Universidad Continental

FACULTAD DE INGENIERÍA

Escuela Académico Profesional de Ingeniería de Sistemas e Informática

Trabajo de Suficiencia Profesional

Implementación del proyecto Wow en el grupo Éxito-Colombia, 2023

Antony Raul Cristobal Zambrano

Para optar el Título Profesional de
Ingeniero de Sistemas e Informática

Huancayo, 2024

INFORME DE CONFORMIDAD DE ORIGINALIDAD DE TRABAJO DE INVESTIGACIÓN

A	:	Decano de la Facultad de Ingeniería
DE	:	Job Daniel Gamarra Moreno Asesor de trabajo de investigación
ASUNTO	:	Remito resultado de evaluación de originalidad de trabajo de investigación
FECHA	:	1 de octubre de 2024

Con sumo agrado me dirijo a vuestro despacho para informar que, en mi condición de asesor del trabajo de investigación:

Título:
Implementación del Proyecto Wow en el Grupo Éxito - Colombia, 2023

Autor:
Antony Raul Cristobal Zambrano – EAP. Ingeniería de Sistemas e Informática

Se procedió con la carga del documento a la plataforma "Turnitin" y se realizó la verificación completa de las coincidencias resaltadas por el software dando por resultado 15 % de similitud sin encontrarse hallazgos relacionados a plagio. Se utilizaron los siguientes filtros:

- Filtro de exclusión de bibliografía SI [ ] NO [X]
- Filtro de exclusión de grupos de palabras menores SI [ ] NO [X]
  Nº de palabras excluidas **(en caso de elegir "SI"):**
- Exclusión de fuente por trabajo anterior del mismo estudiante SI [ ] NO [X]

En consecuencia, se determina que el trabajo de investigación constituye un documento original al presentar similitud de otros autores (citas) por debajo del porcentaje establecido por la Universidad Continental.

Recae toda responsabilidad del contenido del trabajo de investigación sobre el autor y asesor, en concordancia a los principios expresados en el Reglamento del Registro Nacional de Trabajos conducentes a Grados y Títulos – RENATI y en la normativa de la Universidad Continental.

Atentamente,

AGRADECIMIENTOS

A Dios, por darme la fuerza y sabiduría para llegar hasta aquí, a mi madre, por apoyarme siempre en cada paso que doy, y a mis dos hermanos, por ser el motivo para ser un digno ejemplo por seguir.

Con gratitud y admiración, al Dr. Job Gamarra, por su dedicación, compromiso y orientación que hicieron posible la conclusión de esta etapa importante para mí.

A mis docentes, por su valiosa contribución a mi desarrollo profesional.

DEDICATORIA

A mi querida madre, quien me dio todo su apoyo y creyó en mí, haciendo posible todo lo que he conseguido.

A mi abuelo, por sus sabios consejos y enseñanzas, deseándome siempre un mejor futuro.

A mis hermanos, quienes confían en mí y me incentivan a seguir adelante.

A mis maestros, quienes compartieron conocimiento y experiencias importantes para mi vida profesional.

A mi asesor, Dr. Job Gamarra, por ser mi mentor y guiarme durante mi etapa como estudiante.

ÍNDICE DE CONTENIDOS

LISTA DE TABLAS

LISTA DE FIGURAS

RESUMEN

Este informe tiene como propósito describir y detallar las actividades y los procesos necesarios para desarrollar aplicaciones móviles nativas de calidad para proyectos de talla internacional. En la actualidad, el sector *retail* está siendo impulsado por los nuevos canales digitales, que permiten alcanzar un público más amplio y lograr ventas que pueden duplicar las cifras obtenidas en años anteriores. Hoy en día, muchas empresas del sector *retail* cuentan al menos con una página web para exponer y vender sus productos. Sin embargo, con la reciente moda de la movilidad, impulsada por teléfonos inteligentes con acceso a internet, las aplicaciones móviles han adquirido un gran protagonismo. Una aplicación móvil a medida, que exponga los productos de la organización y brinde una experiencia diferente al cliente final para comprar por internet, representa una ventaja significativa sobre los competidores. Este informe describe el proyecto denominado Wow, implementado por la organización Grupo Éxito con el propósito de incrementar las ventas durante el evento comercial conocido como Wow, al que hace alusión el nombre del proyecto. Este evento, que se celebra a nivel nacional en Colombia cada año en agosto, ofrece grandes descuentos en productos de diversas categorías durante una semana en todo el país. La organización Grupo Éxito comenzó con un almacén en Medellín, Colombia, centrado en la venta de textiles y productos de vestuario. Actualmente, la organización tiene presencia en 23 departamentos de Colombia, con 515 almacenes y cerca de 35 000 empleados. Además, está presente en Uruguay, Argentina y otros países latinoamericanos. Su objetivo es mantener el liderazgo en el sector *retail* y brindar una atención de calidad a sus clientes mediante sus diferentes canales de ventas físicas y digitales. Como desarrollador de aplicaciones móviles en el Proyecto Wow, he definido mis funciones y actividades para cumplir con los objetivos establecidos por la organización para este proyecto.

Palabras claves: aplicación móvil, evento Wow, grupo Éxito, proyecto Wow

ABSTRACT

This report aims to describe and detail the activities and processes required to develop high-quality native mobile applications for international projects. Currently, the retail sector is being driven by new digital channels, which allow reaching a broader audience and achieving sales that can double previous figures. Nowadays, many companies in the retail sector have at least a website to display and sell their products. However, with the recent trend of mobility, driven by smartphones with internet access, mobile applications have gained significant prominence. A custom mobile application that showcases the organization's products and provides a unique shopping experience online represents a significant advantage over competitors. This report describes the project called Wow, implemented by Grupo Exito with the aim of increasing sales during the commercial event known as Wow, which the project name refers to. This event, held nationwide in Colombia every August, offers substantial discounts on products from various categories for a week throughout the country. Grupo Exito started with a store in Medellín, Colombia, focused on selling textiles and clothing products. Currently, the organization operates in 23 departments of Colombia, with 515 stores and approximately 35 000 employees. Additionally, it is present in Uruguay, Argentina, and other Latin American countries. Its goal is to maintain leadership in the *retail* sector and provide high-quality customer service through its various physical and digital sales channels. As a mobile applications developer in the Wow Project, I have defined my roles and activities to meet the objectives set by the organization for this project.

Keywords: Grupo Exito, mobile application, Wow event, Wow project

INTRODUCCIÓN

El comercio electrónico en Perú está experimentando un crecimiento notable, con cada vez más personas optando por utilizar canales digitales para adquirir productos y servicios en el sector *retail* (Blacksip, 2019).

Como se mencionó anteriormente, el sector *retail* está siendo impulsado por los canales digitales, logrando ventas aceleradas que eran inimaginables décadas atrás, no solo en Perú, sino también en países como Colombia, Panamá, Ecuador, Chile, Argentina y muchos otros en casi todo el mundo. Diferentes empresas del sector *retail* están aprovechando esta oportunidad para expandir sus ventas y llegar a un público más amplio mediante internet.

La organización Grupo Éxito es una multinacional colombiana líder en el sector *retail* en América Latina, que está adoptando nuevas tecnologías para prepararse ante entornos dinámicos e inesperados, manteniendo así altos estándares de calidad en la atención al cliente.

Preparándose para el evento comercial llamado Wow, que tuvo lugar en Colombia en el 2023, la organización Grupo Éxito se propuso, a inicios de ese año, superar sus ventas totales respecto al año anterior. Para lograr este objetivo, implementaron el proyecto denominado Wow, en alusión al evento comercial mencionado. Este proyecto se enfoca en optimizar su aplicación móvil de ventas, llamada Éxito, para manejar un mayor número de usuarios y desarrollar nuevas funcionalidades que brinden una mejor experiencia de compras por internet.

Este informe detalla el proceso de desarrollo y mantenimiento de la aplicación móvil llamada Éxito, desempeñando el rol de desarrollador de aplicaciones móviles, estructurado en cinco capítulos:

- En el Capítulo I se abordan aspectos generales de la organización, como sus actividades principales, reseña histórica, organigrama, visión y misión, descripción del área donde se realizaron las actividades profesionales, y la descripción del cargo con sus respectivas responsabilidades.

- En el Capítulo II se expone el diagnóstico situacional de la aplicación móvil llamada Éxito, se identifican las necesidades de negocio, se definen objetivos y resultados esperados.

- En el Capítulo III se exponen las bases teóricas y un glosario de términos relevantes empleados en el presente informe.

- En el Capítulo IV se describen las actividades profesionales realizadas, los aspectos técnicos involucrados y el proceso de ejecución detallado.

- Finalmente, en el Capítulo V se presentan los resultados finales, los logros alcanzados, las dificultades enfrentadas, algunas propuestas de mejora y los aportes significativos a la organización.

CAPÍTULO I
ASPECTOS GENERALES DE LA ORGANIZACIÓN

1.1. Datos generales de la organización

La organización Grupo Éxito, que en adelante se mencionará simplemente como Grupo Éxito, es una multinacional colombiana líder en el sector *retail*. Posee marcas reconocidas como Éxito, Carulla, Super Inter, Surtimax, Surtimayorista y la marca de centros comerciales Viva.

Recientemente, Grupo Éxito fue reconocido en la categoría Members por su gestión destacada en dimensiones ambientales, sociales, económicas y de gobierno corporativo, posicionándose dentro del 30 % con mejor desempeño en el sector *retail* de alimentos (Retailer, 2024).

Actualmente, Grupo Éxito tiene presencia en 23 departamentos de Colombia con 515 almacenes y cerca de 35 000 empleados. Además, está presente en Uruguay con 90 tiendas y 6000 empleados, y en Argentina con 25 almacenes y cerca de 2500 empleados.

1.2. Actividades principales de la organización

Como se mencionó anteriormente, Grupo Éxito es una organización que opera en el sector *retail*, centrándose principalmente en la venta al por mayor y al por menor de productos de consumo mediante supermercados e hipermercados bajo las marcas Éxito, Carulla, Surtimax y Super Inter en Colombia. Además, ofrece productos de conveniencia mediante Éxito Express, ubicados estratégicamente para satisfacer las necesidades diarias de los clientes.

El Grupo Éxito también está activamente involucrado en el comercio electrónico. Adicionalmente, ofrece servicios financieros mediante tarjetas de crédito, desarrolla y comercializa productos bajo marcas propias, y participa en programas de sostenibilidad y responsabilidad social corporativa, entre otras actividades.

1.3. Reseña histórica de la organización

- **Fundación y primeros años (1949 - 1970)**

➢ **1949**: Se funda el primer almacén Éxito en Medellín, Colombia por Gustavo Toro Quintero. Inicialmente se centraba en la venta de textiles y productos de vestuario.

➢ **1955**: Se expande la oferta de productos, incluyendo alimentos y otros bienes de consumo diario.

- **Expansión Nacional (1970 - 1999)**

➢ **1970**: La empresa comienza su expansión dentro de Medellín y luego a otras ciudades dentro de Colombia. En esta década se abre el primer supermercado Éxito en Bogotá.

➢ **1994**: Éxito se convierte en la primera cadena de almacenes en Colombia en cotizar en la Bolsa de Valores de Colombia.

- **Alianza y Crecimiento Internacional (2000 - 2010)**

➢ **2000**: El Grupo Casino, un gigante francés del comercio minorista, adquiere una participación significativa en Grupo Éxito, facilitando su expansión y modernización.

➢ **2007**: Grupo Éxito adquiere Carulla Vivero, consolidándose como líder del mercado minorista en Colombia.

- **Diversificación y Sostenibilidad (2010 - 2020)**

➢ **2011**: Expande su presencia internacional al adquirir las operaciones de Grupo Disco y Devoto en Uruguay, y Libertad en Argentina.

➢ **2013**: Inauguración del Centro Comercial Viva Laureles en Medellín, marcando su entrada en el negocio de centros comerciales.

➢ **2016**: Se abre el centro comercial Viva Envigado, uno de los más grandes y modernos de Colombia.

- **2019**: Grupo Éxito implementa su estrategia de omnicanalidad, fortaleciendo su presencia en el comercio electrónico y mejorando la experiencia del cliente tanto en tiendas físicas como en línea.

- **Innovación y Responsabilidad Social (Actualidad)**
- El Grupo Éxito sigue innovando con nuevas tecnologías y servicios, incluyendo soluciones de pago digital, entrega a domicilios y programas de lealtad.

1.4. Organigrama de la organización

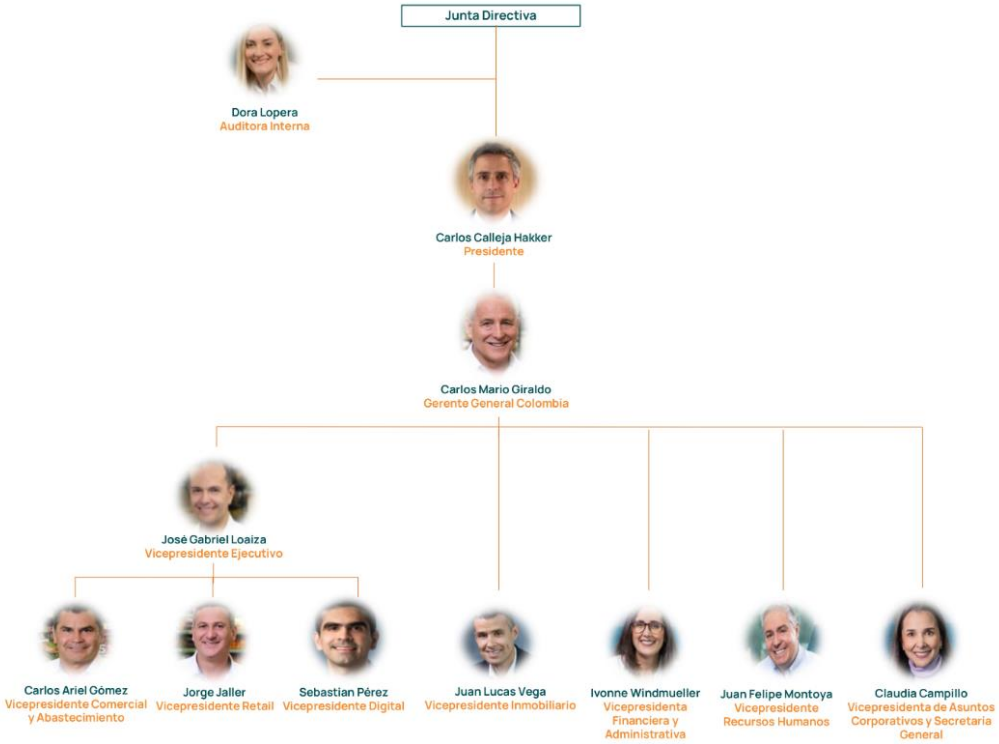


Figura 1. Estructura organizacional del Grupo Éxito
Fuente: Grupo Éxito

1.5. Visión y misión

- **Misión**

Trabajamos para que el cliente regrese, es la misión que tiene el Grupo Éxito de cara a sus clientes finales.

- **Visión**

Para el 2035 ser la mayor cadena de supermercados a nivel internacional, satisfaciendo a nuestros socios y clientes con una compra única de calidad, variedad, precio y servicio, basada en la atención y el compromiso de nuestros trabajadores.

1.6. Descripción del área donde se desarrollaron las actividades profesionales

Según la estructura organizacional del Grupo Éxito, las actividades profesionales se llevaron a cabo en el área de la Vicepresidencia Digital. Esta vicepresidencia lidera la transformación digital de la empresa, impulsando la innovación y la implementación de tecnologías digitales para mejorar la experiencia del cliente.

Recientemente, esta área ha desarrollado y lanzado robustas plataformas de comercio electrónico, conocidas como canales de ventas digitales, que ofrecen una experiencia de compra fluida y personalizada para los clientes, manteniendo la competitividad en un mercado cada vez más digital y centrado en el cliente. Entre estos canales de ventas digitales, los más populares son la aplicación web de ventas llamada exito.com y la aplicación móvil de ventas llamada Éxito, que en adelante se mencionará simplemente como «aplicación móvil».

Es importante destacar que esta vicepresidencia cuenta con diversos proveedores de servicios y recursos humanos, entre los cuales se encuentra Pragma S. A. C. que es una empresa especializada en el sector tecnológico que proporciona servicios y recursos humanos. Mi rol como desarrollador de *software* está vinculado a Pragma S. A. C., la cual se encarga de brindarme un contrato, gestionar mis pagos de nómina y ofrecerme todos los beneficios legales correspondientes.

1.7. Descripción del cargo y responsabilidades

Dentro del área de la Vicepresidencia Digital del Grupo Éxito, desempeñé el cargo de desarrollador de aplicaciones móviles especializado en el sistema operativo Android.

De acuerdo con mi rol, mis responsabilidades fueron:

- Participar en el diseño, desarrollo, implementación y mantenimiento de la aplicación móvil.

- Colaborar estrechamente con equipos multidisciplinarios, incluyendo diseñadores, analistas de negocio y otros desarrolladores

- Implementar prácticas de seguridad robustas para proteger la integridad de los datos de los usuarios y de la aplicación móvil en general.

- Mejorar la calidad de código fuente de la aplicación móvil, implementando pruebas unitarias.

- Solucionar los errores o fallas en la aplicación móvil identificados durante las pruebas internas.

- Participar en procesos de revisión de código fuente de la aplicación móvil

- Proponer mejoras en las prácticas de desarrollo.

CAPÍTULO II
ASPECTOS GENERALES DE LAS ACTIVIDADES PROFESIONALES

2.1. Diagnóstico situacional

El Grupo Éxito cuenta con diferentes canales de ventas digitales, siendo los más importantes su aplicación móvil de ventas llamada Éxito y su aplicación web de ventas exito.com. Siendo esta última el principal canal de ventas digital de la organización, representando casi el 40 % de sus ventas totales. Esta aplicación web es soportada y mantenida por un equipo de valor continuo (EVC) o célula, compuesto por profesionales encargados de desarrollar, innovar y mantenerla.

Por otra parte, su aplicación móvil Éxito es desarrollada y mantenida por el Equipo Apps, el cual está conformado por diferentes equipos de valor continuo (EVC) o células.

La aplicación móvil Éxito, posee una variedad enorme de funcionalidades, en los cuales los principales son un sistema de autenticación, un sistema de pantalla principal parametrizable, sección de pasillos, sección de listado de productos, sección de detalle de productos, sección de lista de deseos, sección de verificación de precios mediante escaneo de código de barras del producto, sistema de entrega a domicilio o recojo en tienda, sistema de carrito de compras, sistema de pasarela de pagos.

Estas características existentes fueron desarrolladas bajo una arquitectura definida. Además, la aplicación cuenta con un proceso de desarrollo, integración continua y despliegue continuo ya definido, así como, con diversos recursos como API, repositorios, analítica, pipelines y ambientes para desarrollo, pruebas y producción. También tienen una versión estable disponible en las tiendas oficiales de Play Store y App Store.

Finalmente, se observó que el proyecto de la aplicación móvil contaba con una baja cobertura de pruebas unitarias, diferentes fallos en producción y los diversos bloques de *code smell* reportados por la herramienta SonarQube.

