



## **"AUTOMERCADEO VIDO MÓVIL"**

### **REPORTE DE RESIDENCIA PROFESIONAL**

**DESARROLLADO EN LA EMPRESA: "SOLUCIONES EN TECNOLOGIAS DE  
INFORMACION (SOTI)"**

**PRESENTA: ANGEL DARIO VIDAÑA VARGAS**

**NO. DE CONTROL: 19130984**

**CARRERA: ING. SISTEMAS COMPUTACIONALES**

**TORREÓN COAH.**

**JUNIO DEL 2024**

## **Agradecimientos**

Quiero comenzar expresando mi más profundo agradecimiento a mis padres, el Ing. Francisco del Ángel Vidaña González y la Mtra. Araceli Vargas Muñoz. Su apoyo incondicional y su ejemplo de dedicación y esfuerzo ha sido el motor que me impulsó a seguir adelante. Su amor y confianza en mí han sido fundamentales para alcanzar cada logro, y les estoy eternamente agradecido por todo lo que han hecho por mí.

Agradezco también a mi hermano, el Lic. Ángel Damián Vidaña Vargas, por ser mi compañero de vida y por su constante apoyo. Su compromiso y determinación siempre han sido una inspiración para mí, y su presencia ha hecho que esta etapa sea una experiencia aún más significativa para mí.

A la Lic. Marcela Delgado Moreno, que siempre me brindó su apoyo y comprensión. El amor y compañía en esta etapa han sido esenciales para mí.

Mi gratitud hacia mi asesora interna, la Dra. Ruth de la Peña Martínez, por su guía experta en esta etapa. Aprecio enormemente su apoyo y orientación.

Gracias a mi asesor externo, el Ing. Osvaldo Esparza, por su colaboración valiosa y asesoramiento y paciencia durante este periodo.

A los catedráticos Ing. Martín Osvaldo Valdés, Mtra Lamia Hamdan, Ing. Armando Ruiz, Ing. Juan Carlos Ulloa, Ing. Ricardo de Ávila, y a todos los demás que fueron parte de mi trayecto en la Ingeniería, gracias por su enseñanza y contribuciones en mi formación académica y profesional.

Agradezco también a mis compañeros Eder Campa, Sebastián Ruiz y a todos los que han compartido esta experiencia conmigo. Su amistad y apoyo han sido invaluables.

Por último, agradezco a los Ingenieros del Poder Judicial Federal, el Ing. Juan Luna y el Ing. Oskar Angulo, por su confianza y por brindarme la oportunidad de crecer en un entorno retador como lo fue el Poder Judicial Federal.

## **Resumen**

La empresa “Soluciones en Tecnologías de Información” se encuentra ubicada en el municipio de Torreón, Coahuila. Dicha empresa se especializa en el área de informática, en donde se desarrolla Software a la medida, venta de equipo de cómputo y control de acceso y vigilancia.

Como desarrollo de la práctica se efectuó la actualización de una aplicación la cual ya está en funcionamiento y en el mercado, en donde se le han agregado diferentes mejoras, módulos y funciones nuevas. Asimismo, se realizaron pruebas exhaustivas, detección de problemas, para después determinar soluciones posibles para la mejora y optimización de esta. De igual forma, se realizaron otras actividades las cuales facilitaron y agilizaron el avance de las tareas y actividades que se proponían.

## **Carta de aceptación:**



Torreón Coahuila a 19 de Marzo del 2024

INSTITUTO TECNOLOGICO DE LA LAGUNA

ATENCION: LUIS FERNANDO MADINAVEITIA SANDOVAL

JEFE DEL DEPARTAMENTO DE GESTION TECNOLOGICA Y VINCULACION

Por medio del presente documento hacemos constar que el C. ANGEL DARIO VIDAÑA VARGAS con el número de control 19130984 de la carrera de ING. EN SISTEMAS COMPUTACIONALES, es aceptado en esta empresa para que realice sus prácticas profesionales, a partir del 06 de Marzo del 2024 en nuestro departamento de desarrollo de sistemas y cumplirá con 500 horas logrando esto tentativamente el día 12 de Junio del 2024.

