

	UNIVERSIDAD FRANCISCO DE PAULA SANTANDER OCAÑA			
	Documento	Código	Fecha	Revisión
	FORMATO HOJA DE RESUMEN PARA TRABAJO DE GRADO	F-AC-DBL-007	08-07-2021	B
Dependencia	Aprobado		Pág.	
DIVISIÓN DE BIBLIOTECA	SUBDIRECTOR ACADEMICO		0(70)	

RESUMEN – TRABAJO DE GRADO

AUTORES	Brayan José Gutiérrez Pinzón		
FACULTAD	Ingenierías		
PLAN DE ESTUDIOS	Ingeniería de sistemas		
DIRECTOR	Luis Fernando Morales Martinez		
TÍTULO DE LA TESIS	Desarrollo de configuraciones que permitan la escalabilidad de la aplicación Agente Polaris en la empresa World Pos Solutions S.A.S		
TITULO EN INGLES	Development of configurations that allow the scalability of the Agente Polaris application in the company World Pos Solutions S.A.S		
RESUMEN			
Este proyecto tiene como finalidad mejorar y optimizar la aplicación Agente Polaris para lograr su adaptación al sistema operativo Android 10. Utilizando metodologías ágiles para el desarrollo de las actividades, reestructurando funcionalidades a nivel de código y documentando los procesos efectuados. Implementado pruebas unitarias y funcionales para garantizar el correcto funcionamiento de la aplicación de gestión de dispositivos móviles y así cumplir con los requisitos del estándar PCI-DSS.			
RESUMEN EN INGLES			
This project aims to improve and optimize the Agente Polaris application to achieve its adaptation to the Android 10 operating system. Using agile methodologies for the development of the activities, restructuring functionalities at code level and documenting the processes carried out. Implemented unit and functional tests to ensure the correct functioning of the mobile device management application and thus comply with the requirements of the PCI-DSS standard.			
PALABRAS CLAVES	SCRUM, Android, software, pruebas unitarias, pruebas funcionales, PCI-DSS, app, móvil, POS, versionamiento, JAVA		
PALABRAS CLAVES EN INGLES	SCRUM, Android, software, unit testing, functional testing, PCI-DSS, app, mobile, POS, versioning, JAVA		
CARACTERÍSTICAS			
PÁGINAS: 70	PLANOS: 0	ILUSTRACIONES: 30	CD-ROM: 0



**Desarrollo de configuraciones que permitan la escalabilidad de la aplicación Agente Polaris
en la empresa World Pos Solutions S.A.S**

Brayan José Gutiérrez Pinzón

Facultad de Ingenierías, Universidad Francisco de Paula Santander Ocaña

Ingeniería de Sistemas

Msc. Luis Fernando Morales Martínez

16 Junio del 2023

Índice

1. Desarrollo de configuraciones que permitan la escalabilidad de la aplicación Agente Polaris en la empresa World Pos Solutions S.A.S.....	10
1.1 Descripción de la empresa	10
1.1.1 Misión	11
1.1.2 Visión.....	11
1.1.3 Objetivos de la empresa	11
1.1.4 Descripción de la estructura organizacional	12
1.1.5 Descripción de la dependencia y/o proyecto asignado	12
1.2 Diagnóstico inicial de la dependencia.....	13
1.2.1 Planteamiento del problema.....	14
1.3 Objetivos de la pasantía	16
1.3.1 General.....	16
1.3.2 Específicos	16
1.4 Descripción de las actividades a desarrollar	17
2. Enfoques referenciales	19
2.1 Enfoque conceptual.....	19
2.2 Enfoque legal	21
3. Informe de cumplimiento del trabajo.....	23
3.1 Presentación de resultados	23
3.1.1 Introducción al proyecto	23

3.1.1.1 Marco de trabajo.....	23
3.1.1.2 Código fuente.....	25
3.1.2 Desarrollar las nuevas funcionalidades en la aplicación Agente Polaris implementando código compatible con el sistema operativo Android 10, considerando los requerimientos propuestos, y así llevar a cabo una transición de forma escalable.	27
3.1.2.1 Analizar los requerimientos actuales para efectuar el desarrollo.....	27
3.1.2.1.1 Requerimientos funcionales.....	28
3.1.2.1.2 Requerimientos no funcionales.....	28
3.1.2.2 Exponer las posibles soluciones e implantar la estimación de tiempos de desarrollo con el equipo de trabajo.	29
3.1.2.2.1 Roles.	29
3.1.2.2.2 Historias de Usuario.....	30
3.1.2.2.3 Product Backlog.....	31
3.1.2.2.4 Sprint planning.....	32
3.1.2.2.5 Daily Scrum..	33
3.1.2.3 Desarrollar las correcciones del error en las notificaciones de descarga y servicio.....	34
3.1.2.4 Desarrollar las correcciones del error en el AlertDialog..	36
3.1.2.5 Desarrollar las correcciones del error en el modo kiosko.....	38
3.1.2.6 Desarrollar las correcciones del error en la solicitud de permisos.....	40
3.1.2.7 Desarrollar las correcciones del error de obtención de coordenadas.....	41
3.1.2.8 Desarrollar las correcciones del error de inicialización del servicio de	

	4
consultas.....	43
3.1.2.9 Realizar el control de cambios periódicos en el repositorio según avance el desarrollo del proyecto.....	45
3.1.3 Realizar los casos de pruebas correspondientes, verificando el correcto funcionamiento en terminales NewPOS 9220 bajo el sistema operativo Android 10.....	46
3.1.3.1 Realizar las pruebas unitarias del desarrollo efectuado..	46
3.1.3.2 Realizar las pruebas funcionales de la aplicación.....	49
3.1.3.3 Registrar y corregir los errores evidenciados en las pruebas.....	51
3.1.4 Elaborar la documentación del desarrollo llevado a cabo, utilizando la herramienta Javadoc, facilitando con ello la orientación de los cambios futuros en el versionamiento.	52
3.1.4.1 Identificar los conceptos adecuados, para el desarrollo de la documentación de una manera técnica y acertada..	52
3.1.4.2 Revisar el código, para la identificación de métodos y clases relevantes utilizados en el desarrollo..	54
3.1.4.3 Realizar reuniones periódicas con el líder del proyecto, con el objetivo de retroalimentar la elaboración de la documentación.	59
4. Diagnóstico final.....	60
5. Conclusiones.....	61
6. Recomendaciones	63
Referencias.....	64

Lista de figuras

Figura 1. Estructura organizacional de la empresa WPOSS.....	13
Figura 2. Certificado de SCRUMstudy.....	25
Figura 3. Certificado de Backtrack Academy.....	27
Figura 4. Videos de capacitación.....	28
Figura 5. Reunión de socialización de la historias de usuario HU-003.....	31
Figura 6. Reunión de refinamiento del product backlog.....	32
Figura 7. Reunión de sprint planning.....	33
Figura 8. Reunión de Daily Scrum.....	34
Figura 9. Estructura básica de una notificación.....	36
Figura 10. Notificaciones del Agente Polaris.....	37
Figura 11. Estructura básica de un AlertDialog.....	38
Figura 12. AlertDialog del Agente Polaris.....	39
Figura 13. Estructura básica para una vista personalizada.....	40
Figura 14. Estructura básica para otorgar permisos.....	42
Figura 15. Estructura básica de la obtención de la geolocalización.....	43
Figura 16. Estructura básica WorkManager.....	45
Figura 17. Sourcetree, interfaz gráfica de Git.....	47
Figura 18. Estructura básica para pruebas con JUnit 5.....	49
Figura 19. Resultado exitoso de pruebas con JUnit 5 en Android Studio.....	49
Figura 20. Resultado fallido de pruebas con JUnit 5 en Android Studio.....	50

Figura 21. Clase con malas prácticas de programación detectado por SonarLint	54
Figura 22. Clase con buenas prácticas de programación detectado por SonarLint	55
Figura 23. Documentación en el código del Agente Polaris utilizando JavaDoc	56
Figura 24. Documentación de paquetes generada por JavaDoc	57
Figura 25. Documentación de clases generada por JavaDoc	57
Figura 26. Documentación de la estructura de las clases generada por JavaDoc	58
Figura 27. Documentación de los métodos generada por JavaDoc	58
Figura 28. Documentación detallada de los métodos generada por JavaDoc	59
Figura 29. Documentación de la estructura jerárquica generada por JavaDoc	59
Figura 30. Reunión periódica.....	60

Lista de tablas

Tabla 1. Matriz DOFA.....	13
Tabla 2. Actividades a realizar en la empresa WPOSS	17
Tabla 3. Roles de SCRUM dentro del proyecto Agente Polaris.....	29
Tabla 4. Estructura de un commit dentro del proyecto Agente Polaris	45
Tabla 5. Check list de pruebas funcionales.....	50
Tabla 6. Tags utilizados para la documentación con JavaDoc	55

Resumen

El presente documento tiene como objetivo describir el proceso llevado a cabo para lograr la escalabilidad de la aplicación Agente Polaris durante la pasantía llevada a cabo en la empresa WPOSS, en la cual se implementaron una serie de mejoras y optimizaciones que permitieron la adaptación al sistema operativo Android 10 y cumplir con los estándares de PCI-DSS.

En este informe se presentan los diferentes procesos realizados que se integran en el proyecto del Agente Polaris perteneciente al área de DMA, basándose en metodologías ágiles para la ejecución y realización de las actividades, la implementación de código fuente que ayude a realizar su respectivo proceso sin restricciones, como lo debe llevar a cabo una aplicación de gestión de dispositivos móviles, reestructurando funcionalidades, aplicando buenas prácticas durante el desarrollo documentando los procesos efectuados y cumpliendo con las respectivas pruebas unitarias y funcionales verificando el correcto funcionamiento del app.

Introducción

La seguridad de los datos y las transacciones financieras es un aspecto crítico en la industria de los pagos electrónicos y es por ello que los estándares de seguridad de datos de la industria de tarjetas de pago (PCI-DSS) se han vuelto cada vez más relevantes para empresas como World Pos Solutions (WPOSS). En particular, la necesidad de actualizar sus terminales de pago con un sistema operativo moderno y robusto se ha convertido en un requisito vital para cumplir con los estándares de seguridad brindando a sus clientes un nivel de protección adecuado.

En el presente trabajo de grado se aborda el proceso para llevar a cabo la integración del sistema operativo Android 10 a la aplicación del Agente Polaris. Este proceso es importante para la industria de los pagos electrónicos, en particular para la industria bancaria, ya que las aplicaciones MDM son fundamentales para garantizar la seguridad de los datos y otros procesos adicionales que lleva a cabo el proyecto, conociendo el uso y adquisición exponencial de estos dispositivos para los diferentes clientes de WPOSS. En consecuencia, es de vital importancia realizar la integración de manera adecuada y eficiente garantizando la escalabilidad permitiendo un resultado óptimo.

1. Desarrollo de configuraciones que permitan la escalabilidad de la aplicación Agente Polaris en la empresa World Pos Solutions S.A.S

1.1 Descripción de la empresa

World Pos Solutions (WPOSS) es una empresa multinacional con un poco más de 25 años de experiencia, a la vanguardia en el desarrollo de soluciones de software y comercialización de dispositivos de pago electrónico como impresoras, lectores de códigos de barras y terminales POS especializados. Con sedes a nivel nacional en las ciudades de Ocaña, Cúcuta, Bucaramanga, incluyendo Bogotá, además de contar con presencia en países como Bolivia, Paraguay, Ecuador y Estados Unidos. (WPOSS, 2021a)

Contando actualmente con productos y servicios que respaldan sus operaciones comerciales, y se presentan en los siguientes apartados mostrados a continuación:

- **Sector bancario:** Interviene en la implementación de servicios de corresponsalía no bancaria, donde se cuenta con la instalación de más de 45.000 puntos de venta en diversas instalaciones a lo largo de Latinoamérica. (WPOSS, 2021a)
- **Redes transaccionales:** Muchos desarrollos tecnológicos se han realizado basándose en la construcción de un amplio portafolio de transacciones como pagos, cobranza, seguros y asistencias, giros, pines y recargas, juegos de azar y loterías. (WPOSS, 2021a)
- **Medios de pago:** Comprometerse con promover el avance tecnológico mediante el desarrollo de nuevas soluciones de inclusión financiera y la expansión de todos los

métodos de pago. (WPOSS, 2021a)

- **E-commerce:** Brindar soluciones que permitan procesar pagos en línea, recurrencias y suscripciones, administrar su cartera y realizar ventas de forma remota, ofreciendo los más altos estándares de seguridad. (WPOSS, 2021a)

1.1.1 Misión

Ser aliado estratégico de nuestros clientes en la innovación y evolución de modelos de negocios de soluciones integrales, que incluyen paquetes completos de productos tecnológicos y servicios asociados, cumpliendo con los estándares de seguridad, altos niveles de servicio y calidad. (WPOSS, 2021b)

1.1.2 Visión

Ser la primera opción como socio estratégico y tecnológico de nuestros clientes en el mercado de América Latina. (WPOSS, 2021b)

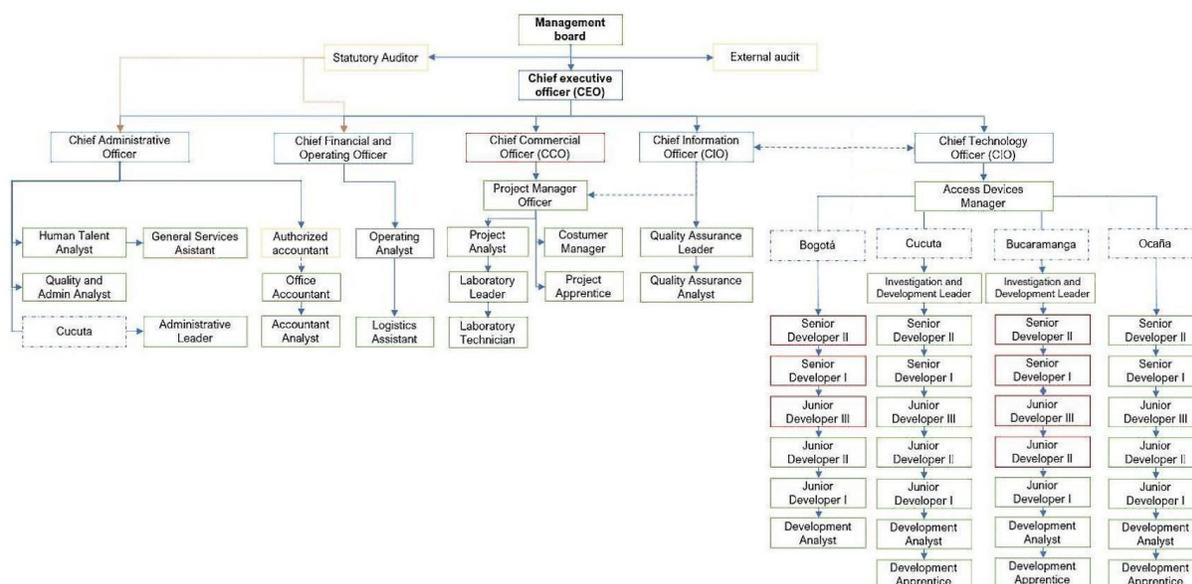
1.1.3 Objetivos de la empresa

- Obtener ventas de 14.000 millones de pesos anuales.
- Poseer un centro de desarrollo basado en altos estándares enfocado a brindar soluciones tecnológicas y de pago.
- Establecer un sistema de gestión de calidad.