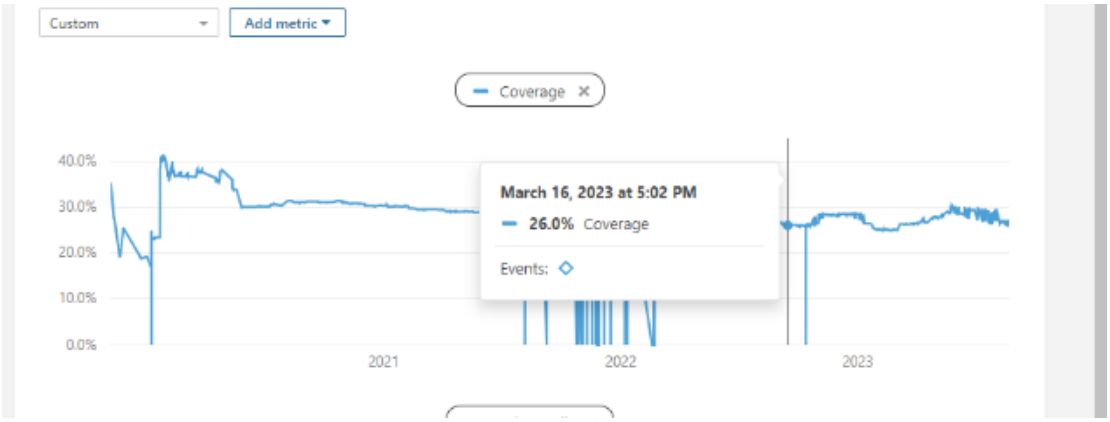


Figura 2. Panel de cobertura de código fuente del proyecto en Sonar Qube
Fuente: Grupo Éxito

Según la figura anterior, el proyecto de la aplicación contaba con un 26 % de cobertura de pruebas unitarias para marzo del 2023, fecha previa al inicio de las actividades profesionales.

2.2. Oportunidades y necesidades de la organización

Una gran oportunidad comercial para el Grupo Éxito es el evento comercial Wow, el cual atrae a una gran cantidad de clientes debido a los descuentos en diversos productos que ofrece la organización. Con el fin de maximizar la disponibilidad de canales de ventas, tanto físicos como digitales, el Grupo Éxito busca preparar su aplicación móvil para atraer a un mayor número de clientes y ofrecerles una experiencia de compra en línea de alta calidad.

A continuación, se detallan las necesidades del Grupo Éxito:

- Contar con una aplicación móvil libre de errores o fallas, que impidan al usuario final concretar sus compras por internet.

- Contar con una aplicación con nuevas características modernas e innovadoras, que permitan al usuario tener una mejor experiencia de compras.

Con el objetivo de satisfacer estas necesidades, el Grupo Éxito creó el Proyecto Wow, conformado por el siguiente Equipo de Valor Continuo (EVC).

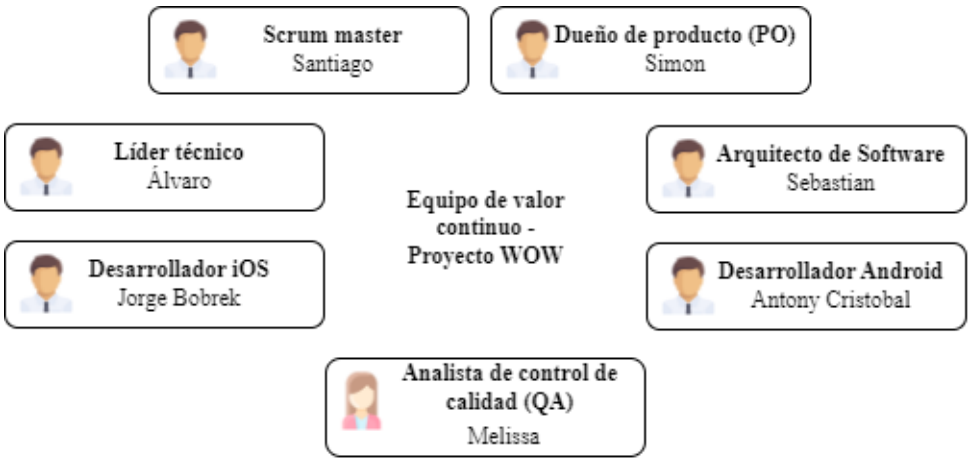


Figura 3. Diagrama del equipo de valor continuo asignado al proyecto Wow
Fuente: Grupo Éxito

2.3. Objetivos de la actividad profesional

El objetivo principal de un desarrollador de aplicaciones móviles en un proyecto es desarrollar nuevas funcionalidades definidas por la organización y mantener la aplicación móvil de manera oportuna. Esto implica:

- Desarrollar e integrar en su totalidad los requerimientos de la organización.
- Aumentar la cobertura de pruebas unitarias del código fuente de la aplicación móvil.
- Reducir la cantidad de reportes de «Code Smell» en Sonar Qube.
- Mantener la arquitectura existente en el proyecto, respetando los principios Solid y aplicando buenas prácticas en la escritura de código.
- Reducir la deuda técnica existente en el proyecto de la aplicación Éxito.
- Solucionar errores y fallas reportadas por el equipo de calidad de manera oportuna.
- Publicar una versión estable de la aplicación en las tiendas de Play Store y App Store, con las nuevas funcionalidades desarrolladas.

CAPÍTULO III
MARCO TEÓRICO

3.1. Bases teóricas

- ***Retail***

El sector *retail*, también conocido como comercio minorista, incluye establecimientos como supermercados, tiendas de marcas, almacenes, centros comerciales y tiendas departamentales. Su objetivo es llevar productos de consumo masivo desde los fabricantes o mediante intermediarios hasta el consumidor final. Este sector es crucial para la economía, ya que se encuentra entre los tres principales generadores de empleo, representando aproximadamente el 10 % del mercado en muchos países (1).

- **Caneles digitales**

Los canales de venta son las vías utilizadas para ofrecer productos y servicios a los consumidores, como sitios web, medios de comunicación o plataformas que permiten interactuar con ofertas. Además, estos canales no solo sirven como escaparate para los clientes que ya están interesados en comprar, sino también como una herramienta publicitaria que atrae a nuevos clientes (2).

- **Aplicaciones móviles nativas / App**

Una app nativa es una aplicación que se encuentra en el escritorio del *smartphone* y se accede mediante su propio icono. Estas aplicaciones se instalan desde tiendas de aplicaciones como App Store o Google Play y están diseñadas específicamente para plataformas como iOS o Android. Este tipo de aplicaciones están creadas para optimizar al máximo las funcionalidades de cada *smartphone*, aprovechando tanto el *hardware* como el *software* del dispositivo (3).

- **Sistemas operativos móviles**

Un sistema operativo móvil ofrece una interfaz de usuario intuitiva y optimizada para pantallas pequeñas, permitiendo a los usuarios navegar por aplicaciones y acceder a información de manera eficiente. Además, proporciona una variedad de funciones y características específicas para el uso móvil, como la conexión a redes inalámbricas, el acceso a aplicaciones en línea y el uso de servicios de ubicación. En el mercado existen diferentes sistemas operativos móviles, como Android, iOS, Windows Mobile y BlackBerry OS, cada uno con sus propias características y beneficios (4).

- **Tiendas de aplicaciones Play Store y App Store**

Una tienda de aplicaciones, es una plataforma que permite a los usuarios buscar e instalar programas en computadoras y dispositivos móviles, donde estas aplicaciones suelen llamarse simplemente apps. A través de una tienda de aplicaciones, es posible buscar, comprar, descargar e instalar apps directamente en el dispositivo de manera más fácil que con la instalación tradicional (4).

- ***Front end***

El *front end*, también llamado «lado cliente», es la parte de una web o aplicación a la que los usuarios tienen acceso y con la que pueden interactuar. Esto incluye los colores y estilos, la estructura de la página, los botones y las acciones disponibles, así como, todo lo que ocurre en pantalla. En resumen, abarca todo lo que el usuario puede experimentar y percibir de manera directa (5).

- ***Scrum***

Scrum es una metodología que facilita la colaboración de equipos para lograr un trabajo de alto impacto. Proporciona un conjunto de valores, roles y directrices que ayudan a los equipos a centrarse en la iteración y la mejora continua en proyectos complejos (6).

- ***Sprint***

En Agile, un *sprint* es un intervalo de tiempo corto y definido en el que se realiza trabajo en un proyecto. Los *sprints* son esenciales en la metodología *Scrum* y suelen durar entre dos y cuatro semanas. (7).

- ***Backlog***

Un *backlog*, también conocido como pila, es una lista organizada que generalmente contiene trabajos, requisitos, acciones pendientes, tareas, entre otros elementos. La estructura

de lista implica que cada elemento sigue al anterior, permitiendo ordenarlos de acuerdo con las necesidades y prioridades del momento (8).

- ***DevOps***

DevOps, que significa Desarrollo y Operaciones, es un enfoque basado en principios ágiles en el cual los *stakeholders* y los departamentos de desarrollo, operaciones y control de calidad trabajan juntos para lograr entregas continuas. Esto permite a la empresa aprovechar rápidamente las oportunidades del mercado y reducir la carga de trabajo (9).

- **UX/UI**

UX, o Experiencia de Usuario, se centra en el diseño de productos y servicios que priorizan las necesidades humanas, con el fin de proporcionar experiencias satisfactorias para los usuarios. Por otro lado, el diseño UI, o Interfaz de Usuario, se enfoca en el diseño gráfico y visual de una aplicación, es decir, cómo se presenta y cómo interactúan los usuarios con ella (10).

- ***Stakeholder***

Un *stakeholder* es cualquier individuo o entidad que tenga un interés o influencia en el proyecto. Pueden incluir clientes, usuarios finales, patrocinadores, gerentes, propietarios de productos, miembros del equipo, entre otros (11).

- **SOLID**

Es un conjunto de cinco principios diseñados para hacer que el desarrollo de *software* sea más comprensible, flexible y fácil de mantener. Estos principios pueden aplicarse en cualquier lenguaje de programación orientado a objetos (OOP); por lo tanto, no se trata de un *framework* o librería específica y no está limitado a una tecnología particular.

S - Principio de Responsabilidad Única: Cada clase debe tener una sola razón para cambiar.

O - Principio de Abierto/Cerrado: Las entidades de *software* deben ser abiertas para la extensión, pero cerradas para la modificación.

L - Principio de Sustitución de Liskov: Los objetos deben poder ser sustituidos por instancias de sus subtipos sin afectar el comportamiento del programa.

I - Principio de Segregación de Interfaces: Las interfaces deben ser específicas para los clientes que las utilizan, evitando interfaces monolíticas.

D - Principio de Inversión de Dependencias: Los módulos de alto nivel no deben depender de módulos de bajo nivel, ambos deben depender de abstracciones (12).

- **API**

Una API (Interfaz de Programación de Aplicaciones, *Application Programming Interface*) es el método mediante el cual dos o más computadoras se comunican, típicamente mediante solicitudes HTTP (13).

- **HTTP**

HTTP, conocido como Protocolo de Transferencia de Hipertexto en inglés, es un protocolo esencial que permite realizar solicitudes de datos y recursos mediante la web. Define reglas y estructuras estándar para la comunicación, creando un lenguaje común entre quien envía y quien recibe la información (14).

- **SDK**

Un SDK es un conjunto de herramientas que los programadores utilizan para desarrollar aplicaciones o funciones específicas para cada sistema operativo (15).

3.2. Glosario de términos

- **MVVM**

Es un patrón de diseño arquitectónico utilizado principalmente en el desarrollo de aplicaciones de *software*. Consta del Modelo (Model), el cual representa los datos y la lógica del negocio de la aplicación. La vista (*View*), el cual es la interfaz de usuario que muestra la información al usuario y captura las interacciones del usuario. El ViewModel *(ViewModel*), quien actúa como intermediario entre la Vista y el Modelo, preparando los datos del Modelo para que la Vista los presente, y también procesa las acciones del usuario en la Vista para actualizar el Modelo si es necesario.

Esto facilita la separación de responsabilidad y ofrece un mayor modularidad en el desarrollo de aplicaciones, lo que resulta en un código más mantenible y escalable.

- **TDD**

TDD (*Test-Driven Development*) es una metodología de desarrollo de *software* donde las pruebas unitarias se escriben antes de implementar el código de producción. El proceso

implica escribir una prueba unitaria que inicialmente falle debido a que la funcionalidad aún no está implementada. Posteriormente, se implementa el código mínimo necesario para que la prueba pase satisfactoriamente. Y finalmente, se refactoriza el código sin cambiar su comportamiento externo, asegurando que sea limpio y eficiente. El TDD promueve un desarrollo más robusto al enfocarse en pruebas desde el inicio, lo que conduce a un código más confiable y menos propenso a errores el cual fue de gran ayuda para aumentar la cobertura de pruebas unitarias del proyecto.

- **PLP**

La presente abreviación se usa muy comúnmente en los equipos de desarrollo dentro de la cuenta del Grupo Éxito, así como, en los mismos representantes del negocio, para referirse a la Pantalla Listado de Productos. Este término hace referencia a la vista en la aplicación que contiene un listado en formato de presentación de grillas con los datos de todos los productos que se pueden mostrar en la vista.

- **PDP**

Esta abreviación, al igual que la anterior, es muy común en todo el equipo involucrado y hace referencia a la Pantalla Detalle de Producto, que representa la vista donde se muestra de forma detallada la información de un producto específico, como la imagen, el precio, el precio por gramo, descuentos, ofertas adicionales, y la opción de agregar al carrito.

- **EVC / Célula**

Esta abreviación hace referencia a Equipo de Valor Continuo, término que, al igual que «Célula», se emplea para referirse a los equipos de desarrollo. Estos equipos de desarrollo están conformados por desarrolladores, un líder técnico, un arquitecto, un *coach* ágil o *scrum master*, un analista de calidad y un dueño de producto (PO).

- **PO**

PO es la abreviatura proveniente del inglés «Product Owner», que hace referencia al dueño del producto. Este actor pertenece al marco *Scrum* y se encarga de representar al lado del negocio, teniendo claras sus necesidades y la capacidad de tomar decisiones basadas en los intereses del negocio.

- **Éxito y Carulla**

Son los nombres de las marcas del Grupo Éxito, que han sido delegados a sus dos aplicaciones móviles, las cuales se encuentran en las tiendas oficiales de aplicaciones móviles.

- **Deuda técnica**

Término empleado para referirse a algunas actividades pendientes, como la solución de fallas o errores reportados que salieron del alcance del *sprint* y que no son prioritarios.

- ***Code Smell***

Término empleado para señalar que el código podría ser mejorado para aumentar su legibilidad, mantenibilidad y calidad general. Algunos ejemplos comunes de *code smells* incluyen código duplicado, métodos o clases excesivamente largos, y uso inapropiado de dependencias. Identificar y corregir estos *Code Smell* ayuda a prevenir problemas futuros y a mantener el código limpio y eficiente.

- **Refactorizar**

Proceso de mantenimiento del código que no está enfocado en añadir o arreglar errores, sino en mejorar la facilidad de comprensión del código o cambiar su estructura y diseño, así como, eliminar código muerto.

CAPÍTULO IV
DESCRIPCIÓN DE LAS ACTIVIDADES PROFESIONALES

4.1. Descripción de actividades profesionales

4.1.1. Enfoque de las actividades profesionales

Como desarrollador de aplicaciones móviles, mi enfoque se basó en emplear las mejores prácticas de desarrollo de código limpio, asegurando la calidad y mantenibilidad exclusivamente de la aplicación móvil Éxito. Temas como bases de datos, microservicios, API, estabilidad de ambientes, pruebas de calidad, procesos DevOps y otras áreas fuera del proyecto de la aplicación móvil no fueron de mi competencia, ya que esas responsabilidades fueron abordadas por otros equipos.

Por otra parte, utilicé técnicas de resolución de problemas como el análisis de causa raíz, y prueba y error.

De igual forma, la colaboración interdisciplinaria también fue fundamental para superar los diversos desafíos técnicos y organizacionales que se presentaron durante la ejecución del Proyecto Wow.

4.1.2. Alcance de las actividades profesionales

- Participar en el desarrollo e implementación de nuevas funcionalidades en la aplicación.

- Dar mantenimiento a la aplicación

- Colaborar estrechamente con equipos multidisciplinarios, incluyendo diseñadores, analistas de calidad y otros desarrolladores.

- Cumplir con los estándares de calidad de *software* establecidos por la organización.

- Realizar pruebas unitarias para garantizar la calidad del código y la funcionalidad de la aplicación.

- Reducir la deuda técnica existente en el proyecto de la aplicación.

- Solucionar errores y fallas reportados por el área de calidad.

- Participar en procesos de revisión de código y proponer mejoras en las prácticas de desarrollo y la infraestructura de la aplicación.

Con este alcance, se garantizó un enfoque pleno en el desarrollo de las actividades del Proyecto Wow, aprovechando el tiempo y los recursos tecnológicos y humanos de manera óptima.

4.1.3. Entregables de las actividades profesionales

- Formulario de definición de terminado (DOD)

- Evidencia de funcionalidad en formato video

- Evidencia de funcionalidad en formato imagen

- Matriz de impacto

- Matriz de riesgos para salida a producción

4.2. Aspectos técnicos de la actividad profesional

4.2.1. Arquitectura de la solución

A continuación, se muestra una imagen representativa de la arquitectura empleada en el desarrollo del proyecto. Es importante mencionar que el diagrama es solo una representación general del diagrama original, con la finalidad de no revelar información sensible del proyecto perteneciente al Grupo Éxito.

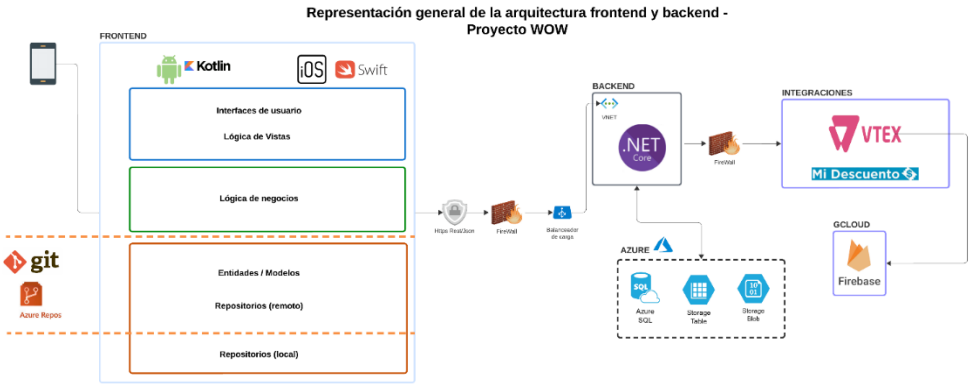


Figura 4. Diagrama de la arquitectura Front-end y Back-end para el proyecto Wow. Fuente: Grupo Éxito

En el lado *Front end* de la aplicación móvil se empleó una arquitectura limpia, segregada por capas: la capa de presentación, la capa de lógica de negocios y la capa de infraestructura. De esta manera, se mantuvo un proyecto bien estructurado, mantenible y escalable.

Por parte del *Backend* de la aplicación, el Grupo Éxito cuenta con servicios que exponen los productos y datos utilizados por sus canales digitales, de tal manera que desde el lado del *Frontend* solo se consumen dichos servicios mediante las API que fueron brindadas por el equipo *Backend*.

