simulando un alto volumen de usuarios y transacciones. Se monitorearon los tiempos de respuesta, la capacidad de procesamiento y la escalabilidad del sistema para identificar posibles cuellos de botella y realizar las optimizaciones necesarias.

Asimismo, se realizaron pruebas de seguridad para detectar vulnerabilidades y asegurar la protección de los datos de los usuarios. Se llevaron a cabo pruebas de penetración para identificar posibles puntos débiles y se implementaron medidas de seguridad adicionales, como la autenticación y autorización robustas, la encriptación de datos y la protección contra ataques comunes.

Todas las pruebas se llevaron a cabo siguiendo metodologías y estándares reconocidos, y se documentaron detalladamente los casos de prueba, los resultados obtenidos y las acciones correctivas tomadas. Se realizaron múltiples ciclos de pruebas y se involucró al equipo de desarrollo y a usuarios beta para obtener retroalimentación adicional y realizar ajustes en base a sus comentarios.

El resultado de este exhaustivo proceso de pruebas y control de calidad fue una aplicación móvil MECAD robusta, confiable y de alto rendimiento. Se logró identificar y solucionar rápidamente los problemas detectados, asegurando una experiencia de usuario satisfactoria y la correcta funcionalidad de la aplicación en diferentes escenarios y dispositivos.

6.13. Optimización del rendimiento

Durante el desarrollo de la aplicación móvil MECAD, se prestó especial atención a la optimización del rendimiento para garantizar una experiencia de usuario fluida y sin interrupciones. Se implementaron diversas estrategias y técnicas para maximizar la velocidad de carga, la capacidad de respuesta y la eficiencia del sistema.

En primer lugar, se llevó a cabo una cuidadosa optimización del código fuente, siguiendo las mejores prácticas de programación. Se realizaron revisiones exhaustivas del código para identificar y corregir posibles problemas de rendimiento, como bucles innecesarios, consultas de base de datos ineficientes o uso excesivo de recursos.

Además, se aplicaron técnicas de caching y almacenamiento en memoria para acelerar el acceso a los datos y minimizar las consultas a la base de datos. Se

implementaron estrategias de cacheo tanto en el backend como en el frontend, utilizando herramientas como Redis y local storage, para almacenar temporalmente datos frecuentemente utilizados y evitar consultas adicionales.

Otra medida importante para mejorar el rendimiento fue la optimización de las imágenes y recursos multimedia. Se utilizó compresión y técnicas de carga progresiva para reducir el tamaño de las imágenes sin comprometer la calidad visual. Además, se minimizaron las solicitudes de recursos externos y se implementaron técnicas de precarga y precarga selectiva para acelerar la carga de la aplicación.

La implementación de técnicas de almacenamiento en caché a nivel de servidor, utilizando herramientas como Varnish, permitió reducir la carga en los servidores y acelerar la entrega de contenido estático a los usuarios. Esto resultó en tiempos de carga más rápidos y una mayor capacidad para manejar un mayor número de usuarios simultáneos.

Por último, se realizaron pruebas exhaustivas de rendimiento y se utilizaron herramientas de monitoreo para identificar cuellos de botella y áreas de mejora. Se llevaron a cabo pruebas de carga simulando diferentes escenarios de uso y se analizaron los resultados para optimizar el rendimiento en función de los picos de demanda.

En resumen, la optimización del rendimiento fue un aspecto crucial en el desarrollo de la aplicación móvil MECAD. Mediante la implementación de técnicas de programación eficientes, estrategias de caching, optimización de recursos multimedia y pruebas exhaustivas de rendimiento, se logró ofrecer una aplicación rápida, receptiva y altamente eficiente, brindando a los usuarios una experiencia de uso óptima y satisfactoria.

6.14. Integración de servicios externos

En el desarrollo de la aplicación móvil MECAD, se hizo hincapié en la integración de servicios externos para enriquecer la funcionalidad y proporcionar una experiencia más completa a los usuarios. Se identificaron varios servicios relevantes que podrían complementar la aplicación y mejorar su utilidad.

En primer lugar, se implementó la integración con servicios de geolocalización, como Google Maps, para permitir a los usuarios encontrar fácilmente los puntos de venta y los productos cercanos a su ubicación. Esto se logró mediante la utilización de API y SDK proporcionados por los proveedores de servicios de geolocalización, lo que permitió obtener datos precisos de ubicación y mostrarlos de forma intuitiva en el mapa de la aplicación.

Además, se estableció la integración con plataformas de pago en línea, como PayPal, para brindar a los usuarios la posibilidad de realizar transacciones seguras y convenientes directamente desde la aplicación. Se implementaron las API y los mecanismos de autenticación necesarios para permitir el procesamiento de pagos de manera segura y confiable.

