

**DESARROLLO DE LA CAPA DE DATOS DEL CLIENTE MÓVIL DE VENTAS
PARA EL SISTEMA MIPYME DE LA EMPRESA CELESTE TEAM S.A.S**



Jhulieth Eliana Dorado Suarez

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Departamento de Telemática

Programa de Ingeniería Electrónica y Telecomunicaciones

Popayán – 2023

**DESARROLLO DE LA CAPA DE DATOS DEL CLIENTE MÓVIL DE VENTAS
PARA EL SISTEMA MIPYME DE LA EMPRESA CELESTE TEAM S.A.S**



Trabajo en modalidad práctica profesional

Estudiante: Jhulieth Eliana Dorado Suarez

Director: Mary Cristina Carrascal Reyes

Asesor de la empresa: Jhon Alex Narváez

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Departamento de Telemática

Programa de Ingeniería Electrónica y Telecomunicaciones

Popayán – 2023

Agradecimientos

A Dios, por ser mi guía, darme sabiduría y tranquilidad en todo momento;

*A mi madre por ser mi compañera, mi ejemplo y las enseñanzas que me ha
brindado con mucho amor;*

*A mi padre por apoyarme, guiarme y enseñarme que todo lo puedo lograr con
esfuerzo y disciplina;*

A Juan Camilo Vásquez por creer en mí, por su apoyo incondicional y consejos;

*A mi familia que me ha impulsado a ser mejor cada día y me ha apoyado en las
adversidades;*

A todos mis amigos por su apoyo y motivación en los días difíciles;

*A Mary Cristina Carrascal por su apoyo y disposición a lo largo del desarrollo de
este proyecto.*

TABLA DE CONTENIDO

1	CAPITULO I. DESCRIPCIÓN Y PLANTEAMIENTO	1
1.1	Descripción.	1
1.2	Planteamiento	2
1.3	Objetivos	3
1.3.1	General:	3
1.3.2	Específicos:	3
1.4	Estructura del documento	4
2	CAPÍTULO II. CONCEPTOS RELACIONADOS	5
2.1	Patrón de diseño MVVM y anotaciones de código	5
2.2	Definiciones claves.	6
3	CAPÍTULO III: FASES DE EJECUCIÓN	9
3.1	Fase de preparación	9
3.2	Fase de implementación y pruebas	9
3.3	Fase de síntesis de la práctica profesional	10
4	CAPÍTULO IV. CARACTERIZACIÓN DEL MÓDULO DE VENTAS	11
4.1	Descripción General del Módulo de Ventas Celeste	11
4.2	Configuración de acceso al sistema	14
4.3	Inicio de sesión	15
4.4	Sucursales	16
4.5	Numeración de documentos asociados a la venta	16
4.6	Cotizaciones	16
4.7	Pedidos	17
4.8	Factura de Venta	18

4.9	Forma de Pago	19
4.10	Forma de Entrega	20
4.11	Patrones de diseño y definición de arquitectura	20
4.11.1	Requerimientos	20
4.11.2	Arquitectura para aplicaciones móviles Android.	22
4.12	Patrones de diseño	27
5	CAPÍTULO V. MARCO DE TRABAJO PARA LA CAPA DE DATOS	28
5.1	Análisis	28
5.2	Descripción del marco de trabajo de la capa de datos para clientes de escritorio	30
5.2.1	Peticiones Generales	31
5.2.2	Tipos de datos de respuesta	33
5.2.3	Pila de peticiones	34
5.2.4	Forma de solicitudes	35
5.3	Adaptación del marco de trabajo para la capa de datos	36
5.3.1	Definición de diseño	36
5.4	Implementación	39
5.4.1	Pila de peticiones para el contexto Android	39
5.4.2	Peticiones generales para el contexto Android	41
5.5	Proceso para realizar una solicitud al servidor	43
6	CAPÍTULO VI. IMPLEMENTACIÓN DE LA CAPA DE DATOS PARA EL MÓDULO DE VENTAS	44
6.1	Marco de trabajo para la implementación del módulo de ventas	44
6.1.1	Asignación de Roles	45
6.1.2	Modelo de Historia de Usuario	46
6.1.3	Product Backlog	47
6.2	Sprint 1.	50
6.2.1	Sprint Planning	50
6.2.2	Sprint Execution	50
6.2.3	Sprint Review	54
6.2.4	Sprint Retrospective	55
6.3	Sprint 2.	55
6.3.1	Sprint Planning	55

6.3.2	Sprint Execution	56
6.3.3	Sprint Review	59
6.3.4	Sprint Retrospective	59
6.4	Sprint 3	60
6.4.1	Sprint Planning	60
6.4.2	Sprint Execution	60
6.4.3	Sprint Review	63
6.4.4	Sprint Retrospective	63
6.5	Sprint 4.	63
6.5.1	Sprint Planning	63
6.5.2	Sprint Execution	65
6.5.3	Sprint Review	67
6.5.4	Sprint Retrospective	67
6.6	Sprint 5.	68
6.6.1	Sprint Planning	68
6.6.2	Sprint Execution	69
6.6.3	Sprint Review	70
6.6.4	Sprint Retrospective	71
6.7	Resultados	71
7	RESULTADOS Y CONCLUSIONES	72
7.1	Cumplimiento de objetivos	72
7.2	Conclusiones	73
8	REFERENCIAS	75
	ANEXOS	77

LISTA DE FIGURAS

Figura 1. Cronograma de la implementación de la capa de datos módulo de ventas.	10
Figura 2. Diagrama de módulos del sistema MYPIME relacionados con la implementación.	13
Figura 3. Interfaz Configuración Empresa.	14
Figura 4. Interfaz Iniciar Sesión.	15
Figura 5. Interfaz Formulario para Cotizaciones.	17
Figura 6. Interfaz Búsqueda de Pedidos.	18
Figura 7. Interfaz Registrar Factura de Venta.	18
Figura 8. Interfaz Formas de pago.	20
Figura 9. Capas de la Clean Architecture.	23
Figura 10. Arquitectura propuesta para la implementación.	25
Figura 11. Diagrama de paquetes de la aplicación móvil Celeste.	26
Figura 12. Diagrama de servicios Celeste.	29
Figura 13. Procesos para realizar una petición al servidor de Celeste.	30
Figura 14. Tabla búsqueda de Facturas de Venta.	31
Figura 15. Tabla búsqueda de Terceros.	32
Figura 16. Tabla edición para ingresar productos.	32
Figura 17. Pila de peticiones ejecutadas por el cliente.	34
Figura 18. Gestión de peticiones con Volley.	37
Figura 19. Pila de peticiones del marco de trabajo para el cliente Android.	40
Figura 20. Procesos para realizar una petición general.	41
Figura 21. Flujo de procesos para realizar una petición al servidor.	43
Figura 22. Representación del Product Backlog.	49
Figura 23. Sprint 1- Sprint Backlog.	50
Figura 24. Sprint 2 - Sprint Backlog.	55
Figura 25. Sprint 3 - Sprint Backlog.	60

Figura 26. Sprint 4 - Sprint Backlog.....	64
Figura 27. Sprint 5-Sprint Backlog.....	68

LISTA DE TABLAS

Tabla 1. Descripción del objeto ParametrosResultadoMetodo.	33
Tabla 2. Forma de solicitud de autenticación.....	35
Tabla 3. Forma de solicitud Post Obj.....	35
Tabla 4. Descripción de los mensajes de red obtenidos en una petición.	40
Tabla 5. Requerimientos para ejecutar una petición general.....	42
Tabla 6. Descripción de los componentes tipo tabla del sistema Celeste.	42
Tabla 7. Descripción de la asignación de roles Scrum.	45
Tabla 8. Descripción HU 76 - Ver información en un formulario.	51
Tabla 9. Descripción de la HU 77 - Cargar datos en un formulario.....	51
Tabla 10. Descripción de la HU 4 - Solicitar código de verificación para configurar empresa.	52
Tabla 11. Descripción de la HU 12 - Validar código de configuración.....	52
Tabla 12. Descripción de la HU 30 - Ingresar credenciales para configurar huella.	53
Tabla 13. Descripción de la HU 29 - Ingresar huella para configurar datos biométricos.....	53
Tabla 14. Descripción de la HU 28 - Ingresar al sistema con huella.....	54
Tabla 15. Descripción de la HU 55 - Visualizar alerta en días de periodo de gracia.	56
Tabla 16. Descripción de la HU 54 – Seleccionar sucursal para ingresar al sistema.	56
Tabla 17. Descripción de la HU 32 - Visualizar menú.....	57
Tabla 18. Descripción de la HU 37 - Ver tabla de terceros registrados.	58
Tabla 19. Descripción de la HU 35 - Registrar Tercero.....	59
Tabla 20. Descripción de la HU 35 - Registrar Producto.	61
Tabla 21. Descripción de la HU 36 - Asignar código de barras a un producto.....	61
Tabla 22. Descripción de la HU 113 - Visualizar PDF.	62

Tabla 23. Descripción de la HU 60 - Filtrar factura de venta por opciones avanzadas.	62
Tabla 24. Descripción de la HU 61 - Asignar tercero cliente a la factura de venta.	65
Tabla 25. Descripción de la HU 63 - Cambiar resolución de una factura de venta.	65
Tabla 26. Descripción de la HU 65 - Agregar Productos.	66
Tabla 27. Descripción de la HU 67 - Cambiar valor unitario de un producto.	66
Tabla 28. Descripción de la HU - Agregar una forma de pago.....	67
Tabla 29. Descripción de la HU 108 - Registrar factura de venta.....	69
Tabla 30. Descripción de la HU 72 - Asignar fecha de vencimiento cotización.....	69
Tabla 31. Descripción de la HU 109 - Registrar Pedido.	70
Tabla 32. Descripción de la HU - Visualizar PDF pedido.....	70

LISTA DE ACRÓNIMOS

MiPyME	Micro, Pequeña y Mediana Empresa
DIAN	Dirección de Impuestos y Aduanas Nacionales
IVA	Impuesto sobre el valor añadido
MVC	Modelo-Vista-Controlador
MVVM	Modelo-Vista-Vista de Modelo
SO	Sistema Operativo
IU	Interfaz de Usuario
UVT	Unidad de Valor Tributario
CRM	<i>Customer Relationship Management</i>
WSC	<i>Web Service Celeste</i>
RMI	<i>Java Remote Method Invocation</i>
REST	<i>Representational State Transfer</i>

1 CAPITULO I. Descripción y planteamiento

En este capítulo se presenta la descripción y el planteamiento del problema abordado del actual trabajo de grado. Adicionalmente se establecen los objetivos y estructura del documento.

1.1 Descripción.

Las Micro, Pequeñas y Medianas Empresas cumplen un papel fundamental en la economía de Colombia ya que aportan un 40% al Producto Interno Bruto (PIB) y generan aproximadamente el 79% del empleo en Colombia[1]. Al tener esto en cuenta, se ha evidenciado la necesidad de garantizar una adecuada gestión administrativa y contable que les permita crecimiento y sostenimiento empresarial. Esto es posible controlando de manera eficiente las operaciones financieras y administrativas. Además, estas empresas deben validar el tipo de responsabilidades tributarias que deben cumplir para evitar sanciones o multas[2].

El proceso de gestión contable consiste en el manejo de la información contable necesaria para realizar reportes auténticos de la actividad y situación actualizada de la empresa a terceros, por ejemplo, administración tributaria e inversionistas. Asimismo, el proceso de gestión administrativa está relacionado con cada una de las actividades involucradas en el funcionamiento de las empresas, ya sea desde la perspectiva legal o propia de la organización. Entre estos procesos se encuentran: procesamiento, liquidación y pago de nómina, contratación y gestión de bonificaciones al personal.

La empresa Celeste Team S.A.S ofrece una solución tecnológica llamada Celeste que se adapta a las necesidades de cada empresa mientras cumple con los requisitos internos y a su vez, la normatividad legal según sus responsabilidades (responsables de IVA, no responsables de IVA, facturación electrónica, etc.). Cada uno de los procesos comerciales (ventas, productos, facturación) pueden tener asociado un proceso contable que requiera configuraciones específicas según corresponda el tipo de procedimiento o normatividad.

De acuerdo con las necesidades que requieran los clientes, Celeste maneja las siguientes versiones:

- **Celeste Mipyme:** Esta versión permite gestionar procesos contables y administrativos de una Mipyme, incluye los módulos de inventario, contabilidad, nómina, compras y facturación electrónica.
- **Celeste Enterprise:** Adicional a las opciones de la versión “Celeste Mipyme”, ofrece el desarrollo personalizado para funcionalidades específicas de

determinados sectores comerciales, por ejemplo, cooperativas de caficultores.

- **Celeste Lite:** Esta versión no tiene disponible los módulos de inventario, contabilidad y nómina. Sólo ofrece la gestión de facturación electrónica.

1.2 Planteamiento

El sistema Celeste está basado en una versión de escritorio con opciones de alojamiento en la nube, de esta manera es posible gestionar adecuadamente la información contable de la empresa, gestionar ventas, compras, gestionar nóminas, procesos normativos, inventario, entre otros. Sin embargo, los clientes manifestaron su preferencia por el uso de dispositivos móviles respecto a los dispositivos de escritorio, de acuerdo con los siguientes escenarios:

- El personal de la empresa se desplaza continuamente a diferentes sectores para poder ofrecer productos o servicios.
- Algunas empresas no cuentan con un punto físico determinado.
- Las empresas cuentan con un número considerado de usuarios, por lo tanto, no es viable instalar un computador de escritorio o laptop por cada uno de ellos, al contrario, requieren el acceso al sistema a través de tabletas o smartphones.
- Los usuarios del sistema han manifestado su preferencia por el uso de aplicaciones móviles.

Esto no es ajeno a la tendencia y versatilidad que ofrecen hoy en día los dispositivos móviles. Tal como indica la actualización **Digital 2022 Global Overview Report [3]**, en Colombia es evidente el impacto que tienen los dispositivos móviles y las aplicaciones. En esta actualización se indica que, la población colombiana conformada por 51.39 millones de personas (Urbanización: 82%), tiene activas 65.75 millones de conexiones móviles (incidencia: 127.9%). De este número de personas, 35.50 millones son usuarios de internet, es decir, el 69.1% de la población total tiene acceso a este servicio, de acuerdo con este porcentaje, el 97.7% de los usuarios de internet, usan teléfonos inteligentes (*smartphone*) frente al 74% de usuarios que usan Laptop u ordenador de escritorio.

Dados estos escenarios de uso, se evidenció la necesidad de implementar una versión móvil que permitiera gestionar de manera ágil procesos de ventas, inventario y cierre de negocios.

Es importante recalcar que el sistema ha sido implementado con librerías y marcos de trabajo propios para el contexto de escritorio. Por lo tanto, para la ejecución de la presente práctica profesional, el equipo de Celeste estableció que se debía hacer un adecuado uso, adaptación e implementación de estos para el contexto móvil.

Al tener en cuenta los anteriores requisitos, la empresa Celeste Team determinó el alcance de la implementación móvil del sistema Celeste en su primera versión, de la siguiente manera:

- Desarrollo del módulo de Ventas: Este es uno de los principales procesos al que los clientes acceden, requiere un alto nivel de complejidad debido a los procesos que implican su desarrollo, tales como: gestión de productos, control de permisos por usuario, configuración de empresas, gestión de sucursales, CRUD de terceros clientes y proveedores, movimiento y control de inventario, gestión de facturas de ventas, cotizaciones, pedidos y gestión de pagos.
- Desarrollo capa de datos: La estructura de la aplicación móvil se dividió en dos capas de acuerdo con la arquitectura recomendada de Android [4], la capa de datos y la capa de interfaz de usuario. De acuerdo con el alcance y definición del problema, se ha determinado que, para efectos del desarrollo del presente trabajo de grado, se debía definir el marco de trabajo para la capa de datos y validarlo a partir de la implementación de la capa de datos del módulo de ventas para el cliente móvil; el desarrollo de la capa interfaz de usuario estará a cargo de otro personal de la empresa.

1.3 Objetivos

1.3.1 General:

Desarrollar la capa de datos para el módulo de ventas del cliente móvil “Celeste” adaptando el marco de trabajo de desarrollo de aplicaciones de escritorio de la empresa Celeste Team S.A.S.

1.3.2 Específicos:

- Caracterizar el módulo de ventas de la empresa Celeste Team S.A.S teniendo en cuenta los patrones arquitectónicos orientados a las aplicaciones móviles.
- Adaptar el marco de trabajo orientado a la capa de datos de aplicaciones de escritorio de la empresa Celeste Team S.A.S para clientes móviles.
- Validar el marco de trabajo a partir de la implementación de la capa de datos en el módulo “Ventas” para el cliente móvil en Android.

1.4 Estructura del documento

A continuación, se describe la estructura del documento del presente trabajo de grado.

Capítulo 1 – Descripción y planteamiento: en esta sección se realiza una breve descripción del sistema Celeste, del planteamiento del problema y el alcance de implementación definido por la empresa Celeste Team.

Capítulo 2 – Conceptos aplicados para la implementación del trabajo de grado: se incluyen los conceptos técnicos que permiten al lector tener una base para comprender el desarrollo del presente trabajo.

Capítulo 3 – Fases de ejecución: se describen las fases de ejecución y la hoja de ruta obtenida de la implementación de la capa de datos del módulo de ventas.

Capítulo 4 – Caracterización del módulo de Ventas: se hace una descripción general del sistema Mipyme con el fin de identificar los procesos de negocio. Además, se analizan los módulos implicados en el desarrollo para definir los patrones de arquitectura y diseño.

Capítulo 5 – Marco de trabajo para la capa de datos: se describe el funcionamiento de la capa de datos del sistema Celeste y se realiza la respectiva adaptación para el contexto de aplicaciones móviles Android.

Capítulo 6 – Implementación de la capa de datos del módulo de Ventas: se presenta el desarrollo de la aplicación móvil implementando el marco de trabajo para la capa de datos.

Capítulo 7 – Conclusiones y resultados: por último, se presentan las conclusiones y se expone el cumplimiento de los objetivos propuestos para el presente trabajo de grado.

2 Capítulo II. Conceptos relacionados

En este capítulo se presentan los conceptos relacionados con la presente práctica profesional, además, se exponen dos artículos que permitieron afianzar los lineamientos clave de desarrollo definidos por la empresa Celeste Team.

2.1 Patrón de diseño MVVM y anotaciones de código

Al tener en cuenta los aspectos de desarrollo de una aplicación, es muy importante definir las ventajas del patrón de arquitectura a utilizar en la implementación. En el estudio, ***Comparative Study of Performance and Productivity of MVC and MVVM design patterns*** [5], se hace una comparación entre los patrones *MVC* y *MVVM* con el fin de determinar cuál patrón es el más adecuado. Aunque se basan en prototipos web, los resultados se pueden aplicar en diferentes contextos, es decir, son un indicador para diferentes tipos de aplicaciones.

El principal objetivo del anterior estudio era determinar el rendimiento y productividad de los prototipos de desarrollo basados en *MVVM* y *MVC*. Se tuvieron en cuenta parámetros como: indicador clave, tiempo de respuesta, línea de código, complejidad y disponibilidad de información. Las variables de estudio tuvieron un peso dependiendo de sus indicadores, la variable de rendimiento tuvo un peso del 70% y la de productividad el 30%.

Los resultados para el indicador de rendimiento establecieron que el patrón *MVVM* presentó mejor comportamiento con un 69.7% que el patrón *MVC* con un 27.94%; Los resultados obtenidos para el indicador de productividad fueron 25.38% para el patrón *MVVM* y 13.67% para el patrón *MVC*. Los autores pudieron determinar que el patrón *MVVM* presenta mejores condiciones de rendimiento y productividad, la diferencia más distintiva del patrón *MVC* es que la acción del usuario afecta al controlador directamente, y para el patrón *MVVM* afecta a la vista.

Por último, una de las características clave de desarrollo para tener en cuenta es el adecuado procesamiento, uso e implementación de anotaciones java, puesto que el modelo de desarrollo de Celeste emplea anotaciones para mantener integración menos detallada y más cercana al elemento de código. Esto permite que los programas Java admitan el procesamiento adicional en tiempo de compilación, tiempo de implementación y tiempo de ejecución.

El procesamiento de anotaciones es un tema ampliamente investigado desde la perspectiva del buen uso, contexto de código y la evaluación de los impactos negativos que puedan generar en los productos de software. En el artículo, ***How does annotations affect Java code readability*** [6], se hace un estudio detallado basado en encuestas con el fin de determinar la legibilidad de los fragmentos de

código con y sin anotaciones, adicionalmente, los resultados también pudieron reflejar los impactos de diseño, mantenimiento y productividad al usarlas.

En el análisis de los estudios relacionados se contrasta la legibilidad de código y la calidad del software al usar o no anotaciones, dónde se evidencia como aspecto positivo que, el uso de anotaciones puede reducir el acoplamiento, proporcionar una evolución más consistente y mejorar potencialmente la calidad del código. Por otro lado, los estudios evidenciaron que esta práctica al usarse en exceso o arbitrariamente puede afectar el tamaño y generar código repetitivo, dificultando la depuración e introduciendo problemas de código.

Los estudios de legibilidad de código se basaban generalmente en percepciones personales más que en la literatura científica, por lo tanto, los autores han ampliado la investigación de tal manera que se puedan definir escenarios claros y casos de estudio con el fin de guiar a diseñadores y usuarios de las API a definir opciones de diseño, ya sea entre una API basada en anotaciones u orientada a objetos.

De los resultados de este estudio los autores concluyeron que una implicación práctica para los diseñadores de API es que las anotaciones no dañaran necesariamente la legibilidad de código, en donde se encontró que una cantidad significativa de personas revelaron una preferencia con respecto a las anotaciones ya que permitían tener una mejor claridad y legibilidad al hacer el código más conciso. Por lo tanto, es necesario definir claramente los objetivos, nombres y alcance que cumplen las anotaciones para que puedan ser fáciles de usar e interpretar, pero específicamente para que sea posible aprovechar todas las ventajas que brindan.

2.2 Definiciones claves.

En esta sección se presentan los conceptos técnicos relevantes relacionados al desarrollo de este trabajo.

Arquitectura de software

La arquitectura de software se puede definir de diferentes maneras según el sistema o contexto de ejecución. Una de las definiciones más acertadas es la propuesta por la norma ISO/IEC/IEEE 42010 en la que se refiere a arquitectura de software como los “conceptos fundamentales o propiedades de un sistema en su entorno, materializados en sus elementos, relaciones y en los principios de su diseño y evolución”[7]. Ampliando un poco esta definición, la arquitectura de software de un sistema representa las decisiones de diseño que ayudan a las partes interesadas a entender y analizar cómo el sistema logrará cualidades como modificabilidad, disponibilidad y seguridad.

Construir una arquitectura de software óptima permite la creación rápida de productos, evolución continua y eficaz del sistema. Para esto es necesario

identificar, priorizar y equilibrar las cualidades significativas desde el punto de vista arquitectónico. Es decir, se deben tomar un conjunto de decisiones importantes sobre la organización del desarrollo de software y cada una de estas decisiones puede tener un impacto en la calidad, la mantenibilidad, el rendimiento y el éxito general del producto final[8].

Patrón de diseño

Para tener un concepto claro y efectivo de lo que es un Patrón de diseño se tiene el propuesto por Gamma et al. en su libro *Design Patterns*: “Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interfaces de usuario”[5].

Los objetivos principales de un patrón de diseño son:

- Proporcionar una base reutilizable de componentes para el diseño de sistemas software.
- Estandarizar los términos comunes entre diseñadores
- Estandarizar la forma de realizar un diseño
- Facilitar la integración de un conocimiento existente para futuros diseñadores.

El amplio uso de patrones de diseño se debe al crecimiento del sector de las tecnologías de la información ya que estas han hecho que las prácticas de desarrollo de software evolucionen, puesto que esto ha permitido agilizar y validar correctamente la ejecución, mejorar la calidad y productividad.

Modelo-Vista-Vista de Modelo

Al tener en cuenta el concepto de Patrón de diseño, se aclara que *MVVM* es un patrón de arquitectura definido como una variación del patrón *MVC*, este patrón se adapta a plataformas de desarrollo de interfaz de usuario modernas. *MVVM* se basa en un mecanismo de separación de la lógica de la interfaz de usuario o de la lógica empresarial, lógica encargada de la recuperación y gestión de datos de la aplicación; de este modo, su objetivo principal es mantener el código de la interfaz de usuario separado de la lógica de la aplicación con el fin de que sea más fácil su administración.

Las clases contenidas en los “Models” representan el modelo de dominio de la aplicación, dentro de este se encuentra el modelo de datos y la lógica empresarial. Este es el primer componente al iniciar una aplicación.

Las vista define la estructura, el diseño y la apariencia de lo que ve el usuario [9].

El modelo de vista contiene la lógica de presentación en la cual se definen las propiedades de las funcionalidades de la aplicación, los cuales son capaces de

enlazar con la vista, es decir, se encarga de la comunicación entre los modelos y las vistas.

Anotaciones de código java

La anotación de código java se agregó a partir de Java 5, esta es una característica del lenguaje de programación que permite agregar metadatos personalizados directamente en el código fuente. Estos metadatos se pueden asociar a clases, miembros, métodos o parámetros.

El objetivo de las anotaciones es delegar lógicas repetitivas, como la inicialización de objetos y el reenvío de solicitudes a compiladores y marcos de tiempo de ejecución, de esta manera se puede reducir la complejidad de programación y mejorar la eficiencia de desarrollo. Muchas de las anotaciones deben cumplir con los siguientes aspectos:

- Sintaxis sencilla y agradable.
- Facilidad de introducción a una base de código.
- Relativamente no invasivas.

Reflexión de Java

Reflexión en Java es el proceso de analizar y modificar todas las capacidades de una clase en tiempo de ejecución. De este modo, reflexión permite que un programa tenga la capacidad de observar y modificar su estructura de forma dinámica. Es así como es posible elegir una clase que se ha escrito previamente en un IDE y después de compilada, mientras el programa se ejecuta, poder modificarla. Por ejemplo, es posible que una clase Java obtenga los nombres de todos sus miembros y los muestre [10].

La capacidad de examinar y manipular una clase desde sí misma puede parecer simple, pero en otros lenguajes esta característica no es posible implementarla. Por ejemplo, en un programa Pascal, C o C++ no es posible obtener información sobre funciones definidas dentro de ese programa.

Un ejemplo de uso de reflexión es JavaBeans, donde los componentes software se pueden manipular visualmente a través de una herramienta. Esta herramienta utiliza reflexión para obtener las propiedades de los componentes de java a medida que se cargan dinámicamente.

3 CAPÍTULO III: Fases de ejecución

Para el desarrollo del presente trabajo se identificaron 3 fases de ejecución.

3.1 Fase de preparación

El objetivo de esta fase fue identificar, contextualizar y caracterizar las funcionalidades actuales del sistema Celeste proyectando los requerimientos del cliente móvil teniendo en cuenta el patrón de arquitectura y tecnologías móviles.