El Grupo Éxito integró los servicios de VTEX, una plataforma de comercio electrónico que proporciona diversas herramientas para la gestión y administración de los productos que el Grupo Éxito ofrece al público en general. Además, VTEX proporciona un modelo de pagos digitales, delegando la responsabilidad de seguridad a este proveedor.

Finalmente, la aplicación implementó los servicios de analítica de *Firebase*, que ayudan al Grupo Éxito a monitorear las preferencias y el comportamiento del usuario en su interacción con la aplicación.

4.2.2. Metodologías

a. *Scrum*

Se eligió *Scrum* como marco de desarrollo ágil en todos los proyectos ejecutados hasta la fecha por varias razones clave.

Primero, *Scrum* proporciona una estructura sólida que facilita la organización y priorización del trabajo mediante su enfoque en *Sprints* y la gestión del *Product Backlog*. Este

enfoque permite una adaptación rápida a los cambios, lo cual es esencial en un entorno tan dinámico como el del *retail*.

Además, *Scrum* promueve la colaboración constante entre los miembros del equipo y con los *Sakeholders*, lo que garantiza que se comprendan y aborden correctamente las necesidades y expectativas del cliente. La transparencia y la comunicación regular mediante eventos como las reuniones diarias (*Daily Meeting*), las revisiones de *sprint* (*Sprint Reviews*) y las retrospectivas (*Sprint Retrospectives*) aseguran que el equipo pueda identificar y resolver problemas de manera proactiva.

Otro factor decisivo para elegir *Scrum* es su enfoque en la entrega continua de valor. Al dividir el trabajo en incrementos manejables, se pueden realizar entregas frecuentes de funcionalidades completas y funcionales. Esto no solo mantiene a los clientes satisfechos al ver resultados tangibles rápidamente, sino que también permite obtener retroalimentación temprana y ajustar el rumbo del proyecto según sea necesario.

Finalmente, la adaptabilidad y la mejora continua fomentadas por *Scrum* son esenciales para enfrentar el entorno VICA (Volatilidad, Incertidumbre, Complejidad y Ambigüedad) que actualmente caracteriza al sector *retail*. La capacidad de adaptarse rápidamente a los cambios del mercado, a las nuevas tecnologías y a las demandas de los clientes es crucial para mantener la competitividad y asegurar el éxito del proyecto.

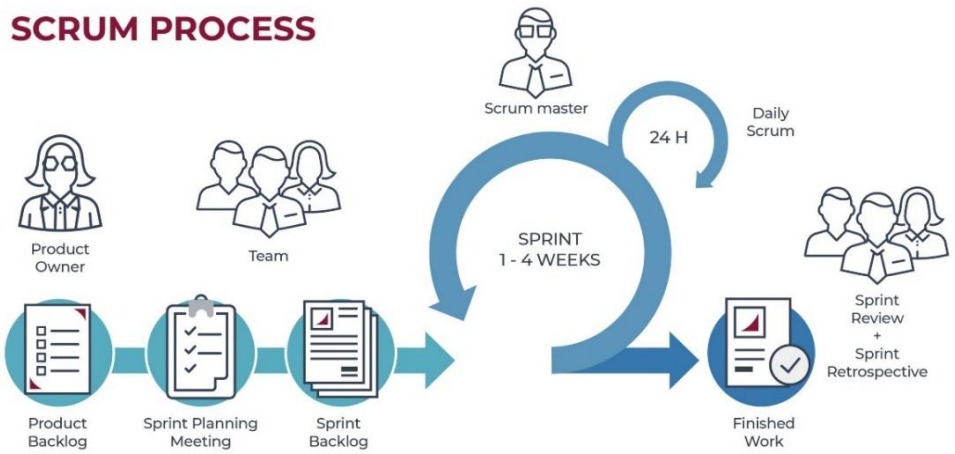


Figura 5. Proceso Scrum como marco de trabajo para el desarrollo ágil de software.
Fuente: Tomada de «Scrum: qué es y por qué se ha convertido en una de las metodologías ágiles más populares» (16)

En el proyecto Wow, se contó con una pila de producto (*Product Backlog*) que contenía las historias de usuario listas para refinar. El proceso *Scrum* comenzaba con la reunión de

refinamiento. En esta, el dueño del producto (*Product Owner*) exponía un conjunto de historias de usuario prioritarias al equipo de desarrollo (analista de calidad y desarrollador de aplicaciones). Durante la reunión, ambos roles expresaban sus dudas e inquietudes, definiendo así los criterios de aceptación para cada historia de usuario.

Posteriormente, se realizaba la reunión de planeación (*Planning*), donde el desarrollador, habiendo previamente definido un conjunto de actividades y tareas para cada historia de usuario refinada, estaba listo para estimar los puntos de esfuerzo para cada historia los cuales estaban representados según la serie de Fibonacci. Una vez asignados los puntos de esfuerzo, el dueño del producto (*Product Owner*) agrupaba un conjunto de historias prioritarias para desarrollarse en el siguiente *Sprint*.

Normalmente, cada *Sprint* abarcaba historias de usuario que sumaban un total de 16 puntos de esfuerzo y tenía una duración de 15 días, destinados al desarrollo de estas historias.

Al finalizar el *Sprint*, se llevaba a cabo una reunión de revisión (*Review*), donde cada desarrollador mostraba las funcionalidades trabajadas durante el *Sprint* anterior, finalizando así un ciclo del proceso *Scrum*.

Si durante la fase de estimación de puntos de esfuerzo, el equipo de desarrollo subestimaba el esfuerzo requerido y no lograba completar una historia de usuario, estas historias se trasladaban al siguiente *Sprint* como prioritarias.

Este proceso se repitió hasta completar todas las historias de usuario en la pila de producto (*Product Backlog*).

A continuación, se muestra la tabla *Backlog* para visualizar de manera general la totalidad de requerimientos delegados a mí, los cuales se abordaron durante toda la ejecución del proyecto.

b. DevOps

Se crearon pipelines utilizando la herramienta Azure DevOps para:

- Compilar el código fuente de cada proyecto de las aplicaciones.
- Ejecutar las pruebas unitarias.
- Validar la cobertura de código con SonarQube.
- Generar el archivo Bundle para Android.
- Publicar los archivos en App Center.

Dependiendo del destino del despliegue, el pipeline actúa de la siguiente manera:

- **Despliegue a QA**: App Center notifica a los analistas de calidad sobre nuevas versiones de la aplicación, permitiéndoles iniciar la etapa de pruebas.

- **Despliegue a Producción**: El pipeline realiza automáticamente el despliegue a las tiendas Play Store.

Estos pipelines se ejecutan automáticamente con cada nuevo despliegue al repositorio remoto Azure Repos. Así, cada vez que se realiza una solicitud de unión a la rama de QA o Producción, el pipeline se ejecuta, garantizando la integración y el despliegue continuo de las actualizaciones de la aplicación.

A continuación, se muestra un gráfico representando este proceso.

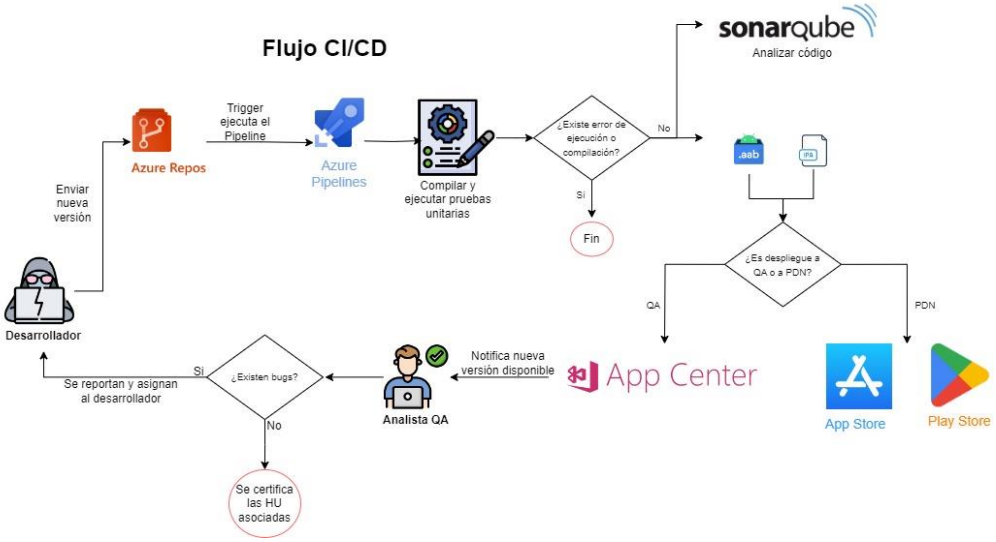


Figura 6. Flujo DevOps (CI/CD) aplicado al proyecto
Fuente: Grupo Éxito

c. Flujo de trabajo colaborativo Git heredado *Gitflow*

GitFlow permitió integrar continuamente código sin ocasionar conflictos con otros desarrollos en paralelo sobre el mismo código fuente de la aplicación. A continuación, se presenta el siguiente gráfico representativo del flujo de trabajo colaborativo *Gitflow*.

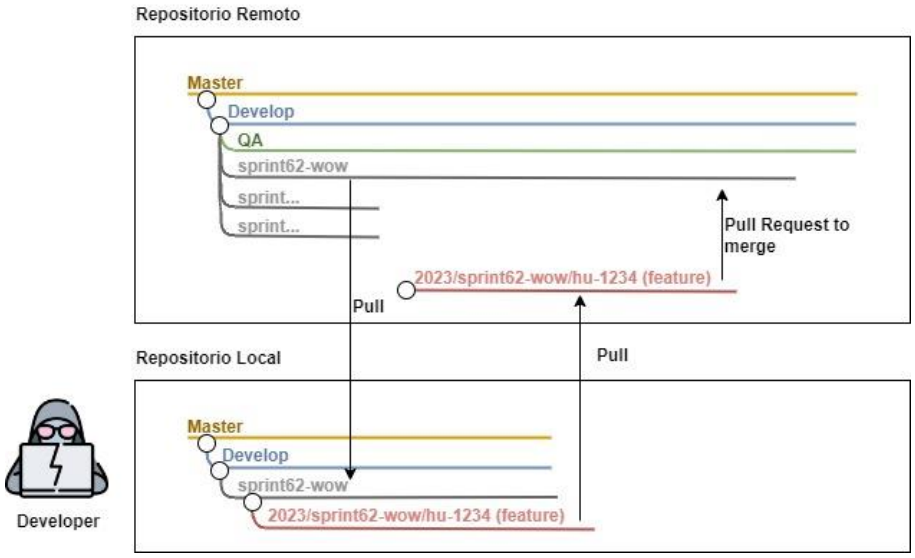


Figura 7. Flujo de trabajo Git heredado GitFlow aplicado en el proyecto

Para desarrollar cada historia de usuario, comencé actualizando la rama local creada para el *sprint* con la rama remota mediante el comando **git pull origin {nombre rama remota}**, posteriormente creé una rama *feature* a partir de esta última rama, teniendo en cuenta la siguiente sintaxis **({año}/{*sprint*}/HU-{Id de la HU})**, empleando el comando **git checkout -b {nombre de la nueva rama}**.

Este Id de la historia de usuario se obtiene de *Azure Boards*, en donde cada historia de usuario posee un identificador creado automáticamente según se muestra en la siguiente figura:

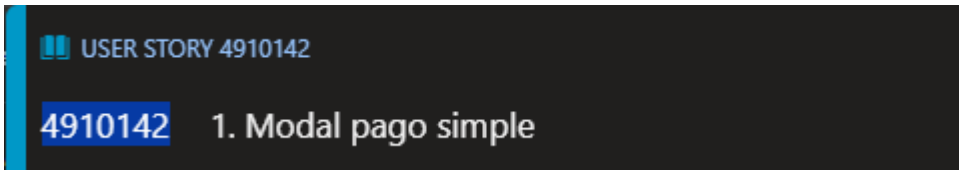


Figura 8. Ubicación del Id de las historias de usuario en Azure Boards

En la rama que creé, comencé escribiendo el código necesario para cumplir con los criterios de aceptación de la historia de usuario. Al finalizar el desarrollo, añadí los archivos nuevos o modificados al área de preparación utilizando el comando **git add.**, seguido del comando **git commit -m "{mensaje del commit}"** para comentar y guardar los cambios en el repositorio git local.

Posteriormente, actualicé mi rama del *sprint* local con la rama remota para asegurarme de tener los últimos cambios, usando el comando **git pull origin {nombre de la rama remota}**. Cuando aparecían conflictos de fusión (merge), los resolvía localmente para mitigar posibles conflictos en el repositorio remoto.

Con los conflictos resueltos, subí mi rama *feature local* al repositorio remoto con el mismo nombre, utilizando el comando **git push origin {nombre de la rama feature local}**. Luego, generé una solicitud de Pull Request (PR) para fusionar mi rama local recientemente subida con la rama remota correspondiente al *sprint*.

Este PR se representaba con un enlace que compartía en los grupos de desarrollo del proyecto, solicitando revisión y retroalimentación para posteriormente completar el proceso de fusión. Repetí este proceso para cada nuevo desarrollo, ajuste o solución de errores, asegurando así que las nuevas versiones de código se actualizarán adecuadamente.

4.2.3. Técnicas

a. Pruebas unitarias

El código fuente de la aplicación móvil tenía inicialmente una cobertura de código del 26 % y aproximadamente 477 reportes de «code smells». Para mejorar estas métricas, cada historia de usuario se desarrolló bajo el enfoque de Desarrollo Guiado por Pruebas (TDD).

Este enfoque implicó que, al comenzar el desarrollo de una nueva característica, primero se piensan y escriben las pruebas unitarias, y luego se desarrolla la lógica necesaria para satisfacer el requerimiento de la historia de usuario. Esta metodología proporcionó la seguridad de no disminuir la cobertura de código existente, y permitió centrarse en escribir pruebas unitarias sobre el código ya existente.

Como resultado, se logró incrementar la cobertura de código y reducir significativamente la cantidad de «code smells», mejorando así la calidad y mantenibilidad del código fuente de la aplicación móvil.

b. Refactorización

Para incrementar la cobertura de pruebas unitarias en el código fuente de la aplicación móvil, en ocasiones fue necesario refactorizar el código existente para facilitar su prueba. La refactorización permitió escribir un código más simple y fácil de entender, sin alterar el comportamiento externo de la aplicación. Este proceso contribuyó significativamente a mejorar la calidad del código y la eficacia de las pruebas unitarias.

c. Gestión de *Backlog*

Para estimar las historias de usuario adecuadamente, la organización contaba con una definición de Pivote, que servía como referencia para puntuar correctamente cada historia de usuario. Además, fue fundamental revisar detalladamente cada historia de usuario junto con los

recursos proporcionados, como API, diseños de interfaz y criterios de aceptación, para poder crear las tareas técnicas por realizar para implementar la funcionalidad.

Si los puntos de esfuerzo asignados a una historia de usuario superaban los 8 puntos, solicité al *Product Owner* dividirla en partes más pequeñas para mantener el desarrollo ágil y la entrega de valor con cada *Sprint*.

Se utilizó un *Product Backlog* para gestionar y priorizar las historias de usuario del proyecto.

El cuadro a continuación muestra el *Product Backlog*, que incluye todas las historias de usuario desarrolladas en el Evento Wow.

Leyenda: Proyecto Objetivo Épica Historia de Usuario
APP WOW
Chequeador
1. Chequeador Mi Descuento
1.1 Chequeador en Mi Descuento - **yo como** usuario **quiero** poder escanear productos en el almacén **para** conocer su precio y activar la oferta
2. Experiencia Carnes
2.1 Experiencia Carnes - PDP - **yo como** usuario de la App **necesito** pedir la carne con mis especificaciones **para** poder comprar como me gusta desde la App
2.2 Experiencia Carnes . PLP - **yo como** usuario de la App **necesito** pedir la carne con mis especificaciones **para** poder comprar como me gusta dede la App
3. Comentarios Frescos 2
3.1 Añadir notas a los productos frescos - Sincronización - **yo como** usuario de la App **necesito** poder añadir notas a mis productos **para** poder comprar como deseo
4. Nuevo Look Lobby
4.1 Añadir acción buscar en teclado de búsqueda - **yo como** usuario de la App **necesito** tener un accionamiento claro en el teclado a la hora de buscar **para** que la busqueda sea mas intuitiva
4.2 Nuevo Look App - **yo como** usuario de la App **necesito** ver un nuevo diseño en la App **para** tener una experiencia diferente y novedosa
5. Reducción deuda tecnica
5.1 Eliminar funcionalidad mi salud - **yo como** usuario **necesito** tener una App ligera, sin funcionalidades obsoletas **para** ocupar menor espacio de almacenamiento
5.2 Eliminar funcionalidad familia - **yo como** usuario **necesito** tener una App ligera, sin funcionalidades obsoletas **para** ocupar menor espacio de almacenamiento
5.3 Eliminar funcionalidad sucursales favoritas - **yo como** usuario **necesito** tener una App ligera, sin funcionalidades obsoletas **para** ocupar menor espacio de almacenamiento
5.4 Actualizar libreria lector codigo de barras - **yo como** usuario **necesito** tener una App ligera, sin funcionalidades obsoletas **para** ocupar menor espacio de almacenamiento
5.5 Depurar code smell - **yo como** usuario **necesito** tener una App ligera, sin funcionalidades obsoletas **para** ocupar menor espacio de almacenamiento
5.6 Incrementar cobertura depruebas unitarias fase 1 - **yo como** usuario **necesito** tener una App de calidad **para** evitar errores durante el uso
5.7 Incrementar cobertura depruebas unitarias fase 2 - **yo como** usuario **necesito** tener una App de calidad **para** evitar errores durante el uso
5.8 Incrementar cobertura depruebas unitarias fase 3 - **yo como** usuario **necesito** tener una App de calidad **para** evitar errores durante el uso
5.9 Incrementar cobertura depruebas unitarias fase 4 - **yo como** usuario **necesito** tener una App de calidad **para** evitar errores durante el uso
5.10 Incrementar cobertura depruebas unitarias fase 5 - **yo como** usuario **necesito** tener una App de calidad **para** evitar errores durante el uso
6. Migración AKS V2
6.1 Migrar servicio getProductDetail - **yo como** usuario de la App **necesito** consumir el nuevo servicio **para** observar productos
6.2 Migrar servicio searchProductGraphQL - **yo como** usuario de la App **necesito** consumir el nuevo servicio **para** buscar productos
6.3 Migrar servicio recommendedProducts - **yo como** usuario de la App **necesito** consumir el nuevo servicio **para** observar productos recomendados
6.4 Migrar servicio getProductsDetail - **yo como** usuario de la App **necesito** consumir el nuevo servicio **para** escanear productos
7. Modularización
7.1 Modularización metodo entrega - **yo como** usuario **necesito** tener una App de mayor calidad **para** optimizar el uso de recursos
8. Metricas
8.1 Metricas en PLP - **yo como** negocio **necesito** tener metricas de las interacciones del usuario en la App **para** tomar decisiones de negocio
8.2 Metricas en PDP - **yo como** negocio **necesito** tener metricas de las interacciones del usuario en la App **para** tomar decisiones de negocio
8.3 Metricas en Checkout - **yo como** negocio **necesito** tener metricas de las interacciones del usuario en la App **para** tomar decisiones de negocio

Figura 9. Backlog del proyecto Wow ordenado y priorizado
Fuente: Grupo Éxito

d. Arquitectura limpia

Se empleó una arquitectura limpia en el código fuente de la aplicación, lo que permitió definir una estructura más organizada del proyecto. Esto no solo facilitó la implementación de nuevas funcionalidades, sino que también mejoró la escalabilidad del proyecto.