Asimismo, se integraron servicios de notificaciones push para mantener a los usuarios informados sobre promociones, actualizaciones de productos o cualquier otra información relevante. Esto se logró mediante la configuración de servicios de notificaciones push proporcionados por plataformas como Firebase, lo que permitió enviar notificaciones instantáneas a los dispositivos móviles de los usuarios.

Otro aspecto importante de la integración de servicios externos fue la conexión con APIs de redes sociales, como Facebook o Apple, para facilitar el inicio de sesión con cuentas existentes y permitir compartir contenido de la aplicación en las redes sociales. Se implementaron los mecanismos de autorización y autenticación necesarios para garantizar la seguridad de los datos del usuario y cumplir con los requisitos de privacidad.

Además, se consideró la integración con servicios de análisis y seguimiento de usuarios, como Google Analytics, para recopilar datos sobre el uso de la aplicación

y obtener información valiosa para tomar decisiones basadas en datos. Se implementaron las configuraciones necesarias para recopilar y analizar métricas relevantes, como la cantidad de usuarios, el tiempo de permanencia en la aplicación y los patrones de uso.

En resumen, la integración de servicios externos fue un componente clave en el desarrollo de la aplicación móvil MECAD. Mediante la implementación de integraciones con servicios de geolocalización, plataformas de pago, notificaciones push, redes sociales y análisis de usuarios, se logró enriquecer la funcionalidad de la aplicación y ofrecer a los usuarios una experiencia más completa y personalizada.

6.15. Implementación de seguridad

La seguridad es un aspecto fundamental en el desarrollo de la aplicación móvil MECAD. Se implementaron diversas medidas y prácticas de seguridad para proteger la información de los usuarios y garantizar la confidencialidad, integridad y disponibilidad de los datos.

En primer lugar, se estableció un sistema de autenticación robusto y seguro. Se implementó un mecanismo de inicio de sesión que requería credenciales válidas, como nombre de usuario y contraseña, para acceder a la aplicación. Se utilizó el algoritmo de hash para almacenar las contraseñas de forma segura en la base de datos, evitando almacenarlas en texto plano.

Además, se implementaron controles de acceso para asegurar que los usuarios solo tuvieran acceso a las funciones y datos correspondientes a su rol y nivel de autorización. Esto se logró mediante la asignación de roles y permisos a cada usuario, lo que garantizaba que solo pudieran realizar acciones permitidas y acceder a la información relevante para su perfil.

Para proteger la comunicación entre la aplicación y los servidores, se implementó el protocolo HTTPS, que encripta los datos transmitidos y asegura que no sean

interceptados ni alterados por terceros. Se obtuvo un certificado SSL para garantizar la autenticidad del servidor y establecer una conexión segura.

Además, se realizaron pruebas exhaustivas de seguridad para identificar posibles vulnerabilidades y brechas en la aplicación. Se emplearon herramientas de prueba de penetración para simular ataques y evaluar la resistencia del sistema. Las vulnerabilidades encontradas fueron corregidas y se aplicaron parches de seguridad de forma regular para mantener la aplicación protegida contra nuevas amenazas.

Por último, se implementó un sistema de copias de seguridad periódicas para asegurar la disponibilidad de los datos en caso de fallos o incidentes. Se estableció un plan de respaldo y recuperación de datos que garantizaba la protección de la información crítica de los usuarios y la capacidad de restaurar los datos en caso de emergencia.

En resumen, la implementación de medidas de seguridad fue una prioridad en el desarrollo de la aplicación móvil MECAD. A través de un sistema de autenticación seguro, controles de acceso, comunicación encriptada, pruebas de seguridad y copias de seguridad regulares, se logró proteger la información de los usuarios y mantener la integridad y confidencialidad de los datos en todo momento.

6.16. Evaluación y mejora continua

La evaluación y mejora continua son elementos fundamentales en el desarrollo de la aplicación móvil MECAD. Durante todo el proceso, se realizaron evaluaciones periódicas para identificar posibles áreas de mejora y realizar ajustes necesarios.

Se estableció un sistema de retroalimentación y seguimiento que permitió recopilar comentarios y sugerencias de los usuarios, tanto internos como externos. Estos comentarios se tuvieron en cuenta para identificar mejoras específicas en la funcionalidad, la usabilidad y el rendimiento de la aplicación.

Además, se realizaron pruebas de aceptación con grupos de usuarios seleccionados para evaluar la experiencia general y recopilar información valiosa sobre posibles problemas o dificultades que los usuarios puedan enfrentar al utilizar la aplicación. Estas pruebas permitieron identificar áreas que requerían ajustes o mejoras para garantizar una experiencia de usuario óptima.

La evaluación continua también incluyó un análisis exhaustivo de las métricas y datos de rendimiento de la aplicación. Se monitorearon aspectos como el tiempo de carga, el tiempo de respuesta, la tasa de errores y el uso de recursos para identificar posibles cuellos de botella y áreas de mejora en el rendimiento. Estos datos se utilizaron como base para la toma de decisiones y la implementación de mejoras en el código y la infraestructura de la aplicación.