Las actividades de esta fase se distribuyen de la siguiente manera:

- Identificar los procesos de negocio involucrados en el módulo ventas del sistema Celeste.
- Explorar la versión de escritorio del sistema Celeste.
- Definir los módulos asociados a la implementación.
- Determinar el patrón de arquitectura de software.
- Identificar los patrones de diseño que se ajusten adecuadamente a la implementación del sistema.
- Implementar prototipos que permitieran determinar el comportamiento del patrón de arquitectura de Software, patrones de diseño y la fuente de datos.
- Explorar la herramienta de gestión Jira-Atlassian para la gestión del marco de trabajo SCRUM.

3.2 Fase de implementación y pruebas

En esta fase se ejecutó la adaptación del marco de trabajo orientado a la capa de datos e implementación del prototipo móvil Celeste aplicando el marco de trabajo SCRUM.

Las actividades de esta fase se distribuyen de la siguiente manera:

- Explorar el código desarrollado en la versión de escritorio para realizar el proceso de incorporación de la versión móvil.
- Configurar del entorno de desarrollo.
- Identificar las tecnologías de gestión de bases de datos locales y remotas.
- Analizar el marco de trabajo de escritorio orientado a la capa de datos.
- Implementación del marco de trabajo para el contexto móvil Android.
- Implementar la capa de datos del módulo Ventas.

De esta fase se obtuvo el cronograma de la implementación de la capa de datos del módulo de ventas aplicando el marco de trabajo SCRUM, como se muestra a continuación:

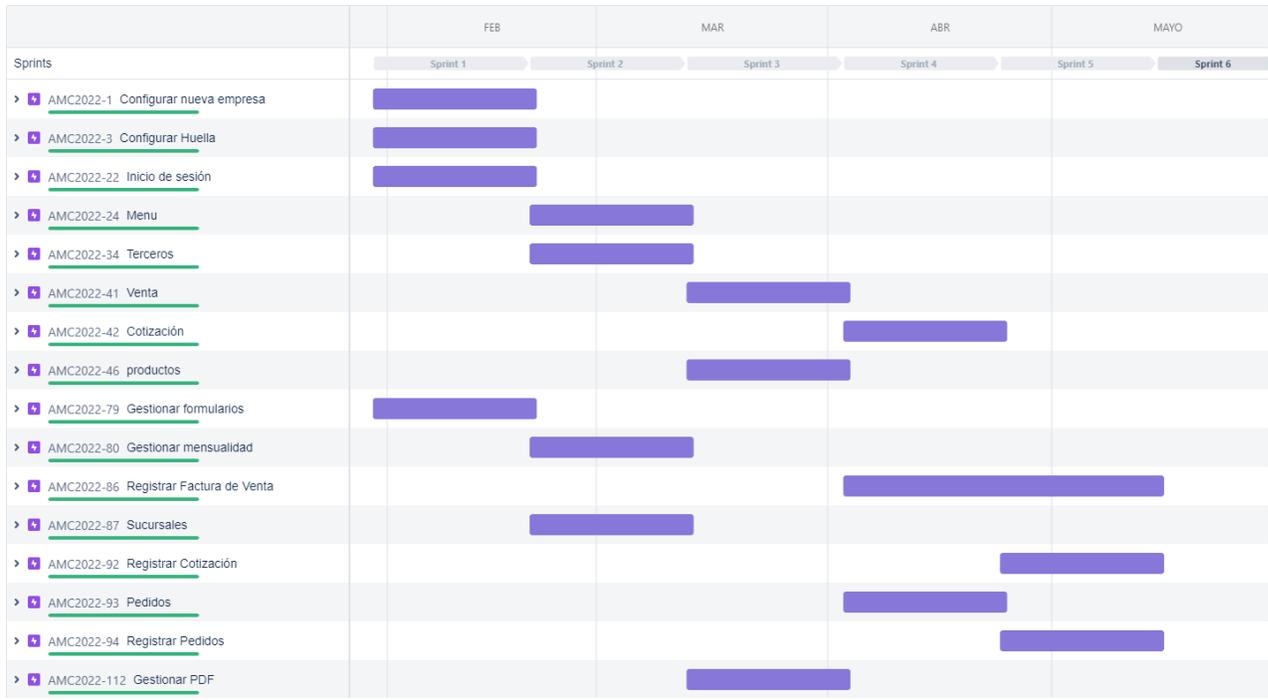


Figura 1. Cronograma de la implementación de la capa de datos módulo de ventas.

Fuente: Realización propia.

3.3 Fase de síntesis de la práctica profesional

Esta fase consistió en la generación de la monografía de trabajo de grado y el proceso de sustentación con el fin de evidenciar el cumplimiento de los objetivos propuestos.

4 Capítulo IV. Caracterización del módulo de ventas

En este capítulo se presenta la caracterización del sistema Celeste orientado al módulo de Ventas. Adicionalmente, se define la arquitectura y los patrones de diseño que se aplicaron en la implementación del proyecto.

4.1 Descripción General del Módulo de Ventas Celeste

Antes de describir el funcionamiento del sistema Celeste es necesario tener en cuenta los siguientes términos:

- **Documento asociado a ventas:** un documento asociado al proceso de venta puede generar movimientos de inventario o caja o no tener movimientos. En este caso los documentos que se tuvieron en cuenta fueron: Factura de Venta, Cotización, Pedido y Comprobante de Venta.
- **Autorización de numeración:** una factura de venta puede tener asociada una autorización de numeración la cual está conformada por un prefijo y un sufijo. Para los casos de las facturas de venta con tipos de resolución POS o Factura Electrónica, las resoluciones son expedidas por la DIAN y tienen configuraciones como fecha y numeración máxima de generación o uso.
- **Numeración de documentos:** para los documentos tipo pedido y cotización se maneja una numeración automática la cual está conformada por un prefijo y un sufijo.
- **Forma de pago:** las formas de pago están asociadas a las facturas de venta y se configuran previamente en el sistema. El usuario puede generar y gestionar cada una de las formas de pago con el fin de realizar los cuadros de caja diarios.
- **Vencimiento de un documento:** los documentos como Pedidos o Cotizaciones tienen una fecha de vencimiento, el cual indica la vigencia de los precios, promociones u otras características aplicadas.
- **Configuraciones del sistema:** las configuraciones permiten habilitar o deshabilitar algunos comportamientos asociados a los módulos del sistema, algunas configuraciones son: IVA, impuesto al consumo, Factura Electrónica, Cantidades de Venta negativos, Control de precios, gestión de listas de precios, entre otros.
- **Listas de precios:** las listas de precio son un conjunto de precios asociados a un tipo de cliente, estas listas permiten vender un producto o servicio a un precio diferente de acuerdo con el cliente (mayorista, distribuidor, etc.).
- **Permisos del sistema:** los permisos están asociados a los usuarios registrados de cada empresa y a las funciones a los que estos usuarios pueden acceder. Estos permisos son gestionados por el administrador y están distribuidos por módulos.

- **Facturador Electrónico:** este término está asociado con las empresas que deben generar y emitir facturas electrónicas en cumplimiento con los requisitos establecidos por la DIAN.
- **Sucursal:** una sucursal es la división de una empresa de acuerdo con la ubicación, tipos de productos u otra característica que requiera la sectorización de dicha empresa.
- **Forma de Entrega:** es un módulo que permite determinar la forma en la que un producto será entregado. Por ejemplo, en bodega, en punto o a domicilio.
- **Usuario del sistema:** los usuarios del sistema no están estrictamente categorizados por el momento. El sistema cuenta con dos tipos de usuarios, Administrador y usuario con permisos limitados.
- **Tercero Referido:** tercero que recomendó el producto o servicio y que, de acuerdo con los módulos habilitados, puede definirse como codeudor de una factura de venta o como mejor convenga. Este tercero se habilita previamente en las configuraciones del sistema.
- **Periodo de gracia:** Es el periodo en días que tiene un usuario para usar el sistema al vencerse la fecha límite de pago.

El flujo o navegación del sistema Celeste inicia con la configuración de una empresa. Este proceso permite obtener los datos y configuraciones necesarias para establecer o iniciar una sesión de cliente. Seguido a esto, se puede encontrar la interfaz de usuario para iniciar sesión, si el usuario ingresa sus credenciales correctamente y cumple las validaciones de sesión, se mostrará el menú con las funciones o módulos a los que éste tiene permiso.

Uno de los módulos principales del sistema Celeste es el módulo Ventas. Este módulo les permite a los usuarios generar un documento asociado a la venta. Cada tipo de documento se gestiona como un módulo diferente, es decir, los pedidos, cotizaciones, facturas de venta y comprobantes de venta, son manejados como módulos por separado.

Los tipos de documentos asociados a la venta se diferencian por 3 características principales:

- **Movimiento de caja:** Es el control de los movimientos de “dinero” registrados como ingresos o egresos, estos movimientos pueden ser en efectivo, transferencia u otra forma de pago registrada en el sistema.
- **Movimiento de inventario:** Es el registro de los movimientos de las cantidades existentes de un producto. Las cantidades pueden estar disponibles o pendientes de acuerdo con el proceso que sea registrado en el sistema.
- **Sin movimientos:** No registra ningún movimiento de caja o inventario.

Los documentos asociados a la venta requieren la asignación de una resolución o numeración, un tercero cliente y una lista de productos, adicionalmente, las facturas de venta tienen asociada una forma de pago y una forma de entrega.

A continuación, se presenta el diagrama de módulos relacionado con la implementación (ver Figura 2).

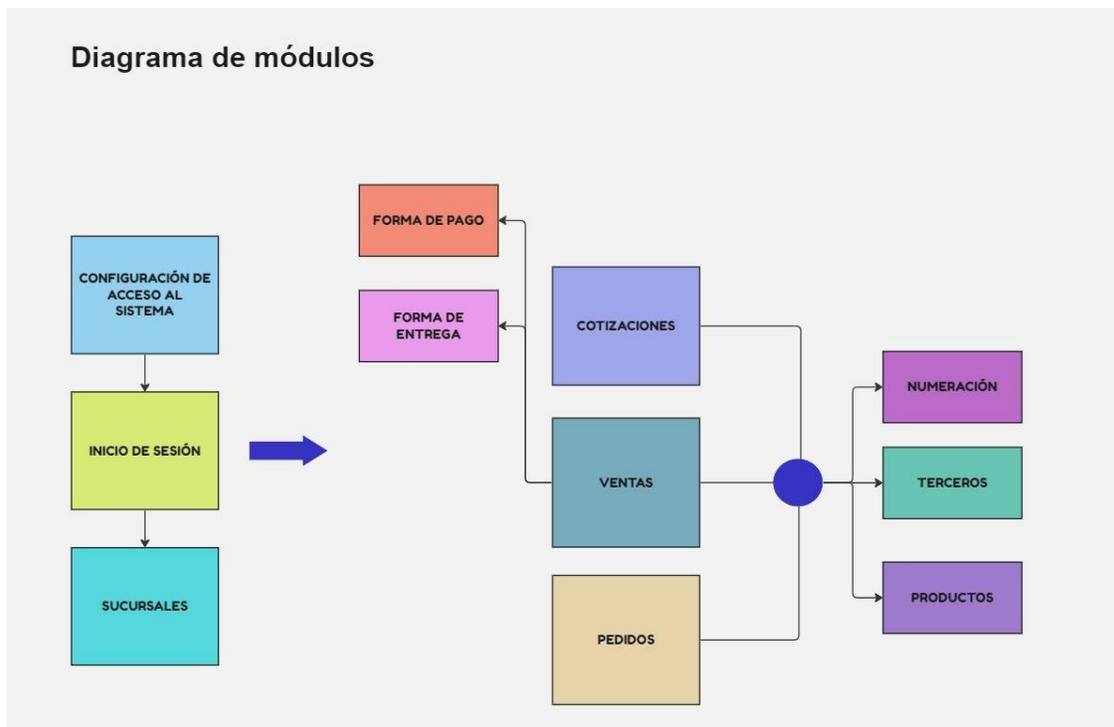


Figura 2. Diagrama de módulos del sistema MYPIME relacionados con la implementación.
Fuente: Realización propia.

Es necesario tener en cuenta que cada uno de los módulos tiene una asignación de permisos y configuraciones previas con el fin de limitar el acceso a determinados usuarios de acuerdo con las funciones definidas por el administrador del sistema.

Adicionalmente, se deben conocer las configuraciones contables que se deban aplicar, según corresponda. Ejemplo: Factura electrónica, tipos de resolución POS, declarantes, no declarantes, impuestos, entre otros.

4.2 Configuración de acceso al sistema

Iniciando con el flujo de la aplicación Celeste se tiene el módulo configuración empresa (ver interfaz de usuario en la Figura 3). Este módulo es el encargado de controlar el acceso de los usuarios al sistema.

Para configurar una empresa el usuario debe ingresar el NIT de la empresa, leer y aceptar los términos y condiciones. Si el NIT ingresado es válido se enviará un código de verificación de seis dígitos al correo del administrador del sistema. En este procedimiento es necesario identificar el dispositivo que ha solicitado un código. Para esto el sistema registra un identificador único por equipo, información hardware y la versión del sistema operativo.

Si el código de verificación ingresado es válido se presenta la interfaz de inicio de sesión. En este proceso se realiza internamente la gestión de los datos necesarios para iniciar sesión, tales como: número de referencia, nombre empresa, identificador y NIT de la empresa, login y clave (usados para la autenticación de peticiones al servidor). Es necesario tener en cuenta que el proceso de configuración se realiza una única vez por equipo y empresa.

 **celeste**
Sorprendentemente Simple

Solicitar Autorización

Para activar el servicio, introduce el Nit de la empresa a la cual desea conectarse.

Se enviará un correo al administrador de la empresa el cual debe autorizar que se tenga acceso a la información privada de la organización.

CEDULA O NIT (*) (Sin puntos y sin DV)

Acepto los términos y condiciones de la licencia de uso
[Ver Terminos y Condiciones](#)

Solicitar Autorización

Figura 3. Interfaz Configuración Empresa.
Fuente: Realización propia.

4.3 Inicio de sesión

El módulo de Inicio de Sesión es el encargado de gestionar el ingreso al sistema para los diferentes usuarios (ver interfaz de usuario en la Figura 4). Para iniciar sesión, el usuario debe seleccionar la empresa e ingresar sus credenciales. En este proceso el sistema organiza los datos **solicitados** que fueron almacenados en la configuración de empresa (número de referencia, nombre, identificador y NIT de la empresa, el login y clave del servicio web).

Si toda la información es correcta se inicia con el proceso de validación de sesión de usuario, de la siguiente manera:

- **Se almacenan los parámetros de sesión:** estos parámetros permiten identificar configuraciones del sistema, permisos y módulos activos para el usuario que ha iniciado sesión.
- **Se válida si el sistema tiene vigente la mensualidad de pago:** el sistema permite validar si un cliente se encuentra en periodo de gracia (se genera una alerta para el usuario), estado suspendido (no es posible brindar ingreso al sistema) o vigente.
- **Se corrobora si el sistema tiene configurada más de una sucursal:** si una empresa tiene configurada más de una sucursal, se presenta el listado de sucursales a las que puede acceder el usuario.

Powered by : www.celeste.com.co

Figura 4. Interfaz Iniciar Sesión.
Fuente: Realización propia.

4.4 Sucursales

Las sucursales hacen referencia a la división programada de una empresa ya sea por ubicación, tipos de productos u otra característica definida por el cliente.

Cada uno de los movimientos registrados en el sistema quedarán guardados en la respectiva sucursal en la que se inició sesión. Esto compete a la gestión de productos, usuarios, terceros, facturas de venta, cotizaciones, pedidos, etc. Es necesario contemplar que cada sucursal tiene un **local** asociado.

4.5 Numeración de documentos asociados a la venta

Las resoluciones o numeración de documentos hacen referencia al proceso de asignación de numeración de las facturas de venta, pedidos o cotizaciones. Esta numeración es obligatoria en cada uno de los registros de los documentos.

Los pedidos y cotizaciones tienen una asignación de numeración automática, y los documentos para registro de ventas, requieren una gestión personalizada por el usuario. El proceso de gestión de resoluciones de factura de venta consiste en la creación de resoluciones de acuerdo con el proceso u obligatoriedad que requiera una empresa. Por ejemplo, los usuarios pueden registrar resoluciones tipo POS o electrónicas, si es que así lo requieren. Este tipo de resoluciones deben ser expedidas por la DIAN y requieren un control de fecha o numeración máximas para utilizar, estos procesos los controla el sistema.

4.6 Cotizaciones

Las cotizaciones son un tipo de documento asociado a la venta que no registran movimientos en la caja o inventario. Para registrar una cotización es necesario indicar el tercero cliente, vendedor, tercero referido (si aplica) y la lista de productos a cotizar (ver Figura 5).

Una cotización debe tener una fecha de vencimiento, de este modo, los usuarios pueden controlar la vigencia de los precios, promociones u otra característica aplicada.

Asimismo, las cotizaciones tienen asociado un estado de documento. El primer estado de una cotización es “Por facturar”, este estado indica que una cotización ha sido registrada pero no se ha convertido en una factura de venta. El segundo estado es “Facturada”, indica que una cotización se ha convertido en una factura de venta, por lo tanto, no es posible realizar otros movimientos con este documento.

Por último, es posible editar una cotización para los casos en los que se requiera agregar o quitar productos, cambiar fecha de vencimiento, cambiar tercero, vendedor o referido.

Figura 5. Interfaz Formulario para Cotizaciones.
Fuente: Realización propia.

4.7 Pedidos

Los pedidos registran movimientos en el inventario y tienen pendiente el proceso de pago. Para registrar un pedido es necesario asignar el tercero cliente, vendedor, tercero referido (si aplica) y la lista de productos. Además, un pedido debe tener asociada una fecha de vencimiento. De esta manera los usuarios pueden controlar la fecha en la que el pedido requiere ser entregado.

Los pedidos tienen 3 tipos de estado (Ver interfaz de búsqueda de pedidos en la Figura 6):

- **Pendiente:** es el estado inicial de todos los pedidos realizados. Indica que tiene pendiente el proceso de facturación de los productos.
- **Anulado:** este estado indica que el proceso del pedido no ha podido ser completado. Al anular un pedido el sistema realiza un ajuste de inventario.
- **Facturado:** el pedido ha completado el proceso de pago y se convierte en una factura de venta. Este estado representa que el documento ha sido cruzado.

Por último, es posible editar un pedido para los casos en los que se requiera agregar o quitar productos, cambiar fecha de vencimiento, cambiar tercero, vendedor o referido.

Menú Búsqueda Pedidos / Celeste MIPYME / DEMO PRUEBAS / Administrador Sistema / Equipo: DESKTOP-38E7L1T-156979

Anular Editar Reimprimir Facturar Enviar Email Movimientos Terminar

Fecha Inicial: 22/03/2023 Fecha Final: 22/06/2023 Estado: Facturados

Buscar Por: General B Avanzada

Todos
Facturados
Pendientes
Anulados

Pedidos Registrados

Prefijo	Número	Fecha	Tercero	Total	Estado
1 PV	9	15-06-2023	Juanita Sofia Paz	106.397.885	10.000 Cruzado
2 PV	10	14-06-2023	Daniel Alejandro Potosi	10.305.494	30.000 Cruzado
3 PV	5	09-05-2023	Daniel Alejandro Potosi	10.305.494	20.000 Cruzado
4 PV	3	29-04-2023	Santiago Medina	80.181.293	810.750 Cruzado
5 PV	2	29-04-2023	Caja De Compensacion Familiar Del Cauca Comfacauc	891.500.182	50.000 Cruzado

Empresa: Administrador Sistema Vendedor: Administrador Sistema Vencimiento: 16/06/2023 Referido: Estado: Cruzado

X	Código	Pedido	Cruzado	Pend.	Descripción	Vr/Unit	% Desc.	Vr/Unit Final	total	IVA
1	274	1	1	0	Gramos pulpa	10.000	0	10.000	10.000	Exclud

Figura 6. Interfaz Búsqueda de Pedidos.
Fuente: Realización propia.

4.8 Factura de Venta

Una factura de venta está compuesta por una resolución, fecha de registro, tercero cliente, tercero referido, lista de productos, formas de pago, forma de entrega y orden de compra, como indica la Figura 7.

Menú Facturación / Celeste MIPYME / DEMO PRUEBAS / Administrador Sistema / Equipo: DESKTOP-38E7L1T-156979

Nuevo Registrar Adjuntar Archivos B/ Venta B/ Cotización Agregar Pedido Cuadre Caja G/ Clientes Anticipos

Número: POSD 2 Fecha:

Factura Cotización Pedido

DATOS DEL CLIENTE

C.C/NIT: 222.222.222.222 - E.C. 1°NOM: Consumidor 2°NOM: 1°AP: Final 2°AP: Registrar

Empresa: Depart: CAUCA Munici: POPAYÁN

Dirección: Celular: Tel: E-Mail: notiene@notiene Clasifi: Sin Cla... Limpiar

Or/Compra:

Eliminar Fila (F9) Vendedor: 1 - Administrador Sis Referido: Cupo: 0 Disponible: 0 Anticipos: 0 Cartera: 0 Cartera Vencida: 0

	Código	Cant.	Descripción	Lote	Vr/Unit	% Desc.	Vr/Unit Final	Total	IVA	Tercero Manc
1		0			0	0	0	0		

Figura 7. Interfaz Registrar Factura de Venta.
Fuente: Realización propia.

La gestión de este módulo está definida por el tipo de cliente (declarantes, no declarantes, facturador electrónico) y las configuraciones del sistema que se requieran. Las principales características para tener en cuenta son las siguientes:

- **Autorización de Numeración o Resoluciones:** las resoluciones se gestionan de acuerdo con el tipo de responsabilidad tributaria que tenga un cliente. Si un usuario configura una resolución POS o Electrónica, el sistema debe realizar algunos ajustes de acuerdo con las configuraciones que sean requeridas. Por ejemplo, las resoluciones tipo POS se manejan en los “puntos físicos” de venta, por lo tanto, son enlazadas a un ÚNICO dispositivo, es decir, en un dispositivo es posible expedir una factura con una resolución POS, sólo si, esta resolución ha sido enlazada previamente con este dispositivo.
- **Facturación electrónica:** las facturas electrónicas son gestionadas por un operador tecnológico que maneja el registro ante la DIAN. Para realizar facturas de venta con este tipo de resoluciones se tienen algunas restricciones. Por ejemplo, se admiten algunos medios de pago específicos, se genera un formato de documento que debe cumplir con los requerimientos de la DIAN, si el monto de la factura supera los 5 UVT vigentes sólo se puede registrar la factura con una resolución electrónica, entre otros.
- **Listas de precio:** las listas presentan las opciones de precios que tiene un producto, que por lo general son 4. Estas listas son gestionadas por el administrador del sistema y son asignadas a los diferentes usuarios a través de los permisos configurados.
- **Tipos de productos:** la gestión de productos está relacionada con cada uno de los tipos de productos, como: productos tipo área, productos con tercero mandato, productos con registro de lote, producto tipo combustible y producto con precio variable. Dependiendo del tipo de producto, se requiere el registro de más o menos datos. Para los que requieren área, es necesario registrar el ancho, el alto y la cantidad por área del producto.

El sistema ha definido un consumidor final en el caso que no sea necesario registrar la información de un tercero cliente. Esta configuración permite que las empresas tengan cargado por defecto este tipo de usuario siempre que ingresan a registrar un nuevo Documento de Venta.

4.9 Forma de Pago

Para finalizar una factura de venta es necesario definir una o varias formas de pago. El cliente puede elegir dentro de las opciones disponibles, los tipos de pago que desea que el sistema admita, como indica la Figura 8.

Es posible combinar una o muchas formas de pago hasta cubrir el valor total de la factura de venta. Cada una de las formas de pago se configuran y solicitan

diferentes datos. Por ejemplo, si se procesa un pago en Efectivo, el sistema solicita los datos de la caja en la que va a ingresar el pago.

Código	Descripción	Valor
1	Efectivo Caja General	20.000

Figura 8. Interfaz Formas de pago.
Fuente: Realización propia.

4.10 Forma de Entrega

La forma de entrega pertenece al módulo Bodega. Este módulo es configurado generalmente para los usuarios que manejan entregas en diferentes sedes o puntos. La gestión de bodega controla las cantidades de los productos disponibles y pendientes por entregar.

4.11 Patrones de diseño y definición de arquitectura

Luego de conocer el funcionamiento general del módulo de Ventas del sistema de Celeste, se describieron los patrones de diseño, arquitectura y las tecnologías a implementar. Para este proceso fue necesario determinar los requerimientos de la organización.

4.11.1 Requerimientos

Se definieron los siguientes requerimientos no funcionales asociados a los requisitos organizativos y del producto:

- **La aplicación debe ser desarrollada para Android:** de acuerdo con una encuesta realizada por Celeste Team, se definió que la primera versión móvil del sistema debería implementarse para dispositivos Android, ya que alrededor de un 82% de los usuarios utilizaban este tipo de sistema operativo. Además, las tecnologías usadas por Celeste Team permitirían un rápido

desarrollo en este tipo de entornos. De esta manera sería posible ofrecer un nuevo tipo de cliente en corto plazo.

- **La implementación debe realizarse en el lenguaje de programación Java:** de acuerdo con la experiencia y comodidad que brindan las tecnologías Java para el equipo de desarrollo de la empresa Celeste Team, se definió que el lenguaje de programación a utilizar para el desarrollo de la aplicación Android sería el lenguaje Java. Android Studio ofrece un soporte nativo sólido para Java y proporciona las herramientas necesarias para desarrollar aplicaciones Android utilizando este lenguaje.
- **La implementación de la aplicación estará guiada por el patrón de diseño MVVM:** la empresa definió al patrón MVVM como patrón de diseño teniendo en cuenta que es uno de los patrones más usados en el desarrollo de aplicaciones Android y ofrece una separación más clara entre la vista y el modelo a través del componente ViewModel.
- **El estilo de arquitectura para la comunicación es REST:** la comunicación e interacción entre el cliente y servidor se gestionó a través de REST como requerimiento del proceso de comunicación al lado del servidor.

Es necesario tener en cuenta que el desarrollo de este proyecto se distribuyó en la adaptación del marco de trabajo orientado a la capa de datos y la validación del marco de trabajo a partir de la implementación de la capa de datos del módulo Ventas del sistema “Celeste”. Es por esta razón que el desarrollo de la aplicación se dividió en proyectos que permitieron la rápida creación de funcionalidades.

4.11.2 Arquitectura para aplicaciones móviles Android.

En el desarrollo de aplicaciones móviles es posible definir una arquitectura de software que cumpla con los principios y prácticas recomendadas. Una de las arquitecturas más usadas en el desarrollo de aplicaciones Android es la *Clean Architecture*, esta arquitectura se enfoca en la implementación de inversión de dependencias y permite la separación del código en diferentes módulos con responsabilidades específicas. Esto facilita el mantenimiento, acoplamiento rápido y facilidad de pruebas [11].