Además, se utilizó una arquitectura de tres capas, conocida como MVVM (Modelo-Vista-ViewModel), lo que resultó en una estructura limpia y ordenada del código fuente de la aplicación.

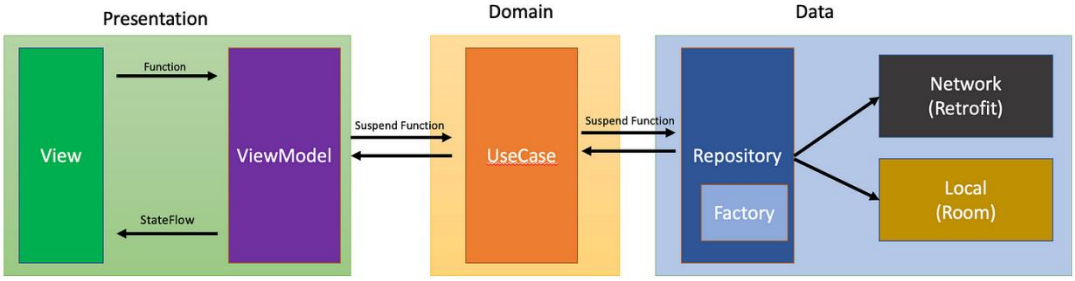


Figura 10. Arquitectura MVVM y Arquitectura Limpia aplicado en el proyecto
Fuente: Tomada de «MVVM Clean Architecture Pattern in Android with Use Cases» (16)

- View: Capa en la que se desarrollaron las interfaces de usuario.
- ViewModel: Capa en la que se desarrolló la gestión de estados para las interfaces.
- UseCase: Capa en la que se desarrollaron los casos de uso de negocio.
- Repository: Capa en la que se desarrollaron las conexiones a fuentes de datos
- Network: Capa en la que se desarrolló el consumo a API y fuentes de datos provenientes del internet
- Local: Capa en la que se desarrolló el consumo de datos locales o en caché.

Gracias a la aplicación de la Arquitectura limpia y la Arquitectura MVVM mencionados previamente, se logró contar con una estructura del código fuente de la aplicación móvil ordenada, simple e intuitiva tal cual se muestra en la siguiente figura.

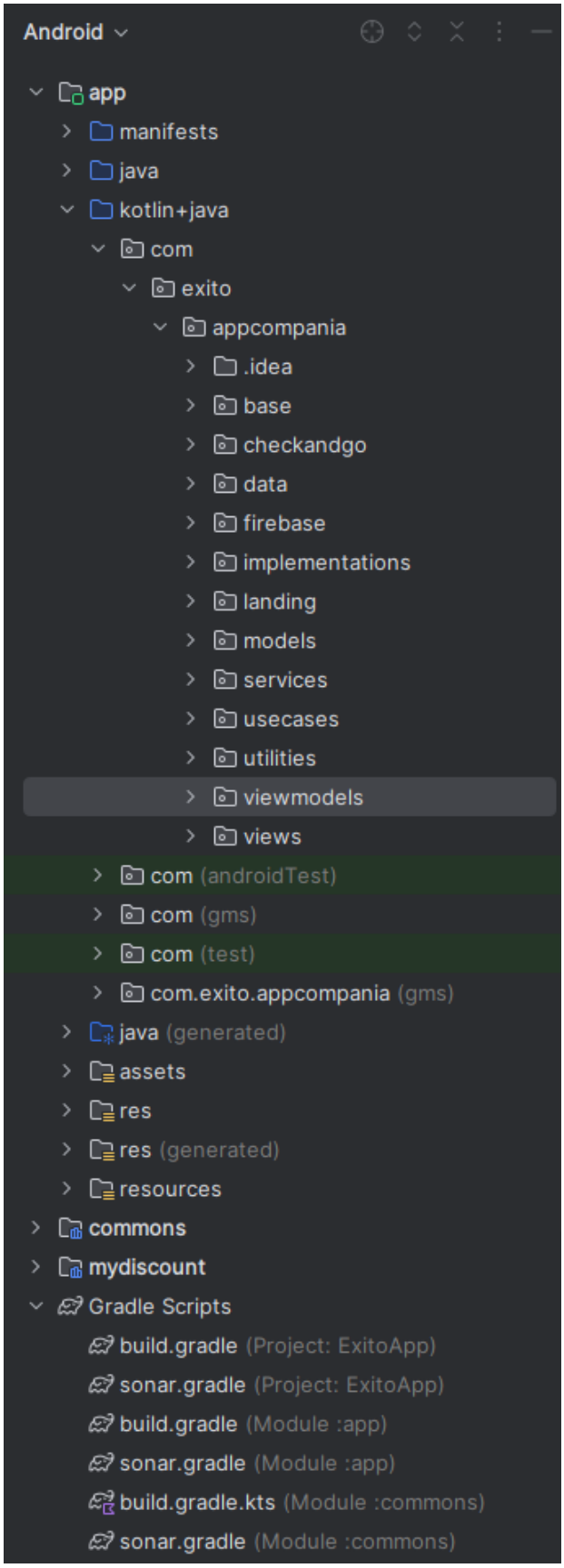


Figura 11. MVVM y Arquitectura limpia en la estructura del proyecto
Fuente: Grupo Éxito

e. Patrón de diseño Observador

Este patrón se aplicó concurrentemente para controlar los estados de la capa de *View Model* y ejecutar métodos en consecuencia del cambio de estado en la vista de la aplicación. Por ejemplo, se utilizó para cambiar el color de un botón al recibir una notificación de cambio de estado.

A continuación, se muestra una clase en donde se empleó el patrón.

```
Antony Cristobal +2
private fun initObserver() {
    viewModel.onProductDetailsRespond.observe(viewLifecycleOwner) { it: Boolean?
        it?.let { it: Boolean
            if (it) {
                viewModel.setProductValuesPartOne()
                viewModel.setProductValuesPartTwo()
                viewModel.setProductValuesPartThree()
                if (isFromPDP == true) {
                    viewModel.setSelectedMeatExpIfExist(item?.itemId ?: "")
                }
                viewModel.loadProductImage( image: item?.imageUrl ?: "")
                viewModel.loadTypeCutSelectedRecyclerView()
                viewModel.loadTypeCutPackRecyclerView()
                try {
                    val actualCategoryList =
                        viewModel.productDetail.categories?.find { cat -> cat.split( ...delimiters: "/").size == 4 }
                    if (actualCategoryList != null) {
                        viewModel.actualCategory = actualCategoryList.split( ...delimiters: "/")[2]
                    }
                } catch (e: Exception) {
                    CrashlyticsControl.sendCrashNotification(viewModel.CrashNotification(message = e.message))
                }
                initContentToShow()
            }
            viewModel.clearOnProductDetailsRespond()
        }
    }
```

Figura 12. Patrón de diseño Observador aplicado en el proyecto
Fuente: Grupo Éxito

Según el bloque de código anterior, se utilizó una variable de tipo estado mutable llamada **onProductDetailsRespond**, perteneciente a la capa de *View-Model*. Cuando cambia el estado de esta variable, se ejecuta una *lambda* que inicia diversos métodos y adicionalmente invoca el método **initContentToShow**, encargado de mostrar todos los componentes de esa vista.

f. Principios Solid

Se aplicaron diferentes principios Solid para diseñar la estructura de las clases, creando *software* eficaz, escribiendo código limpio, flexible, reutilizable y mantenible, y permitiendo una mejor escalabilidad del proyecto, facilitando de esta manera, la implementación de nuevas características en la aplicación a futuro.

A continuación, se mencionan los principios aplicados:

- **O (Open/Close)**

Se emplearon clases base las cuales sirvieron como clases que contienen métodos generales usados constantemente, de esta forma se reusó y optimizó el código escrito.

```
open abstract class BaseCacheParameter<T>(private val maxEntries:Int,private val ttl:Long)  {
    private val storedValue = mutableMapOf<String,T>()
    private val secuence = mutableMapOf<String,Int>()
    private val savedTime = mutableMapOf<String,Long>()

    private var counter = 0
    private var purgeTime :Long = 0L


    Javier Ramirez
    fun storeData(key:String,value:T){
        storedValue.put(key,value)
        secuence.put(key,counter++)
        savedTime.put(key, SystemClock.elapsedRealtime())
        purgueCache()
    }

    Javier Ramirez
    fun retrieveData(key:String):T?{
        purgueCache()
        val value = storedValue[key]
        return value
    }


    Javier Ramirez
    fun setPurgeTimeMillis(millis:Long){
        purgeTime = millis
    }

    Javier Ramirez
    fun invalidateAllCache(){
        storedValue.clear()
        secuence.clear()
        savedTime.clear()
    }

    Javier Ramirez
    private fun purgueCache(){...}
}
```

Figura 13. Principio SOLID Open/Close aplicado en el proyecto
Fuente: Grupo Éxito

- **I (Interface Segregation)**

Se emplearon diferentes interfaces para escuchar la respuesta de las consultas a los servicios usados. Estas interfaces cuentan con métodos necesarios, mejorando la lectura de código y facilitando la creación de pruebas unitarias.

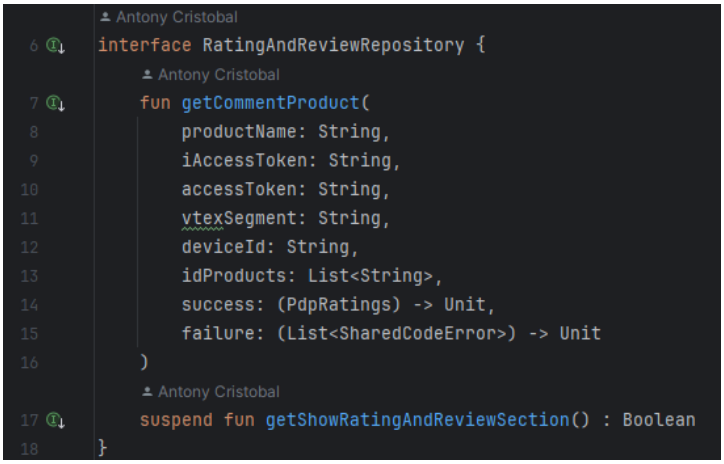


Figura 14. Principio SOLID Interface Segregation aplicado en el proyecto
Fuente: Grupo Éxito

- **D (*Dependency Inversion Principle*)**

Se emplearon interfaces para definir los contratos entre clases, esto se usó comúnmente en los casos de uso que se encargan de hacer consultas a los servicios, creando solo un vínculo de respuesta satisfactoria o errónea, obteniendo de esta forma un bajo acoplamiento entre clases.

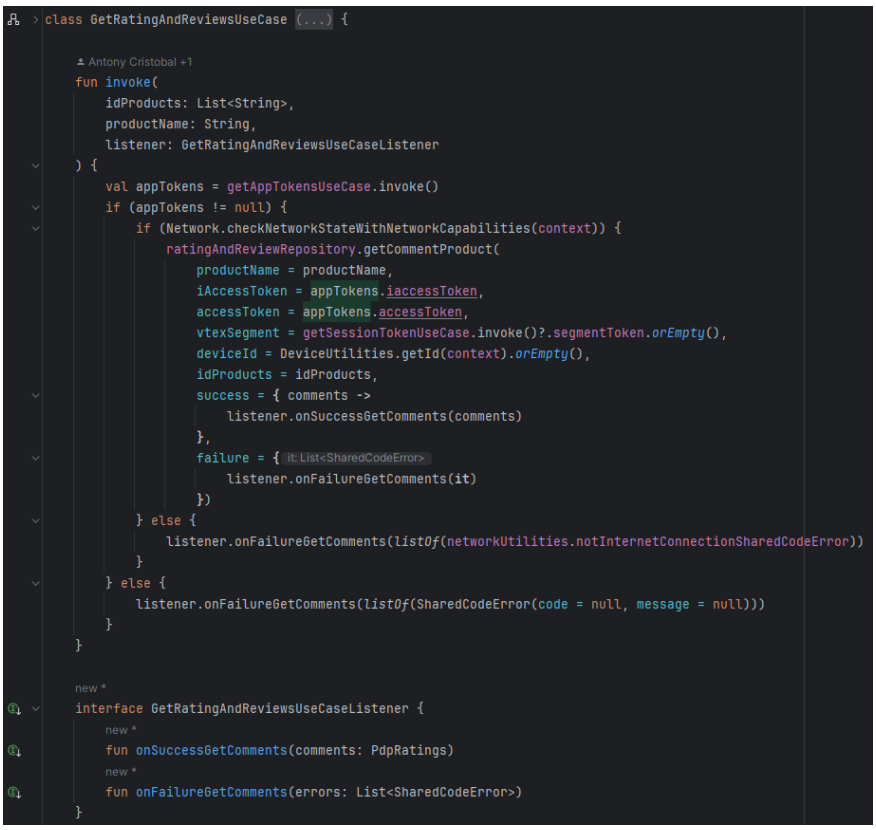


Figura 15. Principio SOLID Dependency Inversion Principle aplicado en el proyecto
Fuente: Grupo Éxito

4.2.4. Instrumentos

a. Android Studio

Este entorno de desarrollo se empleó en el proyecto para editar y compilar el código fuente del proyecto. Gracias a sus completas características, se pudo crear emuladores de diferentes dispositivos móviles en el mercado, con los cuales se pudo ejecutar el proyecto y verificar el correcto funcionamiento de la aplicación en un 97 % de dispositivos Android.

Actualmente, las aplicaciones móviles para Android y Huawei están siendo mantenidas con este entorno de desarrollo.

b. Kotlin

Emplear Kotlin significa tener un código fuente del proyecto legible, simplificado y optimizado, además de los beneficios mencionados anteriormente. Es por eso que este lenguaje fue empleado en el proyecto de la aplicación desde su creación.

Aunque existen librerías desarrolladas con el lenguaje Java, Kotlin puede acceder a las clases y métodos de estas sin inconvenientes.

Gracias a este lenguaje, se pudo encontrar documentación actualizada y una amplia comunidad de soporte para cuando se requería ayuda con algunos temas novedosos y problemas referentes al desarrollo.

c. XML

Parte del proyecto de la aplicación móvil fue desarrollada en XML, específicamente las interfaces de usuario. Si bien esta herramienta es funcional y cumple con el propósito de renderizar la parte gráfica de la aplicación, en el proyecto Wow se empleó Jetpack Compose para el desarrollo moderno y declarativo de estas interfaces.

d. Jetpack Compose

Esta herramienta fue empleada para el desarrollo declarativo de interfaces, haciendo posible la reutilización de código para renderizar la interfaz de usuario. Gracias a contar con una comunidad activa y documentación actualizada de esta herramienta, se logró implementar esta herramienta en las interfaces desarrolladas previamente con XML.

e. Git y GitFlow

Git es una herramienta que se empleó durante el desarrollo del proyecto para el control de versiones y el trabajo colaborativo de manera simultánea, haciendo posible desarrollar e

integrar diversas nuevas funcionalidades desarrolladas por diferentes equipos al mismo tiempo. Para el proyecto Wow se empleó un sistema de *branching* conocido como *GitFlow*, el cual se describirá más adelante.

f. Azure DevOps

Azure DevOps ofreció diferentes herramientas para desarrollar las nuevas funcionalidades de la aplicación, tener un control de versiones del proyecto implementando Git, gestionar el estado del proyecto, desarrollar las ceremonias *scrum* del proyecto y configurar los pipelines para la integración continua y la entrega continua, entre otros.

Adicionalmente, Azure DevOps fue de gran ayuda para registrar el historial de fallas o bugs reportados por el equipo de pruebas de calidad de *software*.

g. SonarLint

SonarLint es una herramienta que se emplea en el entorno de desarrollo Android Studio como *plugin*, permitiendo observar posibles errores de codificación, malas prácticas o malos hábitos al momento de escribir código antes de enviar el código fuente. Esta herramienta también permitió mejorar continuamente la escritura de código limpio, asegurando que se siga las mejores prácticas y se mantenga alta la calidad del código.

h. SonarQube

Esta herramienta en la nube, al igual que SonarLint, permitió escanear el código fuente del proyecto, encontrando y reportando «code smell» y el porcentaje de código cubierto con pruebas unitarias mediante paneles. SonarQube ayudó a ver desde una perspectiva administrativa el nivel de madurez del proyecto y la calidad con la que se estaban desarrollando las nuevas características. Gracias a esto, se pudieron tomar decisiones para incrementar el porcentaje mínimo de código cubierto en el proyecto, cumpliendo así con el propósito de entregar *software* de calidad desde el código fuente.

i. *Flavors*

Los *Flavors* fueron empleados en conjunto con Android Studio. Mediante estos, se logró configurar y separar los ambientes de desarrollo, el ambiente de pruebas de calidad (QA) y el ambiente de producción. Estas configuraciones permitieron definir ciertas variables globales del proyecto, proporcionando gran claridad y flexibilidad durante el desarrollo.

j. Maven Repository

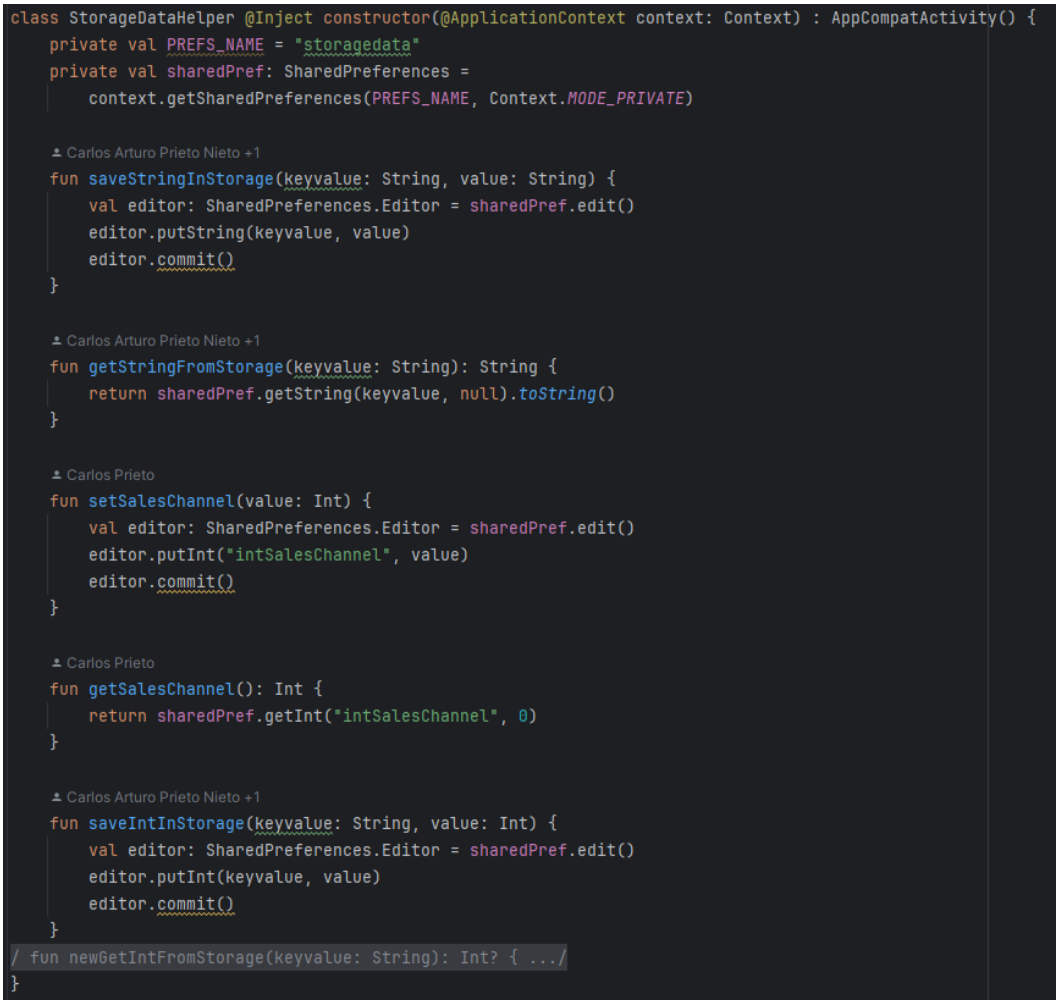
Esta herramienta de gestión de proyectos se empleó para la gestión de dependencias. El proyecto de la aplicación del Grupo Éxito consume los recursos de una dependencia llamada COCO. Esta dependencia es un proyecto desarrollado con Kotlin Multiplatform que sirve como intermediario con el *Backend*. En este proyecto, existen modelos y repositorios que contienen las consultas a los diferentes API brindadas por el equipo *Backend*.