Basándose en los resultados de las evaluaciones y las pruebas realizadas, se implementaron mejoras y ajustes de manera continua a lo largo del desarrollo del proyecto. Se priorizaron las áreas críticas y se asignaron recursos adecuados para abordar los problemas identificados y optimizar la funcionalidad y el rendimiento de la aplicación.

En conclusión, la evaluación y mejora continua fueron pilares clave en el desarrollo de la aplicación móvil MECAD. A través de pruebas de aceptación, análisis de métricas de rendimiento y la recopilación de comentarios de los usuarios, se logró identificar oportunidades de mejora y realizar ajustes necesarios para ofrecer una aplicación de alta calidad y satisfacer las necesidades de los usuarios de manera efectiva.

6.17. Conclusiones del capítulo

Además de los puntos mencionados anteriormente, es importante destacar que el desarrollo de este objetivo específico también se benefició de la colaboración y el

trabajo en equipo. Se fomentó la comunicación abierta y constante entre los miembros del equipo de desarrollo, lo que permitió una mejor comprensión de los requisitos del proyecto y la identificación de soluciones efectivas.

Asimismo, se promovió la adopción de metodologías ágiles, como Scrum, que facilitaron la entrega iterativa de funcionalidades y la rápida adaptación a los cambios y requerimientos del proyecto. Esto permitió una mayor flexibilidad y agilidad en el proceso de desarrollo, asegurando la entrega oportuna de resultados de alta calidad.

Adicionalmente, se implementaron prácticas de gestión de configuración y control de versiones para garantizar la integridad y trazabilidad del código fuente. Esto facilitó la colaboración entre los desarrolladores y evitó conflictos y problemas durante el proceso de desarrollo.

Para desarrollo del backend y frontend de la aplicación móvil MECAD ha sido un proceso integral y detallado que ha buscado cumplir con los objetivos planteados en términos de funcionalidad, usabilidad y calidad.

En cuanto al backend, se ha diseñado una arquitectura basada en microservicios que permite una mayor modularidad, escalabilidad y mantenibilidad del sistema. Se han utilizado tecnologías como Spring Boot y una base de datos relacional para garantizar un desarrollo sólido y eficiente. Además, se han implementado buenas prácticas de programación, como la separación de responsabilidades y la optimización del rendimiento, para asegurar la calidad del código y el funcionamiento óptimo del backend.

Por otro lado, en el frontend se ha utilizado el framework Flutter, aprovechando sus capacidades de desarrollo multiplataforma y su capacidad para crear interfaces de usuario atractivas y responsivas. Se ha seguido el diseño basado en Material Design, asegurando una experiencia de usuario intuitiva y coherente. Se ha prestado especial atención al diseño de la interfaz, la interacción y el flujo de la aplicación, buscando proporcionar una experiencia de usuario satisfactoria y fácil de usar.

Ambas partes, el backend y el frontend, han sido desarrolladas siguiendo un enfoque centrado en la calidad, llevando a cabo pruebas exhaustivas y aplicando buenas prácticas de desarrollo. Se ha trabajado en conjunto con los requerimientos y especificaciones del proyecto, teniendo en cuenta las necesidades de los usuarios y las metas establecidas.

Por último, cabe destacar que se promovió una cultura de mejora continua y aprendizaje dentro del equipo de desarrollo. Se alentó a los miembros del equipo a participar en cursos de formación y capacitación para mantenerse actualizados con las últimas tecnologías y mejores prácticas de desarrollo de software. Esto contribuyó a enriquecer las habilidades técnicas del equipo y a impulsar la calidad y la innovación en el desarrollo de la aplicación.

7. Capítulo VI: Plan de Aseguramiento de la Calidad

El propósito de este plan de aseguramiento de calidad es establecer los lineamientos y las actividades necesarias para garantizar la calidad en el desarrollo de la app tipo e-commerce para la venta directa de productos agrícolas software “MECAD”. El objetivo principal es asegurar que la plataforma cumpla con los estándares y requisitos de calidad establecidos, proporcionando una experiencia segura, eficiente y confiable tanto para los productores como para los consumidores finales. Mediante la implementación de prácticas y procesos de aseguramiento de calidad adecuados, se busca minimizar los riesgos, asegurar la funcionalidad adecuada de la aplicación y promover la satisfacción de los usuarios.

El alcance de este plan es identificar riegos asociados al producto de desarrollo y al proceso de la venta de productos agrícolas como tal y documentar una serie de actividades en forma de checklist, para garantizar la calidad; Abarca todas las fases del proyecto de desarrollo de la app tipo e-commerce, desde la concepción y diseño hasta la implementación y puesta en marcha.