1.1.4 Descripción de la estructura organizacional

Figura 1

Estructura organizacional de la empresa WPOSS



Nota. La imagen muestra la organización estructural en el presente año. (WPOSS, 2022)

1.1.5 Descripción de la dependencia y/o proyecto asignado

El área de desarrollo móvil enfocado a dispositivos Android se destaca por crear y brindar soluciones tecnológicas en el ámbito de software para instrumentos de hardware como lo son los POS, este espacio laboral como lo es DMA se divide en diferentes proyectos los cuales están compuestos por integrantes en los se puede destacar al líder de este y su mano derecha o sub-líder, dentro de todos estos proyectos, se encuentra el Agente Polaris el cual es una aplicación MDM (Mobile Device Management) la cual se ejecuta en segundo plano en variadas terminales bajo las marcas sobresalientes como NewPOS, WizarPOS, Sunmi, Dspread y Tousei, actuando como un apoyo estratégico para las aplicaciones financieras que la empresa también

desarrolla, permitiendo descarga de aplicaciones, llevar el monitoreo general del estado en que se encuentre el dispositivo en cuestión y envío de estos datos mediante un proceso transaccional, que permite de una manera coordinada obtener el control de funcionalidades varias de los terminales punto de venta.

1.2 Diagnóstico inicial de la dependencia

Tabla 1

Matriz DOFA

	Fortalezas (F)	Debilidades (D)
Matriz DOFA	Poseer una aplicación de calidad	Presentación de los requerimientos con una especificación no tan clara
	Contar con herramientas tecnológicas e infraestructura adecuadas en el área de trabajo	Mala documentación o falta de ella en algunos apartados del proyecto
	La utilización del marco de trabajo SCRUM (adaptado a las necesidades de WPOSS) para el desarrollo ágil de software	Confusión dentro de los proyectos en la organización de las actividades
		Poco material que brinde capacitación para instruir al nuevo personal
Oportunidades (O)	FO	DO
Posibilidad de contar con un producto y servicio basado en estándares de alta excelencia, actualizable y escalable	Implantar soluciones de software basadas en un desarrollo ágil, con el objeto de migraciones continuas, para alcanzar un producto escalable	Poner en práctica un registro constante de las actividades realizadas basado en normas de calidad, acompañado de la creación de un material para capacitación basado en un trabajo conjunto con la guía de los líderes del equipo de desarrollo del proyecto actual

Colaborar conjuntamente en equipos de trabajo con un amplio conocimiento en el mercado nacional e internacional	Implementar estándares de control calidad para obtener resultados óptimos acompañado de la intervención y colaboración del equipo de trabajo	Documentar de manera integral las estrategias implementadas en el Agente Polaris, con el objetivo de darle continuidad en la implementación de nuevas características que permitan facilitar los procesos de escalabilidad
Uso e implementación de nuevas tecnologías		
Amenazas (A)	FA	DA
Modificaciones en los estándares de políticas de seguridad y privacidad en el que se basa el proyecto	Apoyarse en planes estratégicos que implanten una directriz fija en la utilización de marcos de trabajo ágiles para evitar retardos en las actividades del Agente Polaris	Implementar una estructura de actividades clara que ayude al desarrollo y eviten el retraso de estas mismas
Los rápidos cambios en la tecnología requieren una actualización constante de hardware y software	Utilizar la excelente estructura del proyecto y herramientas varias del área para lograr un producto adaptable a los cambios	Realizar documentación comprensible para el entendimiento de integrantes futuros y a su vez permitir preservar un producto flexible y ajustable a las diferentes variaciones posteriores
Retrasos en el desarrollo debido a las necesidades cambiantes del cliente		

1.2.1 Planteamiento del problema

La empresa WPOSS, con el afán de mantener sus estándares de calidad para poseer productos y servicios estables, brinda soluciones modernas para el constante beneficio de sus clientes. Además, mantiene sus aplicaciones y hardware libres de ciberamenazas en sus procesos transaccionales. Estos llevan a cabo procedimientos en los cuales se hace el uso de estándares de pagos seguros y seguridad de datos basado en la industria de tarjetas de pago y seguridad de la información, como lo es PCI-DSS (Payment Card Industry - Data Security Standard). (PCI Security Standards Council, 2018) En el presente año se llevó a cabo la publicación de su reciente versión 4.0 y han realizado algunos cambios en los cuales se hace más énfasis en el uso

de sistemas de seguridad robustos y actualizados. (PCI Security Standards Council, 2022)

Debido a esas políticas a cumplir, los distribuidores de terminales POS para WPOSS, como NewPOS, han decidido llevar a cabo la fabricación de dispositivos basados en el sistema operativo (SO) Android 10. Si nombramos algunas de las ventajas al usar este, se enfoca en la privacidad y seguridad, favoreciendo al usuario para obtener mayor protección, transparencia y control (Android Developers, 2019), además de cumplir con los requerimientos de PCI-DSS presentes en su versión actual.

Actualmente, el cliente de WPOSS a implantar la solución se encuentra ubicado en el país de Perú, como herramientas de hardware utiliza estas terminales nombradas en el párrafo anterior en su modelo NewPOS 9220, el cual corre bajo el SO Android 5 y Android 7. Debido a la problemática expuesta en la presente redacción, se requiere realizar una migración o escalabilidad de la aplicación Agente Polaris hacia el SO Android 10 sin afectar el flujo actual que se lleva a cabo en los anteriores SO mencionados. Se espera que este pueda funcionar de una manera correcta y estable como lo realiza actualmente y, a su vez, que se desempeñe de manera sólida en este sistema operativo para poder seguir brindando el apoyo estratégico a la aplicación financiera del cliente una vez se realice el desarrollo de las configuraciones pertinentes.

1.3 Objetivos de la pasantía

1.3.1 General

Desarrollar configuraciones que permitan la escalabilidad de la aplicación Agente Polaris en la empresa World Pos Solutions S.A.S, mediante el uso de herramientas de programación orientada al lenguaje JAVA.

1.3.2 Específicos

- Desarrollar las nuevas funcionalidades en la aplicación Agente Polaris implementando código compatible con el sistema operativo Android 10, considerando los requerimientos propuestos, y así llevar a cabo una transición de forma escalable.
- Realizar los casos de pruebas correspondientes, verificando el correcto funcionamiento en terminales NewPOS 9220 bajo el sistema operativo Android 10.
- Elaborar la documentación del desarrollo llevado a cabo, utilizando la herramienta Javadoc, facilitando con ello la orientación de los cambios futuros en el versionamiento.

1.4 Descripción de las actividades a desarrollar

Tabla 2

Actividades a realizar en la empresa WPOSS

Objetivo General	Objetivos Específicos	Actividades a desarrollar
Desarrollar configuraciones que permitan la escalabilidad de la aplicación Agente Polaris en la empresa World Pos Solutions S.A.S, mediante el uso de herramientas de programación orientada al lenguaje JAVA.	Desarrollar las nuevas funcionalidades en la aplicación Agente Polaris implementando código compatible con el sistema operativo Android 10, considerando los requerimientos propuestos, y así llevar a cabo una transición de forma escalable.	Analizar los requerimientos actuales para efectuar el desarrollo.
		Exponer las posibles soluciones e implantar la estimación de tiempos de desarrollo con el equipo de trabajo.
		Desarrollar las correcciones del error en las notificaciones de descarga y servicio.
		Desarrollar las correcciones del error en el AlertDialog.
		Desarrollar las correcciones del error en el modo Kiosko.
		Desarrollar las correcciones del error en la solicitud de permisos.
		Desarrollar las correcciones del error de obtención de coordenadas.
		Desarrollar las correcciones del error de inicialización del servicio de consultas.
		Realizar el control de cambios periódicos en el repositorio según avance el desarrollo del proyecto.

Objetivo General	Objetivos Específicos	Actividades a desarrollar
	<p>Realizar los casos de pruebas correspondientes, verificando el correcto funcionamiento en terminales NewPOS 9220 bajo el sistema operativo Android 10.</p>	<p>Realizar las pruebas unitarias del desarrollo efectuado.</p> <hr/> <p>Realizar las pruebas funcionales de la aplicación.</p> <hr/> <p>Registrar y corregir los errores evidenciados en las pruebas.</p>
	<p>Elaborar la documentación del desarrollo llevado a cabo, utilizando la herramienta Javadoc, facilitando con ello la orientación de los cambios futuros en el versionamiento.</p>	<p>Identificar los conceptos adecuados, para el desarrollo de la documentación de una manera técnica y acertada.</p> <hr/> <p>Revisar el código, para la identificación de métodos y clases relevantes utilizados en el desarrollo.</p> <hr/> <p>Realizar reuniones periódicas con el líder del proyecto, con el objetivo de retroalimentar la elaboración de la documentación.</p>

2. Enfoques referenciales

2.1 Enfoque conceptual

- **Android:** Actualmente, este es un sistema operativo móvil muy popular, en la actualidad Google es su propietario, basado en el núcleo Linux y que es de código abierto. Esto nos permite el fácil desarrollo de aplicaciones móviles, las cuales son utilizadas en una inmensa cantidad de dispositivos de todo tipo, que corran bajo este sistema operativo. (DiMarzio, 2016) El cual, es una parte fundamental para la integración y el desarrollo de las distintas soluciones de software con los diferentes terminales punto de venta que posee WPOSS.
- **Android Studio:** Este se caracteriza por ser el entorno de desarrollo (IDE) oficial utilizado para el desarrollo de aplicaciones móviles basadas en Android. El cual tiene su base en IntelliJ IDEA, que se caracteriza por ser una herramienta potente en la edición de código. (Android Developers, 2023a) Siendo este un instrumento de apoyo estratégico para el desarrollo de aplicaciones móviles para los distintos clientes de WPOSS.
- **Aplicación móvil:** Estas herramientas se pueden definir como un conjunto de software el cual se basa en la tecnología móvil y que se ejecutan en diferentes dispositivos, como smartphones, smartwatches o tabletas. (Chandran, et al., 2022) Estas aplicaciones se utilizan en diferentes campos de la industria en general y facilitan día a día nuestras tareas cotidianas.
- **Bitbucket:** Este es un servicio de almacenamiento en la nube de repositorios, en su mayoría de código fuente, el cual es utilizado para la gestión de versionamiento de proyectos de software, (Uzayr, 2022) para así, colaborar entre los diferentes miembros de

una empresa u organización, así como lo integra WPOSS en todos sus proyectos de desarrollo de software.

- **Git:** Se basa en una herramienta de código abierto que fue creado por Linus Torvalds, el cual es un sistema de control de versiones que ayuda simplificando la colaboración entre usuarios en un mismo proyecto, facilitando los cambios de estos cuando este crece a gran escala. (Pathak, 2020)
- **Java:** Este es un lenguaje de programación conocido por su larga trayectoria en el desarrollo de software, y que, a día de hoy, sigue siendo una excelente herramienta para llevar a cabo la construcción de sistemas administrativos que actualmente son estables y que, a su vez, este sigue siendo una elección actual para iniciar en el mundo de la programación. (Nadal, 2021) Siendo el lenguaje nativo que se utiliza en la gran mayoría de proyectos de desarrollo móvil en la empresa WPOSS.
- **Javadoc:** Se considera como un formato especial que se utiliza para documentar comentarios de código fuente en el lenguaje de programación Java, el cual, permite una organización y descripción de los diferentes elementos de un proyecto, como lo son las clases o métodos pertenecientes a este. (Brown, et al., 2022)
- **MDM app:** En pocas palabras, podemos definir las aplicaciones de administración de dispositivos móviles como un software el cual utiliza diferentes elementos que ofrecen los smartphones para poder gestionar las diferentes actividades que se puedan ejercer en ellos y así, tener un control más preciso en los múltiples usos que tienen estos en el exponencial sector empresarial. (Barthwal, 2016)
- **POS:** Los dispositivos point of sale (POS) o punto de venta, en español, se caracterizan por ser un instrumento de hardware el cual contiene periféricos, los cuales facilitan los

procesos transaccionales de ventas o compras, mediante software especializado y se utilizan en los diferentes tipos de organizaciones relacionadas con estos procesos.

(Lestaringati, 2018)

- **SonarLint:** Desarrolladora por SonarSource, es una herramienta que se utiliza como extensión en los IDE de desarrollo como JetBrains para ayudar a codificar utilizando o sugiriendo en tiempo real mejores prácticas contextualizando en todo momento al programador sobre el paso recomendado (SonarSource, s.f.) esto con el fin de conseguir un producto final limpio y sostenible a futuro.
- **Sourcetree:** Podemos definir Sourcetree como un software que facilita la tarea de gestión del versionamiento mediante el uso de una interfaz gráfica (GUI), reemplazando los tradicionales comandos de GIT para estas tareas, el cual facilita y mejora la experiencia de usuario para mejorar el flujo de trabajo. (Chakraborty & Aithal, 2022)
- **Scrum:** Esta se define en muchas ocasiones como una metodología, la cual, se basa en un marco de trabajo repetitivo y progresivo, conociéndose como una opción muy popular en la adopción de actividades o proyectos relacionados con el desarrollo de software, dentro de entornos empresariales. (Lasa, et al., 2017)

2.2 Enfoque legal

- **PCI DSS:** En la actualidad, Payment Card Industry Data Security Standard (PCI DSS) se considera como un estándar de seguridad de datos, el cual, se creó por las cinco marcas de pago fundadoras del PCI Security Standards Council, como lo son American Express, Discover Financial Services, JCB International, Mastercard y Visa. Este estándar establece requisitos y procesos estandarizados para la gestión de seguridad transaccional

que permiten salvaguardar la información de los tarjetahabientes. Donde, si una empresa o entidad financiera, contrata un tercero para realizar un software que ayude a suplir sus necesidades operacionales, este último está obligado a respaldar el cumplimiento de PCI DSS. (Calder & Williams, 2019)

- **ISO 8583:** La ISO 8583 se basa en un estándar internacional que define un conjunto de reglas y procedimientos para la transferencia segura y eficiente de mensajes de transacciones financieras entre diferentes sistemas y aplicaciones web o móviles. Esta norma se usa ampliamente en el campo de los pagos electrónicos y es compatible con muchos sistemas de procesamiento de pagos en todo el mundo. (ISO, 2003)

3. Informe de cumplimiento del trabajo

3.1 Presentación de resultados

Antes de comenzar con la implementación y explicación de las distintas soluciones para las actividades efectuadas, es importante destacar el proceso de inducción llevado a cabo en la empresa WPOSS para realizar una iniciación previa dentro del proyecto y área asignada.