Quedando a sus órdenes para cualquier aclaración o duda.

ATENTAMENTE

  
ING. MARTIN HERRERA CASTORENA

Gerente de Desarrollo y asesor externo asignado

Soluciones en Tecnología de Información

## Carta de término:



Torreón Coahuila a 5 de Junio del 2024

INSTITUTO TECNOLOGICO DE LA LAGUNA

ATENCION: SALVADOR VALADEZ GONZÁLEZ

JEFE DEL DEPARTAMENTO DE GESTION TECNOLOGICA Y VINCULACION

Por medio del presente documento hacemos constar que el C. ANGEL DARIO VIDAÑA VARGAS con el número de control **19130984** de la carrera de **ING. EN SISTEMAS COMPUTACIONALES**, realizo satisfactoriamente sus prácticas profesionales en esta empresa, en nuestro departamento de desarrollo cumpliendo un total de 500 horas durante el periodo comprendido del 06 de Marzo del 2024 al 05 de Junio del 2024.

Quedando a sus órdenes para cualquier aclaración o duda.

ATENTAMENTE

  
ING. MARTIN HERRERA CASTORENA

Gerente de Desarrollo y asesor externo asignado

Soluciones en Tecnología de Información

# Evaluación: 26 de Marzo del 2024



## EVALUACIÓN Y SEGUIMIENTO DE RESIDENCIA PROFESIONAL

Nombre del Residente: Angel Dario Vidaña Vargas No. De Control: 19130984

Nombre del Proyecto: Automercadeo VIDO Móvil

Programa Educativo: Ing. En Sistemas Computacionales

Periodo de realización de la Residencia Profesional: Enero-Junio 2024

Calificación Parcial (Promedio de ambas calificaciones) **100**

En qué medida el residente cumple con lo siguiente:

Criterios a Evaluar	Valor	Evaluación
Asiste puntualmente en el horario establecido	5	5
Trabaja en equipo y se comunica de forma efectiva (oral y escrita)	10	10
Tiene iniciativa para colaborar	5	5
Propone mejoras al proyecto	10	10
Cumple con los objetivos correspondientes al proyecto	15	15
Es ordenado y cumple satisfactoriamente con las actividades encomendadas en los tiempos establecidos del cronograma	15	15
Demuestra su liderazgo en su actuar	10	10
Demuestra conocimiento en el área de su especialidad	20	20
Demuestra un comportamiento ético (es disciplinado, acata órdenes, respeta a sus compañeros de trabajo, entre otros)	10	10
Calificación Total:	100	100

Observaciones:

*Martín Herrera Castorena*  
Nombre y Firma del Asesor Externo

Sello de la Empresa, Organismo o Dependencia

**26 - Marzo - 2024**  
Fecha de evaluación

En qué medida el residente cumple con lo siguiente:

Criterios a evaluar	Valor	Evaluación
Asistió puntualmente a las reuniones	10	10
Demuestra conocimiento en el área de su especialidad	20	20
Trabaja en equipo y se comunica de forma efectiva (oral y escrita)	15	15
Es dedicado y proactivo en las actividades encomendadas	20	20
Es ordenado y cumple satisfactoriamente con las actividades encomendadas en los tiempos establecidos en el cronograma	20	20
Propone mejoras al proyecto	15	15
Calificación Total:	100	100

Observaciones:

*Ruth De La Pava M*  
Nombre y Firma del Asesor Interno

**EDUCACIÓN**  
INSTITUTO TECNOLÓGICO DE LA LAGUNA  
DEPARTAMENTO DE  
SISTEMAS Y COMPUTACIÓN  
Sello de la Institución

**26 - Marzo - 2024**  
Fecha de evaluación

## Evaluación: 29 de Abril del 2024



### EVALUACIÓN Y SEGUIMIENTO DE RESIDENCIA PROFESIONAL

Nombre del Residente: Angel Dario Vidaña Vargas No. De Control: 19130984

Nombre del Proyecto: Automercadeo VIDÓ Móvil

Programa Educativo: Ing. En Sistemas Computacionales

Período de realización de la Residencia Profesional: Enero-Junio 2024

Calificación Parcial (Promedio de ambas calificaciones) 100

En qué medida el residente cumple con lo siguiente:

Criterios a Evaluar	Valor	Evaluación
Asiste puntualmente en el horario establecido	5	5
Trabaja en equipo y se comunica de forma efectiva (oral y escrita)	10	10
Tiene iniciativa para colaborar	5	5
Propone mejoras al proyecto	10	10
Cumple con los objetivos correspondientes al proyecto	15	15
Es ordenado y cumple satisfactoriamente con las actividades encomendadas en los tiempos establecidos del cronograma	15	15
Demuestra su liderazgo en su actuar	10	10
Demuestra conocimiento en el área de su especialidad	20	20
Demuestra un comportamiento ético (es disciplinado, acata órdenes, respeta a sus compañeros de trabajo, entre otros)	10	10
Calificación Total:	100	100

Observaciones:

<i>Martín Herrera Costerena</i> Nombre y Firma del Asesor Externo	Sello de la Empresa, Organismo o Dependencia	<i>29 - Abril - 2024</i> Fecha de evaluación
--	--	---

En qué medida el residente cumple con lo siguiente:

Criterios a evaluar	Valor	Evaluación
Asistió puntualmente a las reuniones	10	10
Demuestra conocimiento en el área de su especialidad	20	20
Trabaja en equipo y se comunica de forma efectiva (oral y escrita)	15	15
Es dedicado y proactivo en las actividades encomendadas	20	20
Es ordenado y cumple satisfactoriamente con las actividades encomendadas en los tiempos establecidos en el cronograma	20	20
Propone mejoras al proyecto	15	15
Calificación Total:	100	100

Observaciones:

<i>R. H. Beltrán Peralta</i> Nombre y Firma del Asesor Interno	<b>EDUCACIÓN</b> INSTITUTO TECNOLÓGICO DE LA LAGUNA Sello Departamento de Sistemas Computacionales Revisión 00	<i>29 - Abril - 2024</i> Fecha de evaluación
---	---	---

RGI-DEP-06

## Evaluación 2 de Junio del 2024



### EVALUACIÓN DE REPORTE DE RESIDENCIA PROFESIONAL

Nombre del Residente: Ángel Darío Vidáu Vargas

Nº De Control: 19130984

Nombre del Proyecto: Automercadeo VIDO Móvil

Programa Educativo: Ingeniería en Sistemas Computacionales

Periodo de realización de la Residencia Profesional: Enero – Junio 2024

Calificación Parcial (Promedio de ambas calificaciones)

100

En qué medida el residente cumple con lo siguiente.

Criterios a Evaluar

Valor

Evaluación

Portada	2	2
Agradecimientos	2	2
Resumen	2	2
Índice	2	2
Introducción	2	2
Problemas a resolver priorizándolos	5	5
Objetivos	5	5
Justificación		
Marco Teórico (fundamentos teóricos)	10	10
Procedimiento y descripción de las actividades realizadas	5	5
Resultados, planos, gráficos, prototipos, manuales, programas, análisis estadísticos, modelos matemáticos, simulaciones, normativas, regulaciones y restricciones, entre otros. Sólo para proyectos que por su naturaleza lo requieran estudio de mercado, estudio técnico y estudio económico.	45	45
Conclusiones, recomendaciones y experiencia profesional adquirida	15	15
Competencias desarrolladas y/o aplicadas	3	3
Fuentes de información	2	2
Calificación Total:	100	100