En Clean Architecture es posible definir diferentes capas de acuerdo con las especificaciones de un proyecto, como indica la Figura 9, sin embargo, se pueden agrupar en 5 (las más usadas):

- **Presentación:** capa de interacción de usuario, principalmente son Fragmentos, Actividades, *ViewModels* (intermediario de acuerdo con el patrón de diseño), etc.
- **Casos de uso:** son principalmente las acciones que el usuario puede desencadenar. Estos pueden ser acciones activas (el usuario hace clic en un botón) o acciones implícitas (la aplicación navega a una pantalla).
- **Dominio:** es la capa central de esta arquitectura, contiene la lógica de negocio. Es el módulo Java completo.
- **Datos:** Gestiona los datos de la aplicación teniendo en cuenta las diferentes fuentes de acceso a los datos (servicios web, caché, bases de datos locales, etc.).
- **Marco de Trabajo:** encapsula la interacción con el *framework*, por lo que el resto del código puede ser reutilizable en caso de que se requiera desarrollar la misma aplicación para otra plataforma (una opción real hoy en día con los proyectos).

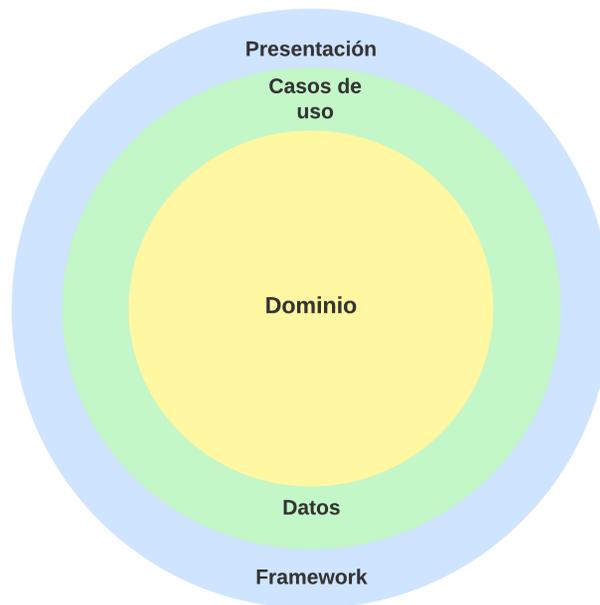


Figura 9. Capas de la Clean Architecture.
Fuente: Realización propia.

La implementación de este tipo de arquitecturas permite que el código sea fácil de probar, desacoplar y mantener, sin embargo, tiene claras desventajas. *Clean Architecture* tiene una curva de aprendizaje ligeramente empinada e implica la creación de múltiples clases, esto puede ocasionar confusión, además, muchas de las capas pueden generar ambigüedades en proyectos cortos o de bajo acoplamiento.

Para determinar adecuadamente la arquitectura a usar, Android ha presentado La guía de arquitectura de apps[4] bajo los principios de solidez y calidad. Esta guía define la experiencia adecuada de usuarios de aplicaciones para dispositivos móviles y los principios específicos que debe cumplir una arquitectura:

- **Separación de problemas:** de acuerdo con la responsabilidad de un *Activity* o *Fragment*, es necesario mantener estas clases limpias para evitar problemas relacionados con el ciclo de vida de los componentes y mejorar la capacidad de pruebas en estas clases.
- **Control de la interfaz de usuario a partir de modelos de datos:** los modelos de datos son independientes de los elementos de IU y otros componentes de la aplicación. Por lo tanto, no deben vincularse a la IU ni al ciclo de vida, ya que estos serán destruidos cuando el sistema operativo decida quitar el proceso de la aplicación de la memoria.

- **Única fuente de información:** si se define un nuevo tipo de dato es necesario asignar un SSOT (*Single Source of Truth*) y sólo este puede modificarlo o mutarlo. SSOT expone funciones o recibe eventos. Esto permite centralizar los cambios a un tipo particular de datos en un sólo lugar, proteger los datos para que no sean manipulados por otro tipo y hace que los cambios en los datos sean fáciles de detectar.
- **Flujo de datos unidireccional:** en el patrón UDF (*Unidirectional Data Flow*) el estado o datos fluye en una sola dirección y los eventos que modifican los datos fluyen en la dirección opuesta. Por ejemplo, los datos de la aplicación generalmente fluyen desde las fuentes de datos hacia la IU. Los eventos del usuario fluyen desde la IU hasta el SSOT, en donde los datos de la aplicación se modifican y se exponen en un tipo inmutable

Al tener en cuenta los principios descritos anteriormente, esta guía define que cada aplicación debe tener al menos dos capas:

- **Capa de Interfaz de Usuario:** se encarga de mostrar los datos en pantalla.
- **Capa de datos:** se encarga de exponer datos y contener la lógica de negocios.

De acuerdo con los requerimientos es posible determinar que:

- El desarrollo implicó bajo acoplamiento puesto que la implementación estaría dividida en diferentes proyectos.
- Fue necesario definir una clara separación de IU y datos.
- Fue necesario implementar nuevas funcionalidades de manera rápida.

Considerando las características descritas anteriormente, se definió que, para la implementación de este proyecto, la arquitectura estaría compuesta por tres capas, como indica la Figura 10. Esto permitió dar cumplimiento a los lineamientos establecidos en la Guía de Arquitectura de Aplicaciones Android, el patrón de arquitectura MVVM y los requerimientos no funcionales de la organización.

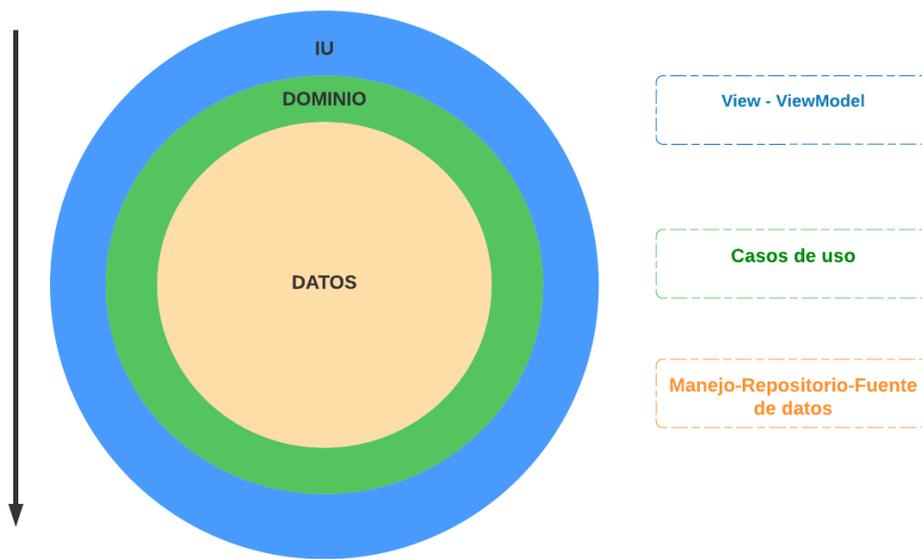


Figura 10. Arquitectura propuesta para la implementación.
Fuente: Realización propia.

- La capa IU (Presentación) contiene los paquetes *View* y *ViewModel* (parte de implementación del patrón de arquitectura MVVM), encargados de renderizar los datos de la aplicación en pantalla y de exponer los datos a la IU mediante lógica, esta capa está atenta a los eventos de la interfaz para actualizarla y reflejar los cambios.
- La capa Dominio es opcional, y representa los casos de uso que contienen lógica empresarial compleja o muy general que puede ser compartida.
- La capa Datos contiene las clases Manejo (en estas clases se desarrolla la lógica de negocio), Repositorios y Fuentes de Datos. Principalmente esta capa es la encargada de llevar a cabo la implementación de la lógica empresarial, el acceso y manejo de las bases de datos locales y remotas, el uso de repositorios que contienen las fuentes de datos y la definición de las clases modelo o parámetros.

La estructura de paquetes está definida por módulos de implementación, como indica la Figura 11, cada módulo contiene el paquete de IU (Control), dominio (si es necesario) y datos. Tal como se indica en la arquitectura, lo esencial es separar la lógica empresarial de la interfaz de usuario.

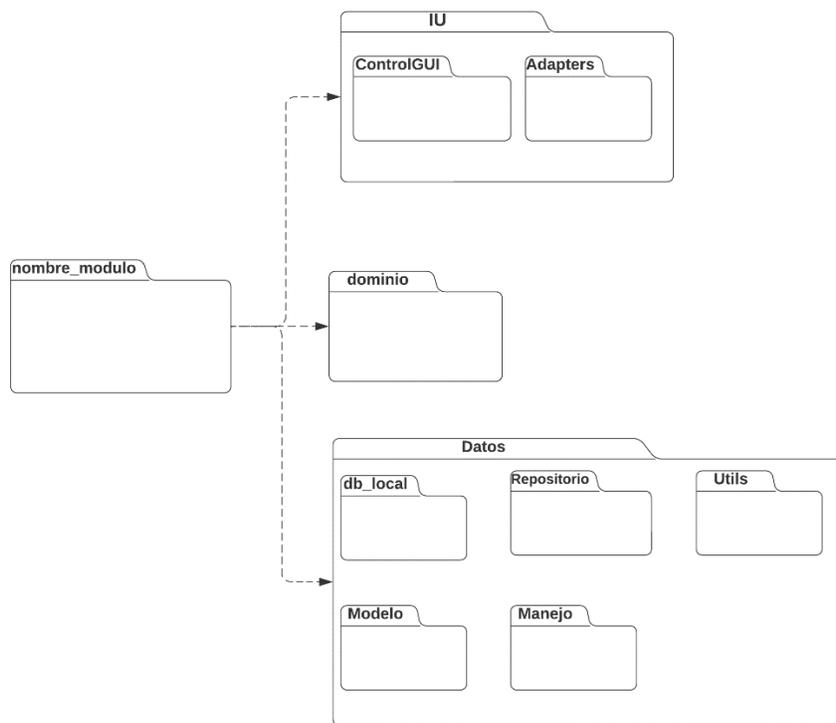


Figura 11. Diagrama de paquetes de la aplicación móvil Celeste.
Fuente: Realización propia.

Para la capa de datos se definieron los siguientes paquetes:

- **Db_local:** se encuentran todas las clases relacionadas a la gestión de base de datos locales.
- **Manejo:** contiene las clases encargadas de la lógica empresarial gestionando las llamadas al servidor.
- **Modelo:** contiene las clases que conforman los atributos o modelos necesarios para un módulo.
- **Repositorio:** la clase de repositorio permite abstraer el acceso a múltiples fuentes de datos, como obtener los datos de la API u obtener los datos de la base de datos de *Room*. Una clase de repositorio proporcionará una API limpia para acceder a los datos del resto de la aplicación. De esta manera, el repositorio contendrá una lógica que decidirá si tenemos que buscar los datos de la red o tenemos que obtener los datos de la base de datos [13].
- **Utils:** este paquete está conformado por una clase que contiene utilidades necesarias para un módulo, tales como: Url de servicio y Constantes.

4.12 Patrones de diseño

La implementación se llevó a cabo aplicando el patrón de arquitectura MVVM que extendió de la arquitectura definida anteriormente. Esto permitió la separación de responsabilidades y la adecuada abstracción de la lógica a implementar.

MVVM separa la Vista de la lógica empresarial aplicando el uso patrones de diseño como *Live Data*. Las clases *ViewModel* están diseñadas para administrar y almacenar información de manera consciente del ciclo de vida permitiendo que los datos se conserven ante los cambios de configuración como rotación de pantalla. Así, *Live Data* permite que cualquier Vista observe cambios en los datos y actualice la IU[11].

La Vista debe observar uno o más *ViewModels* para obtener la información necesaria y actualizar la IU, así puede obtener referencia a diferentes *ViewModels*, pero un *ViewModel* no puede tener ninguna referencia a una Vista. Las clases *Model* recuperan información de su fuente de datos y la exponen a los *ViewModels*.

Para la gestión de la capa de datos se seleccionaron algunos patrones de diseño teniendo en cuenta que algunas arquitecturas o paradigmas de programación implican el uso de determinados patrones de diseño. Excluyendo estos patrones, que han sido expuestos en las secciones correspondientes, se aplicaron dos en específico:

- **Singleton:** es un patrón creacional encargado de crear un objeto único, es decir, la instancia de un objeto se realiza una única vez y si ya ha sido instanciado se proporcionará un acceso al mismo. Este patrón es útil si se requiere que se almacene un mismo objeto único en memoria, por ejemplo, la instancia de Usuario que ha iniciado sesión.
- **Builder:** es un patrón de diseño creacional que permite crear objetos complejos de manera sencilla y organizada. Este patrón se utiliza cuando una clase contiene una gran cantidad de parámetros y es necesario crear una interfaz clara y fácil de usar para construir objetos de esa clase, esto permite que un constructor se encargue de todos los tipos de objetos, ayudándose de constructores concretos encargados de la creación de cada tipo en particular. Un objeto director será el encargado de coordinar y ofrecer los resultados.

5 Capítulo V. Marco de trabajo para la capa de datos

En este capítulo se presenta la adaptación del marco de trabajo orientado a la capa de datos desde el contexto de escritorio al contexto Android.

5.1 Análisis

El modelo de comunicación de datos de Celeste se basa en la arquitectura Cliente-Servidor que utiliza RMI (*Remote Method Invocation*) y REST (*Representational State Transfer*) para el diseño e implementación de los servicios o aplicaciones.

REST es un estilo de arquitectura para desarrollar aplicaciones y servicios web, se basa en 6 aspectos específicos: rendimiento (velocidad de respuesta), facilidad de modificación (evolución de acuerdo con las necesidades), visibilidad (monitoreo del sistema), portabilidad (para diferentes entornos), escalabilidad (control de carga y número de usuarios) y simplicidad (fácil de entender) [14].

RMI es un mecanismo que proporciona la comunicación remota entre programas Java a través de llamadas a métodos de objetos remotos situados en distintas (o la misma) máquinas virtuales. RMI exporta objetos como objetos remotos para que otro proceso remoto puede acceder directamente como un objeto Java. Todos los objetos de este tipo de aplicaciones deben ser implementados en Java.

En el proceso de adaptación del marco de trabajo de la capa de datos fue necesario conocer el modelo de comunicación de los servicios o aplicaciones del sistema Celeste. Este sistema se distribuye en diferentes servicios de autenticación, pagos, CRM, factura electrónica y correos, como indica la Figura 12.

El cliente debe comunicarse a estos servicios siguiendo el estilo arquitectónico REST. Es necesario tener en cuenta que el “servidor Mipyme” es una aplicación Java que utiliza RMI para el proceso de comunicación remota.

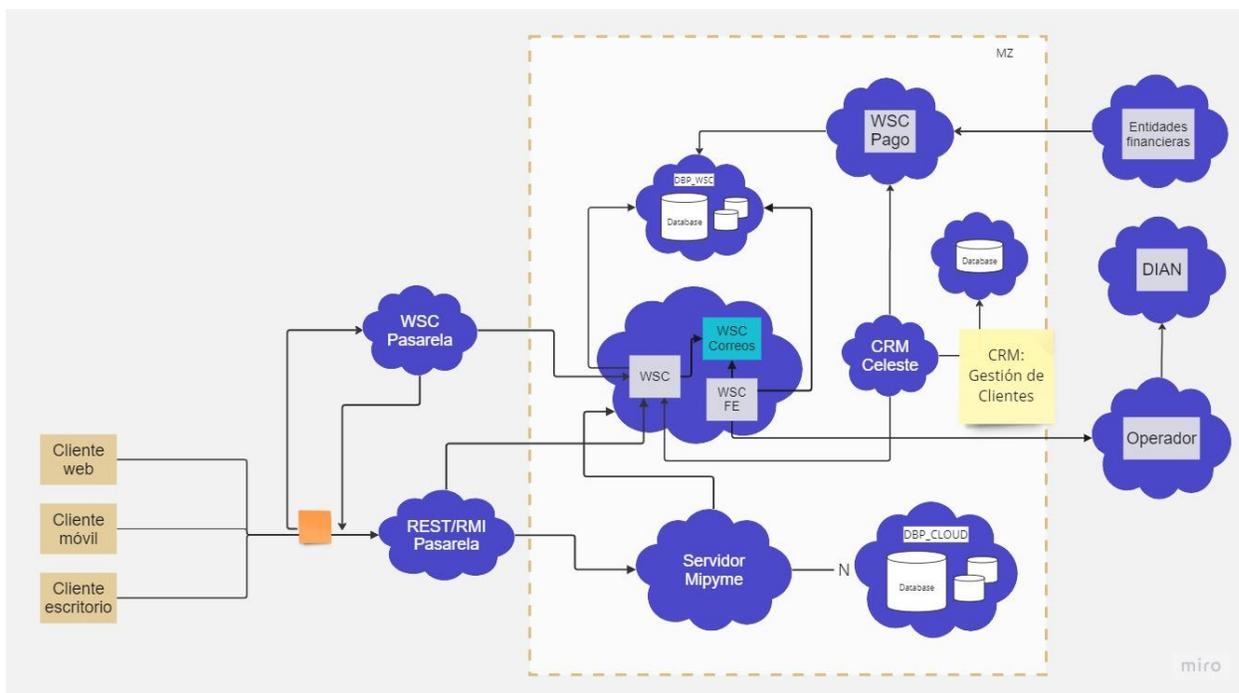


Figura 12. Diagrama de servicios Celeste.

Fuente: Realización propia.

Los clientes utilizan la pasarela REST/RMI que realiza una “traducción” de protocolos entre el cliente REST y el servidor RMI. Los clientes establecen la comunicación a través del servicio “WSC Pasarela” que realiza el proceso de autenticación y control del servicio REST/RMI.

De este modo, el marco de trabajo de la capa de datos del cliente se basa en la gestión de las solicitudes al servidor y la lógica empresarial de cada uno de los módulos. Para realizar y controlar estas solicitudes, el cliente de escritorio utiliza la librería OkHttp que proporciona un cliente Http y permite realizar solicitudes de manera **síncrona(bloqueante)**.

Las solicitudes de red ejecutadas de manera **síncrona** están asociadas a la programación estructurada, la cual se enfoca en la gestión síncrona de la información a través de la ejecución de peticiones de manera bloqueante (espera activa). Este tipo de paradigma de programación no es recomendable para contextos móviles ya que provoca bloqueos en la interfaz de usuario y una experiencia de usuario deficiente.

Por el contrario, para el contexto móvil es recomendado aplicar la programación reactiva. La reactividad se centra en los flujos de datos y el cambio de propagación, es decir, trata de responder a los cambios de valor. El flujo de datos se gestiona de manera asíncrona permitiendo que las tareas se ejecuten de forma no secuencial.

The Reactive Manifiesto[15] establece las siguientes bases para los sistemas reactivos:

- **Responsivos:** aseguran la calidad del servicio cumpliendo oportunamente con los tiempos de respuesta.
- **Resilientes:** se mantienen receptivos incluso cuando se enfrentan a fallas.
- **Elásticos:** se mantienen receptivos incluso ante aumentos en la carga de trabajo aumentando o disminuyendo los recursos asignados.
- **Orientados a mensajes:** garantizan un bajo acoplamiento entre componentes al establecer interacciones basadas en el intercambio de mensajes de manera asíncrona.

Es así como la adaptación del marco de trabajo de la capa datos se enfocó en la gestión asíncrona de las solicitudes al servidor extrayendo los componentes necesarios desde el contexto de escritorio.

5.2 Descripción del marco de trabajo de la capa de datos para clientes de escritorio

Para entender el orden en el que se ejecutan las tareas que permiten completar una solicitud al servidor se debe considerar el flujo de métodos que son usados directamente por el cliente y los métodos encargados de formar las peticiones, como se indica en la Figura 13.



Figura 13. Procesos para realizar una petición al servidor de Celeste

Fuente: Realización propia.

5.2.1 Peticiones Generales

Para realizar las solicitudes al servidor, el cliente accede a los métodos definidos como peticiones generales. Las peticiones generales son una “generalización” de los tipos de datos que necesita procesar el cliente. Estas peticiones se agrupan en los diferentes tipos de datos que están asociados, comúnmente, a los componentes o procesos de Celeste, tales como:

- **Tablas de búsqueda:** este tipo de componente permite visualizar la información de una búsqueda de datos en un ancho de columnas y filas, como indica la Figura 14 y Figura 15. Esta búsqueda puede corresponder a facturas de venta, cotizaciones, pedidos, terceros o productos. Para definir el modelo de datos que se mostrará en una tabla es necesario inicializarla con el tipo de parámetros requerido, por ejemplo: `ParametrosTablaB_Ventas` o `ParametrosTablaB_Terceros`.

Prefijo	Número	Fecha	Nit	Nombre Cliente	Total	Abono	Saldo	Dev	Estado	Estado FE
1	SETP 990.047.238	27-06-2023 17:54:47	222.222.222.222	Consumidor Final	25.000	25.000	0	0	Pagada	Aprobada
2	SETP 990.047.237	27-06-2023 17:42:07	80.181.293	Santiago Medina	28.500	0	28.500	0	Por Cobrar	Aprobada
3	SETP 990.047.236	27-06-2023 17:38:25	10.305.494	Daniel Alejandro Potosi	55.000	55.000	0	0	Pagada	Aprobada
4	SETP 990.047.235	23-06-2023 16:35:38	106.133.333	Tercero Prueba	68.500	68.500	0	0	Pagada	Aprobada
5	SETP 990.047.234	23-06-2023 15:53:10	222.222.222.222	Consumidor Final	30.000	30.000	0	0	Pagada	Aprobada
6	SETP 990.047.233	23-06-2023 15:36:57	10.985.666	Pruebas Email Demo	25.000	25.000	0	0	Pagada	Aprobada
7	SETP 990.047.232	23-06-2023 15:10:56	222.222.222.222	Consumidor Final	500	500	0	0	Pagada	Aprobada
8	SETP 990.047.231	22-06-2023 16:41:18	222.222.222.222	Consumidor Final	30.000	30.000	0	0	Pagada	Aprobada
9	SETP 990.047.230	22-06-2023 16:03:33	222.222.222.222	Consumidor Final	15.000	15.000	0	0	Pagada	Aprobada

Figura 14. Tabla búsqueda de Facturas de Venta.
Fuente: Realización propia.

Excel Nuevo Registrar Actualizar Gestionar Empresas Transferir Tercero Cambiar Tipo Tercero Gestionar Países Gestionar Clasificación

CLIENTES PROVEEDORES TERCEROS

DATOS DEL CLIENTE

C:C/NIT: [] - C.C. 1°NOM: [] 2°NOM: [] 1°AP: [] 2°AP: [] Registrar

Empresa: [] Depart: CAUCA Munici: POPAYÁN

Dirección: [] Celular: [] Tel: [] E-Mail: [] Clasifi: Sin Clasificación

Cupo: [] Vend: [] Retene: Retenciones Obser: []

Configuraciones

	Nit	Dv	Nombre	Empresa	Telefonos	Dirección	Email	Clasificación
1	891.500.182	0	Caja De Compensacion Fc		602 8231868	CALLE 2 NORTE 6 A 5	notificacionesjudicial@	Cliente Mayorista
2	10.305.494	1	Daniel Alejandro Potosi		000000	cl 5	danielpotosi@gmail.c	Sin Clasificación
3	1.098.652.147		Emanuel Palomino	Hidropal	3005698745	PARQUE CENTRAL	.ema@hidropal.com	Sin Clasificación
4	800.251.440	6	ENTIDAD PROMOTORA DE		6016466060	CARRERA 9 # 17AN23	sanitas@	Sin Clasificación
5	10.685.479.685		Jhon Narvaez		3205698714	centro	correo@correo.com	Sin Clasificación
6	89.699.655		Juan Carlos Cruz	Celeste	31526664	Dirección	juan@celeste.com.co	Sin Clasificación
7	106.397.885		Juanita Sofia Paz		3126547415	calle 11 # 8-63	pruebas@gmail.com	Sin Clasificación
8	2.027.961.951		Juan Jimenez	Celeste	3105412365	antonio Nariño	juab@celeste.com.co	Sin Clasificación
9	1.061.708.865		Juan Salazar		31089732154	di	contabilidad@celeste	Sin Clasificación

Figura 15. Tabla búsqueda de Terceros.
Fuente: Realización propia.

- **Tablas de edición:** para crear diferentes documentos de venta, Celeste ordena la información a través de las tablas edición. Este componente está compuesto por un ancho de columnas y filas de acuerdo con los datos que sean necesarios almacenar en cada procedimiento. Por ejemplo, para crear una factura de venta, el usuario ingresa la información de los diferentes productos a través de una tabla edición, como indica la Figura 16.

Nuevo Registrar Adjuntar Archivos B/ Venta B/ Cotización Agregar Pedido Cuadre Caja G/ Clientes Anticipos

Número POSD 2 Fecha Factura Cotización Pedido

DATOS DEL CLIENTE

C:C/NIT: 222.222.222 - C.C. 1°NOM: Consumidor 2°NOM: [] 1°AP: Final 2°AP: [] Registrar

Empresa: [] Depart: CAUCA Munici: POPAYÁN

Dirección: [] Celular: [] Tel: [] E-Mail: notene@notene.cor Clasifi: Sin Clasific... Limpiar

Eliminar Fila (F9) Vendedor Referido Cupo Disponible Anticipos Cartera Cartera Vencida

	Código	Cant.	Descripción	Lote	Vr/Unit.	% Desc.	Vr/Unit Final.	Total	IVA	Tercero Mandato
1		0			0	0	0	0		

OBSERVACIONES: Descuento 0% \$ 0

Cantidades	Peso(Kg)	Excluido	Exento	Gravado	IVA	Ajuste	TOTAL
0	0	0	0	0	0	0	0 COP

Figura 16. Tabla edición para ingresar productos.

Fuente: Realización propia.

- **Autocompletado de datos:** un autocompletado es una pequeña tabla de búsqueda que se usa comúnmente en las tablas edición. Este tipo de componente acelera el proceso de filtrado ya que permite definir un límite de datos a obtener.
- **Gestión de parámetros:** la gestión de parámetros hace referencia a la forma ordenada que se crean, almacenan, eliminan o actualizan datos de un formulario.
- **Método general:** este método permite realizar solicitudes que no están asociadas a ningún componente descrito anteriormente y define como resultado un dato tipo `Object`.
- **Método resultado:** a diferencia del método general, este tipo de método permite obtener como resultado un dato tipo `ParametrosResultadoMetodo`.
- **Ingreso al sistema:** está método está completamente dedicado a establecer el inicio de sesión. Para esto es necesario obtener el Id de sesión y la clave pública (criptografía RSA).
- **Configuración Rest:** este es el primer proceso que se debe realizar en un cliente si se requiere realizar una solicitud al servidor. Aquí se configura la URL de petición de autenticación, el path de servicio, el login y clave que proporcionan el acceso a los recursos del servidor.

5.2.2 Tipos de datos de respuesta

Las peticiones realizadas al servidor establecen un cuerpo de respuesta `String` que generalmente representa una cadena de texto codificada. Esta cadena hace referencia, en la mayoría de los casos, a un objeto. Este objeto debe ser manejado por una librería interna que realiza el proceso de conversión para obtener un tipo de dato `Object` o `ParametrosResultadoMetodo`. Este último es uno de los más usados ya que permite manejar la respuesta teniendo en cuenta otras características, como indica la siguiente tabla:

Tabla 1. Descripción del objeto `ParametrosResultadoMetodo`.