3.1.1 *Introducción al proyecto*

Dentro de la empresa WPOSS es indispensable contar con un personal calificado que adopte el conocimiento previo del flujo de trabajo que se sigue dentro de esta, y así, lograr el éxito para entregar productos de calidad, es por ello, que dentro de las distintas áreas que conforman la sede Ocaña, se realizan distintas capacitaciones que ayuda a un mejor empalme en el conocimiento con respecto a los proyectos, para los nuevos participantes en estos.

3.1.1.1 Marco de trabajo. En primera instancia, cabe destacar que WPOSS adopta las metodologías ágiles para el desarrollo de software. Por ende, estas ayudan a que los procesos dentro de los distintos proyectos sean flexibles, a la vez que sean adaptativos cuando se trata de la orientación al distinto personal en sí, donde, una actividad principal se puede dividir en actividades más pequeñas para que sean repartidas en los distintos miembros del equipo, y con esto, llevar a cabo un fácil manejo en las entregas constantes de software que se realizan gracias a la implementación de estas. (Montero, et al., 2018) Actualmente, en el área de DMA para el proyecto del Agente Polaris se utiliza el marco de trabajo SCRUM, el cual es adaptado a las necesidades de este, es decir, no se realiza la implementación de este al pie de la letra como lo

demanda cualquier guía básica, sino que se omiten ciertos pasos en la ejecución de los proyectos y la realización de sus distintas actividades.

Siendo indispensable que los nuevos integrantes de los equipos adopten los conceptos básicos de SCRUM, se realiza un curso introductorio en la plataforma SCRUMstudy, donde se definen varios conceptos en los cuales se incluyen su marco de trabajo, roles, eventos y artefactos para certificar el uso de este, según lo necesite la empresa para ayudar a cumplir con todas las actividades de desarrollo de software en un ambiente controlado y ordenado.

Figura 2

Certificado de SCRUMstudy



Nota. La imagen muestra el certificado Scrum Fundamentals Certified de SCRUMstudy.

3.1.1.2 Código fuente. Para comenzar con una introducción adecuada al código fuente del Agente Polaris, es necesario tener conocimientos introductorios en la creación de software seguro para evitar vulnerabilidades dentro del proyecto en sí y lograr prevenir riesgos asociados a la seguridad teniendo en cuenta un modelo definido que ayude a cumplir con políticas de protección centradas en aplicaciones móviles MDM, (Cardona, 2017) constituyéndose como una tarea muy importante para esta misma, la cual se debe cumplir a cabalidad, para causar un impacto de confidencialidad hacia el cliente en cuestión, por eso, se realiza un curso con certificación en la plataforma Backtrack Academy sobre la introducción al desarrollo seguro, el cual abarca aspectos fundamentales a tener en cuenta hacia los lineamientos del flujo de desarrollo de software, teniendo en cuenta todas las fases que se involucran en este, y así, tener la capacidad de solucionar temas centrados en la materia si se llega a presentar el caso.

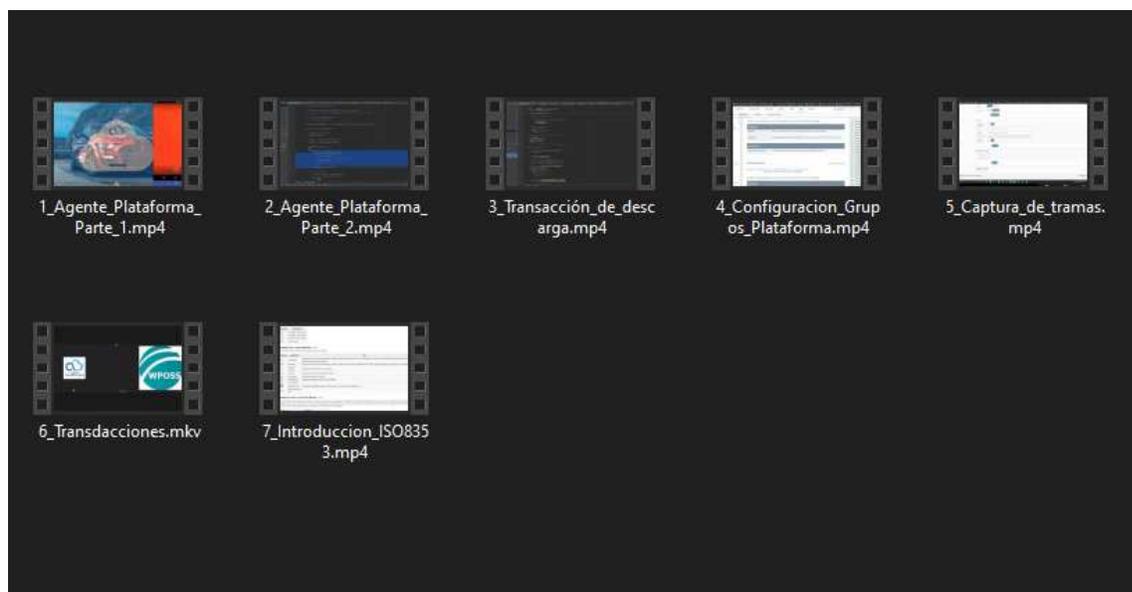
Figura 3*Certificado de Backtrack Academy*

Nota. La imagen muestra el certificado de introducción al desarrollo seguro de la plataforma BackTrack Academy.

Seguido de esto, se empiezan a realizar las capacitaciones que ofrece el área de DMA para el proyecto del Agente Polaris, las cuales consisten en una serie de videos donde se abarcan temas relacionados con la estructura del proyecto base en sí, donde se tocan algunos requerimientos, así, como el flujo de mensajería mediante el estándar ISO 8583, además, de herramientas adicionales que ayudan a suplir necesidades dentro de este como decodificadores y sistemas de control de versionamiento.

Figura 4

Videos de capacitación



Nota. La imagen muestra los videos de las capacitaciones del proyecto Agente Polaris.

3.1.2 Desarrollar las nuevas funcionalidades en la aplicación Agente Polaris implementando código compatible con el sistema operativo Android 10, considerando los requerimientos propuestos, y así llevar a cabo una transición de forma escalable.

3.1.2.1 Analizar los requerimientos actuales para efectuar el desarrollo. Para llevar a cabo la solución a implementar, se requiere tener un análisis previo de estos requerimientos, los cuales no ayudan a profundizar sobre el funcionamiento de la aplicación Agente Polaris para poder realizar de una forma más precisa el cumplimiento de las expectativas del cliente es cuestión. (Rodríguez, 2017) Esta información fue proporcionada por la empresa, que ya define unos requerimientos actuales para el proyecto, los cuales, se deben cumplir a cabalidad para realizar una transición correcta en la adopción del sistema operativo Android 10 y que se

definirán a continuación, dividiéndose en requerimientos funcionales y no funcionales:

3.1.2.1.1 Requerimientos funcionales.

- El protocolo de mensajería utilizado es ISO8583.
- El agente debe permitir la descarga de archivos usando eventos ISO8583.
- El agente debe mantener una sesión para el archivo que se está descargando para que continúe descargando incluso cuando la terminal no está en uso.
- El agente debe permitir que la aplicación descargada se instale y mostrar un mensaje al usuario que permita que el POS se reinicie.
- El agente deberá permitir obtener su rectificación en cada consulta realizada.
- El agente debe permitir que se ejecuten comandos AT para encontrar los datos de la antena a la que está conectado el módem GPRS/3G/4G.
- El agente debe habilitar diversas tecnologías de comunicación como LAN y WIFI.
- El agente debe habilitar la parametrización del switch Polaris IP/puerto.
- El agente debe proporcionar una interfaz a la aplicación host para volver a recibir los datos de contacto (IP/Puerto).

3.1.2.1.2 Requerimientos no funcionales.

- El software de desarrollo debe funcionar independientemente de otras aplicaciones instaladas.
- El proceso de consulta y descarga NO DEBE bloquear la aplicación principal.
- El POS también debe permitir la actualización de la misma aplicación desde un agente Polaris.

3.1.2.2 Exponer las posibles soluciones e implantar la estimación de tiempos de desarrollo con el equipo de trabajo.

3.1.2.2.1 Roles. Antes de conocer cualquier otro proceso relacionado con la planeación de las actividades a ejecutar, es importante tener claro el procedimiento llevado a cabo para la utilización de SCRUM adaptado al proyecto del Agente Polaris. Dentro de la definición del equipo, este se divide en los siguientes roles dentro del proyecto, como se muestra en la tabla siguiente:

Tabla 3

Roles de SCRUM dentro del proyecto Agente Polaris

Rol		Tarea
Product Owner		Representante de los stakeholders. Define los criterios de aceptación.
Scrum Master		Lidera y modera un ambiente laboral óptimo en el equipo.
Equipo Scrum	Equipo de desarrollo	Encargados del desarrollo y pruebas del producto de software.
	Equipo de pruebas	

Nota. La información es tomada de (SCRUMstudy, 2017).

3.1.2.2 Historias de Usuario. Los artefactos de SCRUM se basan en herramientas que son utilizadas para definir y registrar las actividades a realizar durante la ejecución del proyecto y llevar un control de estas. Entre los distintos instrumentos encontramos las historias de usuario (HU) que nos brindan una introducción de lo que requiere y espera de las funcionalidades a implementar el cliente para el producto final a entregar, (Carrasco, et al., 2019) en este caso se elaboraron las respectivas HU por parte del product owner las cuales previamente se socializan con el equipo de desarrollo para determinar cambios o añadir un punto de vista que aporte a estas, si realmente se requiere.

Figura 5

Reunión de socialización de las historias de usuario HU-003

The screenshot shows a Zoom meeting in progress. The main window displays a Microsoft Word document titled "Historia de usuario" (User Story) for "HU-003". The document content is as follows:

Datos de diligenciamiento	
Nombre	[Redacted]
Cargo	Product Owner
Cliente	[Redacted]
Fecha creación	28/02/2023
Fecha de Refinamiento	10/03/2023
Nombre de la historia de usuario	
HU-003	
Descripción	
Yo como [Redacted] quiero poder configurar los terminales NEWPOS9220 en modo quíscico y utilizar solo la aplicación financiera.	
Funcionalidades afectadas	
Modo quíscico	
Anexos	

The meeting interface includes a toolbar at the top with icons for "Controlar", "Separar", "Chat", "Gente", "Participar", "Reaccionar", "Vista", "Aplicaciones", "Más", "Cámara", "Microfono", and "Comparte". A "Mostrar participantes" (Show participants) button is visible. On the right side, there is a vertical list of participant avatars, with the top three being visible.

Nota. La imagen muestra la reunión llevada a cabo por el equipo de SCRUM del Agente Polaris.

3.1.2.2.3 Product Backlog. Al momento de definir una estimación entre las distintas tareas a llevar a cabo durante un sprint encontramos al Product Backlog, este se basa en una lista de todas las actividades priorizadas basadas en los formatos de HU previamente realizadas por el product owner. (Sachdeva, 2016) Junto a este miembro del equipo, se realiza la priorización y estimación debida de cada HU acompañados del equipo de desarrollo y el Scrum Master, para definir el comienzo del sprint y contar así con una especie de cronograma de actividades a realizar totalmente priorizado (Carrasco, et al., 2019) que se consultará y se le dará el debido seguimiento en el desarrollo de estas.

Figura 6

Reunión de refinamiento del product backlog

The screenshot shows a Microsoft Teams meeting window with a Google Sheet open. The sheet is titled 'DESARROLLO MIGRACION ANDROID 10 - AGENTE POLARIS'. It contains a Gantt chart and a resource allocation table. The table has columns for 'MES 1', 'MES 2', 'MES 3', 'MES 4', and 'MES 5', each with sub-columns for days 1-4. The resource 'BRAYAN GUTIERREZ' is assigned to various tasks across these months.

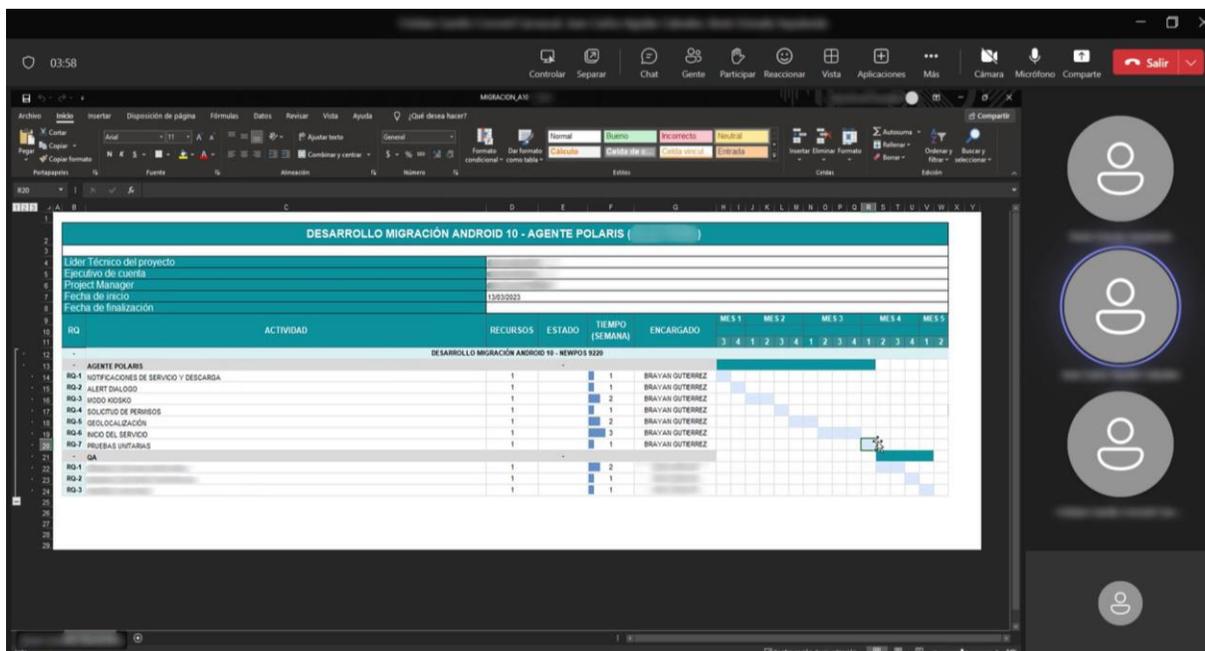
RS	ACTIVIDAD	RECURSOS	ESTADO	TIEMPO (SEMANA)	ENCARGADO	MES 1	MES 2	MES 3	MES 4	MES 5	
DESARROLLO MIGRACION ANDROID 10 - NUESTROS S220						3	4	1	2	3	4
RS1	NOTIFICACIONES DE SERVICIO Y DESCARGA	1	1	1	BRAYAN GUTIERREZ						
RS2	ALISTO DESARROLLO	1	1	1	BRAYAN GUTIERREZ						
RS3	INDIO ANDROID	1	1	2	BRAYAN GUTIERREZ						
RS4	SOLICITO DE PERMISOS	1	1	0	BRAYAN GUTIERREZ						
RS5	RESOLUCION DE ERRORES	1	1	0	BRAYAN GUTIERREZ						
RS6	INDIO DEL SERVICIO	1	1	0	BRAYAN GUTIERREZ						
RS7	PRUEBAS FUTURAS	1	1	0	BRAYAN GUTIERREZ						
RS8	QA	1	1	0							
RS9		1	1	0							
RS10		1	1	0							
RS11		1	1	0							

Nota. La imagen muestra la reunión llevada a cabo por el equipo de SCRUM del Agente Polaris.