Observaciones:

*Martín Herrera Costarera*

Martín Herrera Costarera  
Nombre y Firma del Asesor Externo

Sello de la Empresa, Organismo o Dependencia

3 - Junio - 2024

Fecha de evaluación

Evaluación por el asesor interno

En qué medida el residente cumple con lo siguiente:	
Criterios a evaluar	Valor
Portada	2
Agradecimientos	2
Resumen	2
Índice	2
Introducción	2
Problemas a resolver priorizándolos	5
Objetivos	5
Justificación	
Marco Teórico (fundamentos teóricos)	10
Procedimiento y descripción de las actividades realizadas	5
Resultados, planos, gráficos, prototipos, manuales, programas, análisis estadísticos, modelos matemáticos, simulaciones, normativas, regulaciones y restricciones, entre otros. Sólo para proyectos que por su naturaleza lo requieran estudio de mercado, estudio técnico y estudio económico.	45
Conclusiones, recomendaciones y experiencia profesional adquirida	15
Competencias desarrolladas y/o aplicadas	3
Fuentes de información	2
Calificación Total:	100

Observaciones:

*Ruth Díaz Pérez*

Ruth Díaz Pérez  
Nombre y Firma del Asesor Interno



3 - Junio - 2024

Fecha de evaluación

RGI-DEP-07

Revisión 00

# Índice

<b>Introducción .....</b>	<b>1</b>
<b>Capítulo 1. Generalidades del proyecto .....</b>	<b>3</b>
1.1 Descripción de la empresa.....	3
1.1.1 Soluciones en Tecnologías de Información (SOTI).....	3
1.1.2 Giro de la empresa .....	3
1.1.3 Misión de la empresa.....	3
1.1.4 Visión de la empresa.....	3
1.1.5 Valores de la empresa .....	3
1.2 Problemas a resolver .....	4
1.3 Objetivo general y específicos.....	6
1.3.1 Objetivo general.....	6
1.3.2 Objetivos específicos .....	6
1.4 Justificación .....	7
<b>Capítulo 2. Marco teórico .....</b>	<b>9</b>
2.1. Fundamentos de desarrollo de aplicaciones móviles.....	9
Android Studio .....	9
JAVA .....	9
2.2. Desarrollo de Front-end y Back-end.....	10
Visual Studio Code.....	10
WampServer .....	10
2.3. Gestión de Servicios Web y API .....	10
SoapUI.....	10
2.4. Control de Versiones y Colaboración .....	11
GitHub Desktop .....	11

2.5. Teorías de Sistemas de Información.....	11
<b>Capítulo 3. Procedimiento y descripción de las actividades realizadas</b>	
.....	12
3.1. Introducción .....	12
3.2. Descripción de las actividades realizadas .....	12
3.2.1. Análisis y diagnóstico inicial .....	12
3.2.2. Planificación de actualizaciones y nuevas funcionalidades .....	13
3.2.3. Desarrollo e implementación.....	15
3.2.4. Pruebas y validación .....	23
3.2.5. Documentación y capacitación .....	25
<b>Capítulo 4. Resultados obtenidos</b> .....	26
Conclusiones.....	29
Competencias: .....	31
Referencias bibliográficas .....	32
Anexos .....	33
Código: módulo selección de BD .....	33
Código: módulo BIENVENIDO (nueva lógica).....	37
Código: módulo Agrega Caducidad, cambios en acceso a la BD.....	43
Código: módulo Caducidades, cambios en acceso a la BD.....	44
Código: módulo Introducir Caducidades, cambios en acceso a la BD....	44
Código: módulo Dialogo Cantidad, cambios en acceso a la BD .....	45
Código: módulo Dialogo Precio, cambios en acceso a la BD.....	45
Código: módulo Faltantes Shopper, cambios en acceso a la BD. ....	46
Código: módulo Conteo Clientes, cambios en acceso a la BD. ....	46
Código: módulo Línea Cajas, cambios en acceso a la BD. ....	47