Atributo	Descripción
Fecha	Es posible que la petición requiera enviar fechas que indiquen fecha de última versión, fecha de registro de datos, etc.

Evento	Este evento permite validar si una petición ha generado error, alerta, error de versión, sesión inválida o se ha ejecutado correctamente.
Otros datos	En otros datos es posible enviar objetos.
Mensaje	El servidor puede determinar los mensajes predeterminados que serán mostrados al cliente.

5.2.3 Pila de peticiones

Para realizar una solicitud al servidor, el cliente ejecuta una pila de peticiones tipo POST, para el caso del módulo de ventas. Estas peticiones tienen un orden establecido, como se indica en la Figura 17.

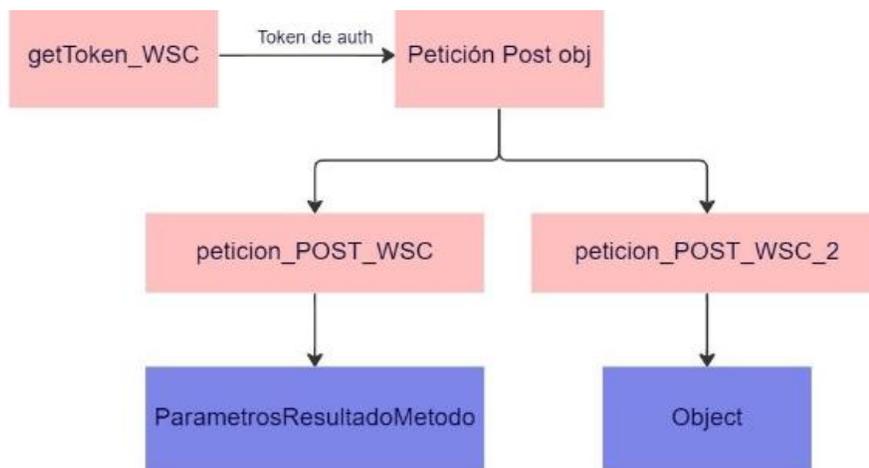


Figura 17. Pila de peticiones ejecutadas por el cliente
Fuente: Realización propia.

- **Petición de autenticación:** para acceder a los servicios de Celeste, el cliente debe realizar un proceso de autenticación a través del método de autenticación basado en token. El usuario provee las credenciales y el servidor generará un token basado en esas credenciales.
- **Petición Post Obj:** esta petición es la encargada de gestionar todas las solicitudes al servidor. Requiere el identificador de la empresa, el token (generado en la primera petición), la url del método y los parámetros requeridos. Estos parámetros se definen con la notación *Object ... params* que permite crear un parámetro de argumentos variables.

- **Peticiones Post WSC y WSC_2:** la Petición Post Obj proporciona un objeto que es utilizado por las peticiones post_wsc (retorna un ParametrosResultadoMetodo) y post_wsc_2 (retorna un Object).

La url del método formada en la petición Post permite identificar el método publicado en el servidor.

url_solicitud_metodo = url_api + path_servicio + url_metodo

- *url_api*: corresponde al dominio o hostname junto con el puerto
- *path_servicio*: ruta del recurso solicitado
- *url_metodo*: consulta de filtrado

5.2.4 Forma de solicitudes

Para formar la petición de autenticación se tuvieron en cuenta los siguientes requisitos:

Tabla 2. Forma de solicitud de autenticación.

Requerimiento	Observaciones
El tipo de respuesta admitido es JSON	Header Accept: application/json
El medio de representación asociada es JSON	Header Content-Type: application/json
El cuerpo de solicitud debe ser un JSON formado por el login WSC y clave WSC	"{"login": \" + login_WSC + "\", \"clave\": \" + clave_WSC + "\"}"
La url está conformada por la url api token y el path de servicio "auth"	url_solicitud_metodo = url_api_token + "auth"

Para realizar la petición Post Obj se tuvieron en cuenta los siguientes requerimientos:

Tabla 3. Forma de solicitud Post Obj.

Requerimiento	Observaciones
El tipo de respuesta admitido es JSON	Header Accept: application/json
El medio de representación asociada es JSON	Header Content-Type: application/json

Se debe tener en cuenta el identificador de la empresa registrada	Header empresa_id
Se debe establecer el Header Authorization	Header Authorization: Token de autenticación
El cuerpo de solicitud debe ser un objeto tipo String formado por los parámetros requeridos por la solicitud	La librería Transporte Objetos de Celeste transforma los parámetros en una cadena como cuerpo de solicitud.
La url del servicio debe estar conformada por la url api, el path de servicio y la url del método	url_api + path_servicio + URLMetodo

5.3 Adaptación del marco de trabajo para la capa de datos

5.3.1 Definición de diseño

Para desarrollar el marco de trabajo de la capa de datos del cliente Android fue necesario definir la biblioteca de Cliente Http. En el contexto Android era posible utilizar OkHttp, Retrofit, Volley, entre otros.

Actualmente una de las bibliotecas más usadas para Android es Retrofit. Esta biblioteca permite realizar peticiones al servidor: *GET, POST, PUT, PATCH, DELETE y HEAD*, y gestionar diferentes tipos de parámetros, convirtiendo automáticamente la respuesta a un tipo de dato.

Algunas características de Retrofit son:

- Aísla el manejador de peticiones en una interfaz
- Usa anotaciones descriptivas (parámetros, cuerpo, cabeceras, etc.)
- Permite enviar las peticiones de manera asíncrona
- Realiza una conversión automática de JSON
- Admite diferentes formatos de datos

Por otra parte, Volley es una biblioteca HTTP optimizada para la gestión de peticiones Android ya que permite el manejo **asíncrono** de las llamadas de red y se integra fácilmente con cualquier protocolo, además, incluye compatibilidad con Strings sin procesar, imágenes y JSON.

Para la definir la biblioteca para el cliente HTTP fue necesario implementar prototipos en los que se validaron los pros y contras de las bibliotecas mencionadas anteriormente. Se determinó que Volley era apropiada para la implantación, ya que esta biblioteca:

- Permite ejecutar automáticamente las solicitudes HTTP en subprocesos (segundo plano) lo que evita que sea bloqueado el hilo principal asociado a la interfaz de usuario garantizando una experiencia fluida en la aplicación.
- Proporciona un mecanismo de manejo de errores que permite gestionar errores de red, errores de servidor y otros problemas de la comunicación REST. Además, realiza intentos automáticos en caso de fallas temporales.
- Almacena en memoria caché las respuestas de las solicitudes REST. Esto puede mejorar el rendimiento de la aplicación ya que permite un acceso rápido a datos previamente cargados.
- Ofrece una interfaz sencilla e intuitiva que simplifica la implementación de comunicaciones REST para Android.
- Facilita la configuración personalizada de las solicitudes.

La gestión de Volley comienza con la generación de un pool de conexiones, las cuales son priorizadas para su ejecución; Luego son seleccionadas por el elemento Cache Dispatcher que permite validar si la respuesta de la petición a realizar se encuentra almacenada en caché debido a solicitudes previas. En caso positivo se procesa la respuesta y se presenta en la interfaz de usuario. En caso negativo se pasa la conexión al pool de peticiones pendientes, en este punto, el elemento Network Dispatcher selecciona las peticiones de la cola para realizar la transacción HTTP, las respuestas obtenidas se almacenan en caché y se procesan para presentarse en la interfaz de usuario, como indica la Figura 18.

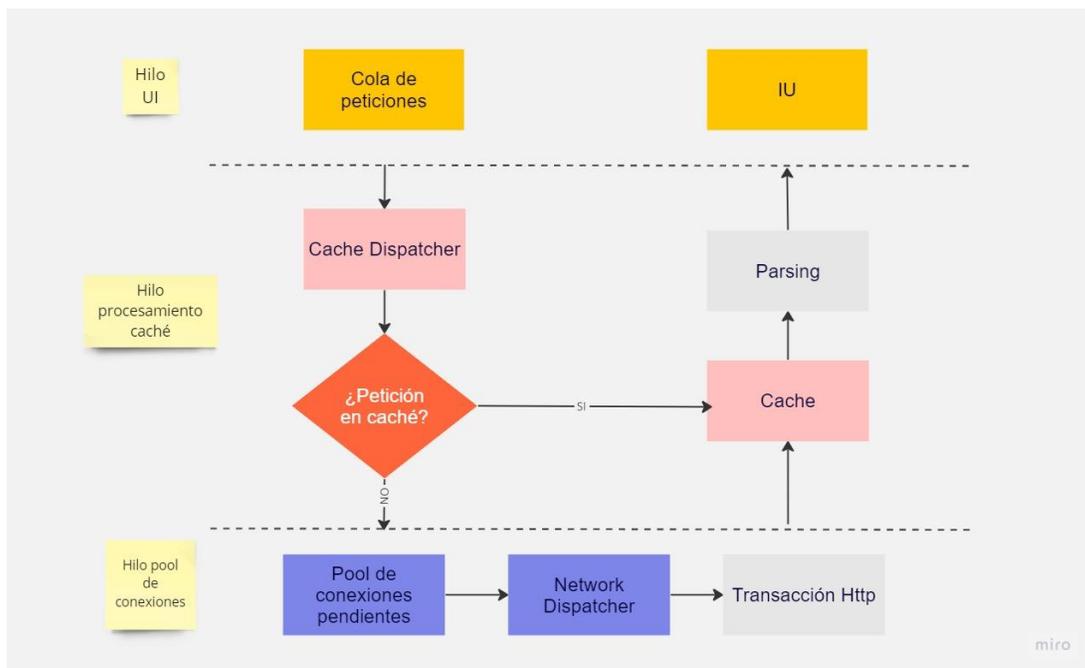


Figura 18. Gestión de peticiones con Volley.
Fuente: Realización propia.

Luego de seleccionar la biblioteca del cliente HTTP, se definió la manera en la que se gestionarían los flujos de datos. Para implementar la programación reactiva se debe considerar el lenguaje de desarrollo, además, se debe tener en cuenta que para este caso el flujo de datos es asíncrono.

En Android existe una variedad de bibliotecas que permiten gestionar la reactividad, RxJava, RxAndroid, LiveData, Coroutines, entre otros. Android proporciona la clase AsyncTask que maneja la ejecución en segundo plano y provee métodos que permiten actualizar la interfaz de usuario desde el hilo principal al completarse la tarea.

Entre las posibles opciones se consideró que, para el lenguaje Java es recomendable el uso de RxJava y RxAndroid (extensión de RxJava para aplicaciones Android) ya que a comparación de la biblioteca AsyncTask, RxJava brinda una mayor flexibilidad en el manejo de flujos de datos continuos, manejo de errores y gestión de memoria [16]. Por otra parte, LiveData está asociado a la gestión de interfaz de usuario a través de MVVM, por lo tanto, se complementa con el manejo de RxJava. Las Coroutines se han descartado ya que son una característica introducida en el lenguaje de programación Kotlin.

RxJava permite la gestión de aplicaciones asíncronas basadas en eventos a través de la implementación del patrón Observable. En este patrón los Observables representan la fuente de datos o eventos que pueden ser observados por uno o varios Observadores[16]. Para este proceso es necesario tener en cuenta las siguientes características:

- **Observable:** es el encargado de emitir un flujo de datos o eventos
- **Operator:** métodos que permiten convertir o modificar los flujos de datos.
- **Observer:** es el consumidor de los datos o eventos (errores y notificación de eventos finalizados) emitidos por el Observable

En conclusión, para usar RxJava se crean Observables, se transforman esos Observables, de acuerdo con las necesidades, para obtener los datos precisos y luego se observan estas secuencias de elementos con el fin de brindar un manejo a los datos o eventos recibidos.

Para definir el hilo en el que se ejecutarán las operaciones asíncronas, RxJava usa una abstracción definida Scheduler que controla el tiempo que consumen diferentes subprocesos con el fin de evitar bloquear el hilo principal de la interfaz de usuario.

- **Scheduler.io:** es uno de los tipos más usados, se usa generalmente para solicitudes de red u operaciones del sistema de base de datos. Utiliza un grupo de hilos reutilizables para ejecutar tareas, si un hilo no se puede reutilizar genera un nuevo subproceso.

- **Scheduler.computation:** está respaldado por un grupo de hilos en donde la cantidad de subprocesos que se pueden usar depende de la cantidad de núcleos presentes en el sistema.
- **Scheduler.NewThread:** genera un nuevo hilo para cada Observable activo, es recomendable usarlo cuando se quiere liberar operaciones que consumen mucho tiempo del subproceso principal a otro subproceso.

Además, RxAndroid facilita el desarrollo para aplicaciones Android con RxJava ya que proporciona integraciones y utilidades específicas de Android que permiten trabajar de una manera más apropiada, tales como:

- **AndroidSchedulers:** maneja un conjunto de Schedulers para gestionar los hilos específicamente de Android (UI Thread e hilos de segundo plano).
- **Looper:** es el responsable de procesar mensajes en el hilo principal gestionados específicamente para Android.

5.4 Implementación

A continuación, se presentan los procesos implementados de acuerdo con los procesos descritos en la sección anterior.

5.4.1 Pila de peticiones para el contexto Android

Primero se implementó la solicitud POST que define una interfaz de respuesta donde el tipo de Observable definido es **Single** (Tipo de Observable RxJava). Este Observable emite un único valor como respuesta a una solicitud de red. Emite OnSuccess para los casos en los que la petición se ha realizado con éxito y OnError si se presenta un error. Asimismo se implementaron los métodos para obtener token y establecer la petición base Post Obj, como indica la Figura 19.

La petición Post General configura la pila de peticiones (que es una estructura que administra y organiza las solicitudes de red mediante el objeto RequestQueue), los Headers, el cuerpo de solicitud y la URL para la petición de autenticación y la petición POST general.

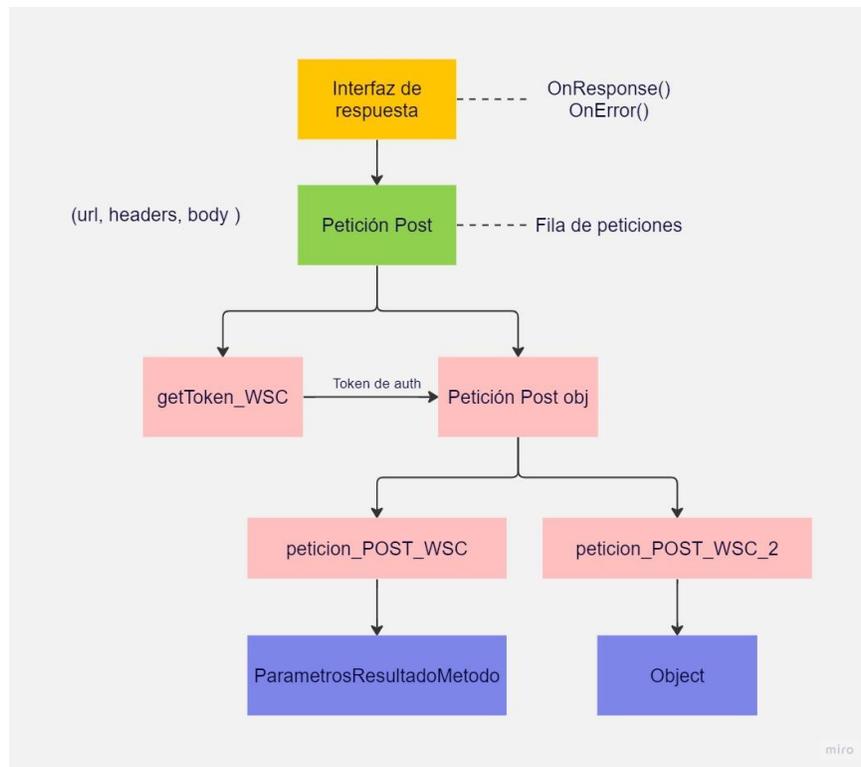


Figura 19. Pila de peticiones del marco de trabajo para el cliente Android.
Fuente: Realización propia.

Los mensajes de red se gestionaron a través de la clase VolleyError (recibida en el caso que se presente un error en la solicitud), esto permite validar de manera específica los diferentes tipos de errores, tal como indica la Tabla 4.

Tabla 4. Descripción de los mensajes de red obtenidos en una petición.

VolleyError	Descripción
NetworkError	Indica que se presentó un error de red al realizar la solicitud
ServerError	Indica que el servidor ha respondido con una respuesta error
AuthFailureError	Se presenta cuando se obtiene un error de autenticación al realizar la solicitud
ParseError	Indica que no se pudo analizar la respuesta del servidor
TimeoutError	Indica que se ha agotado el tiempo de espera de la conexión

5.4.2 Peticiones generales para el contexto Android

Con las peticiones base se procedió a implementar las peticiones generales. Estas peticiones permiten ejecutar cualquier tipo de solicitud asociada a las funciones del sistema (componentes y tipos de respuesta). Es necesario recalcar que, para realizar las peticiones requeridas en la implementación de la aplicación móvil, se debe acceder directamente a las peticiones generales, la cual ejecuta las tareas correspondientes, como indica la siguiente figura:

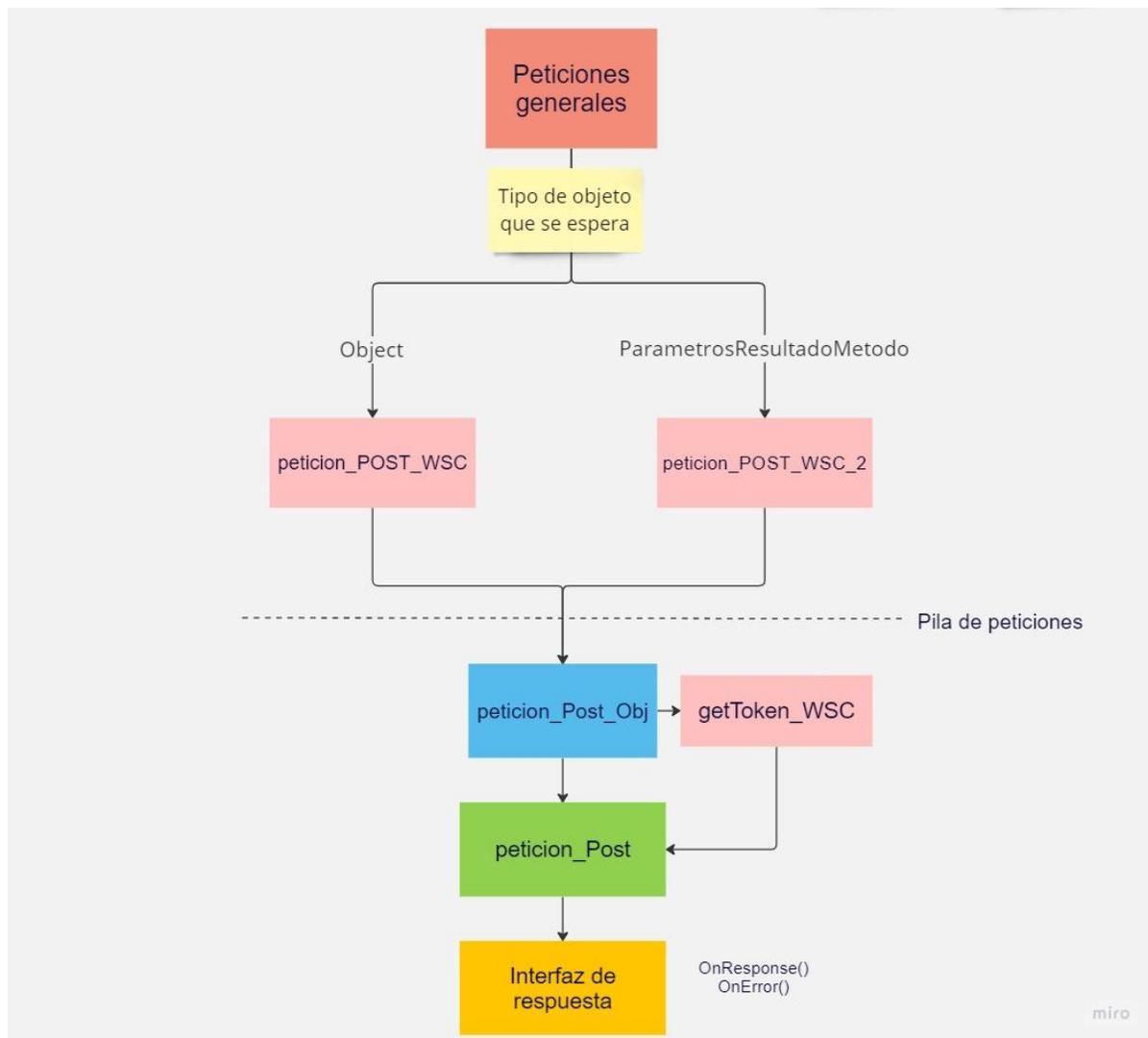


Figura 20. Procesos para realizar una petición general.
Fuente: Realización propia.

Cada una de las peticiones generales debe cumplir con los siguientes requerimientos:

Tabla 5. Requerimientos para ejecutar una petición general.

Requerimiento	Observaciones
Se debe validar la versión y la sesión del cliente	Cada una de las peticiones valida si en el evento obtenido se presentan errores de versión o una sesión inválida. Esto permite emitir un mensaje claro y preciso en el cliente.
Deben emitir el error de la petición de acuerdo con la falla presentada	Se propaga el error desde la petición base.

A continuación, se presentan algunas de las características que se tuvieron en cuenta para la implementación de las peticiones generales.

Los `metodoGeneral` y `metodoResultado` permiten realizar las peticiones que no requieren configuraciones adicionales, para el caso del método general la respuesta es un objeto tipo `ParametrosResultadoMetodo`, para el caso del método resultado se espera un objeto tipo `Object`, cada uno de ellos se gestionó con Rxjava.

Por otro lado, para la gestión del componente de tablas (componente que maneja la presentación y edición de los datos) se tuvo en cuenta el tipo de dato a obtener, como se indica en la siguiente tabla:

Tabla 6. Descripción de los componentes tipo tabla del sistema Celeste.

Componente	Tipo de dato a obtener
Tabla edición	<code>ParametrosTablaEdicion</code>
Autocompletado de datos	<code>ParametrosAutocompletadoDatos</code>
Inicialización de una tabla de búsqueda	<code>ParametrosInicializacionTablaBusqueda</code>
Carga de datos de la tabla de búsqueda	Se espera un arreglo de objetos tipo <code>Object[]</code>

Adicionalmente, para el manejo de formularios o carga de datos es posible utilizar los métodos encargados de cargar la interfaz, actualizar, obtener y eliminar parámetros. Por último, para el ingreso al sistema fue necesario determinar el id de sesión y la clave pública, estos procesos se almacenan en el método `ingresar`.

El proceso de las clases asociadas en la adaptación del marco de trabajo, es posible apreciarlo en la sección de Anexos – anexo 1.

5.5 Proceso para realizar una solicitud al servidor

Para realizar solicitudes al servidor, el cliente debe crear una instancia Singleton de la clase `Conexiones_WSC` para poder gestionar las peticiones. Seguido a esto, es necesario definir las características de la petición a través de la configuración Rest (Url método, Path del servicio, url de la api y credenciales de autenticación). Este procedimiento se realiza una única vez teniendo en cuenta dos flujos, clientes registrados en el sistema y clientes no registrados.

Cumpliendo los anteriores requisitos, el cliente sólo debe acceder a las peticiones generales e invocar el método que sea requerido para obtener la respuesta a su solicitud, como indica la Figura 21.

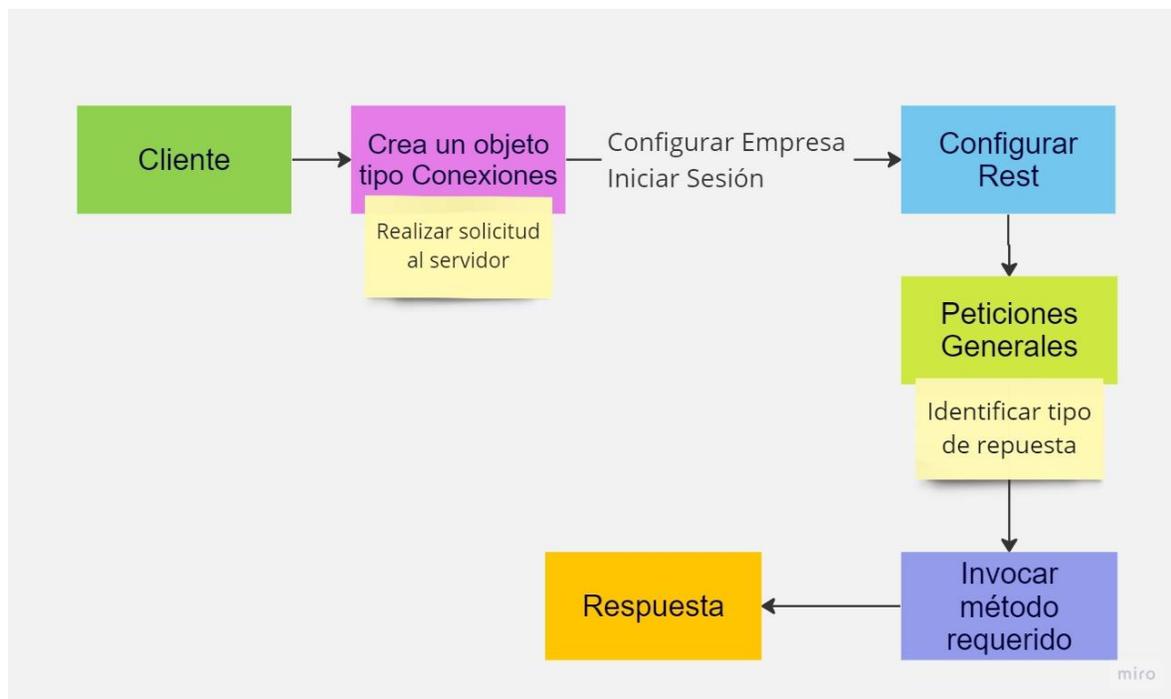


Figura 21. Flujo de procesos para realizar una petición al servidor.
Fuente: Realización propia.

6 Capítulo VI. Implementación de la capa de datos para el módulo de ventas

En este capítulo se presenta la implementación de la capa de la capa de datos del módulo de Ventas, el marco de trabajo utilizado para la implementación y resultados.

6.1 Marco de trabajo para la implementación del módulo de ventas

El marco ágil permite organizar y ejecutar tareas de forma estructurada, lo que garantiza que un producto puede ser entregado de forma rápida y cumpla con una alta calidad, este marco se deriva del manifiesto ágil que se origina de la consigna de tener la capacidad de crear y responder al cambio, priorizando la autoorganización y la colaboración.