3.1.2.2.4 Sprint planning. Los eventos de SCRUM utilizados mientras se llevaba a cabo el proyecto, cumplen como objetivo llevar un control de seguimiento al realizar las actividades periódicas, monitoreando de manera constante los avances dentro del desarrollo de estas. El planeamiento del sprint tiene la finalidad de exponer los resultados del product backlog refinado con anterioridad para poder analizarlo y definir así las responsabilidades que tendrá cada integrante en las diferentes tareas y así, comprometerse para cumplir con el objetivo del sprint y el plan de acción a ejecutar, el cual es organizada por el scrum master. (Sachdeva, 2016)

Figura 7

Reunión de sprint planning



Nota. La imagen muestra la reunión llevada a cabo por el equipo de SCRUM del Agente Polaris.

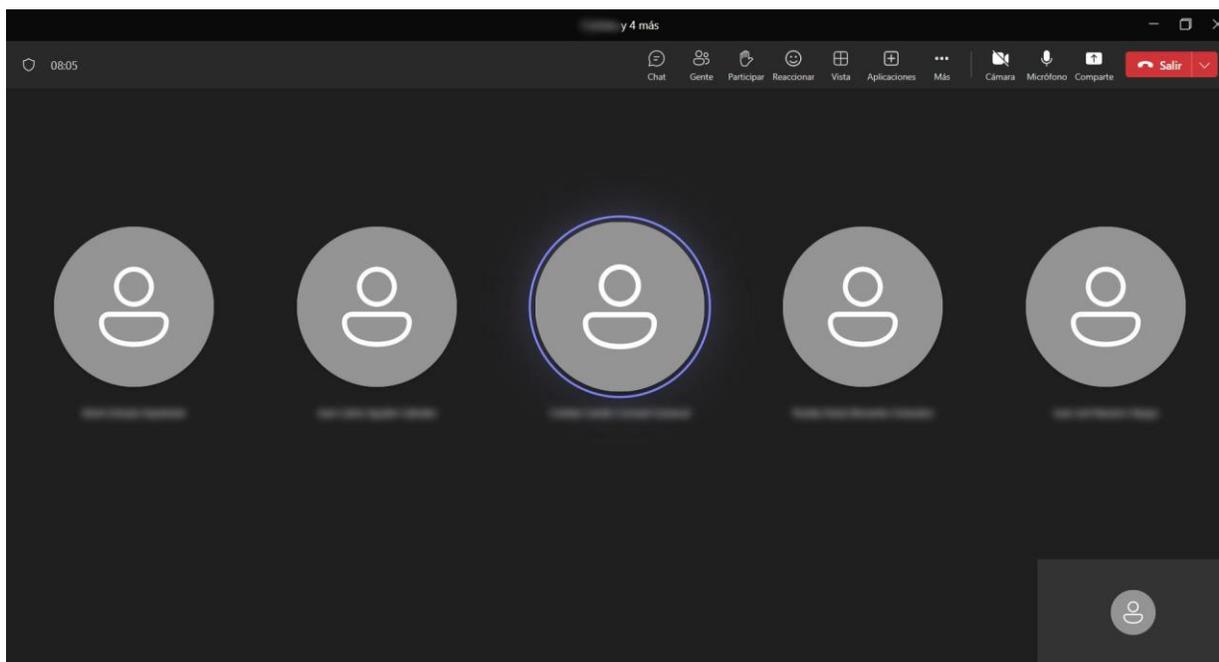
Para esta etapa se realizaron el planeamiento de dos Sprint's en los cuales participó el equipo de desarrollo, donde el primero contaba con el desarrollo de la implementación de las

herramientas para lograr la integración del Agente Polaris al sistema operativo Android 10 y el segundo consistió en la corrección de incidencias que reportó el equipo de QA (Aseguramiento de calidad) al finalizar y evaluar el primer Sprint.

3.1.2.2.5 Daily Scrum. Para verificar el trabajo realizado el día anterior y las actividades a realizar en el transcurso del día, se realiza el uso de reuniones como el Daily Scrum, el cual solo se realiza en un corto periodo de tiempo, (Sachdeva, 2016) monitoreando así el flujo de trabajo que se está llevando a cabo en el sprint.

Figura 8

Reunión de Daily Scrum



Nota. La imagen muestra la reunión llevada a cabo por el equipo de SCRUM del Agente Polaris.

3.1.2.3 Desarrollar las correcciones del error en las notificaciones de descarga y servicio. El Agente Polaris con el afán de lograr comunicar al usuario de los procesos que realiza en su funcionamiento hace uso de las notificaciones Android, actualmente, este hace uso de tres procesos de alertas, entre los cuales encontramos el estado de inicio del servicio, que no indica que esta está activo y funcionando, el segundo trata sobre el proceso de descarga de apk, el cual nos muestra el porcentaje que llevan estas para completarse para su posterior instalación, además de una notificación que nos indica si este proceso de descarga por algún factor interno o externo falla.

En los presentes dispositivos NEW9220 con sistema operativo Android 5 y 7 estas notificaciones que genera el Agente Polaris funcionan sin ningún problema, pero cuando entramos al entorno de Android 10 estas no se están visualizando, así se estén cumpliendo estos procesos mencionados anteriormente, esto se debe a que actualmente se hace uso de las clases Notification.Builder y NotificationManager. La solución planteada es usar o añadir canales de notificación, para crear estas. NotificationChannel es una clase que se incluyó a partir de la versión 8.0 de Android, esta permite una categorización dentro de las distintas notificaciones que genere la aplicación en cuestión, lo cual permite tener un control y modificar de manera independiente cada mensaje de alerta en aspecto visual y de sonido. (Yoshiizumi & Zhang, 2020) Por eso, es indispensable implementar esta solución, la cual también nos ayudará en versiones de android superiores a implantar esta, para lograr una escalabilidad mayor a futuro.

Figura 9*Estructura básica de una notificación*A screenshot of a code editor window with a dark background. The code is in Java and defines a method named showNotification. The code includes comments in Spanish for various fields and methods. The code is as follows:

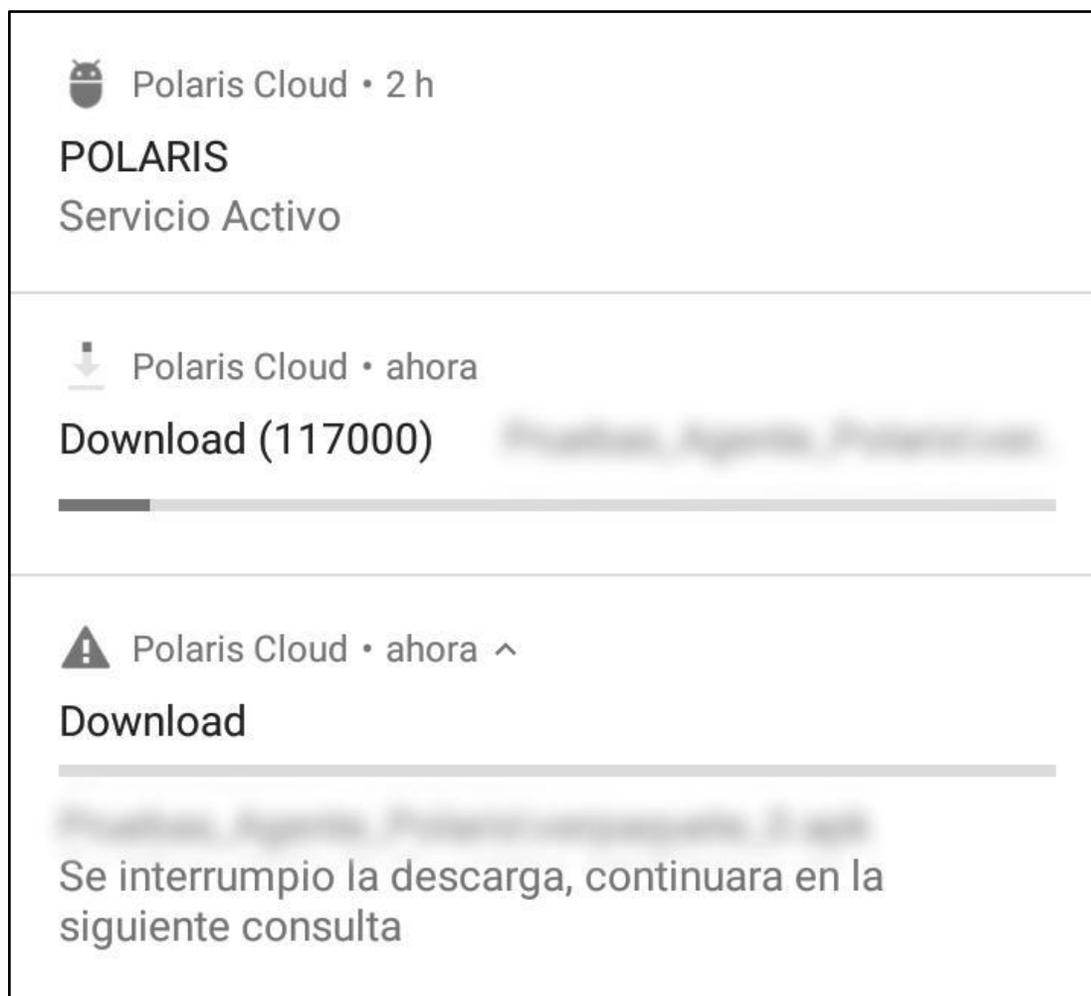
```
public static void showNotification() {
    String channelId = "ID del canal";
    String channelName = "Nombre del canal";
    int importanceNotification = NotificationManager.IMPORTANCE_LOW;
    int notificationID = 0;
    mNotificationManager = (NotificationManager) this.getSystemService(Context.NOTIFICATION_SERVICE);
    NotificationChannel notificationChannel = new NotificationChannel(channelId, channelName,
                                                                    importanceNotification);
    mNotificationManager.createNotificationChannel(notificationChannel);
    mNotifyBuilder = new Notification.Builder(context, ID)
        .setContentTitle("Titulo de la notificación")
        .setContentText("Texto de la notificación")
        .setContentSound(null, 0)
        .setSmallIcon(Icono de la notificación)
        .setAutoCancel(false)
        .build();
    mNotificationManager.notify(notificationID, mNotifyBuilder);
}
```

Nota. La imagen muestra la estructura básica de una notificación utilizando NotificationChannel.

Una vez realizados y añadidos los respectivos ajustes, podemos visualizar las distintas notificaciones que el Agente Polaris alerta en sus distintos servicios y transacciones.

Figura 10

Notificaciones del Agente Polaris



Nota. La imagen muestra las tres distintas notificaciones que genera el Agente Polaris.

3.1.2.4 Desarrollar las correcciones del error en el AlertDialog. Siguiendo con elementos claves, los cuales ayudan a brindar un mejor flujo comunicativo entre el usuario y la app móvil, se encuentran los AlertDialog, estas son ventanas emergentes que ayudan al usuario interactuar y realizar acciones determinadas según la decisión que toma el actor que interactúa, (Android Developers, 2022b) Para este caso, se utiliza para comunicar e informar sobre los

procesos que se realizan con éxito. Actualmente, el Agente Polaris presenta un problema al tratar de generar estos cuadros de diálogo, haciendo que este provoque que se detenga el app, es por eso que se realiza un análisis encontrando la solución que evita que se repita el fallo y este se basa en agregar la constante `TYPE_APPLICATION_OVERLAY`, del objeto instanciado `WindowManager.LayoutParams` al crearlos, esta ayuda a que la alerta se muestre siempre, superponiendo está sobre todas las actividades de las apps, además de evitar que el sistema finalice estas, cuando se tiene poca batería en el dispositivo (Android Developers, 2023d), notando también, que actualmente para en la creación de cada `AlertDialog` se realizaba de manera independiente, por eso, se realiza la creación de un método general el cual se pueda utilizar para la creación de múltiples cuadros de diálogo a partir de este.

Figura 11

Estructura básica de un AlertDialog

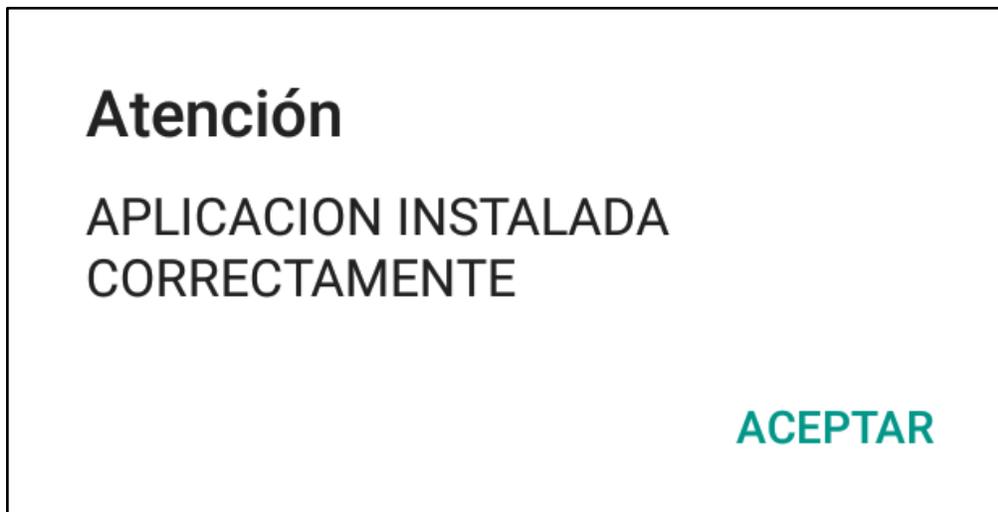
```
public static void showAlertDialog(String message, Context context) {
    AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder(context);
    alertDialogBuilder
        .setTitle("Titulo de la alerta")
        .setMessage(message)
        .setCancelable(false)
        .setPositiveButton("Aceptar", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int id) {
                dialog.dismiss();
            }
        });
    final AlertDialog alertDialog = alertDialogBuilder.create();
    alertDialog.getWindow().setType(WindowManager.LayoutParams.TYPE_APPLICATION_OVERLAY);
    alertDialog.show();
}
```

Nota. La imagen muestra el código de la estructura básica de un `AlertDialog`.