k. Hilt

Es una librería de inyección de dependencias recomendada por Google que permitió gestionar y mantener de forma más sencilla las dependencias del proyecto, facilitando la escritura de código limpio y mantenible. Para el Proyecto Wow, se empleó una arquitectura limpia en la que se utilizó esta librería para inyectar dependencias de manera más sencilla.

l. POO (Programación orientada a objetos)

El proyecto de la aplicación del Grupo Éxito fue concebido bajo el paradigma de la programación orientada a objetos (POO) junto a sus cuatro pilares. Esto permitió crear y manipular objetos junto a sus métodos, desarrollando código reutilizable y escalable.



```
class StorageDataHelper @Inject constructor(@ApplicationContext context: Context) : AppCompatActivity() {
    private val PREFS_NAME = "storagedata"
    private val sharedPref: SharedPreferences =
        context.getSharedPreferences(PREFS_NAME, Context.MODE_PRIVATE)

    Carlos Arturo Prieto Nieto +1
    fun saveStringInStorage(keyvalue: String, value: String) {
        val editor: SharedPreferences.Editor = sharedPref.edit()
        editor.putString(keyvalue, value)
        editor.commit()
    }

    Carlos Arturo Prieto Nieto +1
    fun getStringFromStorage(keyvalue: String): String {
        return sharedPref.getString(keyvalue, null).toString()
    }

    Carlos Prieto
    fun setSalesChannel(value: Int) {
        val editor: SharedPreferences.Editor = sharedPref.edit()
        editor.putInt("intSalesChannel", value)
        editor.commit()
    }

    Carlos Prieto
    fun getSalesChannel(): Int {
        return sharedPref.getInt("intSalesChannel", 0)
    }

    Carlos Arturo Prieto Nieto +1
    fun saveIntInStorage(keyvalue: String, value: Int) {
        val editor: SharedPreferences.Editor = sharedPref.edit()
        editor.putInt(keyvalue, value)
        editor.commit()
    }
/ fun newGetIntFromStorage(keyvalue: String): Int? { .../
}
```

Figura 16. Programación orientada a objetos aplicada al proyecto
Fuente: Grupo Éxito

Como se puede apreciar, en este bloque de código se observa una clase la cual hereda de otra llamada AppCompatActivity(), cuenta con atributos privados y métodos públicos, así como, una inyección de dependencias empleando Hilt.

m. Diseño de interfaces

Para el desarrollo de la aplicación móvil, se crearon diseños de interfaces que representaban la estructura y el aspecto visual de la aplicación.

Estos diseños sirvieron como guía para el desarrollo y aseguraron que la aplicación cumpliera con los requisitos de usabilidad y estética.

Figura 17. Diseño de la interfaz de usuario para la historia de usuario 1.1 - parte 1
Fuente: Grupo Éxito

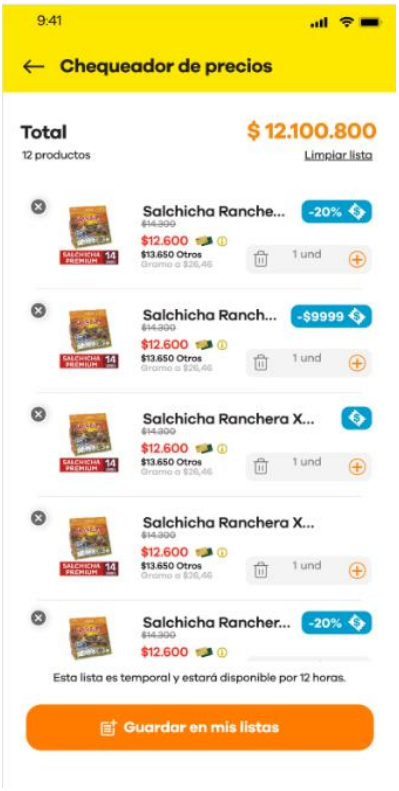


Figura 18. Diseño de la interfaz de usuario para la historia de usuario 1.1 - parte 2
Fuente: Grupo Éxito

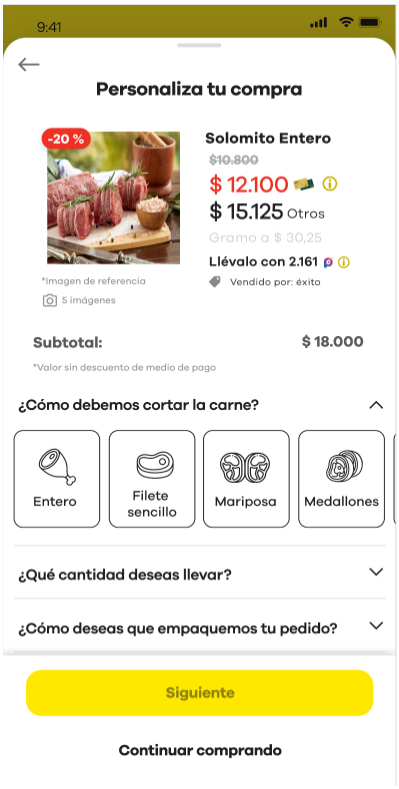


Figura 19. Diseño de la interfaz de usuario de la Experiencia Carnes - paso 1
Fuente: Grupo Éxito

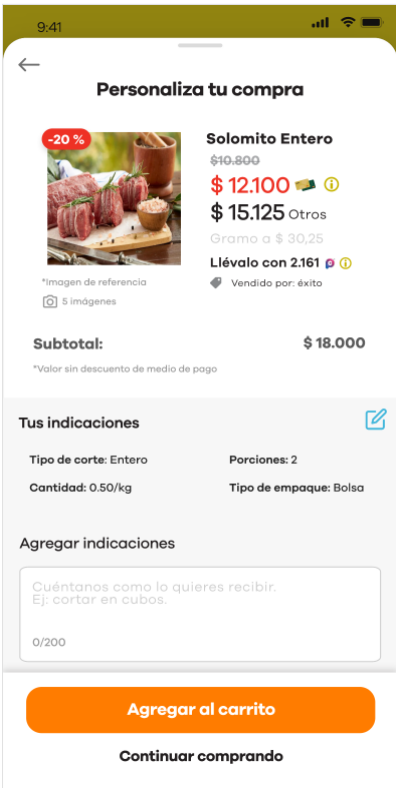


Figura 20. Diseño de la interfaz de usuario de la Experiencia Carnes - paso 2
Fuente: Grupo Éxito

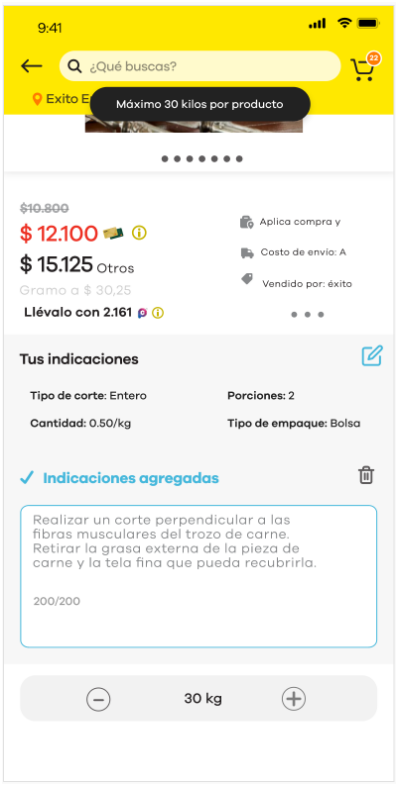


Figura 21. Diseño de la interfaz de usuario para la historia de usuario 3.1
Fuente: Grupo Éxito

Figura 22. Diseño de la interfaz de usuario para la historia de usuario 4.2
Fuente: Grupo Éxito

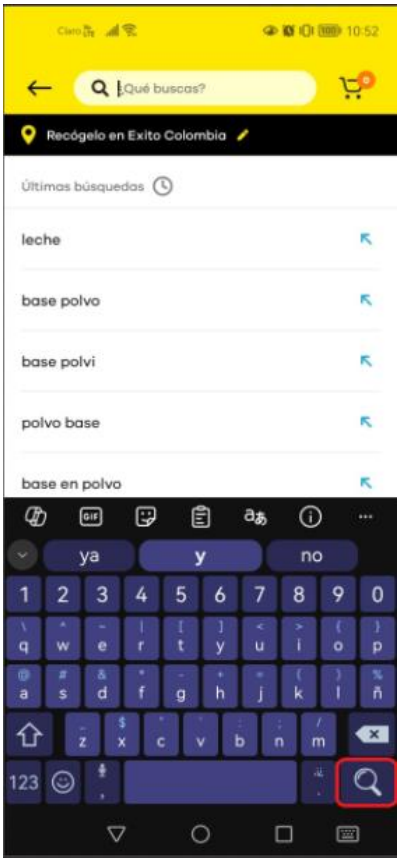


Figura 23. Diseño de interfaz de usuario para la historia de usuario 4.1
Fuente: Grupo Éxito

4.2.5. Equipos utilizados en el desarrollo de actividades

a. *Hardware*

- MacBook Pro M1
- Teléfono Huawei Nova 5T

b. *Software* adicional

- macOS
- Emuladores de Android

c. Materiales adicionales

- Documentación técnica
- Foros de desarrollo de *software*
- Cursos *online*

4.3. Ejecución de actividades profesionales

4.3.1. Cronograma de actividades realizadas

Siguiente página

Tabla 1. ***Cronograma de actividades profesionales desarrolladas***

Actividades profesionales desarrolladas en el 2023	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
Reunión *Kick Off*	X											
Desarrollo Historia de usuario 1.1 Chequeador en Mi Descuento	X											
Desarrollo Historia de usuario 2.1 Experiencia Carnes - PDP	X											
Desarrollo Historia de usuario 2.2 Experiencia Carnes. PLP		X										
Desarrollo Historia de usuario 3.1 Añadir notas a los productos frescos		X										
Desarrollo Historia de usuario 4.1 Añadir acción buscar en teclado de búsqueda			X									
Desarrollo Historia de usuario 4.2 Nuevo Look App			X									

Desarrollo Historia de usuario 5.1 Eliminar funcionalidad mi salud	X			
Desarrollo Historia de usuario 5.2 Eliminar funcionalidad familia	X			
Desarrollo Historia de usuario 5.3 Eliminar funcionalidades sucursales favoritas		X		
Desarrollo Historia de usuario 5.4 Actualizar librería lector código de barras		X		
Desarrollo Historia de usuario 5.5 Depurar code smell			X	
Desarrollo Historia de usuario 5.6 Incrementar cobertura de pruebas unitarias fase 1			X	
Desarrollo Historia de usuario 5.7 Incrementar cobertura de pruebas unitarias fase 2				X
Desarrollo Historia de usuario 5.8 Incrementar cobertura de pruebas unitarias fase 3				X
Primera salida a producción				X

Desarrollo Historia de usuario 5.9 Incrementar cobertura de pruebas unitarias fase 4	X				
Desarrollo Historia de usuario 5.10 Incrementar cobertura de pruebas unitarias fase 5	X				
Desarrollo Historia de usuario 6.1 Migrar servicio getProductDetail		X			
Desarrollo Historia de usuario 6.2 Migrar servicio searchProductGraphQL		X			
Desarrollo Historia de usuario 6.3 Migrar servicio recommendedProducts			X		
Desarrollo Historia de usuario 6.4 Migrar servicio getProductsDetail			X		
Desarrollo Historia de usuario 7.1 Modularización método entrega				X	
Desarrollo Historia de usuario 8.1 Métricas en PLP				X	
Desarrollo Historia de usuario 8.2 Métricas en PDP					X

Desarrollo Historia de usuario 8.3 Métricas en Checkout	X
Segunda salida para producción	X

Fuente: Grupo Éxito

4.3.2. Proceso y secuencia operativa de las actividades profesionales

A. Desarrollo de la historia de usuario 1.1 Chequeador en Mi Descuento

Tabla 2. ***Historia de usuario 1.1 Chequeador en Mi Descuento***

Historia de usuario		
1.1 Chequeador en Mi Descuento		
Como	usuario	
Quiero	poder escanear productos en el almacén	
Para	conocer su precio y activar la oferta	
Completado	**Tarea**	**Tiempo (horas)**
✓	Implementar interfaz para el acceso al servicio getProductsDetail	8
✓	Crear casos de uso	16
✓	Construir interfaz de usuario	24
✓	Crear lógica de interfaz de usuario	8
✓	Implementar pruebas unitarias	24
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	6
	Tiempo total (horas)	**96**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

B. Desarrollo de la historia de usuario 2.1 Experiencia Carnes – PDP

Tabla 3. ***Historia de usuario 2.1 Experiencia Carnes - PDP***

Historia de usuario		
2.1 Experiencia Carnes - PDP		
Como	usuario	
Necesito	pedir la carne con mis especificaciones	
Para	poder comprar como me gusta desde la App	
Completado	**Tarea**	**Tiempo (horas)**
✓	Implementar nuevos modelos de datos para obtener la experiencia carnes del servicio getProductDetail	8
✓	Crear casos de uso	16
✓	Construir interfaz de usuario modal paso 1	8
✓	Construir interfaz de usuario modal paso 2	16
✓	Crear lógica de interfaz de usuario en el PDP	8
✓	Implementar pruebas unitarias	24
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	6
	Tiempo total (horas)	**96**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

C. Desarrollo de la historia de usuario 2.2 Experiencia Carnes. PLP

Tabla 4. ***Historia de usuario 2.2 Experiencia Carnes. PLP***

Historia de usuario		
2.2 Experiencia Carnes. PLP		
Como	usuario	
Necesito	pedir la carne con mis especificaciones	
Para	poder comprar como me gusta desde la App	
Completado	**Tarea**	**Tiempo (horas)**
✓	Implementar nuevos modelos de datos para obtener la experiencia carnes del servicio getProductDetail	8
✓	Crear casos de uso	16
✓	Construir interfaz de usuario modal paso 1	8
✓	Construir interfaz de usuario modal paso 2	16
✓	Crear lógica de interfaz de usuario en el PLP	8
✓	Implementar pruebas unitarias	24
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	6
	Tiempo total (horas)	**96**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

D. Desarrollo de la historia de usuario 3.1 Añadir notas a los productos frescos – Sincronización

Tabla 5. ***Historia de usuario 3.1 Añadir notas a los productos frescos Sincronización***

Historia de usuario		
3.1 Añadir notas a los productos frescos - Sincronización		
Como	usuario	
Necesito	poder añadir notas a mis productos	
Para	poder comprar como deseo	
Completado	**Tarea**	**Tiempo (horas)**
✓	Crear acceso a la cache del dispositivo empleando shared preferences	8
✓	Crear model de datos para almacenar los comentarios	6
✓	Crear casos de uso para almacenar, obtener y sincronizar los comentarios	32
✓	Desarrollar interfaces de la caja de comentarios empleando Jetpack Compose	16
✓	Desarrollar lógica para la interfaz de la caja de comentarios	8
✓	Agregar el envío de comentarios al servicio del checkout	8
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**96**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

E. Desarrollo de la historia de usuario 4.1 Añadir acción buscar en teclado de búsqueda

Tabla 6. ***Historia de usuario 4.1 Añadir acción buscar en teclado de búsqueda***

Historia de usuario		
4.1 Añadir acción buscar en teclado de búsqueda		
Como	usuario	
Necesito	tener un accionamiento claro en el teclado a la hora de buscar	
Para	que la búsqueda sea más intuitiva	
Completado	**Tarea**	**Tiempo (horas)**
✓	Investigar funcionamiento del teclado y limitantes en diferentes dispositivos Android	8
✓	Ajustar *keyboard* en la interfaz principal	8
✓	Ajustar *keyboard* en la interfaz landing	8
✓	Ajustar *keyboard* en la interfaz PLP	8
✓	Ajustar *keyboard* en la interfaz PDP	8
✓	Ajustar *keyboard* en la interfaz pasillos	8
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**66**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

F. Desarrollo de la historia de usuario 4.2 Nuevo Look App

Tabla 7. ***Historia de usuario 4.2 Nuevo Look App***

Historia de usuario		
4.2 Nuevo Look App		
Como	usuario	
Necesito	ver un nuevo diseño en la App	
Para	tener una experiencia diferente y novedosa	
Completado	**Tarea**	**Tiempo (horas)**
✓	Quitar bordes de los *items* del listado atajos de la vista principal	16
✓	Modificar color del *header* en la interfaz vista principal	8
✓	Modificar color del *header* en la interfaz landing	8
✓	Modificar color del *header* en la interfaz PLP	8
✓	Modificar color del *header* en la interfaz PDP	8
✓	Modificar color del *header* en la interfaz pasillos	8
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**74**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

G. Desarrollo de la historia de usuario 5.1 Eliminar funcionalidad mi salud

Tabla 8. ***Historia de usuario 5.1 Eliminar funcionalidad mi salud***

Historia de usuario		
5.1 Eliminar funcionalidad mi salud		
Como	usuario	
Necesito	tener una App ligera, sin funcionalidades obsoletas	
Para	ocupar menor espacio de almacenamiento	
Completado	**Tarea**	**Tiempo (horas)**
✓	Revisar funcionalidad mi salud	8
✓	Eliminar entidades y modelos empleados por la funcionalidad	8
✓	Eliminar interfaces de usuario ligadas a la funcionalidad	8
✓	Eliminar casos de uso	8
✓	Eliminar accesos de datos empleados por la funcionalidad	8
✓	Validar que no se haya alterado la funcionalidad de la App	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**74**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