Para realizarlo, se basa en las normas:

- Norma IEEE 730: Estándar para el Plan de Aseguramiento de la Calidad del Software.
- Norma ISO/IEC 25010: Sistemas y software de ingeniería de calidad - Modelos de calidad del producto.
- Norma ISO/IEC 12207: Procesos del ciclo de vida del software.

- Norma ISO/IEC 9126: Evaluación de la calidad del producto de software.
- Agile Manifesto: Principios y valores ágiles para el desarrollo de software.
- Guía de Verificación y Validación del Software del IEEE (IEEE 1012): Estándar para la verificación y validación del software.
- Guía de Gestión de Riesgos del Software del IEEE (IEEE 1540): Estándar para la gestión de riesgos en proyectos de software.

7.1. Identificación de riesgos

La identificación de riesgos en un plan de aseguramiento de calidad para MECAD es de vital importancia. Permite mitigar los posibles riesgos que podrían afectar la calidad de la aplicación, tomando medidas preventivas para minimizarlos. Además, la identificación de riesgos ayuda a realizar una planificación adecuada de los recursos y las actividades necesarias para asegurar la calidad. Conociendo los posibles riesgos, se pueden asignar los recursos adecuados, establecer plazos y prioridades, y definir estrategias específicas para abordar cada riesgo identificado.

Identificar riesgos también implica establecer un proceso de mejora continua en términos de calidad. Al tener conocimiento de los posibles riesgos, se pueden implementar acciones correctivas y preventivas que ayuden a evitar problemas futuros y optimizar los procesos de desarrollo y entrega de la aplicación. Esto contribuye a la satisfacción del cliente, ya que al anticiparse a los posibles riesgos y asegurar la calidad de la aplicación, se garantiza una experiencia satisfactoria para los usuarios finales.

7.1.1. Riesgos del producto de software

Los riesgos del producto de software se refieren a los posibles problemas o amenazas que pueden afectar directamente al producto de software en sí a la APP MECAD en todo su ciclo de vida. Estos riesgos están relacionados con la calidad, funcionalidad, seguridad, usabilidad y rendimiento del producto. Por ejemplo, un riesgo del producto de software podría ser la presencia de errores o fallas en la aplicación que afecten su funcionamiento o la seguridad de los datos de los diferentes tipos de usuarios como contraseñas para ingresar a la aplicación o datos bancarios que se utilizan a la hora de hacer el pago. Estos riesgos pueden comprometer la satisfacción del cliente, la reputación de la empresa y la eficacia del producto en el mercado.

Tabla 12

Identificación de Riesgos del Producto.

Riesgo	Causa	Impacto	Prioridad
Riesgo de seguridad	Falta de medidas de seguridad adecuadas	Alto: Pérdida de datos, daño a la reputación.	Alta
Riesgo de rendimiento	Ineficiencias en el diseño y la implementación	Moderado: Mala experiencia del usuario	Alta
Riesgo de escalabilidad	Incapacidad para manejar un crecimiento rápido	Alto: Degradación del rendimiento, fallas	Alta
Riesgo de usabilidad	Diseño deficiente e interacción confusa.	Moderado: Mala experiencia del usuario.	Media
Riesgo de calidad del código.	Malas prácticas de desarrollo y mantenimiento.	Moderado: Dificultad de mantenimiento	Media
Riesgo de integración	Falta de compatibilidad con sistemas externos.	Moderado: Funcionalidad limitada o defectuosa.	Media
Riesgo de documentación inadecuada.	Falta de documentación clara y actualizada	Moderado: Dificultades en el mantenimiento.	Baja
Riesgo de gestión de cambios ineficiente	Inadecuada gestión y control de cambios	Moderado: Problemas de alineación y comunicación	Baja

Nota: En esta tabla se relacionan los riesgos identificados para la aplicación. Fuente, elaboración propia.

7.1.2. Riesgos del proceso

Por otro lado, los riesgos del proceso se centran en los posibles problemas o amenazas que pueden surgir durante el desarrollo del software y que están estrechamente ligados con el nicho de ventas digitales de productos agrícolas o al nicho de desarrollo de software. Estos riesgos están relacionados con la gestión del proyecto, la planificación, la comunicación, el control de calidad y otros aspectos del proceso de desarrollo. Por ejemplo, un riesgo del proceso podría ser la falta de recursos adecuados para completar el proyecto a tiempo o la falta de comunicación efectiva entre los miembros del equipo. Estos riesgos pueden afectar la eficiencia y el éxito general del proyecto de desarrollo de software.