SCRUM es un marco de trabajo iterativo e incremental para el desarrollo de proyectos y se estructura en ciclos de trabajo llamados Sprints. Éstos son iteraciones que se suceden una detrás de otra. Al comienzo de cada Sprint, el equipo multifuncional selecciona los elementos (requisitos del cliente) de una lista priorizada. Se comprometen a terminar los elementos al final del Sprint. Durante el Sprint no se pueden cambiar los elementos elegidos. Al final del Sprint, el equipo lo revisa con los interesados en el proyecto, y les enseña lo que han construido.

Los eventos Scrum o ceremonias permiten llevar un registro, control y organización del proceso de desarrollo de inicio a fin.

- **Evento Sprint:** permite definir claramente el incremento de un producto o entregable, este evento está presente en toda la duración del proyecto ya que engloba al resto de eventos.
- **Spring Planning:** permite planificar el trabajo del Sprint, se realiza al principio de cada uno ya con el objetivo de definir lo que se va a hacer y cómo se va a hacer.
- **Daily Scrum:** su objetivo es ordenar e informar sobre el avance y pendientes de las tareas definidas, en este evento se evalúa el Sprint global.
- **Sprint Review:** este evento se comparte el incremento realizado por el Scrum Team con el fin de tomar decisiones oportunas sobre el producto terminado.
- **Sprint Retrospective:** en la retrospectiva se analiza el último Sprint realizado con el fin de generar retroalimentación que permita mejoras para los siguientes Sprints.

6.1.1 Asignación de Roles

A continuación, se presenta la lista de los roles desempeñados en el desarrollo del presente proyecto:

Tabla 7. Descripción de la asignación de roles Scrum.

Personas	Rol	Característica
Javier Castillo	Product Owner	<ul style="list-style-type: none">• Desarrollar y comunicar claramente el Objetivo del Producto.• Comunicar claramente los elementos del Product Backlog.• Asegurar que el Product Backlog sea transparente, visible y comprensible.
Jhon Narvaez	Scrum Master	<ul style="list-style-type: none">• Entrenar a los miembros del equipo.• Ayudar al Equipo a enfocarse en crear Incrementos de alto valor.• Asegurarse de que todos los eventos de Scrum se lleven a cabo y sean positivos, productivos y se mantengan dentro del marco de tiempo.
<ul style="list-style-type: none">• Andres Quemba• Juan Cruz• Jhulieth Dorado	Scrum Team	<ul style="list-style-type: none">• Crear el Sprint Backlog.• Ejecutar las tareas cada día hacia el Sprint Goal.• Responsabilizarse unos a otros como profesionales.

6.1.2 Modelo de Historia de Usuario

Las historias de usuario permiten describir los requerimientos de un cliente en pequeñas unidades de trabajo. Para escribir una HU no se ha definido un estándar, de hecho, sólo se ha establecido una propuesta de formato creada por Mike Cohn:

"yo como usuario
deseo esta funcionalidad
para este beneficio"

Muchos expertos recomiendan que, "cualquier modo de escribir o representar las Historias de Usuario sirve, siempre y cuando el Product Owner y el Equipo de Trabajo tengan la misma imagen mental de lo que se está requiriendo construir"[17].

De este modo, el equipo ha definido la descripción de las historias de usuario solamente con el título, esto requiere un nivel de madurez alto de parte del equipo y el Product Owner.

Nivel de Madurez del Equipo y del Dueño de Producto:
alto

Ejemplo:



Figura 5. Ejemplo de HU – Modo sólo título[17].

Para una adecuada gestión del proyecto se hará uso de la herramienta Jira, en esta herramienta es posible definir una historia de usuario con su respectivo título, descripción, prioridad, responsable, Sprint, identificador (generado automáticamente), puntos estimados por historia, estado, entre otros.

6.1.3 Product Backlog

A partir de los módulos identificados anteriormente, se construyó el Product Backlog, el cual representa las historias de usuario asociadas a la implementación, tal como se presentan en la Figura 22

AMC2022-76	Visualizar datos en un formulario	GESTIONAR FORMULARIOS
AMC2022-77	Cargar datos en un formulario	GESTIONAR FORMULARIOS
AMC2022-78	Ver errores en un formulario	GESTIONAR FORMULARIOS
AMC2022-4	Solicitar código de verificación para configurar empresa	CONFIGURAR NUEVA EMPRESA
AMC2022-12	Validar código de configuración	CONFIGURAR NUEVA EMPRESA
AMC2022-25	Visualizar empresas configuradas	CONFIGURAR NUEVA EMPRESA
AMC2022-27	Ingresar al sistema con credenciales	INICIO DE SESIÓN
AMC2022-30	Ingresar credenciales para configurar huella	CONFIGURAR HUELLA
AMC2022-29	Ingresar huella para verificar datos biométricos	CONFIGURAR HUELLA
AMC2022-31	ingresar al sistema al configurar huella	CONFIGURAR HUELLA
AMC2022-28	Ingresar al sistema con huella	INICIO DE SESIÓN
AMC2022-55	Visualizar alerta de pago de mensualidad en días de gracia	GESTIONAR MENSUALIDAD
AMC2022-56	Visualizar alerta de mensualidad vencida	GESTIONAR MENSUALIDAD
AMC2022-57	Ingresar código de pago de mensualidad	GESTIONAR MENSUALIDAD
AMC2022-54	Seleccionar la sucursal para ingresar al sistema	SUCURSALES
AMC2022-32	Visualizar menú	MENU
AMC2022-58	Buscar opción de sub menú	MENU
AMC2022-37	Ver tabla de terceros registrados	TERCEROS
AMC2022-59	Filtrar por tipo de tercero	TERCEROS
AMC2022-82	Buscar un tercero	TERCEROS
AMC2022-35	Registrar un nuevo tercero	TERCEROS

- AMC2022-36 Actualizar un tercero **TERCEROS**

- AMC2022-47 Ver tabla de productos registrados **PRODUCTOS**

- AMC2022-49 Buscar producto **PRODUCTOS**

- AMC2022-104 Asignar código de barras a un producto **PRODUCTOS**

- AMC2022-51 Crear nuevo producto **PRODUCTOS**

- AMC2022-50 Actualizar producto **PRODUCTOS**

- AMC2022-39 Ver lista de facturas de venta **VENTA**

- AMC2022-38 Buscar Factura de Venta **VENTA**

- AMC2022-60 Filtrar factura de Venta por opciones avanzadas **VENTA**

- AMC2022-83 Reimprimir Factura de Venta **VENTA**

- AMC2022-61 Asignar un tercero cliente a la FV **REGISTRAR FACTURA DE VENTA**

- AMC2022-62 Asignar un referido a la FV **REGISTRAR FACTURA DE VENTA**

- AMC2022-63 Cambiar resolución de una FV **REGISTRAR FACTURA DE VENTA**

- AMC2022-64 Cambiar fecha de una FV **REGISTRAR FACTURA DE VENTA**

- AMC2022-65 Agregar productos a una FV **REGISTRAR FACTURA DE VENTA**

- AMC2022-66 Editar la cantidad de cada producto agregada a la FV **REGISTRAR FACTURA DE VENTA**

- AMC2022-67 Cambiar el valor unitario a un producto en la FV **REGISTRAR FACTURA DE VENTA**

- AMC2022-68 Ver total de una FV **REGISTRAR FACTURA DE VENTA**

- AMC2022-105 Buscar una forma de pago **REGISTRAR FACTURA DE VENTA**

- AMC2022-69 Agregar una forma pago **REGISTRAR FACTURA DE VENTA**

- AMC2022-70 Añadir una observación **REGISTRAR FACTURA DE VENTA**

- AMC2022-85 Agregar una forma de entrega **REGISTRAR FACTURA DE VENTA**

- AMC2022-71 Visualizar los totales de una FV **REGISTRAR FACTURA DE VENTA**

- AMC2022-108 Registrar Factura de Venta **REGISTRAR FACTURA DE VENTA**

- AMC2022-106 Visualizar PDF de la FV realizada **REGISTRAR FACTURA DE VENTA**

- AMC2022-88 Ver tabla de cotizaciones registradas **COTIZACIÓN**

- AMC2022-89 Buscar cotización **COTIZACIÓN**

AMC2022-90	Aplicar filtro de búsquedas avanzadas	COTIZACIÓN
AMC2022-91	Reimprimir Cotización	COTIZACIÓN
AMC2022-43	Asignar tercero cliente a una cotización	REGISTRAR COTIZACIÓN
AMC2022-44	Asignar un tercero referido cotización	REGISTRAR COTIZACIÓN
AMC2022-107	Cambiar fecha de registro de cotización	REGISTRAR COTIZACIÓN
AMC2022-48	Agregar productos a CT	REGISTRAR COTIZACIÓN
AMC2022-72	Agregar una fecha de vencimiento a CT	REGISTRAR COTIZACIÓN
AMC2022-73	agregar días de vencimiento a CT	REGISTRAR COTIZACIÓN
AMC2022-111	Registrar Cotización	REGISTRAR COTIZACIÓN
AMC2022-110	Visualizar PDF de la cotización realizada	REGISTRAR COTIZACIÓN
AMC2022-95	Ver tabla de pedidos registrados	PEDIDOS
AMC2022-96	Buscar pedidos	PEDIDOS
AMC2022-97	Aplicar filtro de búsqueda avanzadas	PEDIDOS
AMC2022-98	Reimprimir pedido	PEDIDOS
AMC2022-99	Asignar tercero cliente a un pedido	REGISTRAR PEDIDOS
AMC2022-100	Asignar tercer referido a pedido	REGISTRAR PEDIDOS
AMC2022-101	Agregar productos a un pedido	REGISTRAR PEDIDOS
AMC2022-102	Agregar fecha de vencimiento del pedido	REGISTRAR PEDIDOS
AMC2022-103	Agregar días de vencimiento del pedido	REGISTRAR PEDIDOS
AMC2022-109	Registrar Pedido	REGISTRAR PEDIDOS
AMC2022-117	Visualizar PDF del pedido realizado	REGISTRAR PEDIDOS
AMC2022-113	Visualizar PDF	GESTIONAR PDF
AMC2022-114	Compartir PDF	GESTIONAR PDF
AMC2022-115	Imprimir PDF	GESTIONAR PDF
AMC2022-116	Descargar PDF	GESTIONAR PDF

Figura 22. Representación del Product Backlog.
Fuente: Realización propia

Una vez definida la lista de historias de usuario se procedió a determinar cada Sprint. Se obtuvo un total de (5) Sprints con un tiempo de ejecución para cada uno de (3) semanas, los cuales tienen un alrededor de 50 puntos por Sprint.

6.2 Sprint 1.

6.2.1 Sprint Planning

Para Sprint 1 se estableció el desarrollo de las tareas correspondientes a registro, inicio de sesión y gestión de formularios, del cual se obtuvo el Sprint Backlog de la Figura 23.

▼ Sprint 1 30 ene – 19 feb (11 incidencias)	
Manejar datos de formularios, configurar empresa e iniciar sesión con credenciales y huella.	
AMC2022-76	Ver información en un formulario GESTIONAR FORMULARIOS
AMC2022-77	Cargar datos en un formulario GESTIONAR FORMULARIOS
AMC2022-78	Ver errores en un formulario GESTIONAR FORMULARIOS
AMC2022-4	Solicitar código de verificación para configurar empresa CONFIGURAR NUEVA EMPRESA
AMC2022-12	Validar código de configuración CONFIGURAR NUEVA EMPRESA
AMC2022-25	Visualizar empresas configuradas CONFIGURAR NUEVA EMPRESA
AMC2022-27	Ingresar al sistema con credenciales INICIO DE SESIÓN
AMC2022-30	Ingresar credenciales para configurar huella CONFIGURAR HUELLA
AMC2022-29	Ingresar huella para verificar datos biométricos CONFIGURAR HUELLA
AMC2022-31	Ingresar al sistema al configurar huella CONFIGURAR HUELLA
AMC2022-28	Ingresar al sistema con huella INICIO DE SESIÓN

Figura 23. Sprint 1- Sprint Backlog.
Fuente: Realización propia

6.2.2 Sprint Execution

Épica Gestión de Formularios: La gestión de formularios permite controlar el flujo de datos a través de la interfaz gráfica de usuario. De esta manera es posible cargar los datos en un formulario, almacenar datos en los parámetros correspondientes, quitar campos con error, habilitar campos, validar campos y enviar mensaje de error, de manera automática.

Para la gestión de formularios se destacaron las siguientes tareas e historias de usuario:

Tabla 8. Descripción HU 76 - Ver información en un formulario.

HU 76: Ver información en un formulario	
Tareas	Descripción
Implementar prototipos con Reflection Java.	En esta incidencia se crearon prototipos que permitieran acceder a los campos de interfaz de usuario y clases parámetro en tiempo de ejecución a través de Reflection Java.
Asignar a cada uno de los campos de una interfaz de usuario el valor de los parámetros obtenidos	Se implementó la funcionalidad que permite cargar en un formulario los datos requeridos de una petición.

Tabla 9. Descripción de la HU 77 - Cargar datos en un formulario.

HU 77: Cargar datos en un formulario	
Tareas	Descripción
Almacenar los datos cargador	En esta tarea se implementó la funcionalidad que permite almacenar los datos de cada campo del formulario en un atributo de acuerdo con el tipo de dato.
Validar formato	Para almacenar los datos, fue necesario validar que el tipo de formato ingresado correspondiera al tipo de variable.
Validar campos obligatorios	En esta tarea se debía validar que todos los campos obligatorios fueran ingresados.
Marcar campos con error	De acuerdo con las validaciones, se implementó la funcionalidad de marcar los campos que tuvieran errores.
Limpiar campos	En esta tarea, cada uno de los campos se limpian o se definen en su valor por defecto al realizar una acción determinada.

Épica configuración empresa: permite al usuario configurar una empresa en un dispositivo móvil solicitando un código de verificación. Para solicitar el código de activación el usuario ingresa el NIT de la empresa, si este NIT es válido, se envía un código de activación al correo registrado del administrador del sistema. Cada solicitud de configuración registra información del dispositivo (identificador único e información del sistema).

En esta sección se destacaron las siguientes tareas e historias de usuario:

Tabla 10. Descripción de la HU 4 - Solicitar código de verificación para configurar empresa.

HU 4: Solicitar código de verificación para configurar empresa	
Tareas	Descripción
Asignar identificador único de dispositivo.	En esta incidencia se implementó la funcionalidad que permite generar un código único de dispositivo y almacenarlo en una base de datos local, esto implica gestionar base de datos local (se utilizó la biblioteca de persistencia Room).
Generar información de dispositivo	En esta incidencia se generó la información de dispositivo (API, modelo, sistema operativo) los cuales son requeridos en la solicitud de configuración empresa.
Realizar petición de configuración empresa	Se implementó la invocación de las peticiones generales y configuración REST para obtener la respuesta a la solicitud.

Tabla 11. Descripción de la HU 12 - Validar código de configuración.

HU 12: Validar Código de configuración	
Tareas	Descripción
Validar código	En esta incidencia se implementa la funcionalidad que permite verificar que el código ingresado para configurar empresa sea válido.

Registrar datos de empresa configurada	Se implementó la funcionalidad que permite registrar los datos obtenidos de la configuración de empresa, los cuales son almacenados en una base de datos local (número de referencia, nombre empresa, login y clave WSC, e identificador de empresa)
--	--

Épica configuración huella: La configuración de huella permite asociar las credenciales biométricas de un usuario con las credenciales de acceso al sistema. En esta sección se destacaron las siguientes tareas e historias de usuario.

Tabla 12. Descripción de la HU 30 - Ingresar credenciales para configurar huella.

HU 30: Ingresar credenciales para configurar huella	
Tareas	Descripción
Verificar credenciales	Se implementó la funcionalidad que permite verificar el ingreso al sistema de un usuario con sus respectivas credenciales. En este caso se acceden a los recursos almacenados en la base de datos local, los cuales contienen, por ejemplo, la clave y login WSC.
Generar flujo de encriptación de datos	En esta incidencia se estableció el flujo de encriptación, en el cual se incorporó el algoritmo de cifrado Cipher utilizando una SecretKey.

Tabla 13. Descripción de la HU 29 - Ingresar huella para configurar datos biométricos.

HU 29: Ingresar huella para configurar datos biométricos	
Tareas	Descripción

Encriptar y codificar credenciales asociadas a datos biométricos	Se implementó la funcionalidad que permite encriptar los datos biométricos relacionados con las credenciales de inicio de sesión, además, se codificaron con la librería interna de transporte objetos.
Almacenar datos encriptados y codificados	Se almacenaron en una base de datos local, los datos asociados al proceso de encriptación y requeridos para el inicio de sesión con huella.

Épica inicio de sesión: El módulo inicio de sesión le permite al usuario ingresar al sistema digitando las respectivas credenciales o huella. En esta épica se destacaron las siguientes tareas e historias de usuario.

Tabla 14. Descripción de la HU 28 - Ingresar al sistema con huella.

HU 28: Ingresar al sistema con huella	
Tareas	Descripción
Control de verificación de datos biométricos	En esta incidencia se verificó que la huella ingresada estuviera asociada al dispositivo.
Obtener datos asociados a la huella ingresada	En esta tarea se decodificaron y desencriptaron los datos asociados a la huella ingresada para realizar el proceso de inicio de sesión.

6.2.3 Sprint Review

Las historias seleccionadas para el desarrollo del Sprint fueron terminadas con éxito. La historia de usuario en la que se presentaron mayores dificultades fue la HU 30 - ingresar huella para configurar datos biométricos, ya que fue necesario realizar una investigación de las técnicas y procesos asociados al flujo de cifrado y encriptación.

6.2.4 Sprint Retrospective

- El ritmo de desarrollo para la ejecución de este Sprint fue muy bueno, pero puede ajustarse, por lo cual se propuso asignar más puntos por Sprint, aproximadamente 45-50 puntos.
- La invocación de los métodos relacionados con las solicitudes al servidor se pudo llevar a cabo de manera sencilla e intuitiva.
- El desarrollo de los métodos relacionados a la gestión de formularios se implementó en un proyecto aparte, en el cual se desarrollan todas las tareas generales asociadas al flujo de implementación.

6.3 Sprint 2.

6.3.1 Sprint Planning

Para el Sprint 2 se definió el desarrollo de las tareas correspondientes a validaciones de inicio de sesión, menú y gestión de terceros, del cual se obtuvo el Sprint Backlog de la Figura 24.

▼ Sprint 2 20 feb – 13 mar (11 incidencias)	
Realizar validaciones de inicio de sesión para mostrar Menu y Gestionar Terceros	
AMC2022-55	Visualizar alerta de pago de mensualidad en días de gracia GESTIONAR MENSUALIDAD
AMC2022-56	Visualizar alerta de mensualidad vencida GESTIONAR MENSUALIDAD
AMC2022-57	Ingresar código de pago de mensualidad GESTIONAR MENSUALIDAD
AMC2022-54	Seleccionar la sucursal para ingresar al sistema SUCURSALES
AMC2022-32	Visualizar menú MENU
AMC2022-58	Buscar opción de sub menú MENU
AMC2022-37	Ver tabla de terceros registrados TERCEROS
AMC2022-59	Filtrar por tipo de tercero TERCEROS
AMC2022-82	Buscar un tercero TERCEROS
AMC2022-35	Registrar un nuevo tercero TERCEROS
AMC2022-36	Actualizar un tercero TERCEROS

Figura 24. Sprint 2 - Sprint Backlog
Fuente: Realización propia

6.3.2 Sprint Execution

Épica Gestionar Mensualidad: La gestión de mensualidad permite validar si un usuario tiene un pago vigente, está en periodo de gracia o suspendido. De esta manera es posible determinar si se permite el ingreso del usuario (vigente o periodo de gracia).

Para la gestión de mensualidad se destacaron las siguientes tareas e historias de usuario:

Tabla 15. Descripción de la HU 55 - Visualizar alerta en días de periodo de gracia.

HU 55: Visualizar alerta en días de periodo de gracia	
Tareas	Descripción
Verificar vigencia	Se implementó la funcionalidad que permite verificar la vigencia de uso del sistema.
Obtener días de gracia	En esta incidencia se debía obtener los días de gracia que tiene el usuario para generar el pago antes de ingresar a periodo de suspensión.

Épica Sucursales: Las sucursales son sedes o secciones en las cuales se divide una empresa. En esta épica se valida si una empresa tiene asignada más de una sucursal, con el objetivo de seleccionar una de ellas para iniciar sesión.

De la épica sucursales se destacaron las siguientes tareas e historias de usuario:

Tabla 16. Descripción de la HU 54 – Seleccionar sucursal para ingresar al sistema.

HU 54: Seleccionar sucursal para ingresar al sistema	
Tareas	Descripción
validar sucursales	Se implementó la funcionalidad que permite validar si una empresa tiene asignada más de una sucursal.

Asignar sucursal para iniciar sesión	Se implementó la funcionalidad que permite determinar en los parámetros de inicialización la sucursal con la cual se desea iniciar sesión. Si sólo hay una por empresa, se debe asignar esta por defecto.
--------------------------------------	---

Épica Menú: La gestión del Menú permite visualizar las opciones de menú y submenú de acuerdo con los permisos de usuario. Además, permite configurar estilos. Es necesario recalcar que, para este proceso se implementaron anotaciones con el fin de identificar las interfaces de usuario que estaban asociadas a una opción de submenú.

Para la gestión del Menú se destacó la siguiente Historia de Usuario:

Tabla 17. Descripción de la HU 32 - Visualizar menú.

HU 32: Visualizar menú	
Tareas	Descripción
Definir anotaciones	En esta tarea se crearon las anotaciones de menú y submenú con cada uno de sus atributos. Las anotaciones se definieron en tiempo de ejecución.
Obtener clases anotadas	Se implementó la funcionalidad que permite obtener las clases anotadas a través de archivos de texto. Es decir, se crearon dos archivos, uno con las clases que usaron la anotación de menú y otro con las que usaron la de submenú.
Implementar procesador de anotaciones	En esta incidencia se gestionó cada una de las anotaciones y se realizaron validaciones de las clases anotadas.

Obtener Menú	Se implementó la funcionalidad que permite leer el archivo con las clases anotadas de menú para acceder a la notación y crear el arreglo de opciones de menú con sus respectivas características (id, nombre, estilo, permisos)
Obtener Submenú	Se implementó la funcionalidad que permite leer el archivo con las clases anotadas de submenú, para acceder a la notación y crear el arreglo de opciones de menú con sus respectivas características (identificador, identificador de menú, nombre, estilo, permisos)

Gestionar Terceros: La gestión de terceros permite visualizar, actualizar, agregar y filtrar un tercero. De esta época se destacaron las siguientes Historias de Usuario:

Tabla 18. Descripción de la HU 37 - Ver tabla de terceros registrados.

HU 37: Ver tabla de terceros registrados	
Tareas	Descripción
Obtener parámetros búsqueda (Inicializar Tabla)	En esta tarea se obtuvo el objeto parámetros asociado a la búsqueda de terceros.
Actualizar Tabla con los terceros registrados	Se implementó la funcionalidad que permitía obtener los terceros registrados a partir del parámetros de búsqueda obtenido en la inicialización de la tabla.

Tabla 19. Descripción de la HU 35 - Registrar Tercero.

HU 35: Registrar Tercero	
Tareas	Descripción
Gestionar peticiones de llenado de cada componente del formulario	Se implementó la funcionalidad que permite llenar cada uno de los campos o componentes requeridos. Por ejemplo, combo de países, municipios y departamentos.
Validar datos del formulario	En esta tarea se realizaron las validaciones de los datos ingresados asociadas al servidor.
Registrar tercero	Se implementó la funcionalidad que permite realizar el registro de un nuevo tercero.

6.3.3 Sprint Review

Las historias seleccionadas para el desarrollo del Sprint 2 se cumplieron en su totalidad con éxito. La historia de usuario que involucró mayor peso del asignado fue HU 34 - Visualizar Menú ya que implicó el procesamiento de anotaciones.

6.3.4 Sprint Retrospective

Al finalizar la ejecución del Sprint 2 se realizó el Sprint Retrospective en el cual se destacaron los siguientes aspectos:

- La modificación del ritmo de desarrollo para la ejecución de este Sprint fue acertada, ya que se pudo cumplir con cada una de las tareas.
- En este Sprint se evidenció la correcta funcionalidad del marco de trabajo de datos puesto que fue posible obtener una amplia lista de terceros y el flujo de trabajo fue rápido. Además, se observaron los mensajes de error en los casos que no se tuvo acceso a internet o no existiera conexión al servidor.

6.4 Sprint 3

6.4.1 Sprint Planning

Para el Sprint 3 se desarrollaron las tareas correspondientes a la gestión de productos, búsquedas de facturas de venta y PDF de documentos de venta. Se obtuvo el Sprint Backlog de la Figura 25.

▼ Sprint 3 13 mar – 3 abr (13 incidencias)	
Gestionar productos, búsquedas de Facturas de Venta y PDF de documentos de venta	
AMC2022-47	Ver tabla de productos registrados PRODUCTOS
AMC2022-49	Buscar producto PRODUCTOS
AMC2022-104	Asignar código de barras a un producto PRODUCTOS
AMC2022-51	Crear nuevo producto PRODUCTOS
AMC2022-50	Actualizar producto PRODUCTOS
AMC2022-39	Ver lista de facturas de venta VENTA
AMC2022-38	Buscar Factura de Venta VENTA
AMC2022-60	Filtrar factura de Venta por opciones avanzadas VENTA
AMC2022-83	Reimprimir Factura de Venta VENTA
AMC2022-113	Visualizar PDF GESTIONAR PDF
AMC2022-114	Compartir PDF GESTIONAR PDF
AMC2022-115	Imprimir PDF GESTIONAR PDF
AMC2022-116	Descargar PDF GESTIONAR PDF

Figura 25. Sprint 3 - Sprint Backlog

Fuente: Realización propia

6.4.2 Sprint Execution

Productos: La gestión de productos le permite al usuario visualizar, buscar, actualizar y registro un producto.

En esta sección se destacaron las siguientes tareas e Historias de Usuario:

Tabla 20. Descripción de la HU 35 - Registrar Producto.

HU 35: Registrar Producto	
Tareas	Descripción
Gestionar componentes	Se implementó la funcionalidad que permite llenar cada uno de los campos o componentes que requieren ejecutar una solicitud al servidor (combo tipo de producto, combo iva, combo impuesto al consumo). En este caso fue necesario analizar los diferentes operadores (zip, flatmap) de RxJava con el fin de mejorar el proceso de solicitudes debido al número de componentes agregados.
Validar datos del formulario	Se implementó la funcionalidad que permite realizar validaciones de los datos ingresados.
Registrar Producto	Se implementó la funcionalidad que permite realizar el registro de un nuevo producto.

Tabla 21. Descripción de la HU 36 - Asignar código de barras a un producto.

HU 36: Asignar código de barras a un producto	
Tareas	Descripción
Determinar biblioteca de lectura de códigos de barra y QR.	Se implementaron prototipos que permitieran determinar la biblioteca de lectura de códigos de barra y QR. De las posibles opciones se seleccionó Zxing ya que permite leer múltiples tipos de códigos y es compatible con diferentes versiones de sistemas operativos.