Código: módulo Personal Anaquel, cambios en acceso a la BD.....	47
Código: módulo Competencias, cambios en acceso a la BD.....	48
Código: módulo Dialogo Buscar Proveedores, cambios en acceso a la BD.....	48
Código: módulo Proceso GAP, cambios en acceso a la BD.....	49
Código: módulo Mistery Shopper, cambios en acceso a la BD.....	49
Código: módulo MS Producto Precio, cambios en acceso a la BD.....	50
Código: implementación de módulo Faltantes Solución.....	50
Código: Adaptador de la lista expandible de Faltantes Solucion.....	85
Código: Adaptador de la lista expandible de Faltantes Solucion después de finaliza la actividad.....	91
Código: Módulo Reporte, clase en la que se visualiza grafica del módulo Faltantes Solucion.....	96
Código: Solucion a Valores nulos de Mistery Shopper.....	110
Código: Solucion a Valores nulos de Producto Precio. ....	111
Código: Implementacion de botón Actualizar en Faltantes Solucion.....	111

## **Índice de ilustraciones**

<b>Ilustración 1.</b> Planificación de actualizaciones y nuevas funcionalidades.....	14
<b>Ilustración 2.</b> Cambios en el back de la aplicación (funciones de los web service) .....	15
<b>Ilustración 3.</b> Actualización de módulos existentes.....	16
<b>Ilustración 4.</b> Corrección de errores.....	17
<b>Ilustración 5.</b> Optimización de la lógica de consulta y acceso a la base de datos.....	17
<b>Ilustración 6.</b> Implementación e integración del módulo "Faltantes Solución".....	18
<b>Ilustración 7.</b> Optimización de la aplicación.....	19
<b>Ilustración 8.</b> Documentación de nuevas funcionalidades y cambios realizados.....	20
<b>Ilustración 9.</b> Cambios en la lógica de la pantalla de "bienvenido" .....	22
<b>Ilustración 10.</b> Uso de emuladores y dispositivos físicos para garantizar la compatibilidad.....	23
<b>Ilustración 11.</b> Validación de la correcta interacción entre la aplicación y los web service.....	24
<b>Ilustración 12.</b> Resultados obtenidos.....	26
<b>Ilustración 13.</b> Resultados obtenidos.....	27
<b>Ilustración 14.</b> Resultados obtenidos.....	27
<b>Ilustración 15.</b> Resultados obtenidos.....	28

## **Índice de tablas**

<b>Tabla 1.</b> Análisis y diagnóstico inicial.....	12
<b>Tabla 2.</b> Planificación de actualizaciones y nuevas funcionalidades. ....	13
<b>Tabla 3.</b> Desarrollo e Implementación. ....	15
<b>Tabla 4.</b> Pruebas y validación.....	23
<b>Tabla 5.</b> Documentación y capacitación.....	25

## **Introducción**

“Si puedes imaginarlo, puedes programarlo” - Alejandro Taboada (1997- 2019).

Actualmente, la empresa SOTI cuenta con una aplicación móvil que está en funcionamiento llamada “VIDO”, la cual se enfoca en el auto-mercadeo en diferentes mercados de la región. Dicha aplicación tiene como objetivo hacer eficientes y mucho más rápidas las diferentes tareas para proveedores. Además, en esta misma se tienen diferentes módulos en los cuales puedes trabajar.

El uso estratégico de esta aplicación por parte de la empresa SOTI ha resultado en mejoras significativas en la productividad y en la experiencia del usuario. La aplicación está disponible exclusivamente para dispositivos Android y se instala a través de un proceso simple, pero seguro.