Una vez implementada la solución, podemos visualizar los distintos AlertDialog que el Agente Polaris genera en sus variados procesos para mantener al usuario informado sobre estos.

Figura 12

AlertDialog del Agente Polaris



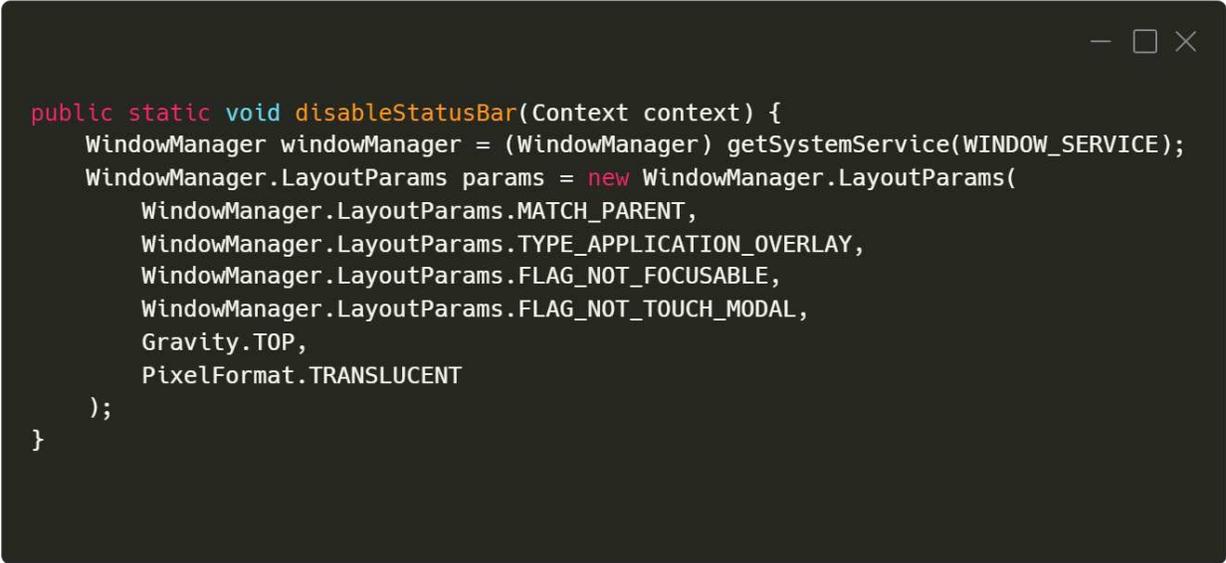
Nota. La imagen muestra cómo se observan los AlertDialog que genera el Agente Polaris.

3.1.2.5 Desarrollar las correcciones del error en el modo kiosko. Esta funcionalidad que se menciona en este apartado es una de las características principales del Agente Polaris, que busca otorgar restricciones específicas al terminal punto de venta en uso. Estas limitaciones tienen como objetivo principal brindar mayor seguridad al asegurarse de que el usuario solo pueda acceder a la aplicación financiera del cliente, (Guido, 2022) la cual, es asignada o parametrizadas desde la plataforma Polaris Cloud, que permite que no se pueda navegar por otras secciones o aplicaciones instaladas en el dispositivo. Para esto, el Agente Polaris lanza la aplicación que se asignó desde la plataforma y procede a bloquear los botones virtuales del POS y su barra de notificaciones.

En la actualidad cuando instalamos la aplicación en un terminal con nivel de API 29 se nota que al asignar este modo no se está realizando el bloqueo de la barra de notificaciones lo cual denota una vulnerabilidad de seguridad, esta deshabilitación de la barra para sistemas operativos Android 5 y 7 se realiza mediante los métodos que provee el SDK de los terminales NEW9220, pero para Android 10 este actualmente no es compatible con estas funciones por lo que se plantea la solución en la cual se crea una vista personalizada invisible, mediante la instancia del objeto `WindowManager.LayoutParams`, al igual que lo hacen los `AlertDialog`, y así, crear una vista que bloquee esta barra de notificaciones, para que no sea accesible.

Figura 13

Estructura básica para una vista personalizada



```
public static void disableStatusBar(Context context) {
    WindowManager windowManager = (WindowManager) getSystemService(WINDOW_SERVICE);
    WindowManager.LayoutParams params = new WindowManager.LayoutParams(
        WindowManager.LayoutParams.MATCH_PARENT,
        WindowManager.LayoutParams.TYPE_APPLICATION_OVERLAY,
        WindowManager.LayoutParams.FLAG_NOT_FOCUSABLE,
        WindowManager.LayoutParams.FLAG_NOT_TOUCH_MODAL,
        Gravity.TOP,
        PixelFormat.TRANSLUCENT
    );
}
```

Nota. La imagen muestra el código de la estructura básica de una vista personalizada.

3.1.2.6 Desarrollar las correcciones del error en la solicitud de permisos. Dentro del mundo de las aplicaciones móviles es común ver que para poder llevar a cabo sus funcionalidades o realizar ciertos procesos, necesitan permisos del sistema para poder ejecutar estas, y que en ocasiones se le solicita consentimiento a los usuarios para la aprobación de los mismos, esto ayuda a que se garantice el cumplimiento de las políticas de seguridad de Android. (De Luca, 2022) La problemática actual se presenta cuando el cliente propone que no se realice la petición de estos permisos que interactúan con el usuario, ya que estos pueden ser rechazados sin intención por parte de los técnicos encargados en un proceso de instalación y ocasionar posibles fallas o restringir funcionalidades en la app, para esto se requiere que una vez se realice el proceso de instalación, estos ya se encuentren aceptados, esto no se debe realizar así y siempre se debe mantener a los usuarios informados de cualquier permiso de acceso que necesitemos del sistema para obtener herramientas adicionales que no se otorgan de manera predeterminada, pero en esta ocasión se realizará para cumplir con las peticiones internas del cliente en sus procesos.

A la hora de implementar la solución aplicando métodos propios de Android no se puede realizar debido a que el sistema no puede realizar la otorgación de permisos en tiempo de ejecución, (Antolínez, 2019) para esto se hace uso del SDK de las terminales NEW9220, este nos provee métodos los cuales, nos permiten iniciar una tarea, en este caso los permisos que queremos que se acepten de manera dinámica en el app, y así, no se tenga que solicitar los permisos al usuario y que este cometa algún error en el proceso de instalación.

Figura 14*Estructura básica para otorgar permisos*


```

        .runOnUiThread(() -> {
            @Override
            public void onFinish() {
                String[] runtimePermission = new String[] {
                    //Permisos a aceptar
                };
                ActivityCompat.requestPermissions(this, runtimePermission);
            }
        });
    });

```

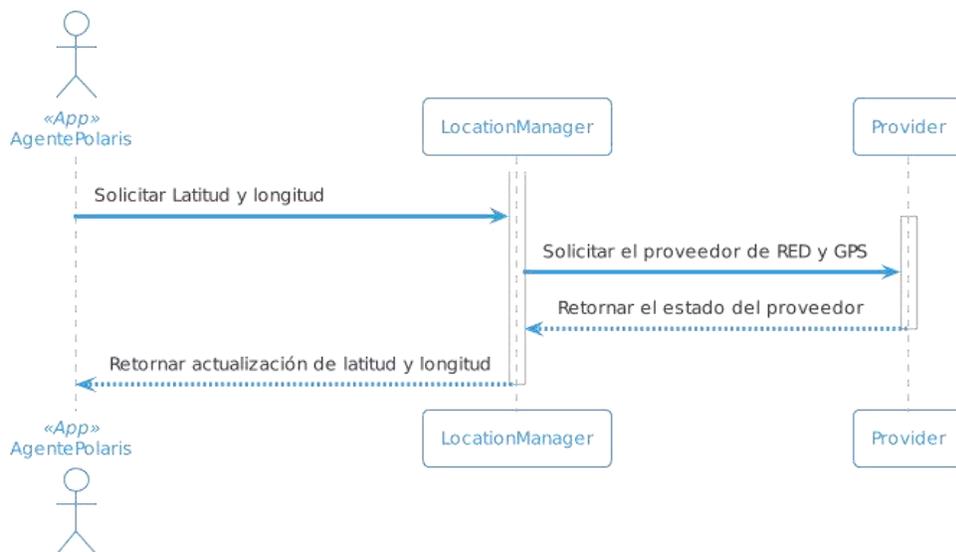
Nota. La imagen muestra el código para otorgar permisos en tiempo de ejecución.

3.1.2.7 Desarrollar las correcciones del error de obtención de coordenadas. Obtener la geolocalización de los terminales NEW9220 es una tarea importante que ejerce la aplicación para su posterior visualización en la plataforma Polaris Cloud, esto se realiza para mantener al cliente informado en todo momento sobre la ubicación de sus terminales y tener un mapeo sobre estas. En la actualidad estas terminales no obtiene las coordenadas de su posición con normalidad en lugares cerrados, a raíz de este caso se está generando desinformación dentro de los estatutos de monitoreo que el cliente maneja, esto se debe a que el Agente Polaris obtiene estos datos mediante GPS, y este, como bien se sabe, en áreas interiores tiende a fallar mucho en cuanto a la precisión de latitud y longitud (Bermudez, 2016) y esto afectará usuarios que posee el cliente con dispositivos que se encuentren físicamente en edificios o lugares subterráneos.

La solución propuesta se basa en una nueva estructura para obtener la ubicación mediante GPS y WIFI, ya que estos dispositivos siempre contarán con conexión a internet para que sus aplicaciones puedan funcionar, entre ellas el Agente Polaris, y así tomar una mejor localización tomando en cuenta cualquiera de los dos aspectos anteriores según esté disponible en el momento, para esta solución a impartir se implementa un LocationManager que nos ayudará a obtener la actualización y acceder a los servicios de localización mediante los recursos de ubicación a utilizar en la terminal, (Android Developers, 2023b) mencionados al principio del párrafo, además, implementado también la clase Provider que es justamente la que nos brinda los proveedores de red y GPS, los cuales ayudarán al LocationManager a la tarea de obtener una presión más exacta de donde se encuentra el dispositivo en cuestión y satisfacer así la problemática presentada para el presente cliente en las terminales actuales bajo sistema operativo Android 10.

Figura 15

Estructura básica de la obtención de la geolocalización



Nota. La imagen muestra el diagrama para obtener la geolocalización mediante GPS y RED.

3.1.2.8 Desarrollar las correcciones del error de inicialización del servicio de consultas. Es importante que las aplicaciones MDM se mantengan en constante funcionamiento para garantizar la seguridad y la gestión adecuada de los dispositivos móviles. El Agente Polaris hace uso de servicios en segundo plano para respaldar su uso constante para el desarrollo de sus distintas tareas y funciones, normalmente, la aplicación puede iniciar este servicio por sí sola aunque se reinicie el terminal, que ocurre cuando se descarga e instala la aplicación financiera o se requiera la activación de este servicio en primer plano, lo cual sucede cuando se realiza una actualización del propio Agente Polaris, en este último caso para el sistema operativo Android 10, se observa que no se está iniciando el background service cuando se realiza esta tarea, esto ocurre debido a que a partir del nivel de API 29 Google realizó algunas restricciones las cuales no permiten que se inicien actividades en segundo plano para brindar al usuario más seguridad y que contenga un contexto más claro al realizar estas tareas bajo su consentimiento en un primer plano. (Android Developers, 2023c)

Para remediar esto, se pretende utilizar funciones alternativas en la creación de tareas de fondo que ayuden a suplir las necesidades actuales, y poder hacer que el Agente Polaris siga funcionando sin restricciones tanto en Android 10, como en sistemas operativos 5 y 7, por eso, se utilizará el API WorkManager que hace parte de la librería de Android Jetpack, que ayuda a que las tareas se realicen bajo el trabajo persistente, siendo más fiable en ejecución constante, funcionando sin importar si la app y el sistema se cierra o se reinicia, además, de poder añadir propias restricciones para el funcionamiento propio de la tarea (Android Developers, 2022b) siendo un reemplazo actual y recomendado a los tradicionales servicios para mantener o ejecutar tareas en segundo plano, añadiendo ventajas realmente positivas que benefician al proyecto

actual, subsanando diferentes aspectos o problemáticas que puedan afectar su funcionamiento a futuro.

Figura 16

Estructura básica WorkManager

```
public class NombreWorker extends Worker {

    public NombreWorker(
        @NonNull Context context,
        @NonNull WorkerParameters params) {
        super(context, params);
    }

    @Override
    public Result doWork() {
        // Trabajo en segundo plano a ejecutar
        return Result.success();
    }
}

public class claseMain {
    Button startButton = findViewById(R.id.start_button);

    startButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            WorkRequest myWorkRequest =
                new OneTimeWorkRequest.Builder(MyWorker.class)
                    .build();

            WorkManager.getInstance(context).enqueue(myWorkRequest);
        }
    });
}
```

Nota. La imagen muestra el código de la estructura básica para realizar un trabajo en segundo plano.

3.1.2.9 Realizar el control de cambios periódicos en el repositorio según avance el desarrollo del proyecto. Ejecutar un flujo de desarrollo ordenado es indispensable para que todo el equipo de trabajo se beneficie de las buenas prácticas de desarrollo que se llevan a cabo al utilizar el versionamiento en los distintos proyectos, lo cual, ayuda al fortalecimiento de la cooperación en el grupo (Zárate, et al., 2016), así como lo realiza la empresa en la actualidad, y para llevarlo a cabo utiliza herramientas como bitbucket, el cual es el repositorio donde se encuentra el proyecto alojado y sourcetree, que se trabaja como la interfaz gráfica que nos ayuda a llevar a cabo las distintas actividades que se realizan diariamente con git y que es indispensable en estos entornos de gestión de proyectos.

Para realizar un push de los commits una vez se terminen los distintos desarrollos se utiliza la siguiente estructura de organización para ayudar a los demás desarrolladores a entender mejor el cambio realizado en el versionamiento del proyecto en general.