Obtener la lectura del código de barras	Se implementó la funcionalidad que permite obtener la lectura de un código de barras o QR para asignar o buscar un producto.
---	--

Gestionar PDF: El proceso de gestión de PDF le permite a un usuario visualizar, imprimir, compartir y descargar un archivo de extensión *.PDF*.

En esta sección se destacó la siguiente Historias de Usuario:

Tabla 22. Descripción de la HU 113 - Visualizar PDF.

HU 113: Visualizar PDF	
Tareas	Descripción
Obtener PDF del documento de Venta	Se implementó la funcionalidad que permite obtener el PDF del documento solicitado. En esta sección se generalizó la petición para cualquier tipo de documento de Venta.
Almacenar PDF	En esta tarea se implementó la funcionalidad que permite leer y almacenar un PDF para que sea posible visualizarlo.

Ventas: El proceso de ventas permite el registro y búsqueda de facturas de ventas. El punto clave de esta sección es el manejo adecuado de todos los módulos relacionados y los permisos o configuraciones relacionadas y de esta manera realizar una factura de venta correctamente.

En la implementación de esta épica se destacó la Historia de Usuario:

Tabla 23. Descripción de la HU 60 - Filtrar factura de venta por opciones avanzadas.

HU 60: Filtrar Factura de Venta por opciones avanzadas	
Tareas	Descripción

Gestionar datos de filtrado	En esta tarea se implementó la funcionalidad de almacenar cada uno de los datos de filtrado y generalizarlo para el funcionamiento de los búsquedas.
Solicitar información filtrada	En esta incidencia se gestionó la petición de búsqueda para los datos filtrados.

6.4.3 Sprint Review

Las historias seleccionadas para el desarrollo del Sprint 3 se cumplieron en su totalidad. Es necesario recalcar que, el proceso de registro de un producto se relacionaba al proceso de registro de Terceros y esto permitió un rápido manejo de las tareas relacionadas con la Gestión de Productos. La historia de usuario en la que se presentó mayor dificultad fue HU 83 - Reimprimir Factura de Venta, ya que implicó el proceso de generalización para los diferentes documentos que se pueden reimprimir.

6.4.4 Sprint Retrospective

Al finalizar la etapa anterior se realizó el Sprint Retrospective del cual se destacó el adecuado manejo del tiempo de ejecución. Se puntualizó que las siguientes iteraciones pueden presentar algunas dificultades ya que se inicia el proceso de Facturación y registro de documentos de Venta. Estos procesos requieren una adecuada configuración y gestión de los componentes asociados al documento de Venta.

6.5 Sprint 4.

6.5.1 Sprint Planning

En la ejecución del Sprint 4 se desarrollaron las tareas correspondientes a la gestión del registro de documentos de Venta. Se debe tener en cuenta que estos documentos tienen componentes comunes que deben organizarse con el objetivo de reutilizarlos.

Por otro lado, la forma de pago debe gestionarse teniendo en cuenta que los tipos de datos que se pueden obtener son diferentes para cada una.

▼ Sprint 4 3 abr – 24 abr (19 incidencias) <small>Gestionar los documentos de Venta y lograr relacionarlos en un mismo control</small>	
AMC2022-88	Ver tabla de cotizaciones registradas
AMC2022-89	Buscar cotización
AMC2022-90	Aplicar filtro de búsquedas avanzadas
AMC2022-91	Reimprimir Cotización
AMC2022-95	Ver tabla de pedidos registrados
AMC2022-96	Buscar pedidos
AMC2022-97	Aplicar filtro de búsqueda avanzadas
AMC2022-98	Reimprimir pedido
AMC2022-61	Asignar un tercero cliente a la FV
AMC2022-62	Asignar un referido a la FV
AMC2022-63	Cambiar resolución de una FV
AMC2022-64	Cambiar fecha de una FV
AMC2022-65	Agregar productos a una FV
AMC2022-66	Editar la cantidad de cada producto agregada a la FV
AMC2022-67	Cambiar el valor unitario a un producto en la FV
AMC2022-68	Ver total de una FV
AMC2022-105	Buscar una forma de pago
AMC2022-69	Agregar una forma pago
AMC2022-70	Añadir una observación

Figura 26. Sprint 4 - Sprint Backlog
Fuente: Realización propia

6.5.2 Sprint Execution

Tabla 24. Descripción de la HU 61 - Asignar tercero cliente a la factura de venta.

HU 61: Asignar tercero cliente a la factura de venta	
Tareas	Descripción
Realizar búsqueda de Terceros	Se implementó la funcionalidad que permite buscar terceros tipo cliente. Para este caso la información obtenida se filtró por el tipo de tercero cliente.
Asignar Tercero a factura de venta	En esta tarea se implementó la funcionalidad que permite almacenar el tercero cliente de una factura de venta y realizar las respectivas validaciones.

Tabla 25. Descripción de la HU 63 - Cambiar resolución de una factura de venta.

HU 63: Cambiar resolución de una factura de venta	
Tareas	Descripción
Obtener resoluciones	Se implementó la funcionalidad que permite obtener las resoluciones registradas en el sistema, adicionalmente, se valida si estas están vigentes y tienen numeración disponible.
Gestionar cambios asociados a la resolución seleccionada	En esta tarea se validó la resolución seleccionada y se establecieron los cambios o configuraciones que deben realizarse en el documento de venta.

Tabla 26. Descripción de la HU 65 - Agregar Productos.

HU 65: Agregar Productos	
Tareas	Descripción
Obtener los parámetros edición de la tabla de productos	Para esta tarea se implementó la funcionalidad que permite obtener los parámetros tabla de la tabla edición de productos.
Realizar la búsqueda de productos	Se implementó la funcionalidad que permite obtener la lista de productos con la información requerida para el proceso de venta.
Agregar producto a la tabla	En esta tarea se realizaron las validaciones del producto agregado asociadas a las configuraciones del sistema.

Tabla 27. Descripción de la HU 67 - Cambiar valor unitario de un producto.

HU 67: Cambiar valor unitario de un producto	
Tareas	Descripción
Obtener valores unitarios de las listas de precio	Se implementó la funcionalidad que permite obtener la listas de precios de un producto teniendo en cuenta los permisos de usuario.
Cambiar valor unitario del producto	En esta incidencia se realizaron las validaciones del precio del producto para determinar si es necesario la verificación del administrador (ingresar clave) y si el precio está dentro de los rangos permitidos.

Tabla 28. Descripción de la HU - Agregar una forma de pago.

HU 35: Agregar una forma de pago	
Tareas	Descripción
Obtener tabla de formas de pago	En esta tarea se implementó la funcionalidad que permite obtener la tabla búsqueda con las formas de pago.
Obtener parámetros tabla edición de formas de pago	Para esta tarea se implementó la funcionalidad que permite obtener los parámetros tabla de la tabla edición de formas de pago.
Agregar forma de pago	En esta tarea se realizaron las validaciones de las formas de pago agregadas teniendo en cuenta las configuraciones del sistema y los montos válidos que se pueden ingresar en una factura de venta.

6.5.3 Sprint Review

Las historias de usuario seleccionadas para el desarrollo del Sprint 4 se cumplieron en su totalidad, sin embargo, es necesario tener en cuenta que, las tareas de búsqueda de los documentos de venta Cotización y Pedidos se pudieron implementar rápidamente ya que se reutilizó el proceso de las facturas de venta. La historia de usuario en la que se presentó mayor dificultad fue la HU 60 – Agregar forma de pago, ya que implicó el proceso de validaciones de usuario y configuraciones de usuario para cada una de ellas.

6.5.4 Sprint Retrospective

Al finalizar la ejecución del Sprint 4 se destacaron los siguientes puntos:

- El tiempo seleccionado para las historias ha sido el adecuado considerando que, desde el Sprint 3 se incrementó el número de incidencias para cada Sprint, puesto que el equipo para ese momento ya tenía una mayor experiencia.
- Se prevé que el Sprint 5 se cumplirá con éxito comparando el esfuerzo requerido del Sprint 4.

6.6 Sprint 5.

6.6.1 Sprint Planning

En la ejecución del Sprint 5 se desarrollaron las tareas correspondientes al registro de factura de venta, pedidos y cotizaciones.



Figura 27. Sprint 5-Sprint Backlog
Fuente: Realización propia

6.6.2 Sprint Execution

Tabla 29. Descripción de la HU 108 - Registrar factura de venta.

HU 108: Registrar factura de venta	
Tareas	Descripción
Realizar validaciones de productos	En esta incidencia se implementó la funcionalidad que permite validar el stock, precio costo-venta, cantidades disponibles para entregar y los datos de un producto.
Realizar validaciones de formas de pago	En esta tarea se valida que, las formas de pago cumplan con los montos de la venta, la información de cada forma de pago fuera correcta y no excediera los montos totales.
Validar resolución de facturación	Se valida que la resolución de una factura de venta cumpla con los requisitos de numeración máxima, fecha de caducidad y esté habilitada para su uso.

Tabla 30. Descripción de la HU 72 - Asignar fecha de vencimiento cotización.

HU 72: Asignar fecha de vencimiento a cotización	
Tareas	Descripción
Validar fecha de vencimiento	Para esta tarea se implementó la funcionalidad que permite verificar que la fecha de vencimiento agregada esté dentro de los rangos máximos configurados y sea igual o posterior a la fecha de registro.

Tabla 31. Descripción de la HU 109 - Registrar Pedido.

HU 109: Registrar Pedido	
Tareas	Descripción
Verificar el tipo de tercero	En esta tarea se valida que el tercero cliente del pedido no sea el tercero varios y la información del tercero sea válida.
Realizar las validaciones de cantidades disponibles	En esta incidencia es posible verificar el movimiento de inventario de las cantidades pendientes por entregar de cada pedido.

Tabla 32. Descripción de la HU - Visualizar PDF pedido.

HU 117: Visualizar PDF pedido	
Tareas	Descripción
Obtener PDF	Se implementó la funcionalidad que permite obtener el documento PDF del pedido, teniendo en cuenta la fecha de registro, el tipo de resolución y el formato del documento (media carta o tiquete).

6.6.3 Sprint Review

Las historias de usuario seleccionadas para el desarrollo del Sprint 5 fueron terminadas con éxito. Es necesario considerar que, en la ejecución de este Sprint se realizó un proceso de generalización de las funcionalidades de los diferentes documentos de venta.

La historia de usuario que presentó mayor dificultad fue la HU 108 – Registrar Factura de Venta, ya que de esta funcionalidad se realizó la generalización del registro de las cotizaciones y pedidos.

6.6.4 Sprint Retrospective

Al finalizar la ejecución del Sprint 5 se destacaron los siguientes puntos:

- El tiempo seleccionado para las historias ha sido el adecuado considerando que fue posible ejecutar todas las tareas asociadas del Sprint.
- Algunas historias de usuario implicaban el desarrollo de una funcionalidad asociada a otra historia, por lo tanto, fue posible generalizar determinadas funcionalidades como, validaciones de productos y resoluciones.
- Los errores presentados fueron solucionados constantemente en equipo, lo que permitió que su solución fuera rápida.
- Al finalizar este Sprint la aplicación tuvo un funcionamiento adecuado y la socialización de los resultados se realizó satisfactoriamente.

6.7 Resultados

Uno de los procesos de diseño que se realizaron fueron los diagramas de clases de los principales módulos implicados en el desarrollo. Estos diagramas están consignados en la sección de Anexos desde el Anexo 2 hasta el Anexo 10.

Cada una de las funcionalidades fueron probadas teniendo en cuenta los criterios de los respectivos módulos. Las consideraciones más relevantes para los casos de prueba se encuentran la sección de Anexos – Anexo 2.

Adicionalmente, es posible apreciar las principales funcionalidades implementadas en la sección de Anexos – Anexo 12

Por último, se determinó que el soporte de la aplicación Celeste App aplicaría desde la versión de API 26 o posteriores, esta caracterización se hizo considerando la versión de Android más usada por los clientes de Celeste Team y las versiones mínimas recomendadas por Android.

7 Resultados y conclusiones

En este capítulo se presentan los resultados obtenidos de la caracterización e implementación de la capa de datos del módulo Ventas para dispositivos móviles Android, las validaciones para el cumplimiento de los objetivos propuestos y las conclusiones obtenidas a lo largo del desarrollo y culminación del actual trabajo.

7.1 Cumplimiento de objetivos

De la presente práctica profesional realizada en Celeste Team S.A.S se cumplieron los siguientes objetivos (general y específicos).

En primer lugar, se realizó la socialización del sistema Celeste en su versión de escritorio con el fin de obtener las bases de negocio y desarrollo que permitieran iniciar la caracterización del módulo de ventas y la definición de la arquitectura, los patrones de diseño y tecnologías. Para la caracterización se realizaron las validaciones del funcionamiento de los módulos relacionados a la implementación con el fin de determinar el flujo de trabajo.

La definición de la arquitectura implicó la implementación de prototipos que permitieran evidenciar su adecuado funcionamiento, pero principalmente, cumpliera con los requisitos de la empresa. Al obtener la arquitectura, se realizó la respectiva socialización con el equipo de desarrollo, en la cual se concluyó que, la arquitectura seleccionada es la adecuada y es posible aplicarla ya que está relacionada con el modelo de desarrollo que lleva la empresa.

Gracias al proceso de caracterización del módulo de ventas Celeste, el análisis y diseño de la arquitectura de desarrollo se pudo dar por cumplido el primer objetivo específico *“Caracterizar el módulo de ventas de la empresa Celeste Team S.A.S teniendo en cuenta los patrones arquitectónicos orientados a las aplicaciones móviles”* (ver Capítulo IV. Caracterización del módulo de ventas).

Posterior a la caracterización y definición de la arquitectura, se llevó a cabo la adaptación del marco de trabajo desde el contexto de escritorio al contexto móvil Android. En esta sección se realizó el análisis de las características relacionadas al marco de trabajo de datos de escritorio, del cual se destaca, la ejecución síncrona de peticiones al servidor a través de la librería OkHttp ya que desde el contexto móvil no es recomendable, ni permitido, el bloqueo de la interfaz de usuario. Con el análisis e implementación del marco de trabajo de la capa de datos para aplicaciones móviles Android se da por cumplido el segundo objetivo específico *“Adaptar el marco de trabajo orientado a la capa de datos de aplicaciones de escritorio de la empresa Celeste Team S.A.S para clientes móviles”* (ver Capítulo V. Marco de trabajo para la capa de datos).

Una vez implementado el marco de trabajo para la capa de datos de aplicaciones móviles se realiza el proceso de validación a partir de la implementación de las funcionalidades de los módulos definidos en la sección 4, con el fin de determinar si el uso y funcionamiento del marco de trabajo era el adecuado.

A lo largo de la implementación se evidenció que el desarrollo de las funcionalidades fue rápido, la aplicación del marco de trabajo era entendible y la gestión de peticiones al servidor era sencilla y casi imperceptible. Al obtener la aplicación móvil para Android en su primera versión se da por cumplido el tercer objetivo específico “Validar el marco de trabajo a partir de la implementación de la capa de datos en el módulo “Ventas” para el cliente móvil en Android”.

7.2 Conclusiones

El desarrollo de la presente práctica profesional tuvo como objetivo apoyar el desarrollo de la aplicación Celeste App para brindar una versión móvil del sistema. En este proceso se aplicaron conocimientos asociados al diseño de arquitecturas, desarrollo de aplicaciones Android con Java, manejo de Reflexión Java, procesamiento de anotaciones, manejo de base de datos locales y solicitud de peticiones aplicando la arquitectura REST.

Ahora se presentan las conclusiones obtenidas durante el desarrollo y culminación de la práctica profesional:

- La definición acertada de la arquitectura de desarrollo se dio gracias a la adecuada caracterización y análisis de los módulos implicados en la implementación. En este proceso se indagó sobre el entorno de negocio, y principalmente, se analizó el código de la aplicación de escritorio. Esto aclaró los principales lineamientos de desarrollo que se debían seguir en la aplicación móvil de acuerdo con los criterios de desarrollo de la empresa Celeste Team.
- La definición de la arquitectura permitió la clara separación de la capa de presentación y la capa de datos, esto con el objetivo de relacionar los marcos de trabajo para cada una de las capas y separar responsabilidades a lo largo de la implementación.
- Aplicar adecuadamente los requerimientos de la empresa y el modelo de programación de Celeste permitió que el equipo de desarrollo encargado de la aplicación de escritorio afianzara rápidamente el proceso de desarrollo para el cliente móvil Android.
- El marco de trabajo para aplicaciones móviles permitió que en la implementación del cliente Android el proceso de realizar solicitudes al servidor fuera casi imperceptibles. Esto facilitó la entrega de múltiples funcionalidades ya que el desarrollador no debía encargarse de formar una solicitud de red o realizar tareas repetitivas.

- Se validó el marco de trabajo para la capa de datos implementando el módulo de ventas de la aplicación Celeste. En esta sección se destaca la correcta elección de la biblioteca de cliente OkHttp ya que esta permitió manipular la forma de realizar solicitudes al servidor facilitando la ejecución de las solicitudes de red para los diferentes encabezados y tipos de respuesta.
- Cada una de las tecnologías y herramientas aprendidas ampliaron los conocimientos técnicos, personal, laboral, analítico y de toma de decisiones. Estos fueron reflejados en la definición de la arquitectura y patrones de diseño. En el análisis, elección e implementación de las tecnologías aplicadas en la adaptación del marco de trabajo para la capa de datos del cliente móvil Celeste. En el uso del marco de trabajo ágil Scrum y su gestión a través de la herramienta Jira. En la caracterización e implementación del módulo de ventas de la aplicación Celeste App.

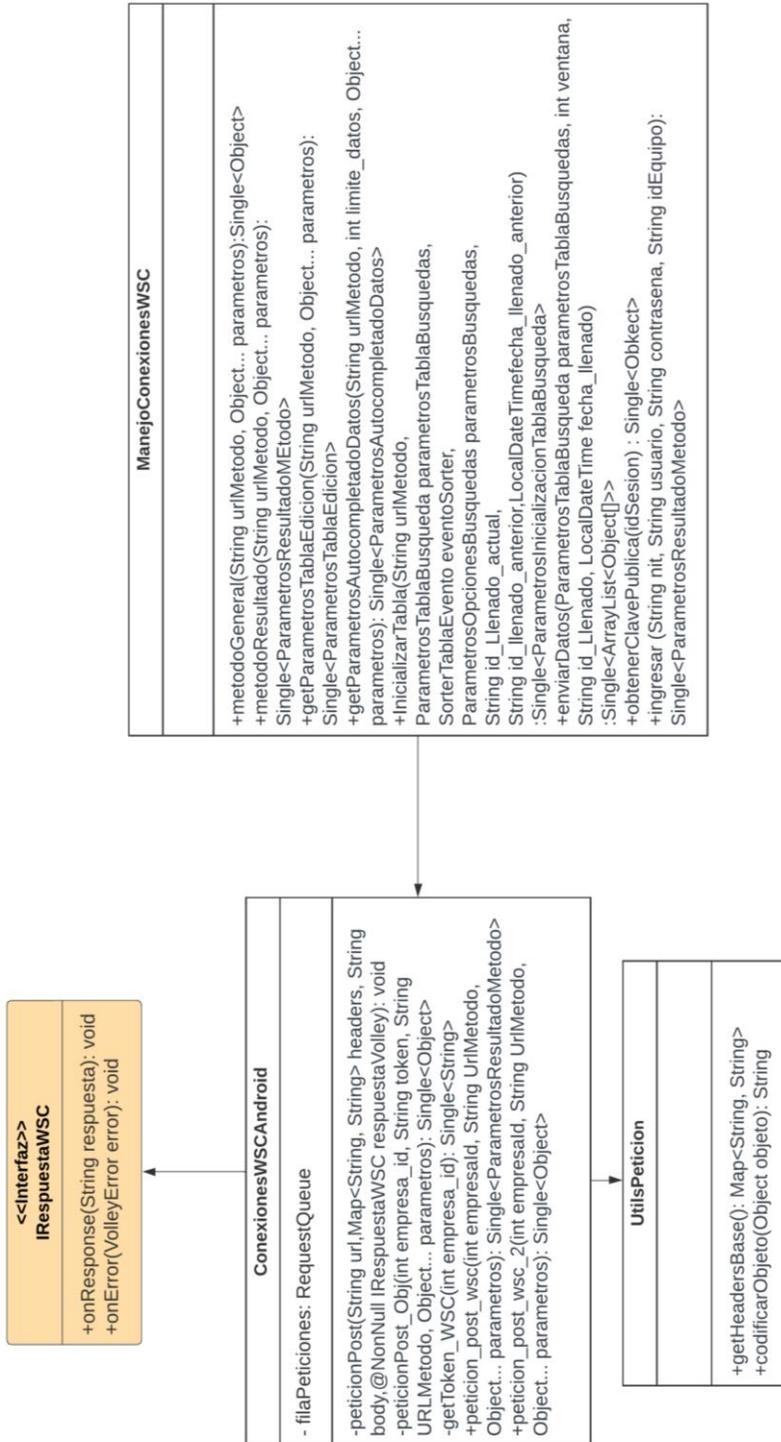
8 Referencias

- [1] «MiPyme Vive Digital - MINTIC - Vive Digital». <https://mintic.gov.co/portal/vivedigital/612/w3-propertyvalue-7235.html> (accedido 19 de junio de 2023).
- [2] «Importancia del Software Contable en las Empresas | Loggro», 20 de mayo de 2019. <https://loggro.com/blog/articulo/cual-es-la-importancia-del-software-contable-para-las-pyme-en-colombia/> (accedido 19 de junio de 2023).
- [3] «Digital 2021: Global Overview Report», *DataReportal – Global Digital Insights*. <https://datareportal.com/reports/digital-2021-global-overview-report> (accedido 4 de julio de 2022).
- [4] «Guía de arquitectura de apps | Desarrolladores de Android», *Android Developers*. <https://developer.android.com/jetpack/guide?hl=es-419> (accedido 11 de octubre de 2022).
- [5] Gloria Arcos-Medina, Jorge Menéndez, y Javier Vallejo, «Comparative Study of Performance and Productivity of MVC and MVVM design patterns», *KnE Eng.*, vol. 3, n.º 1, ene. 2018, doi: 10.18502/keg.v1i2.1498.
- [6] E. Guerra *et al.*, «How does annotations affect Java code readability?», In Review, preprint, nov. 2022. doi: 10.21203/rs.3.rs-2256326/v1.
- [7] dsgg18, «ISO/IEC/IEEE 42010:2011 – Descripción de arquitectura – Parte I: Introducción», *LIFE.ART.TECH*, 16 de agosto de 2017. <https://lifeartech.wordpress.com/2017/08/16/isoiecieee-42010-descripcion-de-arquitectura-intro/> (accedido 23 de marzo de 2023).
- [8] «Software Architecture & Design Introduction». https://www.tutorialspoint.com/software_architecture_design/introduction.htm (accedido 23 de marzo de 2023).
- [9] michaelstonis, «Modelo-Vista-Modelo de vista», 28 de noviembre de 2022. <https://learn.microsoft.com/es-es/dotnet/architecture/maui/mvvm> (accedido 24 de marzo de 2023).
- [10] «Uso de la reflexión de Java». <https://www.oracle.com/technical-resources/articles/java/javareflection.html> (accedido 24 de marzo de 2023).
- [11] F. Hermoza, «Android y el patrón MVVM», *Medium*, 25 de mayo de 2020. <https://fahedhermoza.medium.com/android-y-el-patr%C3%B3n-mvvm-b5e71f5e8ec7> (accedido 16 de mayo de 2023).

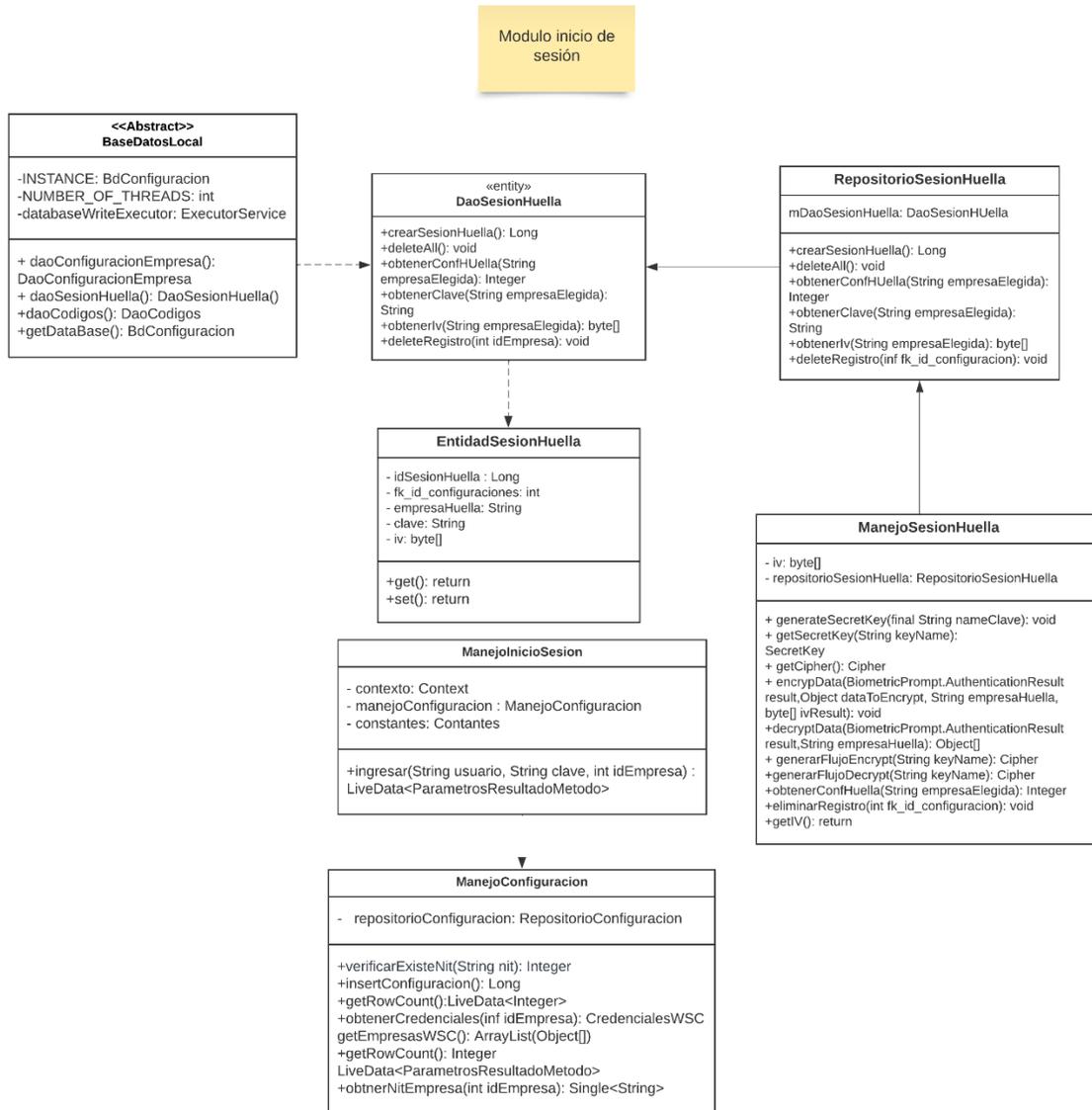
- [12] ADX 2099, «Clean Architecture ¿Qué es? y ¿Cómo funciona?», *Medium*, 14 de abril de 2020. <https://adx2099.medium.com/clean-architecture-qu%C3%A9-es-y-c%C3%B3mo-funciona-4a1778360419> (accedido 20 de junio de 2023).
- [13] «How to Build a Simple Note Android App using MVVM and Room Database?», *GeeksforGeeks*, 27 de julio de 2021. <https://www.geeksforgeeks.org/how-to-build-a-simple-note-android-app-using-mvvm-and-room-database/> (accedido 9 de agosto de 2023).
- [14] H. Patricio, «Entendiendo REST: Arquitectura cliente-servidor», *The Dojo MX Blog*, 4 de julio de 2019. <https://blog.thedojo.mx/2019/07/04/entendiendo-rest-arquitectura-cliente-servidor.html> (accedido 29 de mayo de 2023).
- [15] «The Reactive Manifesto». <https://www.reactivemanifesto.org/> (accedido 5 de junio de 2023).
- [16] «Home», *GitHub*. <https://github.com/ReactiveX/RxJava/wiki/Home> (accedido 9 de junio de 2023).
- [17] «Lecciones Aprendidas en Desarrollo de Software». <http://www.lecciones-aprendidas.info/search?q=Modos+de+Representaci%C3%B3n+de+las+Historias+de+Usuario> (accedido 23 de mayo de 2023).