H. Desarrollo de la historia de usuario 5.2 Eliminar funcionalidad familia

Tabla 9. ***Historia de usuario 5.2 Eliminar funcionalidad familia***

Historia de usuario		
5.2 Eliminar funcionalidad familia		
Como	usuario	
Necesito	tener una App ligera, sin funcionalidades obsoletas	
Para	ocupar menor espacio de almacenamiento	
Completado	**Tarea**	**Tiempo (horas)**
✓	Revisar funcionalidad familia	8
✓	Eliminar entidades y modelos empleados por la funcionalidad	8
✓	Eliminar interfaces de usuario ligadas a la funcionalidad	8
✓	Eliminar casos de uso	8
✓	Eliminar accesos de datos empleados por la funcionalidad	8
✓	Validar que no se haya alterado la funcionalidad de la App	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**74**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

I. Desarrollo de la historia de usuario 5.3 Eliminar funcionalidad de sucursales favoritas

Tabla 10. ***Historia de usuario 5.3 Eliminar funcionalidad de sucursales favoritas***

Historia de usuario		
5.3 Eliminar funcionalidad de sucursales favoritas		
Como	usuario	
Necesito	tener una App ligera, sin funcionalidades obsoletas	
Para	ocupar menor espacio de almacenamiento	
Completado	**Tarea**	**Tiempo (horas)**
✓	Revisar funcionalidad de sucursales favoritas	8
✓	Eliminar entidades y modelos empleados por la funcionalidad	8
✓	Eliminar interfaces de usuario ligadas a la funcionalidad	8
✓	Eliminar casos de uso	8
✓	Eliminar accesos de datos empleados por la funcionalidad	8
✓	Validar que no se haya alterado la funcionalidad de la App	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**74**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

J. Desarrollo de la historia de usuario 5.4 Actualizar librería de lector código de barras

Tabla 11. ***Historias de usuario 5.4 Actualizar librería de lector código de barras***

Historia de usuario		
5.4 Actualizar librería de lector código de barras		
Como	usuario	
Necesito	tener una App ligera, sin funcionalidades obsoletas	
Para	ocupar menor espacio de almacenamiento	
Completado	**Tarea**	**Tiempo (horas)**
✓	Eliminar dependencia anterior de lector de código de barras	2
✓	Importar dependencia nueva del lector de código de barras	6
✓	Eliminar clases y métodos provenientes de anterior dependencia	8
✓	Emplear las nuevas clases y métodos de la nueva dependencia	8
✓	Crear interfaces para abstraer y segregar el control de la cámara del dispositivo	8
✓	Validar que no se haya alterado las funcionalidades que emplearon la anterior librería	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**66**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

K. Desarrollo de la historia de usuario 5.5 Depurar code smell

Tabla 12. ***Historia de usuario 5.5 Depurar code smell***

Historia de usuario		
5.5 Depurar code smell		
Como	usuario	
Necesito	tener una App ligera, sin funcionalidades obsoletas	
Para	ocupar menor espacio de almacenamiento	
Completado	**Tarea**	**Tiempo (horas)**
✓	Eliminar dependencias innecesarias en todo el proyecto	8
✓	Eliminar importaciones no usadas en todo el proyecto	8
✓	Eliminar código no usado en todo el proyecto	8
✓	Refactorizar clases con alta complejidad ciclomática en todo el proyecto	8
✓	Refactorizar clases con alta complejidad anidada en todo el proyecto	8
✓	Emplear abstracciones y principios SOLID en todo el proyecto	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**74**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

L. Desarrollo de la historia de usuario 5.6 Incrementar cobertura de pruebas unitarias fase 1

Tabla 13. ***Historia de usuario 5.6 Incrementar cobertura de pruebas unitarias fase 1***

Historia de usuario		
5.6 Incrementar cobertura de pruebas unitarias fase 1		
Como	usuario	
Necesito	tener una App de calidad	
Para	evitar errores durante el uso	
Completado	**Tarea**	**Tiempo (horas)**
✓	Incrementar pruebas unitarias en la capa de ViewModel del módulo PDP	16
✓	Incrementar pruebas unitarias en la capa de modelos del módulo PDP	16
✓	Incrementar pruebas unitarias en la capa de casos de uso del módulo PDP	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**66**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

M. Desarrollo de la historia de usuario 5.7 Incrementar cobertura de pruebas unitarias fase 2

Tabla 14. ***Historia de usuario 5.7 Incrementar cobertura de pruebas unitarias fase 2***

Historia de usuario		
5.7 Incrementar cobertura de pruebas unitarias fase 2		
Como	usuario	
Necesito	tener una App de calidad	
Para	evitar errores durante el uso	
Completado	**Tarea**	**Tiempo (horas)**
✓	Incrementar pruebas unitarias en la capa de ViewModel del módulo PLP	16
✓	Incrementar pruebas unitarias en la capa de modelos del módulo PLP	16
✓	Incrementar pruebas unitarias en la capa de casos de uso del módulo PLP	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**66**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

N. Desarrollo de la historia de usuario 5.8 Incrementar cobertura de pruebas unitarias fase 3

Tabla 15. ***Historia de usuario 5.8 Incrementar cobertura de pruebas unitarias fase 3***

Historia de usuario		
5.8 Incrementar cobertura de pruebas unitarias fase 3		
Como	usuario	
Necesito	tener una App de calidad	
Para	evitar errores durante el uso	
Completado	**Tarea**	**Tiempo (horas)**
✓	Incrementar pruebas unitarias en la capa de ViewModel del módulo Landing	16
✓	Incrementar pruebas unitarias en la capa de modelos del módulo Landing	16
✓	Incrementar pruebas unitarias en la capa de casos de uso del módulo Landing	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**66**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

O. Desarrollo de la historia de usuario 5.9 Incrementar cobertura de pruebas unitarias fase 4

Tabla 16. ***Historia de usuario 5.9 Incrementar cobertura de pruebas unitarias fase 4***

Historia de usuario		
5.9 Incrementar cobertura de pruebas unitarias fase 4		
Como	usuario	
Necesito	tener una App de calidad	
Para	evitar errores durante el uso	
Completado	**Tarea**	**Tiempo (horas)**
✓	Incrementar pruebas unitarias en la capa de ViewModel del módulo Pasillos	16
✓	Incrementar pruebas unitarias en la capa de modelos del módulo Pasillos	16
✓	Incrementar pruebas unitarias en la capa de casos de uso del módulo Pasillos	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**66**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

P. Desarrollo de la historia de usuario 5.10 Incrementar cobertura de pruebas unitarias fase 5

Tabla 17. ***Historia de usuario 5.10 Incrementar cobertura de pruebas unitarias fase 5***

Historia de usuario		
5.10 Incrementar cobertura de pruebas unitarias fase 5		
Como	usuario	
Necesito	tener una App de calidad	
Para	evitar errores durante el uso	
Completado	**Tarea**	**Tiempo (horas)**
✓	Incrementar pruebas unitarias en la capa de ViewModel del módulo recomendados para ti	16
✓	Incrementar pruebas unitarias en la capa de modelos del módulo recomendados para ti	16
✓	Incrementar pruebas unitarias en la capa de casos de uso del módulo recomendados para ti	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**66**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

Q. Desarrollo de la historia de usuario 6.1 Migrar servicio getProductDetail

Tabla 18. ***Historia de usuario 6.1 Migrar servicio getProductDetail***

Historia de usuario		
6.1 Migrar servicio getProductDetail		
Como	usuario	
Necesito	consumir el nuevo servicio	
Para	observar productos	
Completado	**Tarea**	**Tiempo (horas)**
✓	Crear nuevos modelos para mapear los datos del nuevo servicio	8
✓	Crear acceso para la conexión a la nueva API	8
✓	Crear caso de uso	8
✓	Emplear el nuevo caso de uso en la vista PDP	8
✓	Refactorizar código	8
✓	Implementar pruebas unitarias	8
✓	Verificar correcto funcionamiento de la vista PDP	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**82**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

R. Desarrollo de la historia de usuario 6.2 Migrar servicio searchProductGraphQL

Tabla 19. ***Historia de usuario 6.2 Migrar servicio searchProductGraphQL***

Historia de usuario		
6.2 Migrar servicio searchProductGraphQL		
Como	usuario	
Necesito	consumir el nuevo servicio	
Para	buscar productos	
Completado	**Tarea**	**Tiempo (horas)**
✓	Crear nuevos modelos para mapear los datos del nuevo servicio	8
✓	Crear acceso para la conexión a la nueva API	8
✓	Crear caso de uso	8
✓	Emplear el nuevo caso de uso en las vistas PLP y buscador	8
✓	Refactorizar código	8
✓	Implementar pruebas unitarias	8
✓	Verificar correcto funcionamiento de las vistas PLP y buscador	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**82**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

S. Desarrollo de la historia de usuario 6.3 Migrar servicio recommendedProducts

Tabla 20. ***Historia de usuario 6.3 Migrar servicio recommendedProducts***

Historia de usuario		
6.3 Migrar servicio recommendedProducts		
Como	usuario	
Necesito	consumir el nuevo servicio	
Para	observar productos recomendados	
Completado	**Tarea**	**Tiempo (horas)**
✓	Crear nuevos modelos para mapear los datos del nuevo servicio	8
✓	Crear acceso para la conexión a la nueva API	8
✓	Crear caso de uso	8
✓	Emplear el nuevo caso de uso en las secciones recomendados para ti	8
✓	Refactorizar código	8
✓	Implementar pruebas unitarias	8
✓	Verificar correcto funcionamiento de las secciones recomendados para ti	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**82**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

T. Desarrollo de la historia de usuario 6.4 Migrar servicio getProductsDetail

Tabla 21. ***Historia de usuario 6.4 Migrar servicio getProductsDetail***

Historia de usuario		
6.4 Migrar servicio getProductsDetail		
Como	usuario	
Necesito	consumir el nuevo servicio	
Para	escanear productos	
Completado	**Tarea**	**Tiempo (horas)**
✓	Crear nuevos modelos para mapear los datos del nuevo servicio	8
✓	Crear acceso para la conexión a la nueva API	8
✓	Crear caso de uso	8
✓	Emplear el nuevo caso de uso en la vista del chequeador de precios	8
✓	Refactorizar código	8
✓	Implementar pruebas unitarias	8
✓	Verificar correcto funcionamiento de vista chequeador de precios	16
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**82**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

U. Desarrollo de la historia de usuario 7.1 Modularización método entrega

Tabla 22. ***Historia de usuario 7.1 Modularización método entrega***

Historia de usuario		
7.1 Modularización método entrega		
Como	usuario	
Necesito	tener una App de mayor calidad	
Para	optimizar el uso de recursos	
Completado	**Tarea**	**Tiempo (horas)**
✓	Crear y configurar un nuevo proyecto llamado deliveryMethod	4
✓	Crear interfaz de usuario del módulo método de entrega empleando Jetpack Compose	24
✓	Implementar una arquitectura MVI	16
✓	Crear conexiones a la API selectDeliveyMethod	4
✓	Crear casos de uso	16
✓	Implementar el proyecto nuevo como dependencia en el proyecto de la aplicación	4
✓	Implementar pruebas unitarias	8
✓	Desplegar al área de QA	8
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**94**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

V. Desarrollo de la historia de usuario 8.1 Métricas en PLP

Tabla 23. ***Historia de usuario 8.1 Métricas en PLP***

Historia de usuario		
8.1 Métricas en PLP		
Como	negocio	
Necesito	tener métricas de las interacciones del usuario en la App	
Para	tomar decisiones de negocio	
Completado	**Tarea**	**Tiempo (horas)**
✓	Implementar los servicios de Google y Firebase en el proyecto para cada ambiente	6
✓	Crear clase abstracta para enviar la analítica para el módulo PLP	6
✓	Configurar todos los eventos de acuerdo a la guía de métricas	16
✓	Enviar analítica en cada interacción de los elementos en el módulo PLP según la guía de métricas	24
✓	Implementar pruebas unitarias	16
✓	Verificar el envío de analítica en la herramienta de debug view en Firebase	16
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**94**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

W. Desarrollo de la historia de usuario 8.2 Métricas en PDP

Tabla 24. ***Historia de usuario 8.2 Métricas en PDP***

Historia de usuario		
8.2 Métricas en PDP		
Como	negocio	
Necesito	tener métricas de las interacciones del usuario en la App	
Para	tomar decisiones de negocio	
Completado	**Tarea**	**Tiempo (horas)**
✓	Crear clase abstracta para enviar la analítica para el módulo PDP	6
✓	Configurar todos los eventos de acuerdo a la guía de métricas	16
✓	Enviar analítica en cada interacción de los elementos en el módulo PDP según la guía de métricas	24
✓	Implementar pruebas unitarias	16
✓	Verificar el envío de analítica en la herramienta de debug view en Firebase	16
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**88**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

X. Desarrollo de la historia de usuario 8.3 Métricas en Checkout

Tabla 25. ***Historia de usuario 8.3 Métricas en Checkout***

Historia de usuario		
8.3 Métricas en Checkout		
Como	negocio	
Necesito	tener métricas de las interacciones del usuario en la App	
Para	tomar decisiones de negocio	
Completado	**Tarea**	**Tiempo (horas)**
✓	Crear clase abstracta para enviar la analítica para el módulo del Checkout	6
✓	Configurar todos los eventos de acuerdo a la guía de métricas	16
✓	Enviar analítica en cada interacción de los elementos en el módulo del Checkout según la guía de métricas	24
✓	Implementar pruebas unitarias	16
✓	Verificar el envío de analítica en la herramienta de debug view en Firebase	16
✓	Crear DOD	1
✓	Grabar evidencia de solución	1
✓	Solucionar bugs	8
	Tiempo total (horas)	**88**
	Puntos de esfuerzo	**8**
	Prioridad	**Alta**

CAPÍTULO V
RESULTADOS

5.1. Resultados finales de las actividades realizadas

Las actividades como desarrollador de aplicaciones móviles se desarrollaron de manera conforme, respetando arquitecturas definidas por la organización, cumpliendo los estándares de calidad, y adaptándome completamente al proyecto, llegando a conocer cada una de las funcionalidades vigentes hasta la fecha.

En cuanto a los objetivos logrados, se expone lo siguiente:

Tabla 26. ***Objetivos logrados***

Objetivo	Indicador	Porcentaje de cumplimiento
Desarrollar e integrar en su totalidad los requerimientos de la organización.	Azure Boards	100 %
Aumentar la cobertura de pruebas unitarias del código fuente de la aplicación móvil.	Sonar Qube	90 %
Reducir la cantidad de reportes de «Code Smell» en Sonar Qube.	Sonar Qube	90 %
Mantener la arquitectura existente en el proyecto, respetando los principios SOLID y aplicando buenas prácticas en la escritura de código.	Aprobación de Pull Request	100 %
Reducir la deuda técnica existente en el proyecto de la aplicación Éxito.	Azure Boards	100 %

Solucionar errores y fallas reportadas por el equipo de calidad de manera oportuna.	Azure Boards	100 %
Publicar una versión estable de la aplicación en las tiendas de Play Store y App Store, con las nuevas funcionalidades desarrolladas.	Play Store	100 %

5.2. Logros alcanzados

Mi experiencia de trabajar en proyectos de este calibre, me permitió lograr lo siguiente:

- Desarrollar con más potencia mis habilidades técnicas gracias a los nuevos retos y oportunidades de mejora que de manera necesaria tuve que afrontar.

- Escalar un nivel completo de *seniority*, es decir pasé de estar en una categoría de desarrollador *advance* a desarrollador *senior*.

- Mejorar mis habilidades blandas como la comunicación, el liderazgo y las capacidades de colaborar activamente en equipos de alto rendimiento, independientemente de la cultura o nacionalidad que poseo, siendo una experiencia única e inolvidable para mi persona.

- Recibir buenas críticas por parte de otros equipos dentro del Grupo Éxito, convirtiéndome en una referencia. Esto me ha permitido mantenerme dentro del equipo de las Apps del Grupo Éxito y tomar otros proyectos dentro de la organización.

- Entregar una aplicación mucho más robusta, mantenible, escalable y de calidad, evidenciando el correcto empleo de buenas prácticas y una adecuada arquitectura.

- Disminuir la deuda técnica del proyecto que se acumuló con el tiempo.

- Aumentar la cobertura de pruebas unitarias que daban un mal aspecto en cuanto a calidad de *software* de la aplicación.

- Salir a producción de manera exitosa, brindando una app operativa, funcional, libre de fallas, estable y robusta.

5.3. Dificultades

- Al inicio de cada *sprint*, la entrega inoportuna de diseños representaba un impedimento en ciertas etapas, dificultando la continuación del desarrollo de las historias de usuario y ocasionando, en ocasiones, demoras en el despliegue a QA.

- Al realizar las pruebas durante el desarrollo, la falta de configuración de los datos de productos presentaba de igual forma un impedimento para culminar la historia de usuario.

- En la ejecución del pipeline para realizar los despliegues a QA o Producción, los tiempos de compilación y ejecución eran considerablemente largos, llegando a durar hasta una hora y treinta minutos debido a la gran envergadura del proyecto. Esto representó un riesgo significativo para cumplir con el cierre de las historias de usuario.

- En ocasiones, el equipo de *Backend* desplegó ajustes que provocaron caídas en los servicios del ambiente de desarrollo, generando impedimentos al momento de realizar pruebas dentro de la aplicación.

5.4. Planteamiento de mejoras

5.4.1. Metodologías propuestas

- **Kanban**

Se usa para gestionar y optimizar el flujo de trabajo en proyectos, facilitando la visualización de todas las tareas y su estado en un tablero, identificando cuellos de botella y optimizando el flujo de trabajo. También permite adaptarse rápidamente a cambios en prioridades y condiciones. Además, aumenta la transparencia y la comunicación dentro del equipo y promueve la entrega continua de valor mediante la gestión eficiente del trabajo en progreso.

- **Monorepo**

Como estrategia de Git ayuda a consolidar todos los proyectos en un único repositorio, lo que facilita la gestión y el seguimiento del código. Además, optimiza la integración continua y los despliegues, aumentando la eficiencia y la transparencia en el desarrollo de *software*.

- **Modularización**

Al dividir la aplicación en módulos independientes, se mejora la organización del código, facilitando la reutilización de componentes y promoviendo el desarrollo en paralelo por diferentes equipos. Esto reduce el tiempo de compilación, ya que solo los módulos modificados

necesitan ser recompilados. Además, mejora la capacidad de pruebas, permitiendo enfoques más granulares y específicos para cada módulo. También contribuye a la escalabilidad y mantenibilidad del proyecto, facilitando la identificación y resolución de problemas, y finalmente permite una mayor flexibilidad en la implementación de nuevas funcionalidades sin afectar otras partes del sistema.

- **Framework de desarrollo Flutter**

Flutter es un framework de desarrollo que permite crear aplicaciones multiplataforma, es decir con un solo código fuente, se puede desarrollar aplicaciones móviles para Android y iOS, e incluso para aplicaciones web. De esta manera reducimos costos a la organización, y obtendremos una aplicación homologada en diseño y estructura interna para cualquier sistema operativo.

5.4.2. Descripción de la implementación

- **Kanban**

Este método de gestión de proyectos se podría emplear para trabajar los insumos para el desarrollo de nuevas funcionalidades con el equipo de diseño, de manera que los recursos se entreguen de forma más ágil y rápida. Esto no reemplazaría el uso de *Scrum*, sino que, por el contrario, se emplearían ambas metodologías en conjunto, logrando así unificar ambas en la popular metodología *Scrumban*, la cual es utilizada por muchos proyectos de gran envergadura.