Tabla 13

Riesgos del proceso

Riesgo	Causa	Impacto	Prioridad
Riesgo de planificación deficiente	Estimaciones inexactas, falta de experiencia en planificación	Alto: Desviación de plazos, retrasos en el proyecto	Alta
Riesgo de falta de seguimiento y control	Falta de herramientas adecuadas para seguimiento y control	Moderado: Pérdida de visibilidad, desviaciones	Alta
Riesgo de falta de competencias y habilidades	Falta de capacitación o experiencia en el equipo de desarrollo	Moderado: Calidad deficiente, errores	Media
Riesgo de mala gestión de recursos	Asignación inadecuada de recursos humanos y tecnológicos	Moderado: Ineficiencia, sobrecarga	Media
Riesgo de falta de mejora continua	Falta de retroalimentación y mejora en los procesos de desarrollo	Bajo: Estancamiento, falta de eficiencia	Baja
Riesgo de mala logística de distribución	Falta de coordinación en la entrega de productos	Alto: Retrasos en las entregas, insatisfacción del cliente	Alta
Riesgo de falta de conocimiento del mercado	Desconocimiento de las necesidades y preferencias de los clientes	Moderado: Producto no adecuado, baja demanda.	Media

Nota: En esta tabla se relacionan los riesgos identificados para los procesos del proyecto. Fuente, elaboración propia.

7.2. Actividades, resultados y tareas

7.2.1. Garantías de la aplicación

A continuación, se listan por cada uno de los riesgos identificados para el producto de software, actividades y tareas que ayudan a mitigar y reducir la probabilidad y/o impacto del riesgo:

i) Riesgo de seguridad:

- ✓ Utilizar prácticas de diseño seguro para desarrollar una arquitectura de software resistente a ataques y vulnerabilidades.

- ✓ Realizar pruebas de seguridad exhaustivas para identificar vulnerabilidades y debilidades en el software.
- ✓ Incluir pruebas de penetración, pruebas de inyección de código, pruebas de seguridad de la capa de aplicación, entre otras.
- ✓ Incluir el proyecto en actividades de ataque continuo para identificar nuevas vulnerabilidades del sistema.
- ✓ Aplicar buenas prácticas de codificación segura, como la validación de entrada, el manejo seguro de datos confidenciales y la prevención de inyecciones de código malicioso.
- ✓ Utilizar sonarQube para identificar posibles vulnerabilidades en el código.
- ✓ Implementar un modelo de control de acceso adecuado que limite los privilegios de los usuarios y garantice la seguridad de los datos.
- ✓ Aplicar principios de mínimo privilegio y separación de funciones para evitar el acceso no autorizado.
- ✓ Implementar mecanismos de protección de datos, como cifrado de datos en reposo y en tránsito, para garantizar la confidencialidad y la integridad de la información.
- ✓ Establecer un proceso para aplicar actualizaciones y parches de seguridad de forma regular y oportuna.
- ✓ Mantener el software actualizado con las últimas correcciones de seguridad.
- ✓ Proporcionar capacitación continua (cada 5 meses) en seguridad a los miembros del equipo de desarrollo para promover una cultura de seguridad y conciencia de los riesgos.
- ✓ Realizar auditorías periódicas de seguridad para evaluar y verificar el cumplimiento de los controles de seguridad establecidos.

ii) Riesgo de rendimiento y escalabilidad:

- ✓ Implementar arquitecturas limpias y enfoques como DDD para separar el dominio de la tecnología.
- ✓ Identificar los requisitos de rendimiento del sistema, como tiempos de respuesta, capacidad de carga y escalabilidad.
- ✓ Utilizar herramientas de análisis de rendimiento, como Apache JMeter, para simular cargas de trabajo y evaluar el rendimiento del sistema.
- ✓ Utilizar patrones de diseño y técnicas de optimización de código para mejorar el rendimiento del software.
- ✓ Utilizar índices eficientes y consultas optimizadas para mejorar el rendimiento de las operaciones de base de datos identificarlos por medio de Workbench.
- ✓ Utilizar herramientas de orquestación de contenedores, como Kubernetes, para facilitar la escalabilidad horizontal y la distribución eficiente de los componentes del sistema.
- ✓ Monitoreo del estado del servicio en tiempo real por medio de prometheus y grafana.

iii) Riesgo de usabilidad:

- ✓ Realizar entrevistas y encuestas a los usuarios potenciales para comprender sus necesidades, expectativas y preferencias.
- ✓ Hacer un prototipo con figma para recibir retroalimentación antes del desarrollo.
- ✓ Realizar evaluaciones de accesibilidad para garantizar que el producto cumpla con las pautas de accesibilidad, como las establecidas por el W3C.