Anexos

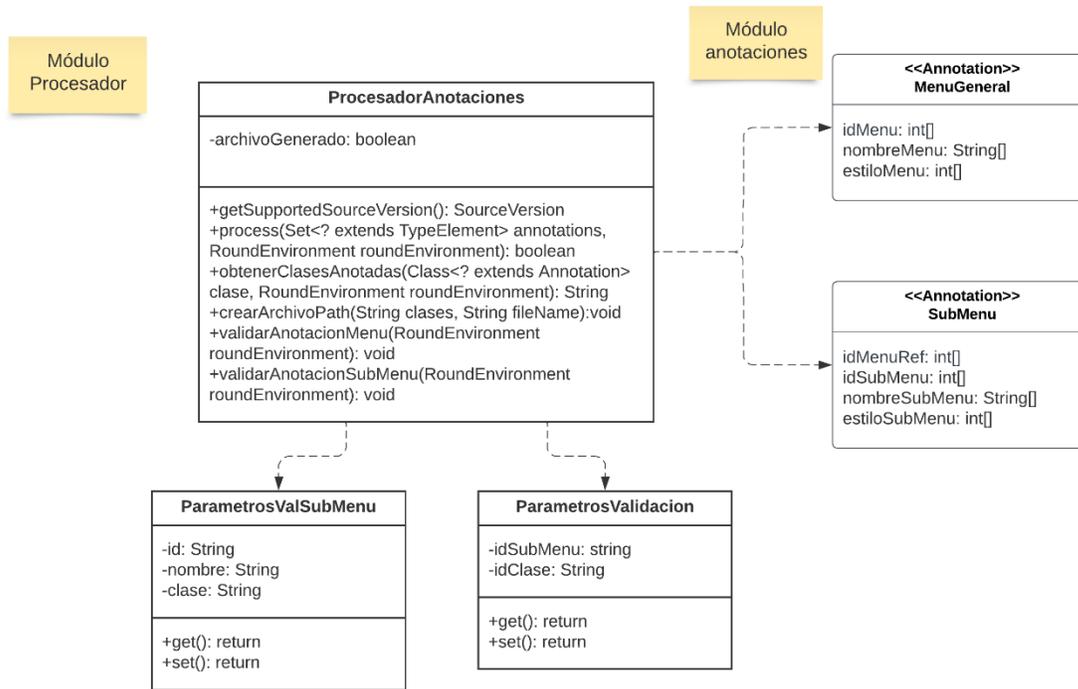
Anexo 1. Diagrama de clases para el diseño del marco de trabajo para el contexto móvil Android.



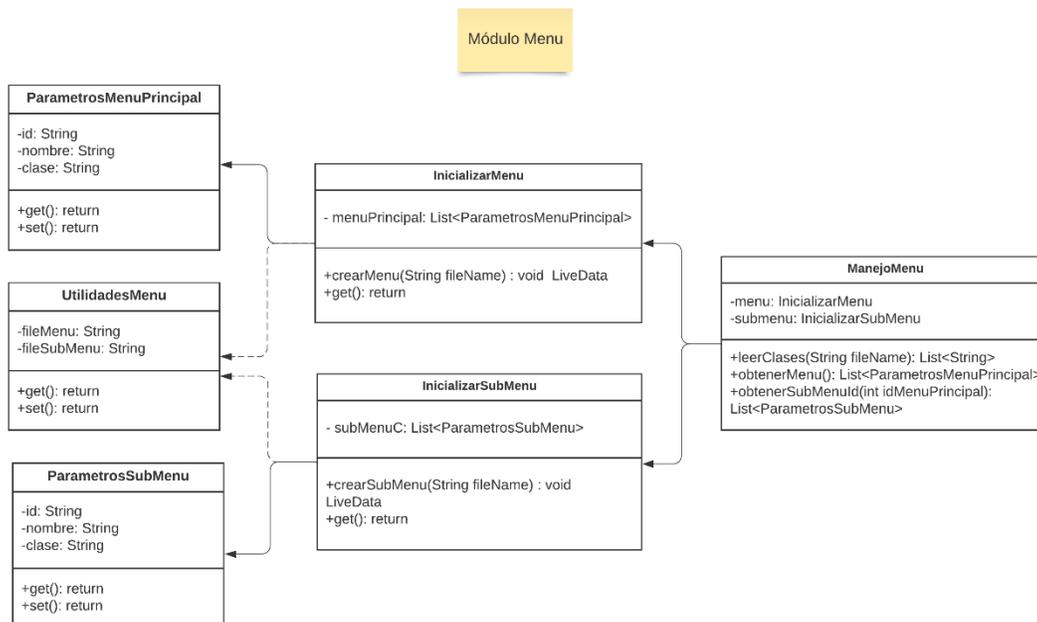
Anexo 3. Diagrama de clases para el diseño del módulo inicio de sesión y configuración de huella.



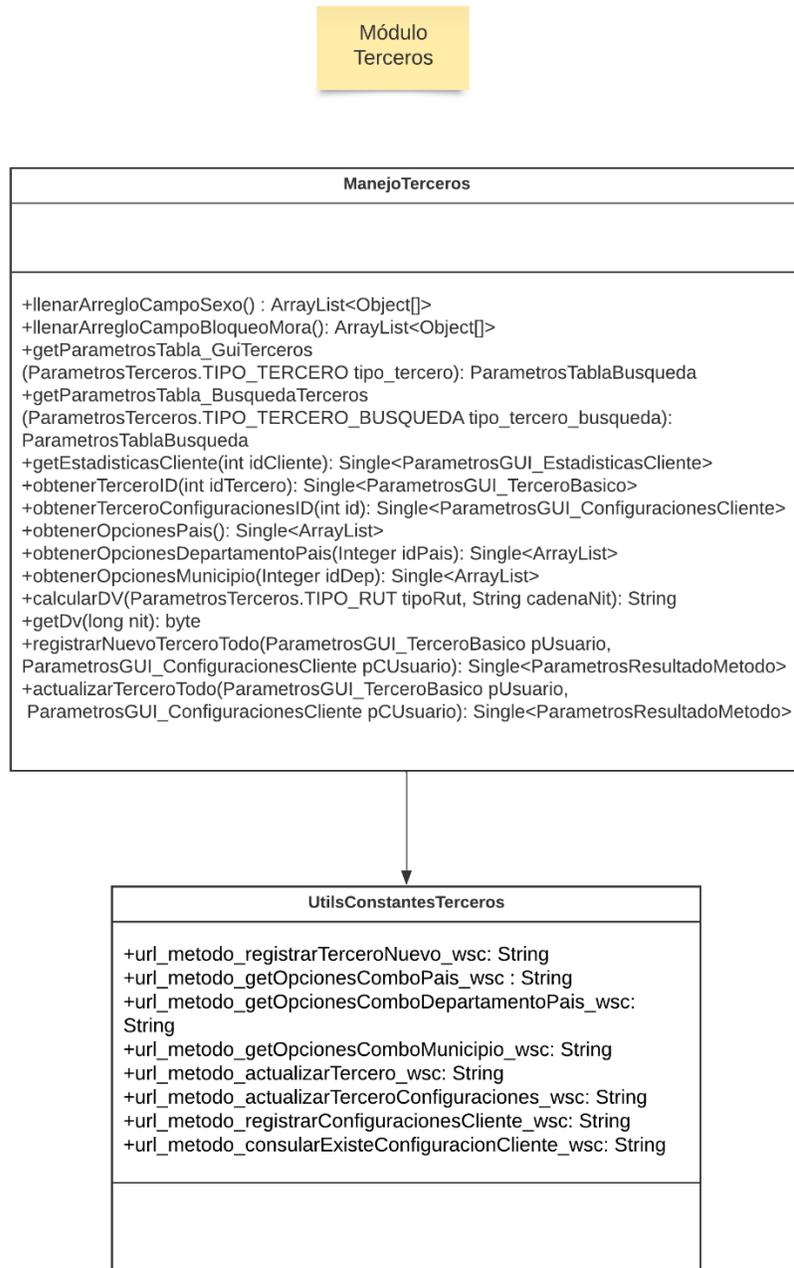
Anexo 4. Diagrama de clases para el diseño del módulo de procesamiento de anotaciones.



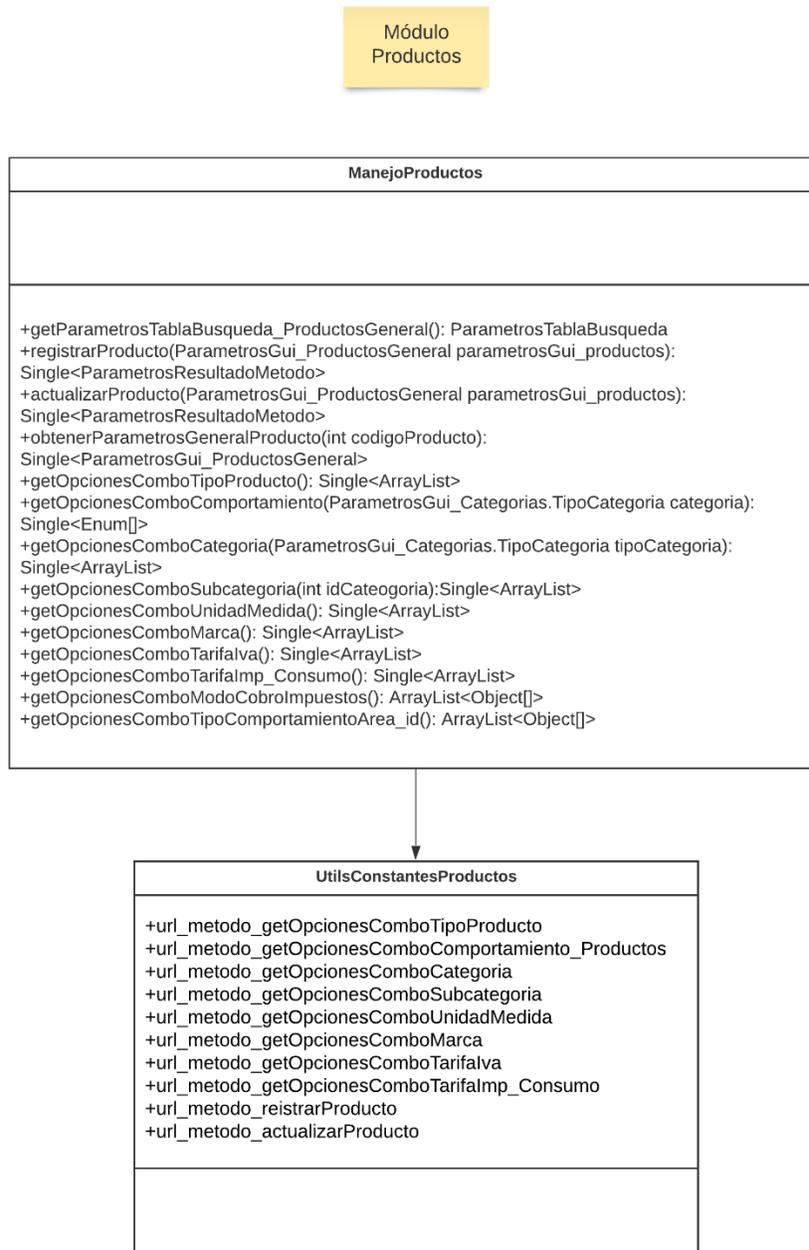
Anexo 5. Diagrama de clases para el diseño del módulo menú.



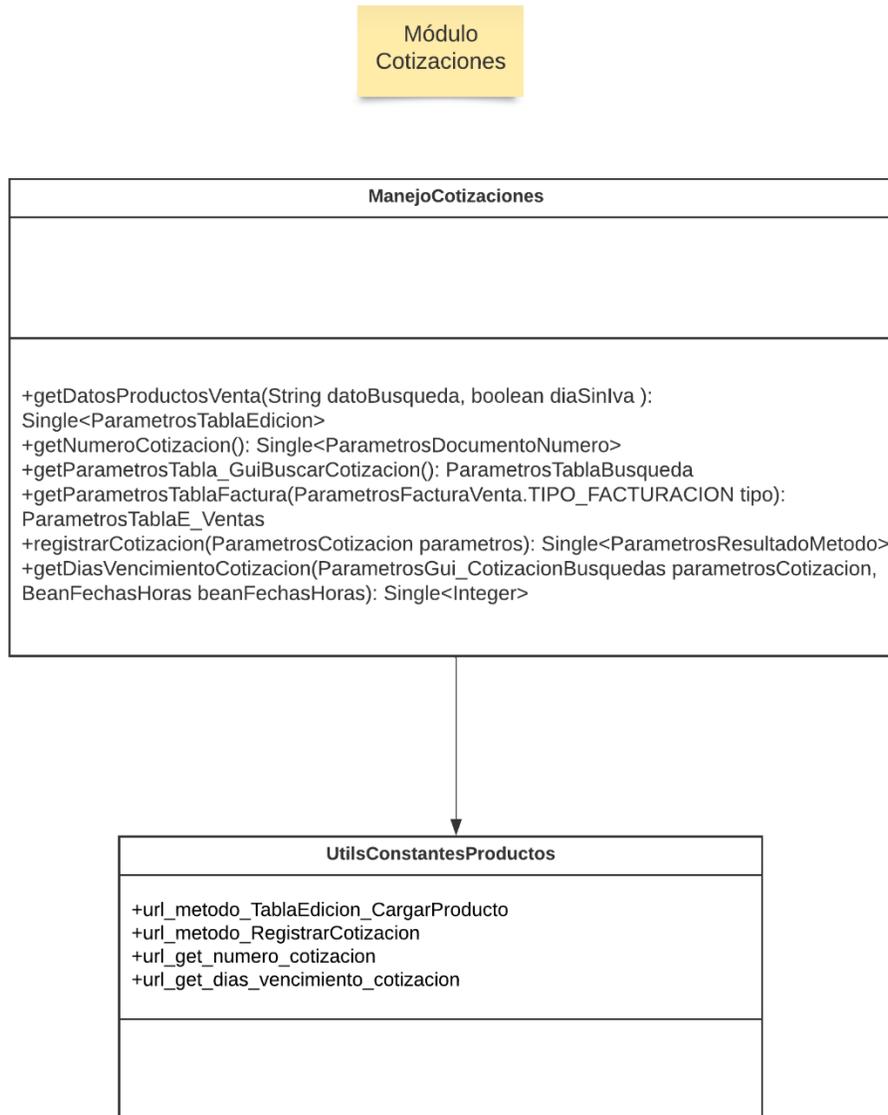
Anexo 6. Diagrama de clases para el diseño del módulo de terceros.



Anexo 7. Diagrama de clases para el diseño del módulo de productos.



Anexo 8. Diagrama de clases para el diseño del módulo cotizaciones.

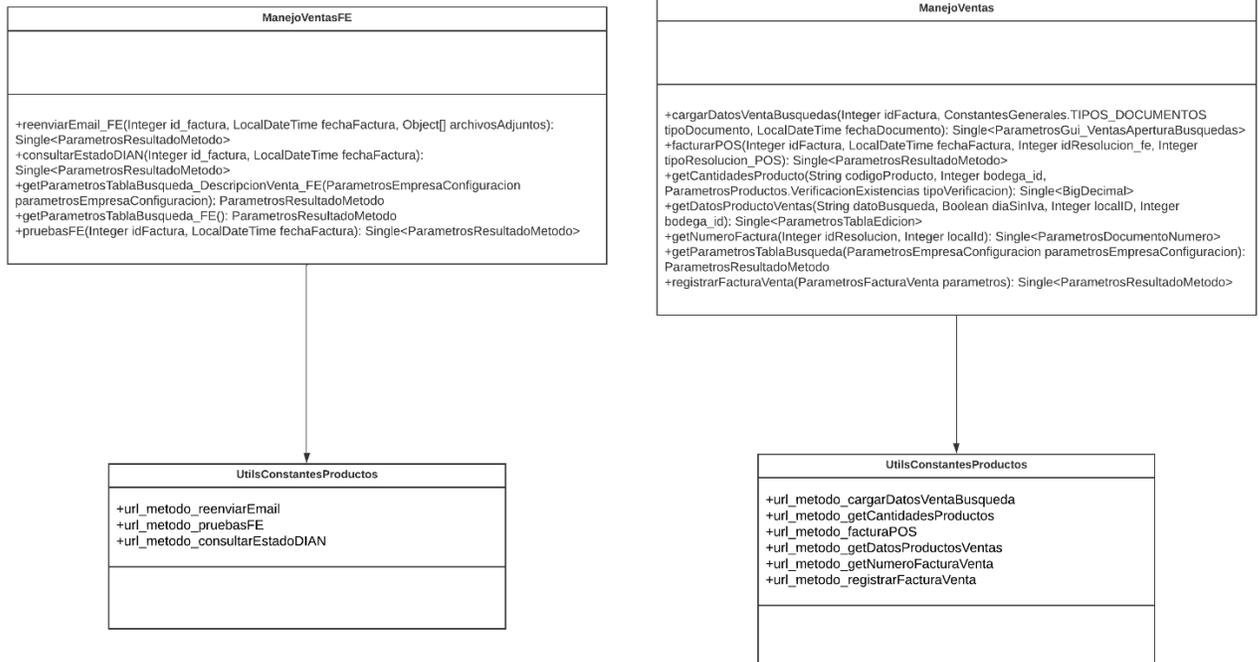


Anexo 9. Diagrama de clases para el diseño del módulo pedidos.



Anexo 10. Diagrama de clases para el diseño del módulo ventas.

Módulo Ventas



Anexo 11. Criterios asociados a las validaciones de las funcionalidades principales implementadas de la aplicación móvil Celeste App.

EPICA	Configurar Empresa	
SPRINT	1	
ESCRITO POR	Jhulieth Eliana Dorado	
TEST #	FUNCIONALIDAD	CRITERIO
1	Solicitar código de configuración	Ingresar NIT no existente
2	Solicitar código de configuración	Ingresar NIT válido
3	Solicitar código de configuración	Ingresar NIT de empresa ya configurada
4	Solicitar código de configuración	Solicitud sin internet
7	Ingresar código	Enviar código inválido
8	Ingresar código	Enviar código sin internet
9	Ingresar código	Enviar código válido
12	Generar nuevo código	El usuario no ha ingresado el código en el tiempo esperado

EPICA	Inicio de sesión y configuración de huella	
SPRINT	1-2	
ESCRITO POR	Jhulieth Eliana Dorado	
TEST #	FUNCIONALIDAD	CRITERIO
1	Iniciar sesión con credenciales	Ingresar usuario o contraseña incorrectos
2	Iniciar sesión con credenciales	Iniciar sesión sin conexión a internet

3	Iniciar sesión con credenciales	Credenciales correctas, más de una sucursal, mensualidad al día
4	Iniciar sesión con credenciales	Credenciales correctas, sólo una sucursal, mensualidad al día
5	Iniciar sesión con credenciales	Credenciales correctas, más de una sucursal, días de gracia
6	Iniciar sesión con credenciales	Credenciales correctas, más de una sucursal, mensualidad vencida
7	Iniciar sesión con credenciales	Credenciales correctas, una sucursal, días de gracia
8	Iniciar sesión con credenciales	Credenciales correctas, una sucursal, mensualidad vencida
9	Configurar Huella	El dispositivo no tiene hardware de huella
10	Configurar Huella	No se ha configurado huellas en el dispositivo
11	Configurar Huella	Ingresar usuario o contraseña incorrectos
12	Configurar Huella	Credenciales correctas, huella incorrecta
13	Configurar Huella	Credenciales correctas, huella correcta, mensualidad al día, más de una sucursal
14	Configurar Huella	Credenciales correctas, huella correcta, mensualidad al día, una sucursal
15	Configurar Huella	Credenciales correctas, huella correcta, días de gracia, más de una sucursal
16	Configurar Huella	Credenciales correctas, huella correcta, mensualidad vencida, más de una sucursal
17	Configurar Huella	Credenciales correctas, huella correcta, días gracia, una sucursal
18	Configurar Huella	Credenciales correctas, huella correcta, mensualidad vencida, una sucursal
19	Gestionar Huella	Eliminar huellas registradas en el dispositivo
20	Gestionar Huella	Se agrega huella en el dispositivo
21	Gestionar Huella	2 o más huellas configuradas: Se elimina una huella

EPICA	Gestionar Menú	
SPRINT	2	
ESCRITO POR	Jhulieth Eliana Dorado	
TEST #	FUNCIONALIDAD	CRITERIO
1	Buscar en menú principal	Buscar por caracteres desconocidos u opciones no existentes
2	Buscar en menú principal	Buscar por opciones existentes
3	Orden de menú Principal	De acuerdo con el id ingresado en la anotación
4	Navegar a opciones de submenú	El menú principal no tiene asociado un submenú
5	Buscar en submenú	Buscar por caracteres desconocidos u opciones no existentes

EPICA	Gestionar Terceros	
SPRINT	2	
ESCRITO POR	Jhulieth Eliana Dorado	
TEST #	FUNCIONALIDAD	CRITERIO
1	Visualizar terceros registrados	Gestionar permisos de usuario
2	Buscar tercero	Mediante Voz o escrito
3	Crear Tercero	No ingresar todos los datos requeridos
4	Crear Tercero	Solicitar peticiones de llenado en orden arbitrario
5	Crear Tercero	Tipo de identificación NIT
6	Crear Tercero	Tipo de identificación RUT
7	Actualizar Tercero	Tipo proveedor
8	Actualizar Tercero	Tipo Cliente
9	Seleccionar país y departamento	Asociar país con departamento

10	Seleccionar país y departamento	Asociar departamento con municipio
----	---------------------------------	------------------------------------

EPICA	Gestionar Productos	
SPRINT	3	
ESCRITO POR	Jhulieth Eliana Dorado	
TEST #	FUNCIONALIDAD	CRITERIO
1	Visualizar productos registrados	Sin permisos para visualización
2	Buscar producto	Por voz, código de barras o escrito
3	Crear producto	Solicitar peticiones de llenado en orden arbitrario
4	Crear producto	Tipo de producto área
7	Crear producto	Tipo de producto con Tercero Mandato
11	Crear producto	Tipo de producto combustible
14	Crear producto	Tipo de producto final
15	Crear producto	Tipo de producto lote
17	Crear producto	Sin contabilidad asignada
18	Crear producto	Con contabilidad asignada
19	Actualizar Producto	Con referencia de precios unitarios
20	Actualizar Producto	Llenar combos de información en orden

EPICA	Gestionar PDF	
SPRINT	3	
ESCRITO POR	Jhulieth Eliana Dorado	
TEST #	FUNCIONALIDAD	CRITERIO
1	Visualizar PDF de documento de venta al registrar documento	Plantilla no encontrada
2	Visualizar PDF de documento de venta al registrar documento	PDF no encontrado
5	Compartir PDF	Validar con aplicaciones instaladas
6	Compartir PDF	PDF no coincide
7	Descargar PDF	Almacenar en carpeta descargas
8	Imprimir PDF	Tipo de documento pedido
11	Imprimir PDF	Tipo de documento cotización
12	Imprimir PDF	Tipo de documento factura de venta

EPICA	Gestionar Documento de Venta	
SPRINT	3	
ESCRITO POR	Jhulieth Eliana Dorado	
TEST #	FUNCIONALIDAD	CRITERIO
2	Filtrar búsqueda	Por fecha
3	Filtrar búsqueda	Por estado
4	Filtrar búsqueda	Por estado factura electrónica
5	Filtrar búsqueda	Por opciones avanzadas
6	Filtrar búsqueda	Por estado de cartera

7	Filtrar búsqueda	Por forma entrega
8	Filtrar búsqueda	Por sucursal
9	Filtrar búsqueda	Por estado entrega
10	Reimprimir documento	Formato de impresión Media Carta
11	Reimprimir documento	Plantilla no encontrada

EPICA	Gestionar factura de venta	
SPRINT	4	
ESCRITO POR	Jhulieth Eliana Dorado	
TEST #	FUNCIONALIDAD	CRITERIO
1	Cambiar tipo de documento de venta (pestañas)	Solicitar resolución entre el cambio de pestaña
2	Cambiar tipo de documento de venta (pestañas)	Asignar tercero cliente
3	Gestionar resolución del documento venta	Numeración de resolución vigente y vencida
4	Gestionar fecha de documento	Permisos de edición de fecha
5	Gestionar fecha de documento	Para factura electrónica validar rangos de fecha para edición
7	Asignar un tercero a la venta	Tipo de tercero Cliente
8	Crear nuevo tercero desde la búsqueda	Validar permiso para crear tercero desde el registro
9	Visualizar estadísticas cliente	Cliente con cupo, anticipos y cartera
10	Cargar Búsqueda referidos	Tipo de tercero que aplica para ser referido
11	Seleccionar un producto en la búsqueda	Producto con Tercero Mandato, lote y área