Tabla 4

Estructura de un commit dentro del proyecto Agente Polaris

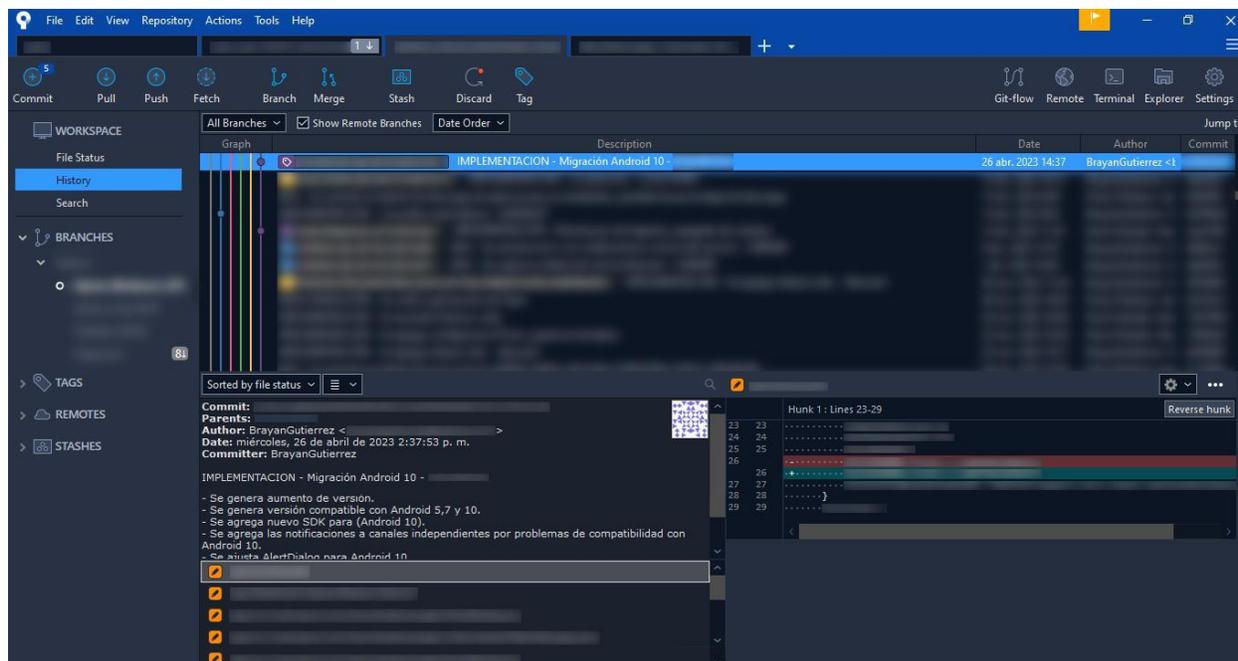
Nombre de la estructura		Descripción	
Título	Breve resumen de los cambios realizados, el título se divide en tipo de mensaje y el asunto	Tipo de mensaje	- Implementación - Bug - Refactorización
		Asunto	Descripción breve del cambio efectuado
Cuerpo	Explicación opcional más detallada que proporciona contexto e información adicional		
Pie	Describe los temas afectados que se utilizan para el seguimiento de eventos		

Nota. La información es tomada de (WPOSS, 2022).

Y de una manera gráfica dentro de Sourcetree se puede ver cómo se implementa esta estructura de commits en la siguiente figura.

Figura 17

Sourcetree, interfaz gráfica de Git



Nota. La imagen muestra el repositorio del Agente Polaris con la estructura de commits que utiliza la empresa WPOSS.

3.1.3 Realizar los casos de pruebas correspondientes, verificando el correcto funcionamiento en terminales NewPOS 9220 bajo el sistema operativo Android 10

3.1.3.1 Realizar las pruebas unitarias del desarrollo efectuado. Dentro de los procesos en los cuales se requieran resultados de calidad en el producto de software es indispensable la realización de pruebas unitarias por parte del equipo de desarrollo en todos sus trabajos en

cuanto a codificación dentro del proyecto, para lograr cumplir de igual manera con los requerimientos establecidos de este mismo y darle un valor agregado al producto final (Barrera, 2018).

Actualmente, la empresa WPOSS no hace uso de pruebas de este tipo dentro del proyecto del Agente Polaris, por lo cual se realiza o se hace uso de los test unitarios implementado el framework JUnit 5, el cual se basa en tags para funcionar y realizar el testing de código Java (Castaño, 2019) y el cual es una herramienta que posee por defecto el propio IDE de desarrollo utilizado, como lo es Android Studio. Estas pruebas se realizaron con el fin de comprobar que los elementos como notificaciones, cuadros de diálogo y funcionalidades varias mencionadas en los apartados anteriores que se implementaron para su compatibilidad con el sistema operativo Android 10, se hayan creado de manera satisfactoria y que no ocurriera ningún error de cualquier tipo al tratar de ejecutarse, es por eso, que para las pruebas se siguió la siguiente estructura de código que se evidencia en la figura siguiente.

Figura 18*Estructura básica para pruebas con JUnit 5*

```

public class NotificationServiceTest {

    private ClaseInstanciada claseInstanciada;

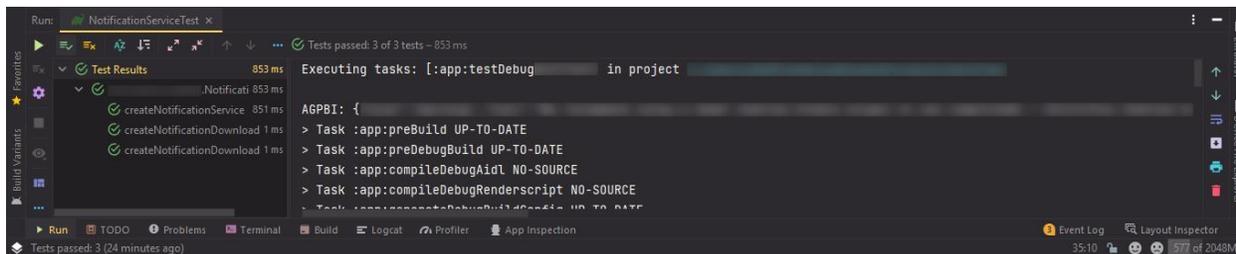
    @Before
    public void setUp() {
        claseInstanciada = new ClaseInstanciada();
        claseInstanciada.onCreate();
    }

    @Test
    public void createNotificationService() {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            NotificationManager notificationManager = claseInstanciada.getSystemService(NotificationManager.class);
            NotificationChannel notificationChannel = notificationManager.getNotificationChannel(ClaseInstanciada.channelId);
            assertNotNull(notificationChannel);
            assertEquals(notificationChannel.getName(), "Nombre del Canal");
            assertEquals(notificationChannel.getImportance(), NotificationManager.IMPORTANCE_DEFAULT);
        }
    }
}

```

Nota. La imagen muestra la estructura de unit test para la creación de un notificationChannel.

Posteriormente dentro del entorno de desarrollo podemos observar de manera visual si el test unitario pasó de forma satisfactoria o no.

Figura 19*Resultado exitoso de pruebas con JUnit 5 en Android Studio*

Nota. La imagen muestra los resultados de test unitarios para la creación de los canales de notificaciones.

Si intencionalmente manipulamos el código para poder visualizar los errores que podremos presentar al momento de crear notificaciones cuando el canal de notificaciones en nulo podemos obtener los siguientes resultados

Figura 20

Resultado fallido de pruebas con JUnit 5 en Android Studio

```

Tests failed: 1, passed: 2 of 3 tests - 881 ms
Test Results
  .NotificationServiceTest 881 ms
  createNotificationService 880 ms
  createNotificationDownload 1 ms
  createNotificationDownloadE 0 ms

Cannot invoke "android.app.NotificationManager.getChannel(String)" because "notificationManager" is null
java.lang.NullPointerException: Create breakpoint : Cannot invoke "android.app.NotificationManager.getChannel(String)"
    at com.wposs.unittest.NotificationServiceTest.createNotificationService(NotificationServiceTest.java:29) <34 internal>
    at jdk.proxy1/jdk.proxy1.$Proxy2.processTestClass(Unknown Source) <7 internal lines>
    at worker.org.gradle.process.internal.worker.GradleWorkerMain.run(GradleWorkerMain.java:69)
    at worker.org.gradle.process.internal.worker.GradleWorkerMain.main(GradleWorkerMain.java:74)
  
```

Nota. La imagen muestra los resultados de test unitarios para la creación de los canales de notificaciones.

3.1.3.2 Realizar las pruebas funcionales de la aplicación. El ciclo del desarrollo de proyectos dentro de la empresa WPOSS finaliza cuando el equipo de aseguramiento de calidad realiza las pruebas funcionales del software en cuestión para verificar que su ejecución se realice de manera adecuada, cubriendo diferentes escenarios en los cuales se pueda ver afectado el uso operativo de este. La importancia de estos tests ayuda a realizar diferentes simulaciones basadas en comportamientos reales, los cuales podría tomar la aplicación que ayudan a identificar la detección de errores a mayor escala (Macías, et al., 2016) y cubrir un gran campo basado en los requerimientos y verificar el cumplimiento de estos a su vez.

Igualmente, luego de terminar la implementación de software, es importante que los programadores hagan la realización de estas pruebas y lograr así identificar de igual manera

errores con la aplicación, sin abarcar un gran campo general como lo realiza el área de QA, que en muchas ocasiones realizan un barrido general de la aplicación para verificar que no se afecten otras funcionalidades, pero sí verificar que la construcción de software implementada dentro del proyecto no lo haga. Es por eso que se realizan las siguientes actividades de testing basándose en un check list el cual expone casos que pueden abarcar el desarrollo efectuado, para posteriormente poder corregir los errores encontrados en el análisis llevado a cabo.

Tabla 5

Check list de pruebas funcionales

ID	Caso de prueba	Check
1	Validar el mensaje en la barra de notificaciones que indica que el Agente se encuentra funcionando.	✓
2	Actualización por descarga remota.	✓
3	Validar que al realizar la instalación del Agente por primera vez se acepten los permisos de manera automática.	✓
4	Validar que el Agente sólo haga una descarga por consulta.	✓
5	Mensaje en pantalla luego de actualización de versión: "AGENTE POLARIS ACTUALIZADO!"	✓
6	Validar que al realizar cambios en la configuración de AgentePolaris éste no se detenga.	✓
7	Al realizar consulta manual, luego de la instalación (Registrar equipo) se aplican los cambios en Polaris Cloud Service.	✓
8	Al cerrar el proceso del Agente, no se interrumpe la descarga.	✓
9	Validar en la barra de notificaciones que el mensaje de descarga se vaya actualizando según avanza esta.	✓
10	Validar mensaje de respuesta en barra de notificaciones cuando falla la instalación de la aplicación. "Se interrumpió la descarga, continuará en la siguiente consulta".	✓

ID	Caso de prueba	Check
11	Mensaje en pantalla luego de instalar la aplicación financiera: "APLICACIÓN INSTALADA CORRECTAMENTE".	✓
12	Al realizar la instalación de la app financiera, luego de que se reinicie la terminal, el Agente se inicia de manera automática.	✓
13	Validar mensaje de respuesta cuando falla la instalación de la aplicación: "ERROR EN LA INSTALACIÓN EL ARCHIVO SERÁ DESCARGADO NUEVAMENTE EN LA PRÓXIMA CONSULTA".	✓
14	Validación en pantalla de mensaje de alerta configurado en plataforma. Nota: Al borrar este mensaje en plataforma, no se debe visualizar en la terminal.	✓
15	Validar modo Kiosko	✓
16	Validar que al reiniciar el POS el tiempo de activación del modo kiosko sea mínimo.	✓
17	Validar que al reiniciar el POS envíe consulta después de 15 segundos.	✓
18	Validar que al desactivar el modo kiosko no quede bloqueada la barra de notificaciones, ni las demás funcionalidades del POS.	✓
19	Validar que el servicio de geolocalización envíe su última posición capturada.	✓

Nota. La información es tomada de (WPOSS, 2023).

3.1.3.3 Registrar y corregir los errores evidenciados en las pruebas. Durante la realización de las pruebas unitarias y funcionales no se encontraron errores significativos en la aplicación Agente Polaris. Esto es un indicativo de que el desarrollo efectuado se llevó a cabo de manera correcta, siguiendo estándares de calidad.

Las pruebas unitarias demostraron que cada componente de la aplicación se creó de acuerdo a lo esperado en terminales con nivel de API 29, sin afectar el funcionamiento de estos en los sistemas operativos anteriores donde estos ya trabajaban. No se encontraron fallos en el

funcionamiento individual de las funciones afectadas con la escalabilidad, lo cual es un resultado muy satisfactorio.

Por otra parte, las pruebas funcionales evidenciaron que la aplicación Agente Polaris cumple con los requisitos establecidos que maneja el cliente actualmente y se comporta de manera estable en diferentes escenarios que propone el checklist de pruebas. Se evaluaron las funcionalidades afectadas por el desarrollo y se verificó la correcta interacción entre los componentes. No se encontraron errores significativos en estas pruebas, lo que refuerza la calidad del desarrollo.

Estos resultados pueden ser evidenciados en los dos apartados anteriores donde se demuestra el resultado arrojado por las pruebas.

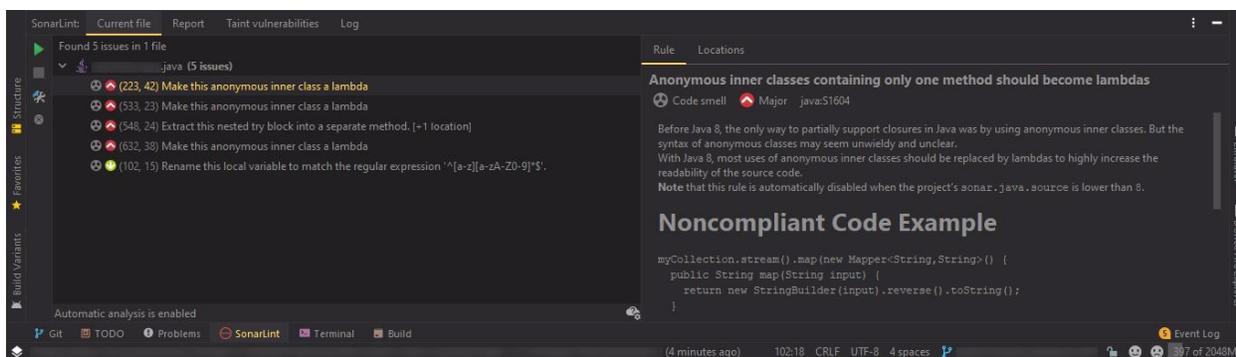
3.1.4 Elaborar la documentación del desarrollo llevado a cabo, utilizando la herramienta Javadoc, facilitando con ello la orientación de los cambios futuros en el versionamiento.

3.1.4.1 Identificar los conceptos adecuados, para el desarrollo de la documentación de una manera técnica y acertada. Para el uso de buenas prácticas dentro del proyecto del Agente Polaris estas se pueden aplicar dentro del código fuente utilizando SonarLint, que es utilizado para obtener las mejores decisiones de programación en tiempo real, para corregirlas en el momento, sin tener que esperar para hacer un análisis general al finalizar el proceso de codificación. Al utilizar esta herramienta se mejora la calidad del código fuente y se reducen tanto errores como problemas en la etapa de desarrollo. Basta con solo instalar la extensión en nuestro IDE de desarrollo Android Studio y podremos hacer uso de esta funcionalidad.

Si tenemos código mal estructurado dentro de la clase o archivo, SonarLint dictará un diagnóstico de manera inmediata, donde nos comunicará donde se encuentra el error o donde se debe implantar la mejora, añadiendo ejemplos de cómo solucionar los distintos casos que esta nos marque.