- ✓ Diseñar y desarrollar el producto para que sea compatible con diferentes dispositivos y tamaños de pantalla.
- ✓ Utilizar herramientas de diseño responsivo.
- ✓ Utilizar herramientas de gestión de proyectos, como Trello para registrar y dar seguimiento a los cambios y mejoras realizadas.
- ✓ Realizar pruebas de aceptación de usuario para validar que el producto es pertinente.
- ✓ Utilizar TestRail para planificar, ejecutar y documentar las pruebas de aceptación de usuario.

iv) Riesgo de calidad del código:

- ✓ Establecer estándares de codificación como clean code y principios SOLID.
- ✓ Realizar revisiones de código estático utilizando SonarQube para identificar problemas de calidad y buenas prácticas de codificación.
- ✓ Implementar pipelines de despliegue automático y establecer ambientes productivos y preproductivos para desarrollo y pruebas (DEV, QA y PDN).
- ✓ Desarrollar pruebas unitarias exhaustivas con una cobertura de mínimo el 60% del código antes de subir un reléase a QA.
- ✓ Utilizar sistemas de control de versiones Git para gestionar el código fuente de manera eficiente y asegurarse de que se sigan las prácticas adecuadas de control de cambios.
- ✓ Utilizar metodología de trunk-base para git con el fin de no combinar errores en la rama principal y subirlos eventualmente a producción.

- ✓ Hacer uso de los pull request para complementar la tarea anterior pull request que además debe ser aprobado por un miembro diferente del equipo de desarrollo.

v) Riesgo de integración:

- ✓ Diseñar y ejecutar pruebas de integración exhaustivas para validar la interoperabilidad de los sistemas.
- ✓ Utilizar estándares como REST, SOAP o MQP para la comunicación entre sistemas.
- ✓ Desarrollar servicios y adaptadores que faciliten la comunicación y la integración con los sistemas externos.
- ✓ Diseñar y ejecutar pruebas de integración exhaustivas para validar la interoperabilidad de los sistemas.
- ✓ Utilizar herramientas de automatización de pruebas como SoapUI, Postman o JMeter para realizar pruebas de integración de manera eficiente.

vi) Riesgo de documentación inadecuada:

- ✓ Establecer estándares claros y consistentes para la documentación del proyecto, incluyendo formatos, estructuras y convenciones.
- ✓ Utilizar plantillas y guías para facilitar la creación de documentos coherentes y completos.
- ✓ Capturar y documentar de manera exhaustiva los requerimientos del sistema, incluyendo funcionalidades, restricciones y criterios de aceptación.
- ✓ Utilizar Trello para mantener un registro centralizado de los requerimientos.
- ✓ Detallar el diseño y la arquitectura del sistema, incluyendo diagramas UML, descripciones de componentes y relaciones entre ellos por medio de lucidchart para diagramar.
- ✓ Centralizar documentación en confluence.

vii) Riesgo de gestión de cambios ineficiente:

- ✓ Definir un proceso claro y estructurado para solicitar, evaluar, aprobar y realizar cambios en el software.
- ✓ Establecer roles y responsabilidades claras en el proceso de gestión de cambios.
- ✓ Establecer un proceso de revisión y aprobación de cambios por parte de un equipo técnico o comité designado.

7.2.2. Garantías del proceso

i) Riesgo de planificación deficiente:

- ✓ Utilizar la metodología SMART (Específico, Medible, Alcanzable, Realista, Tiempo) para definir los objetivos.
- ✓ Desarrollar un plan de proyecto detallado que incluya actividades, plazos, recursos asignados y responsabilidades.
- ✓ Utilizar Microsoft Project crear y gestionar el plan de gestión del proyecto.
- ✓ Establecer un proceso para monitorear el progreso del proyecto y realizar ajustes cuando sea necesario.
- ✓ Utilizar herramientas de colaboración en línea como Microsoft Teams para realizar las reuniones.
- ✓ Realizar análisis de riesgos y planificación de contingencias
- ✓ Elaborar matrices de riesgos.

ii) Riesgo de falta de seguimiento y control:

- ✓ Se implementa la metodología scrum como principal actividad de mitigación de este riesgo.

- ✓ Definir métricas y criterios de éxito para evaluar el progreso del proyecto.
- ✓ Realizar auditorías periódicas para evaluar el cumplimiento de los estándares de calidad y las mejores prácticas.
- ✓ Utilizar Microsoft Power BI, Tableau o Excel para visualizar y analizar los datos.

iii) Riesgo de falta de competencias y habilidades:

- ✓ Realizar un análisis de las competencias y habilidades necesarias para el proyecto.
- ✓ Determinar las brechas existentes en el equipo y las áreas en las que se requiere mejorar.
- ✓ Establecer un plan de capacitación para el equipo, identificando los temas relevantes para el proyecto.
- ✓ Asignar recursos y tiempo para el desarrollo de habilidades específicas.
- ✓ Facilitar el acceso a recursos de aprendizaje, como cursos en línea, tutoriales, documentación y libros relacionados con las competencias necesarias.
- ✓ Utilizar plataformas de aprendizaje en línea, como Udemy, Coursera o LinkedIn Learning, para ofrecer cursos específicos.
- ✓ Realizar evaluaciones regulares del desempeño del equipo para identificar áreas de mejora y oportunidades de desarrollo.
- ✓ Establecer programas de incorporación y formación para los nuevos miembros del equipo.

iv) Riesgo de falta de mejora continua:

- ✓ Seguir adecuadamente la metodología scrum que promueve la mejora continua.

v) Riesgo de mala logística de distribución:

- ✓ Realizar un análisis detallado de los requerimientos de distribución del producto.
- ✓ Diseñar un plan de distribución eficiente que tome en cuenta la ubicación de los proveedores, clientes y puntos de distribución.
- ✓ Utilizar herramientas de seguimiento y monitoreo en tiempo real, como RFID (Identificación por Radiofrecuencia) o GPS, para rastrear los envíos y garantizar la entrega oportuna.
- ✓ Establecer acuerdos y contratos claros con los proveedores y socios de distribución, definiendo las responsabilidades y expectativas.
- ✓ Realizar inspecciones de calidad en los productos antes de su envío, asegurando que cumplan con los estándares establecidos.
- ✓ Implementar mecanismos de retroalimentación de los clientes para evaluar la satisfacción y calidad de los productos entregados.
- ✓ Establecer indicadores clave de desempeño (KPIs) para medir la eficiencia y efectividad de la logística de distribución.

vi) Riesgo de falta de conocimiento del mercado

- ✓ Realizar un análisis exhaustivo del mercado objetivo, identificando las necesidades, preferencias y comportamientos de los clientes potenciales.
- ✓ Utilizar herramientas de investigación de mercado, como encuestas, entrevistas y análisis de datos, para obtener información relevante sobre el mercado y los competidores.

- ✓ Identificar y definir claramente los perfiles de los usuarios del producto, considerando aspectos demográficos, preferencias y necesidades específicas.
- ✓ Realizar pruebas de usabilidad con usuarios representativos del mercado objetivo, para evaluar la facilidad de uso y la satisfacción con el producto.
- ✓ Realizar análisis de datos del mercado, como tendencias de consumo, preferencias de compra y patrones de comportamiento, para identificar oportunidades y ajustar la estrategia de negocio.

7.3. Medición de la calidad

La medición de la calidad con métricas es una práctica fundamental en el aseguramiento de la calidad de un producto de software, ya que permite evaluar objetivamente diferentes aspectos de su rendimiento y conformidad con los requisitos establecidos. Las métricas proporcionan datos cuantitativos que permiten medir y comparar el grado de calidad del producto en diferentes etapas de su desarrollo. A continuación, se listan las métricas a a tener en cuenta en la medición de la calidad para el proyecto:

Tabla 14

Métricas de aseguramiento de la calidad

Nombre	Métrica	Umbral de aceptación
Seguridad	Número de vulnerabilidades identificadas en pruebas de seguridad.	Menor o igual a 5 vulnerabilidades críticas por versión del software
Rendimiento	Tiempo de respuesta promedio del sistema	Menor o igual a 2 segundos para las transacciones más frecuentes
Usabilidad	Índice de satisfacción del usuario (evaluado mediante encuestas o pruebas de usabilidad).	Mayor o igual al 80% de satisfacción del usuario.
Calidad del código	Cobertura de pruebas unitarias	Mayor o igual al 65% de cobertura de pruebas unitarias

Compleción de la documentación	Porcentaje de documentos completados según la planificación	Mayor o igual al 85% de los documentos completados según la planificación.
Efectividad de la gestión de cambios	Porcentaje de cambios implementados correctamente sin causar impacto negativo	Mayor o igual al 90% de cambios implementados correctamente sin impactos negativos.
Variación en la duración de las tareas	Porcentaje de variación en la duración de las tareas en comparación con la planificación inicial	Menor o igual al 10% de variación en la duración de las tareas.
Cumplimiento de actividades de seguimiento y control	Porcentaje de actividades de seguimiento y control realizadas según lo programado.	Mayor o igual al 95% de actividades de seguimiento y control realizadas según lo programado
Capacitación y desarrollo del equipo	Porcentaje de empleados que han recibido capacitación relevante	Mayor o igual al 80% de empleados que han recibido capacitación relevante.
Utilización de recursos	Porcentaje de utilización de recursos según la planificación.	Utilización de recursos mayor o igual al 90% según la planificación
Implementación de acciones de mejora	Porcentaje de acciones de mejora implementadas	Utilización de recursos mayor o igual al 90% según la planificación
Cumplimiento de plazos de entrega	Porcentaje de entregas realizadas dentro de los plazos acordados	Mayor o igual al 95% de entregas realizadas dentro de los plazos acordados.
Evaluación de la satisfacción del cliente	Calificación promedio de satisfacción del cliente en una escala de 1 a 10.	Calificación promedio de satisfacción del cliente mayor o igual a 8

Nota: Esta tabla contiene las métricas de aseguramiento de la calidad del software. Fuente, elaboración propia.