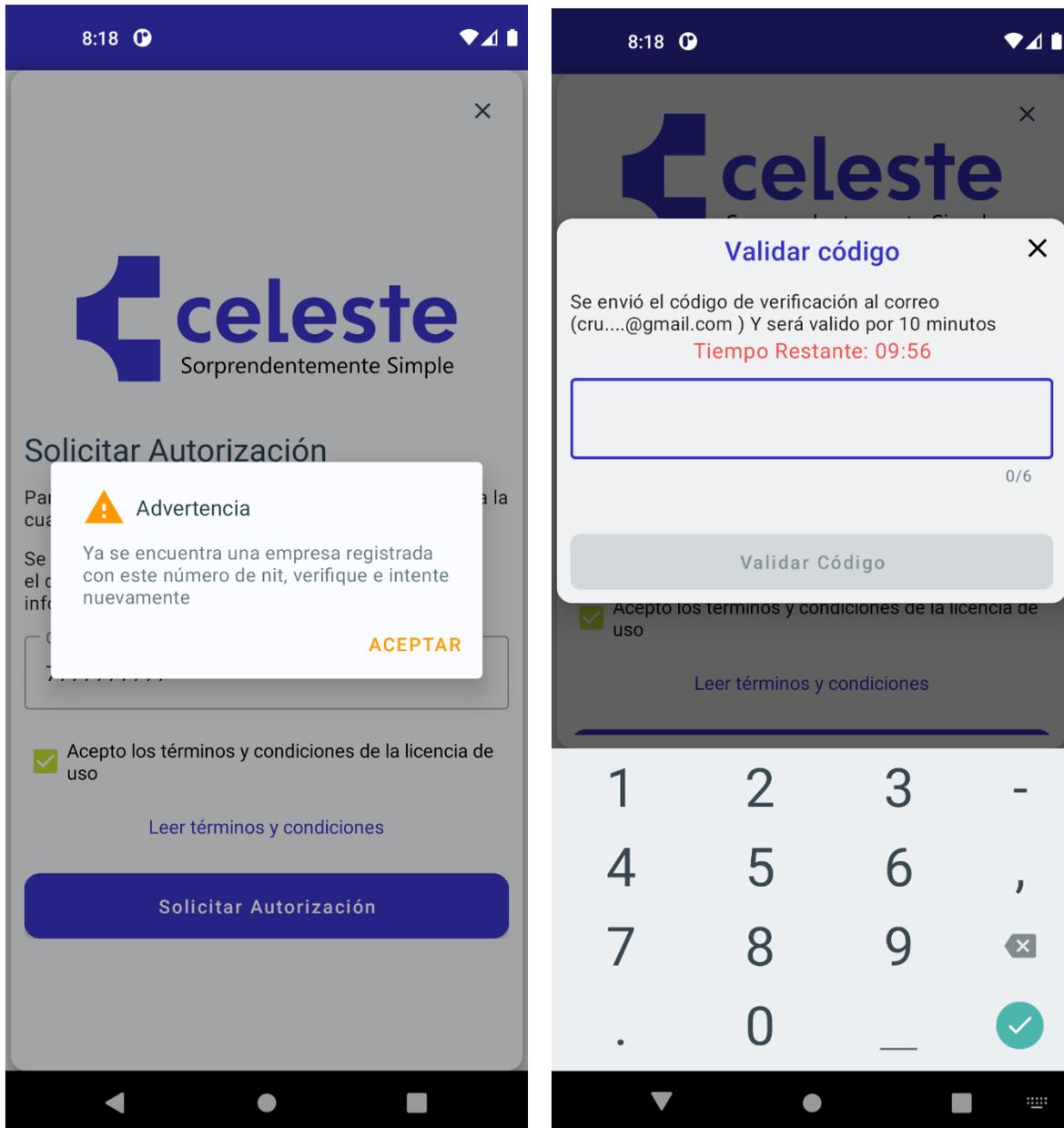
13	Cargar un producto escaneando código	Código de barras no encontrado
16	Gestionar productos agregados, gestionar cantidades	Cantidades superiores a 0 e inferiores a cantidad disponible (según configuración)
18	Gestionar productos agregados, gestionar cantidades	Unidad de medida que permite decimales
19	Gestionar productos agregados, gestionar valor unitario	Cambiar valor unitario (permiso todos)
20	Gestionar productos agregados, gestionar valor unitario	Cambiar valor unitario (permiso sólo tabla)
21	Gestionar productos agregados, gestionar valor unitario	Cambiar valor unitario (permiso sólo lista)
22	Gestionar productos agregados, gestionar valor unitario	Cambiar valor unitario (permiso ninguno)
23	Gestionar productos agregados, gestionar valor unitario	configuración verificar costo venta
26	Gestionar productos agregados	Eliminar Producto
31	Alertas de productos a facturar	Cantidades no disponibles o precios no permitidos
34	Registrar Productos	Condición UVT POS

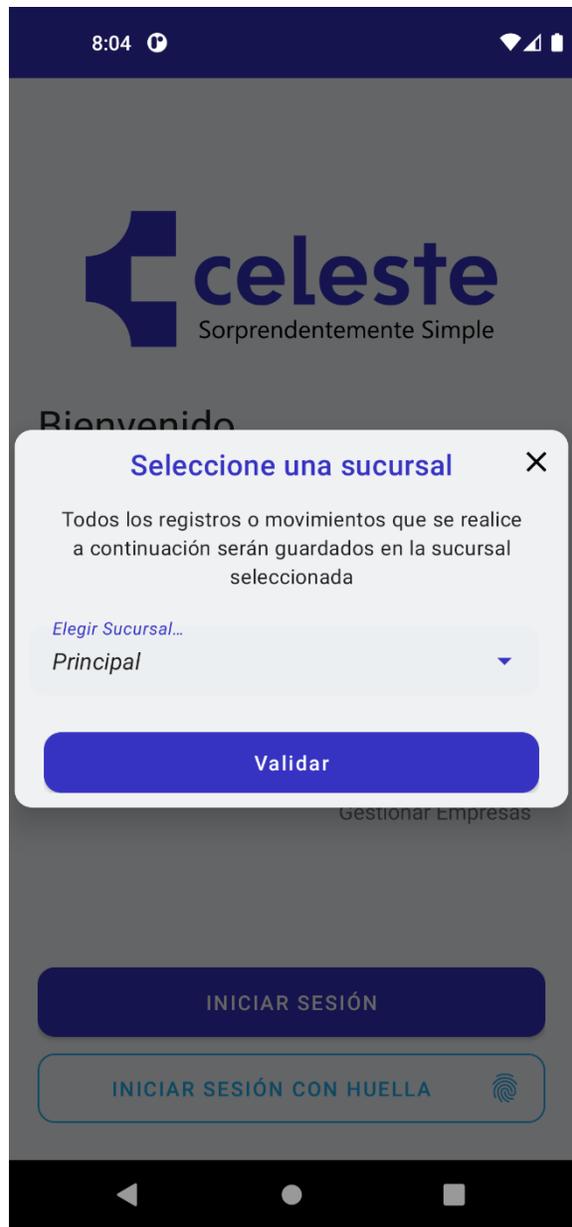
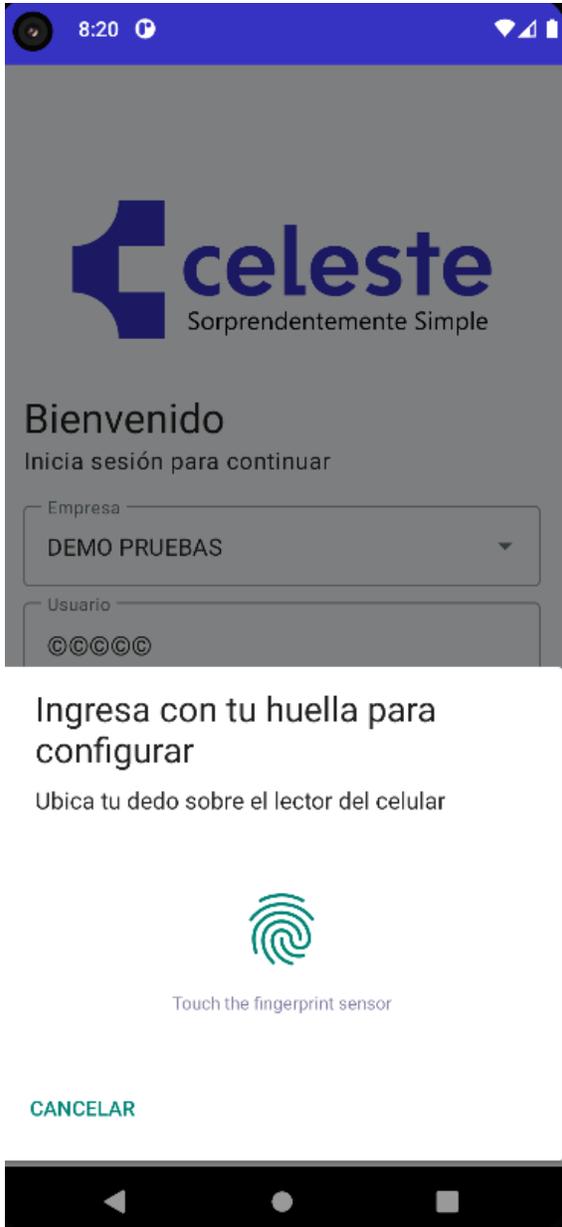
EPICA	Registrar factura de venta	
SPRINT	5	
ESCRITO POR	Jhulieth Eliana Dorado	
TEST #	FUNCIONALIDAD	CRITERIO
1	Ver totales factura	Validar la configuración del sistema descuento
2	Ver totales factura	configuración Impuestos (iva, impuesto al consumo)
3	Ver totales factura	ajuste
4	Agregar forma pago	Ver estadísticas anticipos en forma pago
5	Agregar forma pago	Forma de pago efectivo
6	Agregar forma pago	Forma de pago crédito
7	Agregar forma pago	Forma de pago tarjeta
8	Agregar forma pago	Forma de pago cruce con anticipos
9	Agregar forma pago	Forma de pago cheque
10	Agregar forma pago	Forma de pago consignación
11	Agregar forma pago	Forma de pago transferencia
12	Agregar forma pago	Forma de pago devoluciones
13	Agregar forma pago	Forma de pago notas crédito
14	Agregar forma pago	Forma de pago rete fuente
15	Agregar forma pago	Forma de pago rete iva
16	Agregar forma pago	Forma de pago rete ica

17	Agregar forma pago	Forma de pago estampillas
18	Agregar forma pago	Forma de pago bono de descuento
19	Agregar forma pago	Forma de pago otras formas pago
20	Agregar forma entrega	Verificar módulo de bodega
22	Registrar factura	Totales no coinciden
24	Registrar factura	Productos no cuentan con cantidades disponibles
26	Registrar factura	Los pagos ingresados no cubren el total de la factura
27	Registrar factura	Configuración contable no coincide

EPICA	Registro de pedidos	
SPRINT	5	
ESCRITO POR	Jhulieth Eliana Dorado	
TEST #	FUNCIONALIDAD	CRITERIO
1	Editar fecha documento	Fecha de registro de acuerdo con los rangos permitidos
2	Cambiar precio unitario	Desde la tabla, lista precio, todos o ninguno
3	Asignar tercero, referido y vendedor	Asociar cada búsqueda al tipo de tercero
4	Agregar productos	Tener en cuenta el tipo de producto agregado
5	Asignar días vencimiento	Validar fecha vencimiento superior a la de registro

Anexo 12. Funcionalidades principales de la aplicación Celeste App.





8:04

Buscar Documento de Venta

Cliente, Número, C.C/NIT ...

Emanuel Palomino **Por Cobrar**
 COAN-7 \$25,000
 09-08-2023 11:28 AM Saldo: \$24,987
 Abono: \$13 1 de 111

Juan Carlos Cruz **Pagada**
 COAN-6 \$5,000
 08-08-2023 05:53 PM 2 de 111

Juan Carlos Cruz **Pagada**
 COAN-5 \$1,000,000
 08-08-2023 11:07 AM 3 de 111

Emanuel Palomino **Pagada**
 COAN-4 \$5,000
 08-08-2023 11:03 AM 4 de 111

Consumidor Final **Pagada**
 COAN-3 \$20,000
 04-08-2023 09:21 AM 5 de 111

Consumidor Final **Pagada**
 CONT-12 \$30,000
 04-08-2023 09:17 AM 6 de 111

Consumidor Final **Pagada**
 COAN-2 \$20,000
 04-08-2023 09:17 AM 7 de 111

8:05

Documento PDF

Formato de Venta FEC (contingencia) T80

DEMO PRUEBAS
 Demo Pruebas
 7.777.777.777 No Responsable de IVA e INC
 7777777
 Direccion pruebas
 Autorización Factura de Contingencia 2528 de 2023-07-27
 Vence 2024-07-27. Numeración COAN del 1 Hasta 3450000

FACTURA ELECTRONICA DE CONTINGENCIA
 NUMERO: COAN 7
 FECHA EXPEDICION: 09/08/2023 11:28 AM
 VENDEDOR: Administrador Sistema

CLIENTE
 NIT: 1098652147 NOM: Emanuel Palomino
 TEL: 3005698745
 DIR: PARQUE CENTRAL CANDELARIA VALLE DEL CAUCA Colombia

PRODUCTOS Y/O SERVICIOS

COD: 295 - Cargador kinotec
 IVA: 19 %
 CAN: 1 V/U: 25.000 V/T: 25.000

EXCLUIDO:	0
EXENTO:	0
GRAVADO:	21.008,4
IC:	0
IVA:	3.991,6
TOTAL:	\$ 25.000

FORMAS DE PAGO
 * Crédito: \$25.000

* Reimpresión



8:07

Agregar un tercero

Identificación

Tipo Tercero
Cliente & Proveedor

NIT/Cedula - DV

Tipo Identificación
C.C.

Nombre

Primer Nombre Segundo Nombre

Primer Apellido Segundo Apellido

Nombre Empresa

Dirección

Departamento
CAUCA

Municipio
POPAYÁN

Registrar

8:07

Seleccionar Departamento

Buscar...

ANTIOQUIA

ATLÁNTICO

BOGOTÁ D.C

BOLÍVAR

BOYACÁ

CALDAS

CAQUETÁ

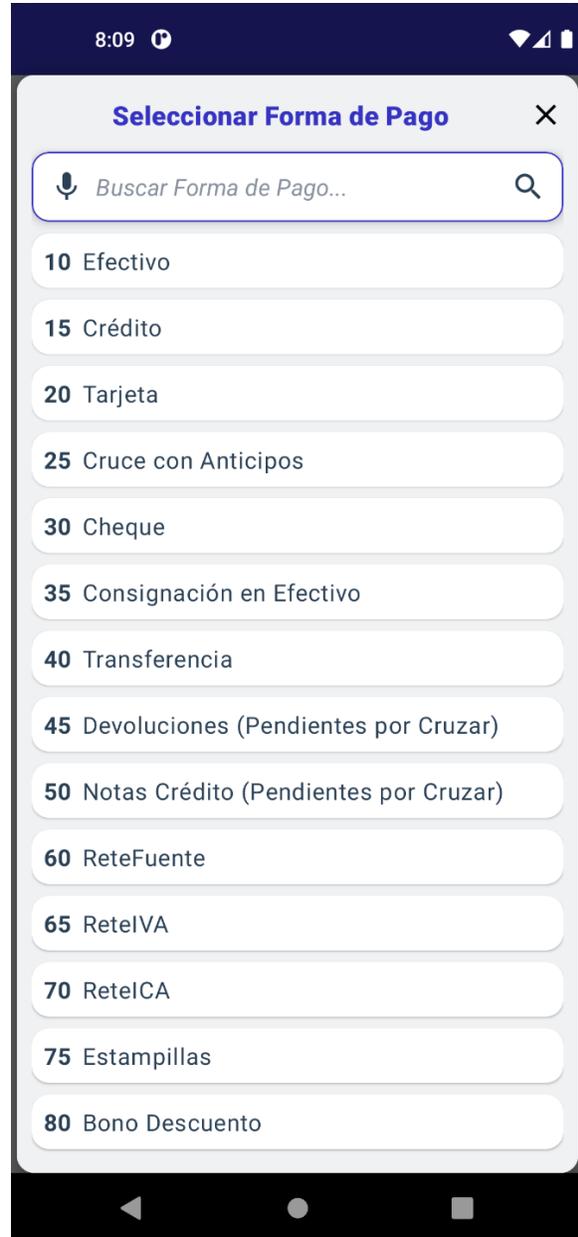
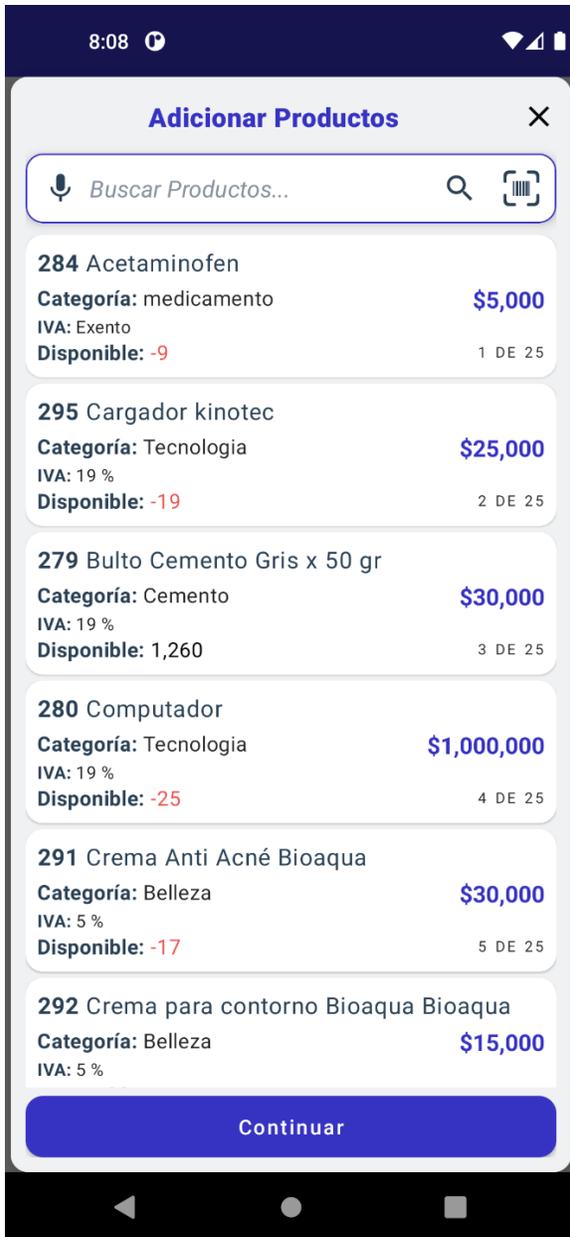
AMAZONAS

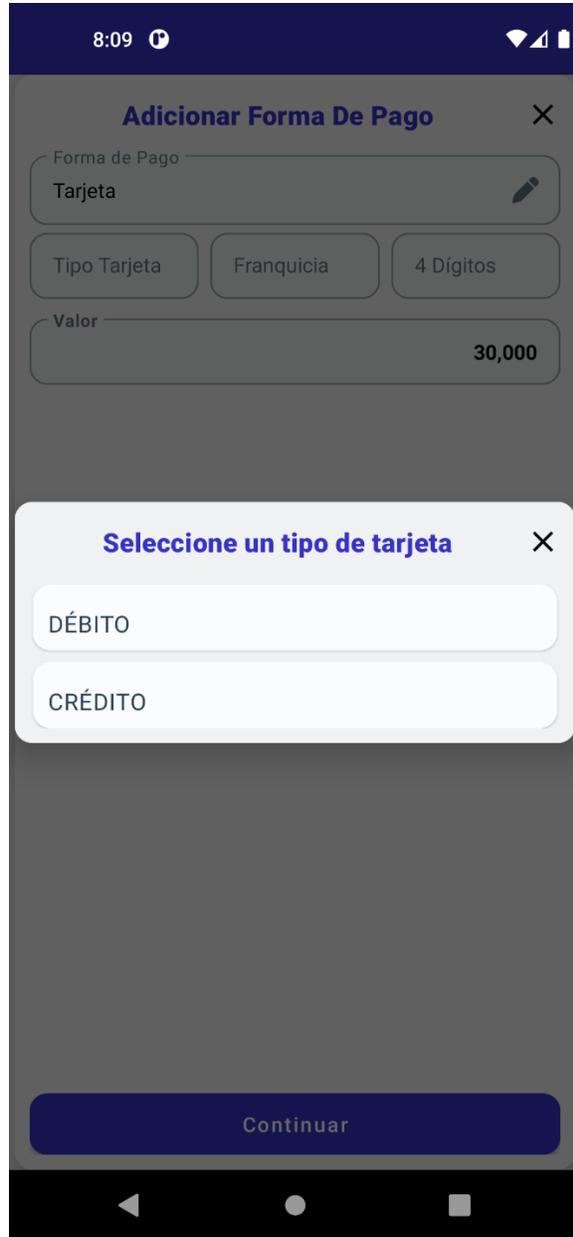
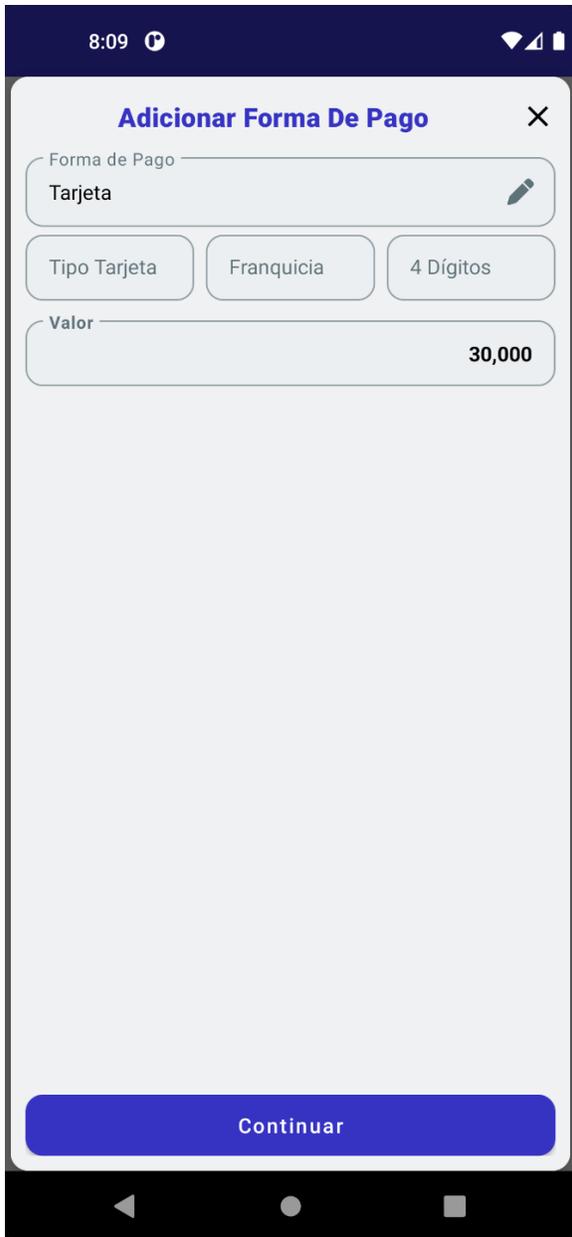
ARAUCA

ARCHIPIÉLAGO DE SAN ANDRÉS

CASANARE

CESAR





8:11

Buscar Pedidos

Tercero, Número, C.C/NIT ...

Demo Pruebas Por Cruzar
 Nit: 7777777777 \$110,000
 PV-17 04-08-2023 04:00 PM 1 de 9

Santiago Medina Por Cruzar
 Nit: 80181293 \$25,000
 PV-16 27-07-2023 10:32 PM 2 de 9

Demo Pruebas Por Cruzar
 Nit: 7777777777 \$580,000
 PV-15 19-07-2023 10:35 AM 3 de 9

Emanuel Palomino Por Cruzar
 Nit: 1098652147 \$25,000
 PV-14 13-07-2023 03:27 PM 4 de 9

Daniel Alejandro Potosi Por Cruzar
 Nit: 10305494 \$25,000
 PV-13 12-07-2023 02:54 PM 5 de 9

Juan Carlos Cruz Por Cruzar
 Nit: 89699655 \$30,000
 PV-12 11-07-2023 09:10 AM 6 de 9

Daniel Alejandro Potosi Por Cruzar
 Nit: 10305494

8:11

Documento PDF

Formato Pedido Ventas T80

DEMO PRUEBAS
 Demo Pruebas
 7.777.777.777 No Responsable de IVA e INC
 7777777
 null
 Direccion pruebas
 PEDIDOS VENTAS

NUMERO: PV17
 FECHA EXPEDICION: 04/08/2023 04:00 PM
 VENCIMIENTO: 05/08/2023 03:59 PM
 VENDEDOR: Administrador Sistema

CLIENTE
 NIT: 7777777777 NOM: Demo Pruebas
 TEL: 7777777
 DIR: Direccion pruebas POPAYÁN CAUCA Colombia

PRODUCTOS Y/O SERVICIOS

COD: 295 - Cargador kinotec
 IVA: 19 %
 CAN: 1 V/U: 25.000 V/T: 25.000

COD: 290 - Jabón Bioaqua Bioaqua
 IVA: 5 %
 CAN: 1 V/U: 25.000 V/T: 25.000

COD: 293 - Palleta de sombras Harmony Tredy
 IVA: Exento
 CAN: 1 V/U: 60.000 V/T: 60.000

EXCLUIDO:	0
EXENTO:	60.000
GRAVADO:	44.817,92
IC:	0
IVA:	5.182,08

8:14

Buscar Cotizaciones

Nombre Cliente, Número, C.C./NIT

Juan Carlos Cruz Por Facturar
Nit: 89699655
COT-27
01-08-2023 02:14 PM **\$25,000** 1 de 8

Juan Carlos Cruz Por Facturar
Nit: 89699655
COT-25
18-07-2023 10:09 AM **\$25,000** 2 de 8

Demo Pruebas Por Facturar
Nit: 7777777777
COT-24
14-07-2023 10:21 AM **\$640,000** 3 de 8

Juan Carlos Cruz Por Facturar
Nit: 89699655
COT-23
13-07-2023 03:48 PM **\$25,000** 4 de 8

Daniel Alejandro Potosi Por Facturar
Nit: 10305494
COT-22
11-07-2023 09:10 AM **\$2,000,000** 5 de 8

Consumidor Final Por Facturar
Nit: 222222222222
COT-16
04-07-2023 11:34 AM **\$30,000** 6 de 8

Santiago Medina Por Facturar
Nit: 80181202

8:14

Documento PDF

Formato Cotización T80

DEMO PRUEBAS
Demo Pruebas
7.777.777.777 No Responsable de IVA e INC
7777777
null
Direccion pruebas
COTIZACIÓN

NUMERO: COT24
FECHA EXPEDICION: 14/07/2023 10:21 AM
VENCIMIENTO: 15/07/2023 10:21 AM
VENDEDOR: Administrador Sistema

CLIENTE
NIT: 7777777777 NOM: Demo Pruebas
TEL: 7777777
DIR: Direccion pruebas POPAYÁN CAUCA Colombia

PRODUCTOS Y/O SERVICIOS

COD: 276 - Bulto PAPA
IVA: Excluido
CAN: 1 V/U: 50.000 V/T: 50.000

COD: 287 - Metros este es de prueba para ver la concatenacion Marca
IVA: 19 %
CAN: 1 V/U: 10.000 V/T: 10.000

COD: 286 - Impresora Bluetooth dig-381 Digital Pos
IVA: 19 %
CAN: 1 V/U: 580.000 V/T: 580.000