La aplicación también está diseñada para ser intuitiva. Por ejemplo, la funcionalidad de inicio de sesión se ha incluido opciones de autenticación biométrica, lo que refuerza la seguridad sin sacrificar la velocidad de acceso. Una vez dentro de la aplicación, los usuarios pueden navegar por un sistema de módulos claramente delineados que abarcan desde el registro de entrada y salida del personal, hasta la verificación y el registro de productos.

Como características de la aplicación tenemos el uso de las geovallas, las cuales nos sirven para definir límites geográficos virtuales en un área del mundo real. Lo anterior funciona mediante GPS o alguna otra tecnología de localización en tiempo real, la cual es la encargada de saber con exactitud cuando el usuario que tiene el dispositivo móvil entra y sale de un área específica, conocida como geovallas.

La aplicación contiene en su mayoría webservice ya que implementarlos en una aplicación ofrece numerosos beneficios, incluyendo la separación de la lógica de cliente y servidor, lo que facilita la actualización y el mantenimiento sin afectar la experiencia del usuario. En adición, mejora la gestión de la seguridad y el acceso a recursos remotos, y varios de los webservice se pueden reutilizar en diferentes momentos de la aplicación

facilitando y reduciendo los costos de desarrollo y operación, haciendo que sea mucho más ágil y con una fácil adaptación.

Esta aplicación actualmente ya está siendo usada y puesta a prueba en el mercado por una empresa la cual le trabaja a Soriana y a otros comercios importantes de la región. Esta aplicación puede entrar en funcionamiento en casi cualquier parte del mundo, ya que los módulos que están dentro de la aplicación se moldean a las necesidades de quien lo utiliza, por lo que no es ningún problema que alguna otra compañía o comercio contrate los servicios de VIDO.

El propósito de la aplicación es optimizar las operaciones de mercadeo en establecimientos comerciales, permitiendo a los usuarios gestionar el inventario, controlar caducidades y precios, así como realizar auditorías de competencia directamente desde sus dispositivos móviles. A través de funciones avanzadas como el escaneo de productos, registro biométrico para seguridad y autenticación, y monitoreo de tareas de personal, la aplicación facilita una gestión eficiente y en tiempo real de los recursos y actividades en el punto de venta, mejorando así la eficacia operativa y el servicio al cliente.

# **Capítulo 1. Generalidades del proyecto**

## **1.1 Descripción de la empresa**

**1.1.1 Soluciones en Tecnologías de Información (SOTI).** Empresa especializada en el área de informática, la cual se dedica a hacer software a la medida, venta de equipo de cómputo, control de acceso, vigilancia y soporte técnico.

**1.1.2 Giro de la empresa.** Desarrollo de software y tecnología.

**1.1.3 Misión de la empresa.** Que todos nuestros clientes aprovechen al máximo los recursos tecnológicos para realizar sus labores diarias de manera eficiente y ahorrando costos de operación, logrando con esto posicionarnos como la empresa de soluciones tecnológicas de preferencia en el mercado.

**1.1.4 Visión de la empresa.** Ser reconocido como un proveedor de tecnologías de información que ofrece soluciones y ayuda a nuestros clientes a realizar sus labores diarias y con esto lograr que ellos sean nuestra mejor recomendación hacia más clientes y mercados.

## **1.1.5 Valores de la empresa**

**Responsabilidad.** SOTI se responsabiliza no solo de entregar productos y servicios de alta calidad, sino también de asegurar que estos contribuyan positivamente a la operación diaria de sus clientes. Esto implica una gestión ética y sostenible de sus operaciones, garantizando que todas las soluciones cumplan con las expectativas y necesidades del cliente.

**Compromiso.** La empresa se compromete con sus clientes al proporcionar un soporte continuo y dedicación a la excelencia. Este compromiso se traduce en una búsqueda constante de mejoras y adaptaciones.

**Profesionalismo.** SOTI valora profundamente el profesionalismo en todos sus empleados y servicios. Su profesionalismo asegura que todos los proyectos se manejen

con la máxima eficacia y se entreguen puntualmente, cumpliendo con los estándares más altos de la industria.