Figura 21

Clase con malas prácticas de programación detectado por SonarLint

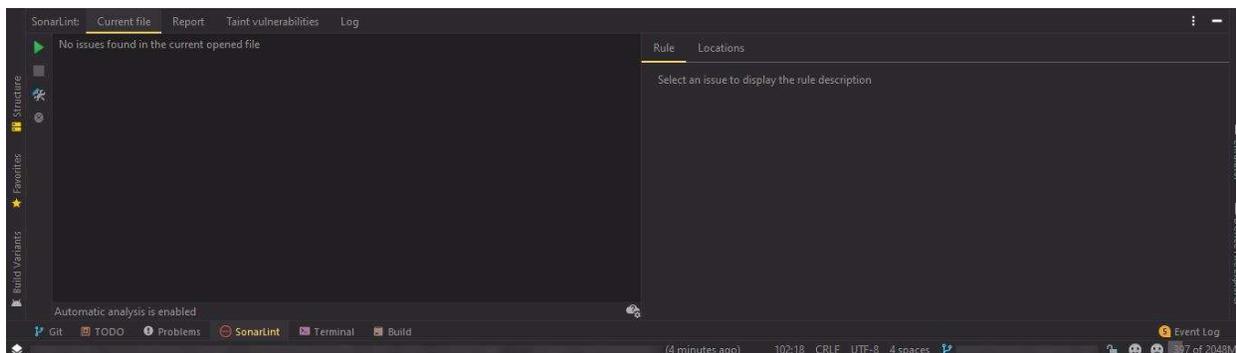


Nota. La imagen muestra los errores y soluciones a aplicar dentro del código.

Una vez solucionadas las sugerencias marcadas por esta herramienta implantando código eficiente dentro del proyecto, nos procederá a indicar que todo se encuentra bien dentro de la clase o archivo actual.

Figura 22

Clase con buenas prácticas de programación detectado por SonarLint



Nota. La imagen muestra la interfaz con un mensaje indicando que no se encuentran errores.

3.1.4.2 Revisar el código, para la identificación de métodos y clases relevantes utilizados en el desarrollo. La documentación en los proyectos de software se considera una parte fundamental para poder lograr así un producto más completo y de crecimiento exponencial con el tiempo, además de ayudar al equipo de trabajo para que se tenga una mayor comprensión de lo que se realizó en la codificación del software. Para lograr esto, se pueden hacer uso de distintas herramientas que ayudan a lograr este objetivo de buenas prácticas, entre estas destaca el uso de JavaDoc, el cual es un instrumento que nos ayudará a cumplir con la tarea, basándose en comentarios para detallar el funcionamiento de las distintas clases y métodos empleados (Delgado, 2017) dentro de la escalabilidad de la aplicación Agente Polaris hacia el sistema operativo Android 10, los comentarios se asignan antes de cada clase y método añadiendo tags que nos ayudan a comprender mejor las entradas y salidas de estos, además, de contar con una descripción y el objetivo de su creación.

Tabla 6

Tags utilizados para la documentación con JavaDoc

Tag	Descripción
@author	Nombre del autor.
@param	Descripción y tipo de parámetros que recibe el método.
@return	Describe que retorna el método aplicado en el desarrollo. Se debe omitir el uso de este para los métodos que retorna void y también para constructores.
@version	Versión de los métodos y clases empleados en el desarrollo.

Nota. La información es obtenida de (Oracle, s.f.)

Figura 23

Documentación en el código del Agente Polaris utilizando JavaDoc

```

package ...;

import ...;

/**
 * This class defines the activation and deactivation of the kiosk mode of the terminal
 * @author RPOSS
 * @version ...
 */

/**
 * Method activating the kiosk mode
 * @param ... global interface about the application environment
 * @param ... : deactivate the notification bar and the navigation buttons;
 * @param ... : activate the notification bar and the navigation buttons
 * @return returns a ... value indicating the deactivation of the notification bar and
 * the navigation buttons
 */

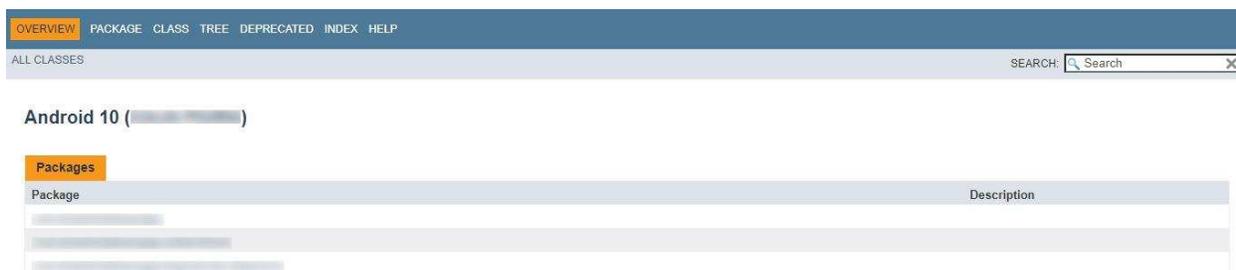
```

Nota. La imagen muestra los distintos tags usados para la documentación.

Una vez se documentaron correctamente los métodos y clases afectados en la migración realizada hacia el sistema operativo Android 10, es decir, solo se realizará el proceso que se alteró o añadió al realizar la escalabilidad, siguiendo los estamentos que nos propone JavaDoc se procede a generar un documento en formato HTML que contendrá toda la información de las especificaciones anteriores, ofreciendo una interfaz intuitiva que permite navegar por estos aspectos de forma rápida.

Figura 24

Documentación de paquetes generada por JavaDoc



Nota. La imagen muestra el documento visualizando los paquetes editados.

Figura 25

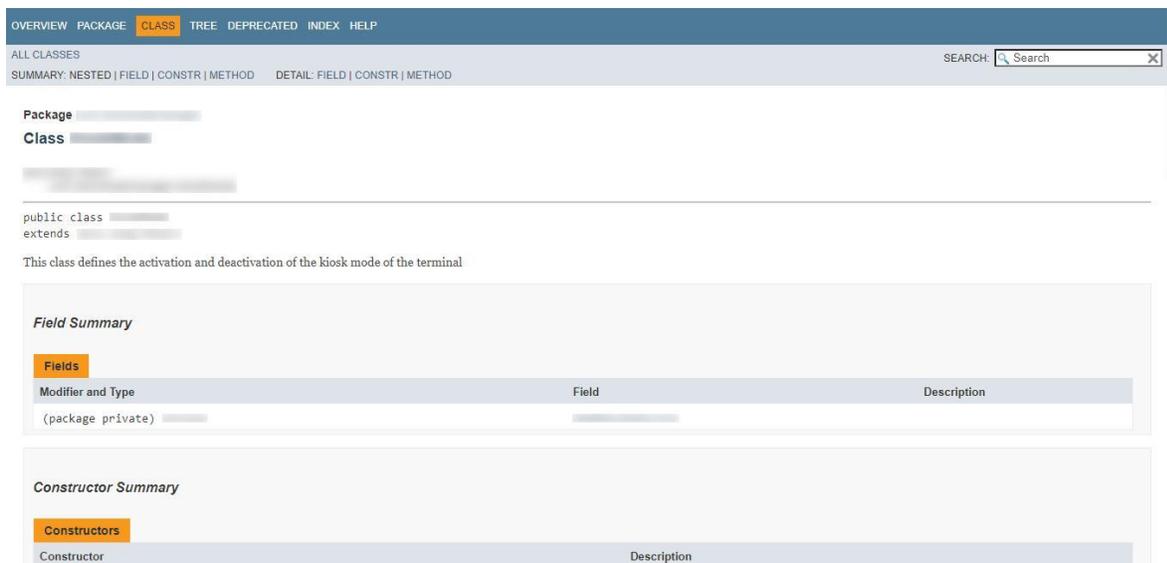
Documentación de clases generada por JavaDoc



Nota. La imagen muestra el documento visualizando las clases editadas.

Figura 26

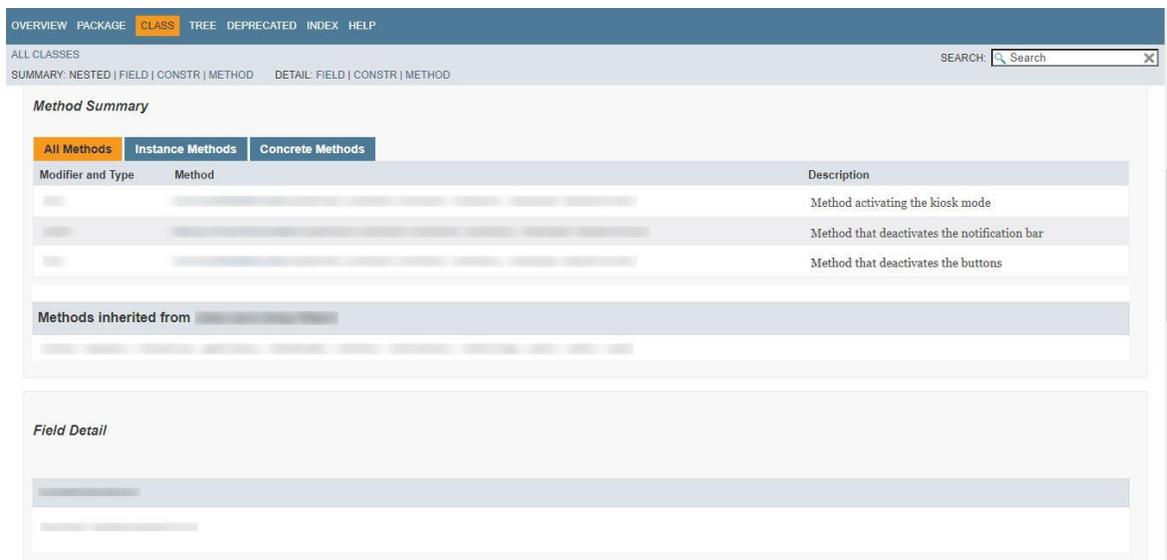
Documentación de la estructura de las clases generada por JavaDoc



Nota. La imagen muestra el documento describiendo la estructura general de la clase editada.

Figura 27

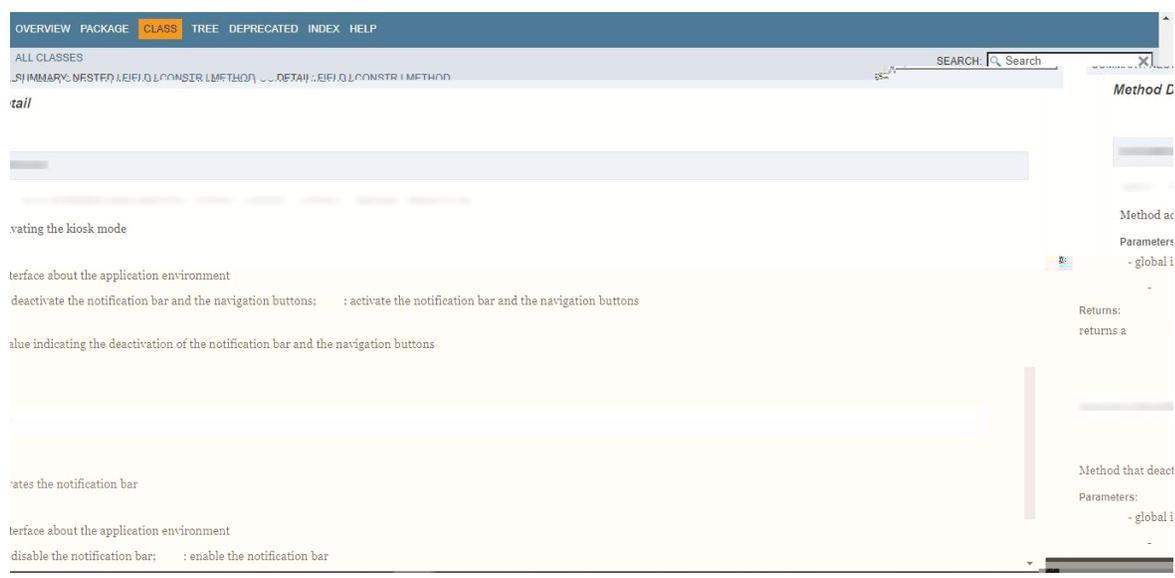
Documentación de los métodos generada por JavaDoc



Nota. La imagen muestra el documento describiendo métodos de la clase editada.

Figura 28

Documentación detallada de los métodos generada por JavaDoc



Nota. La imagen muestra el documento describiendo los detalles de los métodos de la clase editada.

Figura 29

Documentación de la estructura jerárquica generada por JavaDoc

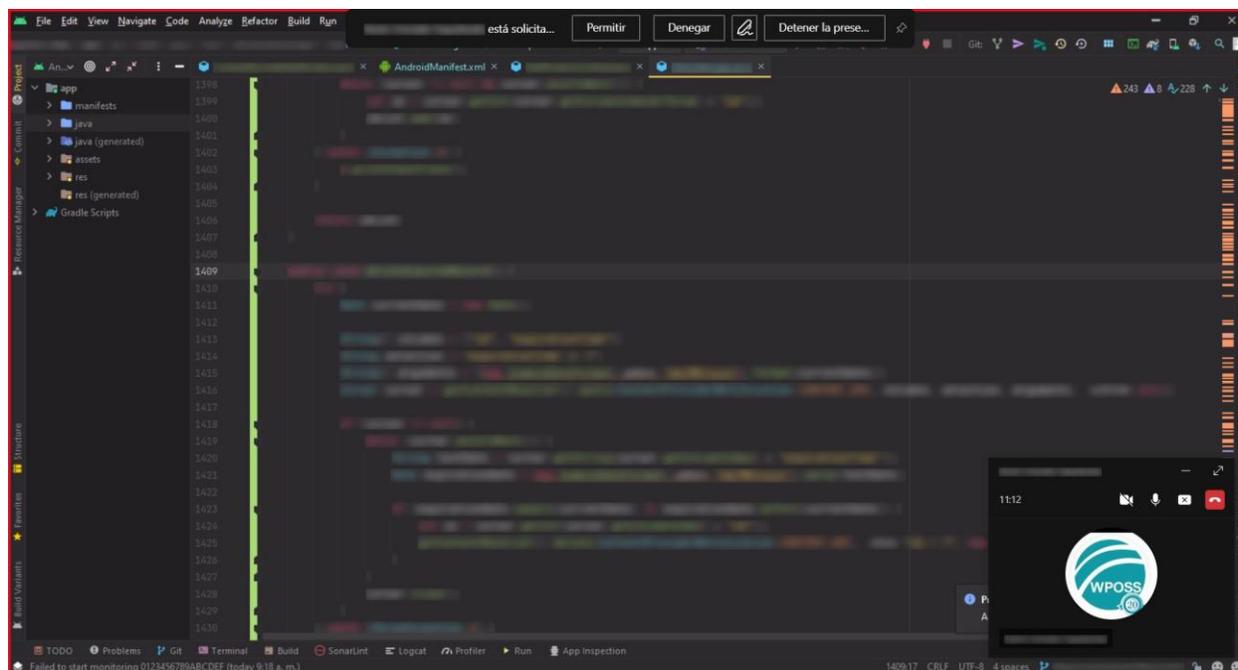


Nota. La imagen muestra el documento visualizando la estructura jerárquica de los paquetes editados.