- **Monorepo**

De acuerdo al gran problema actual que se vive en el mantenimiento del repositorio remoto en Azure Repository, emplear Monorepo como estrategia de ramificación sería la mejor opción para los proyectos futuros, de esta manera evitamos crear ramas para cada *sprint*, los cuales se pueden resumir a una única llamada *Trunk*, el cual contendrá una versión final tanto para desplegar a QA o a Producción.

- **Modularización**

Dado que el código fuente de la aplicación cuenta con una gran cantidad de funcionalidades, se recomienda comenzar a segregar los módulos de la aplicación en proyectos internos con compiladores independientes. Esto ayudará a evitar largas esperas para compilar el proyecto y reducirá el tiempo de ejecución de los pipelines para despliegues. Para implementar esta estrategia, se debe separar la aplicación por módulos y crear nuevos subproyectos dentro de la aplicación, configurando su propio entorno de compilación. Luego, se migrará todo el código correspondiente a cada módulo y se crearán pruebas unitarias

específicas para esos módulos. De esta manera, cualquier nuevo desarrollo o mantenimiento en un módulo particular solo afectará a ese subproyecto.

- **Flutter**

Este framework, muy popular en la actualidad, es una excelente opción para reducir costos de desarrollo dentro de la organización. Migrar a Flutter implicaría una reingeniería completa, comenzando desde cero para replicar la aplicación utilizando el nuevo lenguaje Dart y las prácticas recomendadas por el framework. Este proceso incluiría la creación de un nuevo proyecto desde cero, adoptando la programación de interfaces declarativas, desarrollando recursos y configurando los entornos de ambiente. Además, se establecería un único pipeline y se configuraría un proceso de integración continua, posiblemente empleando la estrategia de branching Monorepo. La implementación se beneficiaría de una comunidad activa y abundante documentación, facilitando la integración de todas las características actualmente presentes en la aplicación desplegada en la tienda Play Store.

5.5. Análisis

En el análisis de este trabajo, se observó que la organización cuenta con diferentes equipos especializados en actividades específicas. Sin embargo, la creciente complejidad del código fuente de la aplicación, a medida que se añaden nuevas funcionalidades, ha llevado a la obsolescencia de muchas otras. Esto ha generado la necesidad de eliminar el código deprecado para mantener un código limpio y optimizado. A pesar de esto, en las actividades definidas en los *Sprint*s, no se considera esta tarea de mantenimiento, dejándola no priorizada.

Además, se ha notado que los proyectos actuales de desarrollo de *software* requieren cada vez más conocimientos técnicos, lo que demanda capacitaciones continuas. Invertir en cursos o charlas es fundamental para proporcionar al negocio soluciones más eficientes y optimizadas.

Finalmente, se observó que, a mayor complejidad del proyecto, mayor es la propensión a incurrir en malas prácticas de desarrollo. Es crucial mantener una conciencia ética y exigirnos revisar y refactorizar el código entregado para evitar la acumulación de deuda técnica en el futuro.

5.6. Aportes del bachiller en la organización

- **Incremento en Ventas**
  - Durante el Evento Wow de 2023, el Grupo Éxito superó el objetivo de duplicar las ventas con un incremento del 136 % respecto al año anterior, tal como se evidencia en el anexo 9.

  - La aplicación móvil experimentó un crecimiento del 176 % en usuarios activos, reflejando un notable éxito en la atracción y retención de clientes, tal como se evidencia en el anexo 9.

- **Mejora en la calidad del código**
  - Se incrementó la cobertura de código en un 90 %, tal como se detalla en el Anexo 8, asegurando una mayor robustez y facilidad de mantenimiento para las nuevas funcionalidades.

  - Se redujo la cantidad de reportes de «Code Smell» en SonarQube de 477 a 48, lo que representa una disminución de casi el 90 %, como se evidencia en el Anexo 8. Esto contribuye a un código más limpio y menos propenso a errores.

- **Reducción de deuda técnica**
  - Se logró una reducción del 100 % en la deuda técnica del proyecto, gracias a la refactorización del código, la adopción de buenas prácticas de desarrollo y la integración de nuevas tecnologías.

- **Innovaciones y nuevas ideas**
  - Se introdujeron ideas y soluciones innovadoras, incluyendo el uso de Jetpack Compose para la creación de interfaces de usuario. Esto permitió una programación declarativa moderna y eficiente, mejorando la experiencia de desarrollo y la calidad de la interfaz de usuario.

- **Resultados adicionales**
  - Se solucionaron errores y fallas reportadas oportunamente, culminando el proyecto con 0 fallas reportadas por el equipo de calidad.

  - La versión final de la aplicación, con las nuevas funcionalidades desarrolladas, fue publicada exitosamente en las tiendas Play Store y App Store, tal como se evidencia en el anexo 7.

Estos logros reflejan el impacto positivo del proyecto y demuestran el valor añadido a la organización, superando las expectativas y dejando una huella significativa en el Grupo Éxito.

CONCLUSIONES

Al finalizar el Proyecto Wow, se puede concluir que las tecnologías, metodologías y herramientas empleadas fueron las adecuadas y resultaron ser de gran ventaja para desarrollar el *backlog* definido. La implementación de prácticas como TDD (desarrollo guiado por pruebas) contribuyó significativamente a la mejora de las métricas en *SonarQube*, reflejando una madurez del proyecto y generando una perspectiva de entrega de valor más sólida para el Grupo Éxito.

El código entregado no solo cumple con las directrices establecidas por la arquitectura propuesta, sino que también ha demostrado una alta calidad y optimización. La utilización de herramientas modernas y enfoques innovadores, como *Jetpack Compose* para las interfaces de usuario, ha aportado a la creación de una experiencia de usuario más fluida y eficiente.

Además, el *software* entregado ha tenido un impacto significativo en los objetivos de negocio del Grupo Éxito. La organización no solo alcanzó el objetivo de incrementar las ventas en un 100 % respecto al año anterior, sino que superó esta meta con un aumento del 136 %. Asimismo, se logró un incremento del 176 % en usuarios activos, tal como se detalla en el anexo 10. Estos resultados subrayan la importancia de entregar un *software* de alta calidad y valor, que permite al cliente seguir escalando y obteniendo resultados sobresalientes.

En resumen, el Proyecto Wow ha demostrado que una combinación adecuada de tecnologías, prácticas de desarrollo y metodologías ágiles puede llevar a un éxito notable, contribuyendo significativamente a los objetivos estratégicos del Grupo Éxito.

RECOMENDACIONES

Para aquellos colegas que están comenzando a desarrollar proyectos de *software* similares, es crucial tomar conciencia sobre la importancia de entregar código de calidad. Adoptar buenas prácticas y seguir lineamientos de desarrollo, junto con la implementación de un patrón de arquitectura adecuado, facilitará la escalabilidad y el mantenimiento del proyecto a lo largo del tiempo.

Además, es fundamental desarrollar y poner en práctica habilidades blandas para manejar cualquier novedad o desafío que pueda surgir durante el ciclo del proyecto. Mantener una actitud abierta al diálogo y demostrar profesionalismo que en todo momento contribuirá con un ambiente de trabajo más efectivo y colaborativo.

Se recomienda el uso de un marco de desarrollo ágil como *Scrum*, que facilita la gestión del proyecto, permitiendo al equipo adaptarse de manera más flexible a los cambios inesperados y mejorar la eficiencia en la ejecución del proyecto.

Por último, se aconseja utilizar *Git* como sistema de control de versiones, acompañado de una estrategia de *branching* adecuada a las necesidades del equipo. Esta herramienta no solo facilita el trabajo colaborativo y la integración de nuevas características desarrolladas simultáneamente, sino que también es ampliamente empleada en proyectos de gran envergadura, siendo una opción flexible y eficaz en diferentes repositorios remotos populares.

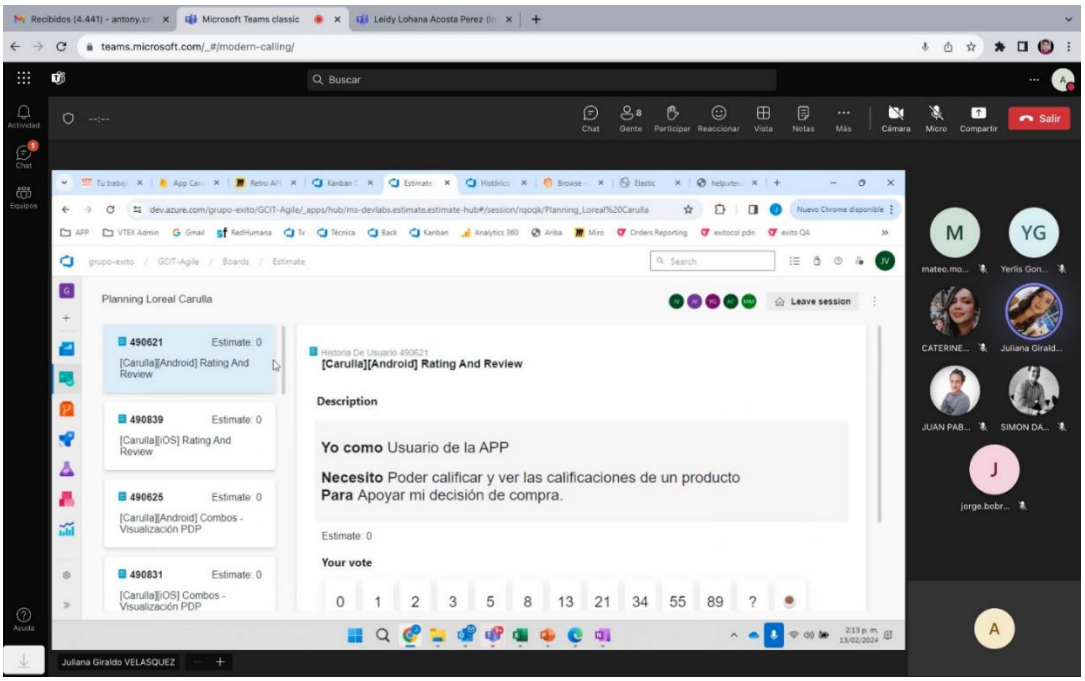
REFERENCIAS

1. **ARREOLA**. *El retail ha muerto, ¡que viva el retail!* 2017 [fecha de la consulta: 28 de febrero de 2024]. Disponible en: https://www.forbes.com.mx/el-*retail*-ha-muerto-que-viva-el-*retail*/
2. **Hotmart**. *¿Cuáles son los canales de venta más eficientes hoy en día?* 2022 [fecha de la consulta: 28 de febrero de 2024]. Disponible en: https://hotmart.com/es/blog/canales-de-venta
3. **RODRÍGUEZ**. *Qué son las apps nativas: todo lo que necesitas saber*. 2020 [fecha de la consulta: 28 de febrero de 2024]. Disponible en: https://vanadis.es/que-son-las-apps-nativas-todo-lo-que-necesitas-saber/
4. **Alegsa**. *Definición de sistema operativo móvil*. 2023 [fecha de la consulta: 3 de marzo de 2024]. Disponible en: https://www.alegsa.com.ar/Dic/sistema_operativo_movil.php#h0
5. **INDEED**. *Qué es front end: todo lo que debes saber.* 2023 [fecha de la consulta: 4 de marzo de 2024]. Disponible en: https://www.indeed.com/orientacion-profesional/desarrollo-profesional/que-es-front-end
6. **MARTINS**. *Scrum: conceptos clave y cómo se aplica en la gestión de proyectos.* 2023 [fecha de la consulta: 5 de marzo de 2024]. Disponible en: https://asana.com/es/resources/what-is-scrum
7. **KEEPCODING**. *¿Qué es un sprint en Agile?* 2024 [fecha de la consulta: 5 de marzo de 2024]. Disponible en: https://keepcoding.io/blog/que-es-un-*sprint*-en-agile/
8. **ÁLVAREZ**. *Fundamentos de Scrum Framework: Product Backlog vs Sprint Backlog*. 2023 [fecha de la consulta: 7 de marzo de 2024]. Disponible en: https://netmind.net/es/fundamentos-de-scrum-framework-product-*backlog-sprint-backlog*/
9. **MIROSLAWA**, **HERNÁNDEZ, SOLÍS, ÁLVAREZ**. *¿Qué es DevOps? Definición y características.* 2023 [fecha de la consulta: 4 de marzo de 2024]. Disponible en: https://innovaingenieria.uagro.mx/innova/index.php/innova/article/download/134/86/
10. **GALLEGOS**. *¿Qué es el diseño UX/UI y cuáles son sus ventajas?* 2022 [fecha de la consulta: 5 de marzo de 2024]. Disponible en: https://www.gluo.mx/blog/que-es-diseno-ux-ui-y-cuales-son-sus-ventajas
11. **SANCHEZ**. *¿Qué es un Stakeholder?* 2023 [fecha de la consulta: 7 de marzo de 2024]. Disponible en: https://medium.com/m %C3 %A1s-tech/qu %C3 %A9-es-un-stakeholder-4d5b85122318#:~:text=En %20*Scrum* %2C %20un %20Stakeholder %20se %20refiere %20a %20cualquier %20persona %20o,miembros %20del %20equipo %2C %20entre %20otros

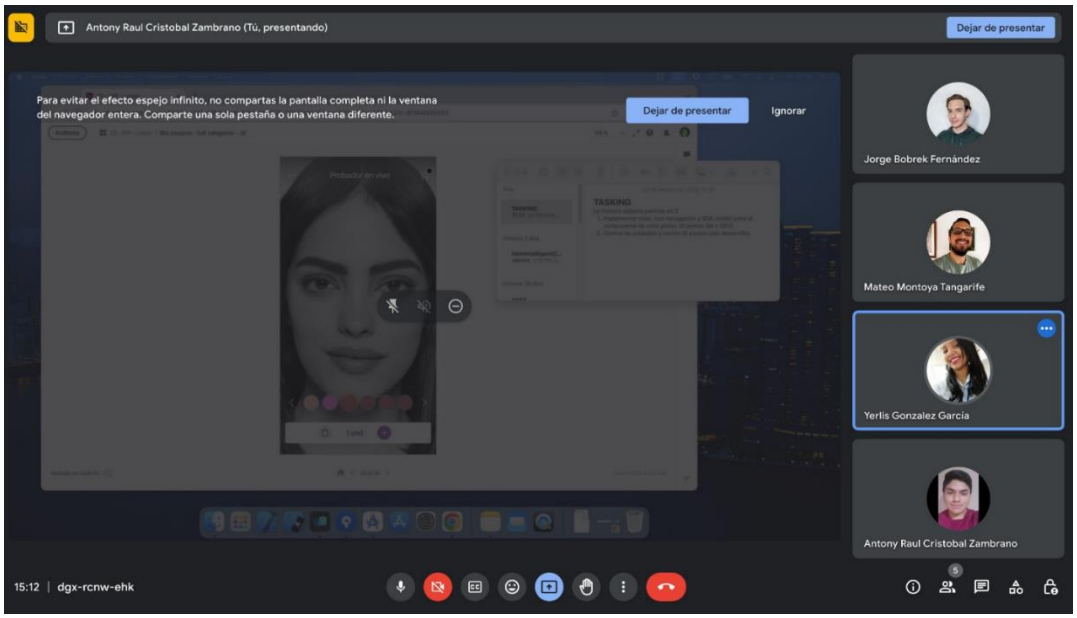
12. **LEGORRETA**. *Principios SOLID: Qué son y para qué sirven.* 2019 [fecha de la consulta: 8 de marzo de 2024]. Disponible en: https://medium.com/@nandy_x/principios-solid-que-son-y-para-que-sirven-1e4799daebf5
13. **MARTINEZ**. *API: Un Recorrido Introductorio.* 2024 [fecha de la consulta: 8 de marzo de 2024]. Disponible en: https://medium.com/@jaredcsv/apis-un-recorrido-introductorio-97464cd44c99
14. **MADEO**. *Introducción al protocolo HTTP*. 2022 [fecha de la consulta: 8 de julio de 2024]. Disponible en: https://medium.com/@diego.coder/un-vistazo-al-protocolo-http-y-https-1919ed0d725
15. **LEIRO**. *¿Qué es un SDK?* 2022 [fecha de la consulta: 9 de marzo de 2024]. Disponible en: https://actualizatec.medium.com/qu %C3 %A9-es-un-sdk-56184db155c5
16. **AUSUM Cloud**. *Scrum: qué es y por qué se ha convertido en una de las metodologías ágiles más populares.* 2020 [fecha de la consulta: 21 de febrero de 2024]. Disponible en: https://ausum.cloud/scrum-metodologia-agil-mas-popular-en-empresas/
17. **RAJ**. *MVVM Clean Architecture Pattern in Android with Use Cases.* 20 23 [fecha de la consulta: 16 de agosto de 2024]. Disponible en: https://medium.com/@ami0275/mvvm-clean-architecture-pattern-in-android-with-use-cases-eff7edc2ef76

ANEXOS

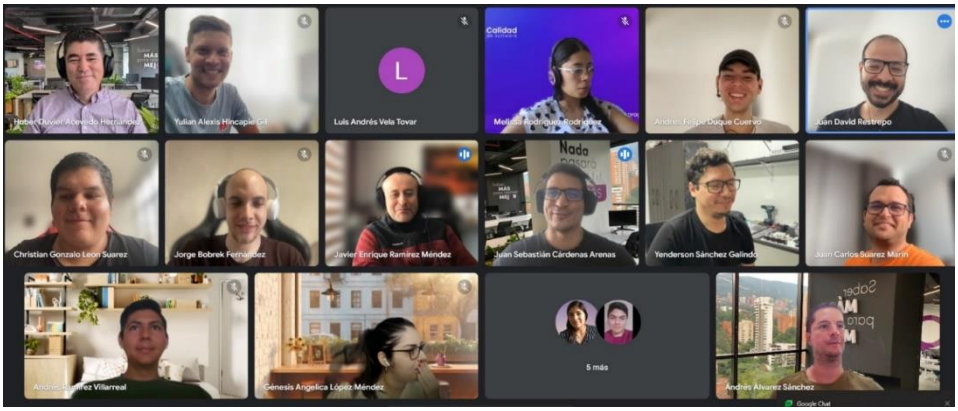
Anexo 1. Reunión de refinamiento



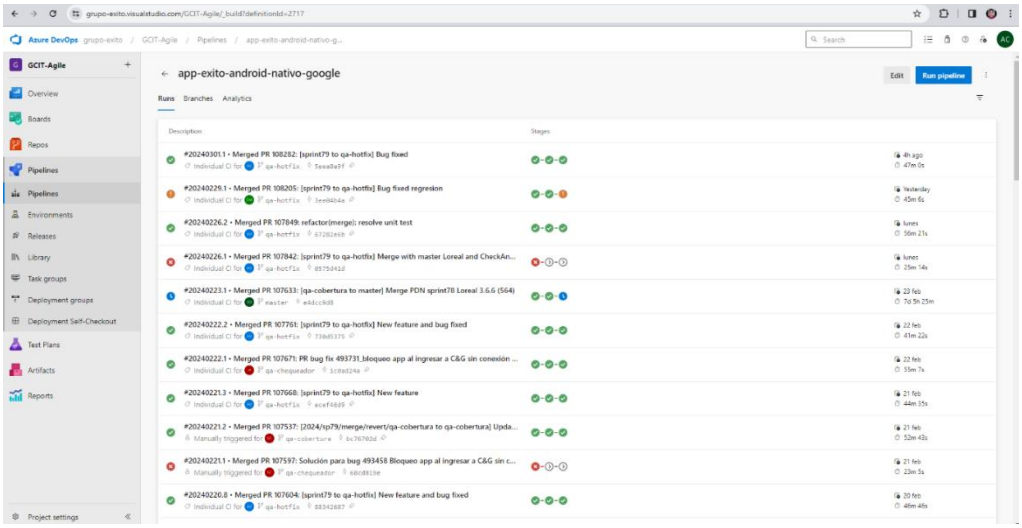
Anexo 2. Reunión de creación de tareas con el equipo de QA