7.4. Conclusiones del capítulo

Este capítulo establece las bases para asegurar la calidad del desarrollo de la aplicación MECAD, aplicando prácticas y procesos adecuados, y siguiendo normas y estándares reconocidos en la industria. Esto contribuirá a minimizar los riesgos, asegurar la funcionalidad adecuada de la aplicación y promover la satisfacción de los usuarios. El propósito es asegurar que la plataforma cumpla con los estándares

y requisitos de calidad, proporcionando una experiencia segura, eficiente y confiable para los productores y consumidores.

Se identifican riesgos, se dan acciones concretas para reducir o mitigarlos y se establecen métricas mara medir la calidad tanto del producto como del proceso.

El plan se basa en diversas normas y estándares, como el IEEE 730, ISO/IEC 25010, ISO/IEC 12207, ISO/IEC 9126, IEEE 1012 y IEEE 1540, que proporcionan directrices y principios para la gestión de calidad, verificación y validación del software, y gestión de riesgos.

8. Conclusiones

Durante el desarrollo de este trabajo de grado, se ha tenido la oportunidad de aplicar y consolidar los conocimientos adquiridos durante la Especialización en Ingeniería de Software. A través de la investigación de referentes conceptuales y antecedentes, el análisis y la implementación de la aplicación móvil MECAD, hemos enfrentado desafíos técnicos y metodológicos que han permitido fortalecer las habilidades y competencias en el campo del desarrollo de software.

Trabajar en este proyecto ha brindado una experiencia enriquecedora y significativa, donde se ha aprendido a gestionar de manera efectiva los recursos disponibles, a planificar y organizar el trabajo en equipo, y a enfrentar y resolver problemas de manera creativa. Además, se ha tenido la oportunidad de crear un producto que busca acercar los productores agrícolas con los usuarios finales aumentando los beneficio tanto para unos como para otros, con una solución que realmente aporte valor a su trabajo y mejore la experiencia en la comercialización de sus productos.

Esta experiencia de trabajo de grado ha proporcionado no solo la oportunidad de aplicar los conocimientos técnicos adquiridos, sino también de desarrollar habilidades de comunicación, trabajo en equipo y resolución de problemas.

Es motivo de orgullo el resultado alcanzado con la confianza en que el proyecto MECAD contribuirá de manera significativa al sector agrícola y abrirá nuevas oportunidades en el campo del desarrollo de aplicaciones móviles.

9. Anexos

9.1. Anexo A. Manual de técnico

9.2. Anexo B. Manual de usuario

El manual de usuario será una guía que le permitirá a quien haga uso de la aplicación bien sea productor o consumidor realizar las diferentes transacciones.

10. Referencias

Anaya, N., & Salmon, C. (s/f). *Conceptos de NativeScript*.

Anshari, M., Almunawar, M. N., Masri, M., & Hamdan, M. (2019). Digital marketplace and FinTech to support agriculture sustainability. *Energy Procedia*, *156*, 234–238. https://doi.org/10.1016/j.egypro.2018.11.134

Bautista Sánchez, A. (2020). *Desarrollo de aplicaciones móviles*. http://www.upo.es/MoleQla

Bórnez, F. (2022). *Agricultura Digital. El camino hacia la agricultura sostenible*. www.regaber.com

Jablonski, A. (s/f). *Business models : strategies, impacts and challenges*.

Jimenez-Torres, V., Tello-Borja, W., & Rios-Patiño, J. I. (s/f). Lenguajes de Patrones de Arquitectura de Software: Una Aproximación Al Estado del Arte Pattern Languages Software Architecture: An Approach To State of the Art. *Scientia et Technica Año XIX*, *19*(4).

Ministerio de Agricultura y desarrollo Rural, Departamento Administrativo de Ciencia, T. e I. (Colciencias), & Corporación Colombiana de Investigación Agropecuaria (Corpoica). (s/f). *PECTIA - Plan Estratégico de Ciencia, Tecnología e Innovación del Sector Agropecuario Colombiano (2017-2027)*.

Molina Ríos, J. R., Honores Tapia, J. A., Pedreira-Souto, N., & Pardo León, H. P. (2021). Estado del arte: metodologías de desarrollo de aplicaciones móviles. *3C Tecnología_Glosas de innovación aplicadas a la pyme*, *10*(2), 17–45. https://doi.org/10.17993/3ctecno/2021.v10n2e38.17-45

Oiver Pérez. (2011). *Cuatro enfoques metodológicos para el desarrollo de Software RUP – MSF – XP - SCRUM*.

Ramírez-Olivera Gabriela, Benítez-Guerrero Edgard, & Mezura-Godoy Carmen. (s/f). *Mapeo sistemático de la literatura sobre experiencia de usuario en sistemas de recomendación*.

Ramón Toniut, H. (2021). La Transformación Del Modelo De Negocios en La Era Digital en Los Retails De Indumentaria. *Palermo Business Review*, 73–96.

Simmonds, N. (2019). *Análisis de frameworks para desarrollo de aplicaciones móviles y web Nicolás Simmonds Samper*.