3.1.4.3 Realizar reuniones periódicas con el líder del proyecto, con el objetivo de retroalimentar la elaboración de la documentación. Para concretar y verificar que todas las actividades se realicen de manera correcta, aplicando las diferentes herramientas que ayuden a llevar a cabo el cumplimiento de los objetivos propuestos con el equipo de trabajo, se realizaron reuniones periódicas con el líder del proyecto para verificar y orientar al desarrollo para que este se esté realizando con las mejores prácticas y cumplir con cada una de las actividades propuestas de desarrollo, solucionando dudas y planteando mejoras en el proceso. Se lograron completar dos reuniones durante toda la ejecución de la escalabilidad del Agente Polaris hacia el sistema operativo Android 10.

Figura 30

Reunión periódica



Nota. La imagen muestra una reunión llevada a cabo con el líder del proyecto.

4. Diagnóstico final

Llevado a cabo el trabajo de pasantías en la empresa WPOSS, se destaca el valor de enseñanza adquirida al trabajar con un personal capacitado en la tarea del desarrollo de software. Durante este tiempo, se lograron fortalecer conocimientos en la adopción del marco de trabajo SCRUM, acondicionado a los diferentes proyectos dentro de la empresa, participando en los procesos de la planificación y elaboración del trabajo mediante esta metodología, permitiendo trabajar en un ambiente colaborativo aportando habilidades y conocimientos para cumplir los objetivos del proyecto Agente Polaris perteneciente al área de DMA.

Además, de adquirir experiencia en el desarrollo de aplicaciones móviles Android, implementando buenas prácticas, documentando los desarrollos elaborados y añadiendo pruebas tanto unitarias como funcionales para verificar la correcta ejecución de estas, destacando de igual manera la importancia de realizar revisión previa y estructurar correctamente el código para llegar siempre a la solución más refinada, implementando también el uso de herramientas de control de versiones para mejorar la calidad del código y optimizar los tiempos de desarrollo.

Todo lo aprendido durante las pasantías permitió un crecimiento personal y laboral en el campo del desarrollo de software, una rama muy demandada en la carrera de ingeniería de sistemas. Esto permitirá ser un mejor profesional en el futuro y tener un desempeño destacado en el ámbito laboral.

5. Conclusiones

Lograr implementar la escalabilidad del sistema operativo Android 10 en el proyecto del Agente Polaris, trae consigo beneficios importantes para la seguridad de los dispositivos móviles y la gestión de la seguridad para el cliente perteneciente a la empresa World Pos Solutions, radicando este en el cumplimiento de los estándares de calidad que exige PCI-DSS. Para lograr esto, es necesario llevar a cabo diferentes configuraciones a nivel de software que permitan generar las características de seguridad necesarias para cumplir con estos lineamientos, sin afectar el trabajo actual que realiza la aplicación en sus terminales con sistemas operativos Android 5 y 7.

Cumplir con un marco de trabajo ágil para la ejecución de las actividades logra efectividad en la realización y entrega de estas, permitiendo la organización y ahorro de tiempo, ayudando a la par con la verificación de los requerimientos actuales que opera el cliente permitiendo que estos se cumplan a cabalidad sin ninguna alteración y siguiéndolos al pie de la letra para presentar finalmente un producto adecuado. Contando de igual manera con un ciclo de trabajo de desarrollo de software que abarca desde el análisis previo para implantar las soluciones, su ejecución con la orientación y ayuda del equipo, hasta los tests para validar y asegurar la calidad de las soluciones planteadas, es fundamental para garantizar el éxito en la escalabilidad en los diferentes terminales NewPOS 9220 con sistema operativo Android 10. Además de contar con un proceso documental importante para asegurarse de que todos los procesos se registran de manera completa y precisa a lo largo de la implementación de soluciones utilizando herramientas que ayudan a llevar a cabo estas tareas facilitando la comprensión para los integrantes presentes y futuros del proyecto.

Durante mi experiencia en las pasantías, tuve la oportunidad de aplicar los conocimientos y habilidades adquiridos a lo largo de mi formación universitaria en un ambiente laboral real. A través de esta oportunidad, pude obtener una valiosa experiencia profesional que me permitirá fortalecer mis habilidades y destrezas en futuros desafíos.

6. Recomendaciones

Adoptar una metodología o marco de trabajo ágil única para la planificación y ejecución de las actividades, siguiendo estas al pie de la letra puede beneficiar de manera significativa, mejorando la colaboración en equipo, además de poder realizar entregas más rápidas facilitando el desarrollo de las actividades, ya que, si solo se utilizan características principales de una o varias, puede generar confusión y retrasos dentro del equipo en sí.

Mantener las capacitaciones y documentación actualizada cada cierto tiempo dentro del proyecto es fundamental, partiendo de que estas son herramientas realmente importantes para garantizar en sí, el crecimiento y expansión en los desarrollos de software presentes y futuros, además de facilitar la inclusión de nuevo personal para realizar un empalme adecuado.

Referencias

Android Developers. (2019). *Android 10 for Developers*.

<https://developer.android.com/about/versions/10/highlights#foldables>

Android Developers. (2022). *Cómo programar tareas con WorkManager*.

<https://developer.android.com/topic/libraries/architecture/workmanager?hl=es-419>

Android Developers. (2022). *Cuadros de diálogo*.

<https://developer.android.com/guide/topics/ui/dialogs?hl=es-419>

Android Developers. (2023). *Introducción a Android Studio*.

<https://developer.android.com/studio/intro?hl=es-419>

Android Developers. (2023). *LocationManager*.

<https://developer.android.com/reference/android/location/LocationManager>

Android Developers. (2023). *Restricciones para iniciar actividades en segundo plano*.

<https://developer.android.com/guide/components/activities/background-starts?hl=es-419>

Android Developers. (2023). *WindowManager.LayoutParams*.

<https://developer.android.com/reference/android/view/WindowManager.LayoutParams>

Antolínez Ladino, D. A. (2019). *Vulnerabilidades y seguridad en el sistema operativo android*.

[Tesis de especialización, Universidad Piloto de Colombia]. Repositorio Institucional

UPC. <http://repository.unipiloto.edu.co/handle/20.500.12277/6341>

Barrera Quinche, W. H. (2018). *Aplicación móvil de información de frecuencias de cooperativas de transporte interprovincial evaluado bajo pruebas unitarias automatizadas*. [Tesis de pregrado, Escuela Superior Politécnica de Chimborazo]. DSpace ESPOCH.

<http://dspace.esPOCH.edu.ec/handle/123456789/9113>

Barthwal, D. (2016). Mobile device management (MDM) in organizations. *ResearchGate*.

10.13140/RG.2.1.2467.7362

Bermudez, E. R. (2016). Prototipo para la localización en interiores con BLE Beacons. *U@CSIS*, 1(1), 32-37.

<http://investigacionsis.fuac.edu.co/html/RepositorioOJS/ojsfuac/ojs/index.php/UACISIS/article/view/26>

Brown, N. C., Weill-Tessier, P., Sekula, M., Costache, A. L. & Kölling, M. (2022). Novice use of the Java programming language. *ACM Transactions on Computing Education*, 23(1), 1-24. <https://doi-org.sibdigital.ufpso.edu.co/10.1145/3551393>

Calder, A. & Williams, G. (2019). *PCI DSS: a pocket guide* (6^a ed.). It Governance Publishing Ltd.

https://books.google.com.co/books?id=8D6uDwAAQBAJ&lpg=PA1&ots=XNCybPAyG_&dq=pci%20dss&lr&hl=es&pg=PA4#v=onepage&q=pci%20dss&f=false

Cardona Perdomo, R. A. (2017). Análisis de riesgos sobre dispositivos móviles empresariales.

<http://repository.unipiloto.edu.co/handle/20.500.12277/6375>

- Carrasco, M., Ocampo, W., Ulloa, L., & Azcona, J. (2019). Metodología Híbrida de Desarrollo de Software combinando XP y SCRUM. MIKARIMIN. *Revista científica Multidisciplinaria*, 5(2), 109-116. https://www.researchgate.net/profile/Jon-Azcona-Esteban-2/publication/336588210_METODOLOGIA_HIBRIDA_DE_DESARROLLO_DE_SOFTWARE_COMBINANDO_XP_Y_SCRUM/links/5da724db299bf1c1e4c81709/METODOLOGIA-HIBRIDA-DE-DESARROLLO-DE-SOFTWARE-COMBINANDO-XP-Y-SCRUM.pdf
- Castaño Vasquez, M. (2019). *Validador de requerimientos para la ejecución automática de pruebas unitarias mediante un pipeline*. [Tesis de pregrado, Pontificia Universidad Católica del Ecuador Sede Ambato]. Repositorio PUCESA. <https://repositorio.pucesa.edu.ec/handle/123456789/2811>
- Chakraborty, S. & Aithal, P. S., (2022). A Practical Approach To GIT Using Bitbucket, GitHub and SourceTree. *International Journal of Applied Engineering and Management Letters (IJAEML)*, 6(2), 254-263. <https://doi.org/10.5281/zenodo.7262771>
- Chandran, V. P., Balakrishnan, A., Rashid, M., Pai Kulyadi, G., Khan, S., Devi, E. S., Nair, S., & Thunga, G. (2022). Mobile applications in medical education: A systematic review and meta-analysis. *PLoS One*, 17(3), e0265927. <https://doi.org/10.1371/journal.pone.0265927>
- De Luca, G. (2022). *Hacia un prototipo certificado del sistema de permisos de Android 10*. [Tesis de licenciatura, Universidad Nacional de Rosario]. Repositorio Hipermedial UNR. <http://hdl.handle.net/2133/24770>

- Delgado Castellanos, S. (2017). *ToraDocu—Generacion automática de casos de test con oráculos: precondiciones y postcondiciones*. [Tesis de pregrado, Universidad Politécnica de Madrid]. Archivo Digital UPM. <https://oa.upm.es/47223/>
- DiMarzio, J. (2016). *Beginning Android Programming with Android Studio*. Wrox. <https://books.google.com/books?id=9ukkdQAAQBAJ&lpg=PR1&ots=BWkkVdr8IE&dq=Android%20Programming%20for%20Beginners&lr&hl=es&pg=PP1#v=onepage&q&f=false>
- Guido, S.M. (2022). *Prototipo de quiosco electrónico responsivo para servicios de recursos humanos*. [Tesis de licenciatura, Universidad Autónoma de Ciudad de Juárez]. Recursos Electrónicos UACJ. <http://hdl.handle.net/20.500.11961/6265>
- ISO. (2003). *ISO 8583-3:2003*. <https://www.iso.org/obp/ui/es/#iso:std:iso:8583:-3:ed-2:v1:en>
- Lasa Gómez, C., Heras del Dedo, R. & Álvarez García, A. (2017). *Métodos ágiles: Scrum, Kanban, Lean*. Difusora Larousse - Anaya Multimedia. <https://elibro-net.sibdigital.ufpso.edu.co/es/lc/ufpso/titulos/122933>
- Lestaringati, S. I. (2018, August). Mobile point of sale design and implementation. *IOP Conference Series: Materials Science and Engineering* (Vol. 407, No. 1, p. 012094). [10.1088/1757-899X/407/1/012094](https://doi.org/10.1088/1757-899X/407/1/012094)
- Macías, A., Delgado, M., Fajardo, J. & Larrosa, D. (2016). Generación automática de combinaciones de valores para pruebas funcionales utilizando metaheurísticas. *Revista Cubana de Ingeniería*, 7(2), 48-54. <https://rci.cujae.edu.cu/index.php/rci/article/view/530>

- Montero, B. M., Cevallos, H. V., & Cuesta, J. D. (2018). Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software. *Espiraes revista multidisciplinaria de investigación*, 2(17), 114-121.
<https://gc.scalahed.com/recursos/files/r161r/w25597w/438760423-269-823-1-PB-pdf.pdf>
- Nadal, M. (2021). *Curso de programación Java*. Difusora Larousse - Anaya Multimedia.
<https://elibro-net.sibdigital.ufps.edu.co/es/lc/ufps/titulos/215497>
- Oracle. (s.f.). *How to write doc comments for the JavaDoc tool*.
<https://www.oracle.com/technical-resources/articles/java/javadoc-tool.html>
- Pathak, A. (2020). Introduction to Git for beginners. *XRDS: Crossroads, The ACEM Magazine for Students*, 26(4), 54-59. <https://doi-org.sibdigital.ufps.edu.co/10.1145/3398459>
- PCI Security Standards Council (PCI SSC). (2018, agosto). *Glossary of Payment and Information Security Terms*. https://www.pcisecuritystandards.org/wp-content/uploads/2022/05/Small_Merchant_Glossary_of_Payment_and_Information_Security_Terms.pdf
- PCI Security Standards Council (PCI SSC). (2022). *Requirements and Testing Procedures - V4.0*. https://listings.pcisecuritystandards.org/documents/PCI-DSS-v4_0.pdf
- Rodríguez Barajas, C. T. (2017). Impacto de los requerimientos en la calidad de software. *Tecnología Investigación y Academia*, 5(2), 161–173.
<https://revistas.udistrital.edu.co/index.php/tia/article/view/7607>

- SCRUMstudy. (2017). *Una guía para el cuerpo de conocimiento de SCRUM (Guía SBOK™) (3ª ed.)* SCRUMstudy™. <https://www.scrumstudy.com/SBOK/SCRUMstudy-SBOK-Guide-3rd-edition-spanish.pdf>
- Sachdeva, S. (2016). Scrum Methodology. *International Journal Of Engineering And Computer Science*, 5(16792), 16792-16800.
https://www.academia.edu/26010951/Scrum_Methodology
- Sonar Source. (s.f.). *SonarLint*. <https://www.sonarsource.com/products/sonarlint/#learn>
- Uzayr, S. (2022). *Mastering Git: A Beginner's Guide*. CRC Press.
<https://doi.org/10.1201/9781003229100>
- WPOSS. (2021, 15 febrero). *¿Quiénes somos?* <https://www.wposs.com/quienes-somos/>
- WPOSS. (2021, 4 febrero). *Sobre nosotros*. <https://www.wposs.com/>
- Yoshiizumi, K., & Zhang, J. (2020). Building a Smartphone Push Notification Management System. *International Symposium on Affective Science and Engineering ISASE2020* (pp. 1-4). Japan Society of Kansei Engineering. <https://doi.org/10.5057/isase.2020-C000050>
- Zárate, J. S., Ponce, B. H., Alvarez, N. S., & Barahona, B. V. (2016). Seguimiento de proyectos de programación. Una aplicación de GitHub en la educación. *ReCIBE. Revista electrónica de Computación, Informática, Biomédica y Electrónica*, 5(3).
<https://www.redalyc.org/journal/5122/512253114003/>