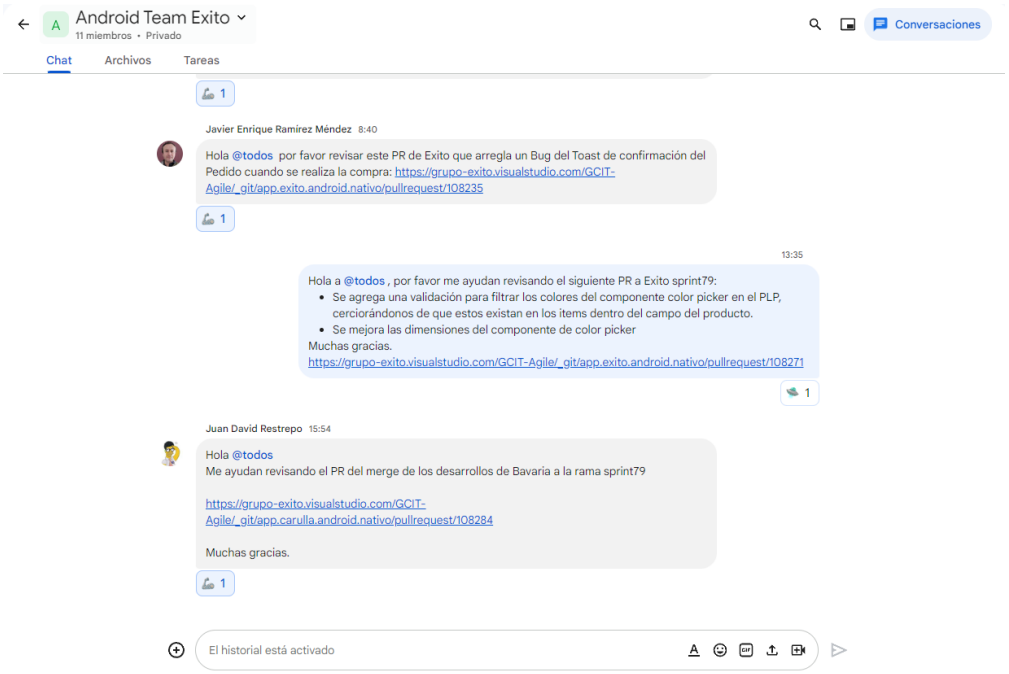
Anexo 3. Reunión de alineación del equipo general



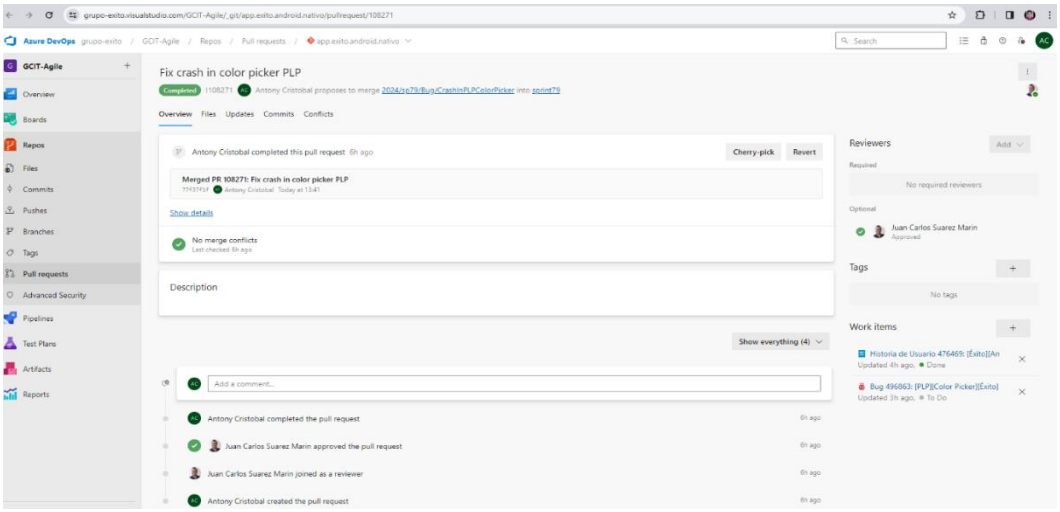
Anexo 4. Proceso de ejecución de pipelines de diferentes proyectos



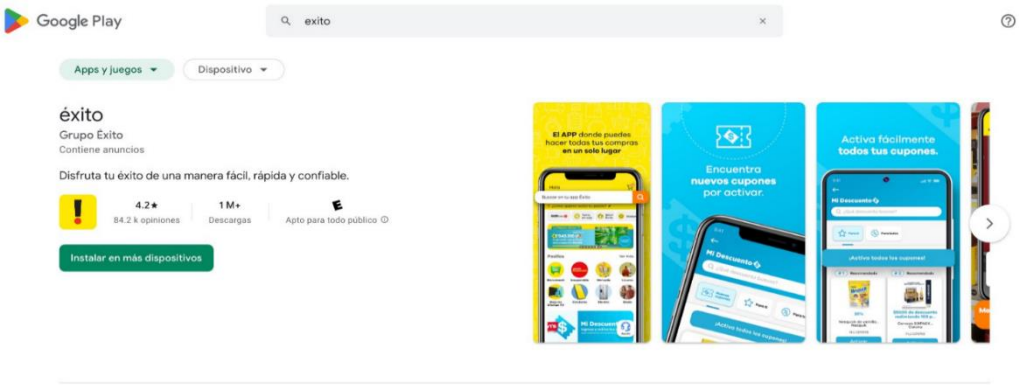
Anexo 5. Solicitud de aprobación Pull Request al equipo de desarrolladores



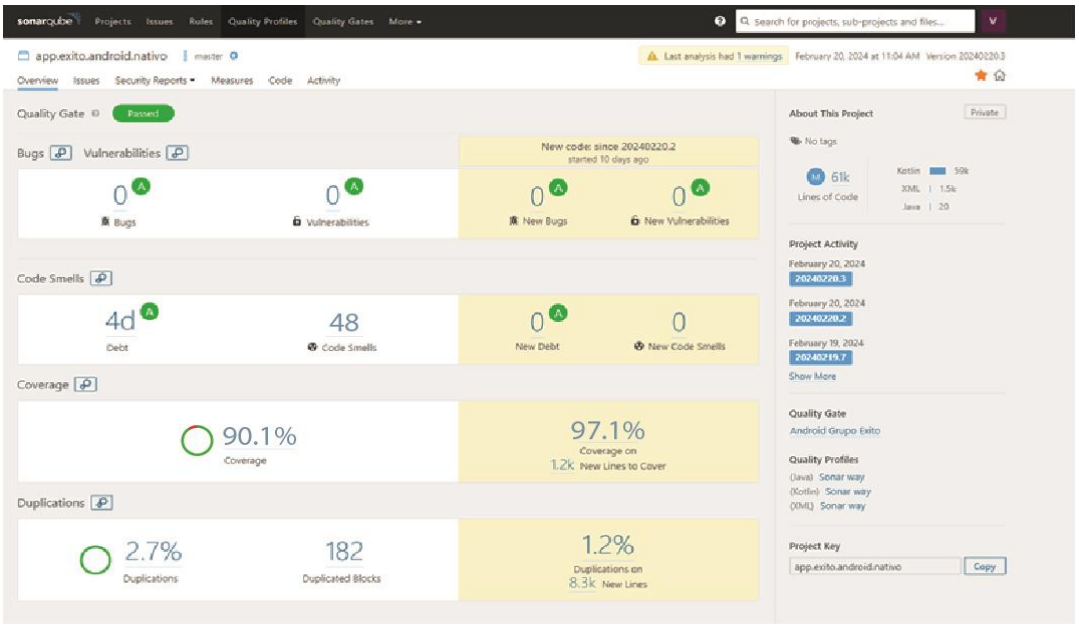
Anexo 6. Aprobación de la solicitud de Pull Request



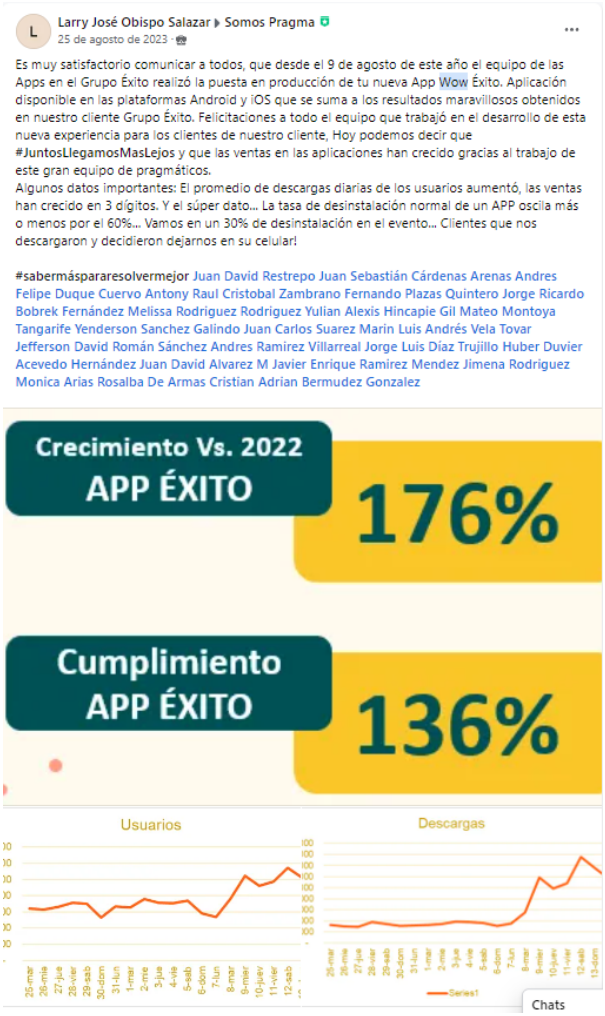
Anexo 7. Aplicación Éxito en la tienda Play Store.



Anexo 8. Métricas de Sonar Qube después del Proyecto Wow.

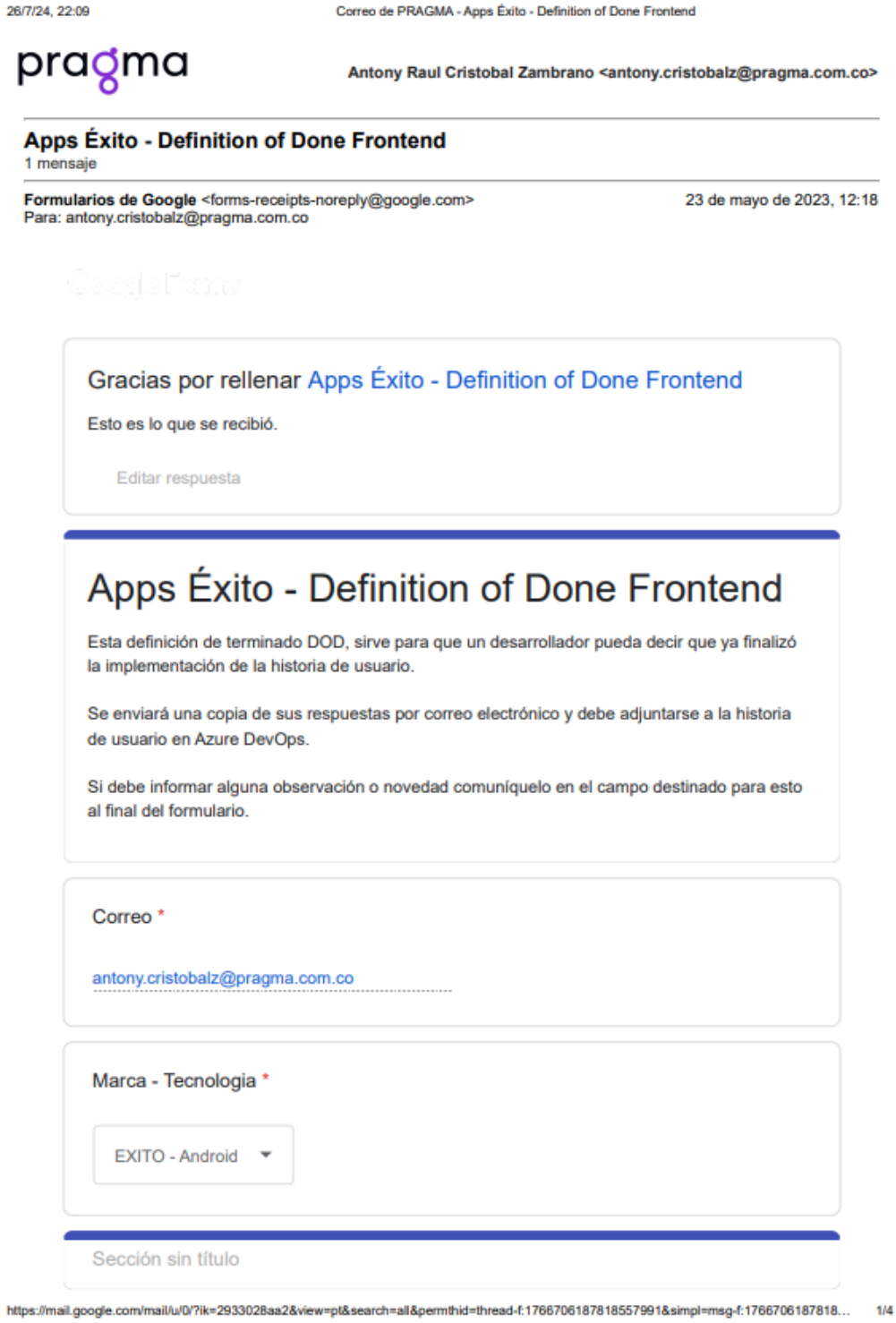


Anexo 9. Métricas de las ventas obtenidas en el evento Wow 2023.



Anexo 10. Reunión general del equipo multidisciplinario de Apps del Grupo Éxito.

Anexo 11. Formulario de definición de terminado (DOD)



26/7/24, 22:09 Correo de PRAGMA - Apps Éxito - Definition of Done Frontend

pragma

Antony Raul Cristobal Zambrano <antony.cristobalz@pragma.com.co>

Apps Éxito - Definition of Done Frontend
1 mensaje

Formularios de Google <forms-receipts-noreply@google.com> 23 de mayo de 2023, 12:18
Para: antony.cristobalz@pragma.com.co

Gracias por rellenar Apps Éxito - Definition of Done Frontend

Esto es lo que se recibió.

Editar respuesta

Apps Éxito - Definition of Done Frontend

Esta definición de terminado DOD, sirve para que un desarrollador pueda decir que ya finalizó la implementación de la historia de usuario.

Se enviará una copia de sus respuestas por correo electrónico y debe adjuntarse a la historia de usuario en Azure DevOps.

Si debe informar alguna observación o novedad comuníquelo en el campo destinado para esto al final del formulario.

Correo *

antony.cristobalz@pragma.com.co

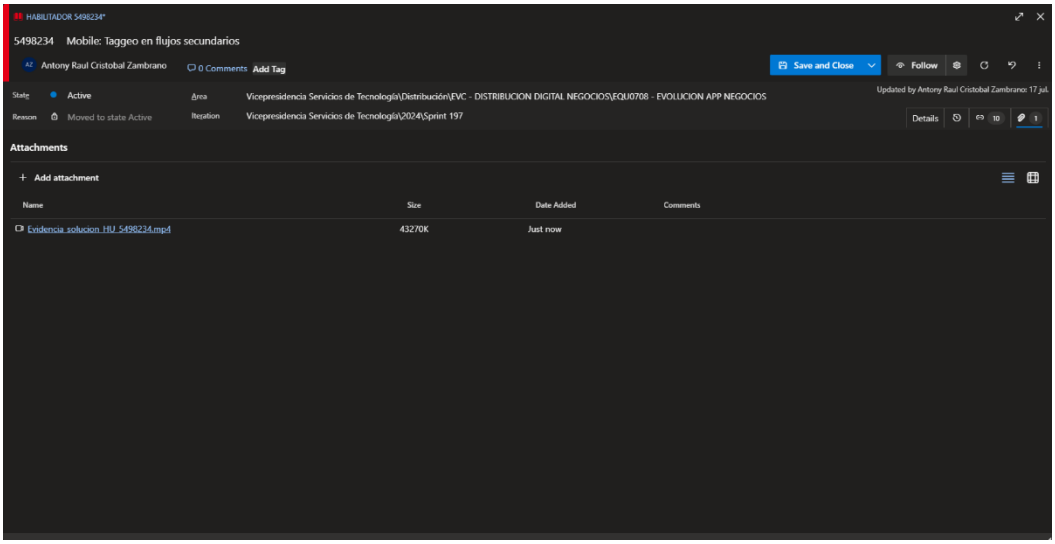
Marca - Tecnologia *

EXITO - Android

Sección sin título

https://mail.google.com/mail/u/0/?ik=2933028aa2&view=pt&search=all&permthid=thread-f:1766706187818557991&simpl=msg-f:1766706187818... 1/4

Anexo 12. Evidencia adjuntada a la historia de usuario



Anexo 13. Matriz de impacto

IMPACTO	NIVEL
MUY BAJO	0
BAJO	1
MEDIO	2
ALTO	3
EXTREMO	4

MATRIZ DE RIESGOS		
N°	EVENTO	IMPACTO
1	Almacenamiento de información confidencial	0
2	Dependencia de pruebas de performance	0
3	Dependencia de servicios externos	2
4	Version estable	0
5	Version certificada	0

Anexo 14. Matriz de riesgo

IMPACTO	NIVEL
MUY BAJO	0
BAJO	1
MEDIO	2
ALTO	3
EXTREMO	4

MATRIZ DE IMPACTO											
HU	MODULOS DE LA APP										
	PLP	PDP	PASILLOS	CHECKOUT	METODO DE ENTREGA	RECOMENDADOS PARA TI	BUSCADOR	LANDING	CHEQUEADOR	MIS LISTAS	CDP
Historia de usuario 1.1	0	0	0	0	0	0	0	0	3	0	0
Historia de usuario 2.1	0	2	0	0	0	0	0	0	0	0	0
Historia de usuario 2.2	2	0	0	0	0	0	0	0	0	0	0
Historia de usuario 3.1	2	2	0	0	0	0	0	0	0	0	0
Historia de usuario 4.1	0	0	0	0	0	0	2	0	0	0	0
Historia de usuario 4.2	1	1	1	1	1	0	1	1	0	0	1