**Innovación.** La visión de SOTI de ser un líder reconocido en el campo de la tecnología de información se apoya fuertemente en su capacidad de innovar. Esto no solo implica seguir las últimas tendencias tecnológicas, sino también ser pioneros en el desarrollo de nuevas soluciones que permitan a sus clientes estar a la vanguardia. La innovación está en el corazón de su modelo de negocio, permitiéndoles explorar nuevas áreas de crecimiento y ofrecer soluciones creativas y efectivas.

## 1.2 Problemas a resolver

La aplicación requiere una serie de modificaciones significativas, incluyendo una importante actualización para abordar un problema crucial: la limitación de su base de datos, lo que restringía considerablemente su potencial de comercialización.

El nuevo módulo, "Faltantes Solución", fue diseñado e implementado desde cero, representando un significativo desafío técnico y de integración con el resto de la aplicación móvil. Este módulo interactúa directamente con el módulo preexistente "Faltantes Shopper" y tiene como objetivo principal ofrecer soluciones específicas a productos que no están disponibles o que presentan algún tipo de problema.

### Funcionalidades del módulo Faltantes Solución:

**Visualización de productos:** muestra una tabla interactiva con los productos listados en el módulo "Faltantes Shopper". Estos productos son aquellos que han sido identificados previamente como faltantes o problemáticos.

**Asignación de soluciones:** permite al usuario seleccionar cada producto y asignar una solución adecuada desde una lista predefinida de acciones correctivas. Cada solución asignada cambia de color a verde en la interfaz para indicar que el problema ha sido atendido.

**Registro de actividad:** el módulo registra automáticamente la hora de inicio cuando el usuario comienza a trabajar en la solución de los faltantes. Una vez que todos los productos han sido procesados y el usuario presiona el botón "Finalizar", el sistema marca la hora de finalización.

**Interfaz de usuario intuitiva:** la interfaz ha sido diseñada para ser intuitiva y fácil de usar, asegurando que el proceso de solución sea eficiente y que los estados de los productos se visualicen claramente.

Se actualizaron y corrigieron más errores que se tenían dentro de la aplicación, las cuales se fueron resolviendo conforme se iban detectando y avanzando en el proyecto, entre ellos destacan los siguientes:

**Impresión doble de cliente:** en la pantalla principal, se tiene que seleccionar el cliente al que se va a auditar, aparece una lista en donde salen los clientes disponibles y si por alguna razón esa lista se cierra y se vuelve a abrir, los clientes salen duplicados, por lo que es molesto a la hora de volver a querer seleccionar algún cliente.

**Impresión doble de soluciones:** de la misma manera que en la impresión doble de cliente, se ve en este caso. Si por alguna razón se cierra la lista donde se imprimen las soluciones disponibles que se agregaron en el módulo nuevo de Faltantes Solución, las soluciones se duplicaban, y era molesto el estar viendo dos veces cada solución cada vez que se quería seleccionar alguna de estas.

**Mal funcionamiento de módulos pre-actualización:** se hicieron los cambios correspondientes en el para que los WebService y la base de datos, fueran reconocidas correctamente por la aplicación.

**Nueva lógica para consulta de módulos:** se hizo un cambio nuevo en la forma en la que se consultaban los módulos, ya que la aplicación se contrata dependiendo de las necesidades del cliente, por lo que antes se tenía que contratar forzosamente el módulo de Asistencia. Lo que hacía era marcar la entrada del auditor y una vez marcada la entrada, se desplegaban los módulos, entonces si no se había hecho este proceso, no

había forma de que los módulos dentro de la aplicación se mostraran. Entonces, para evitar que forzosamente se adquiera el módulo de asistencia, se implementó un WebService el cual llevaba dos parámetros que van directamente asignados al cliente (este se selecciona desde un principio), “N” y “S”. La lógica era la siguiente: si el parámetro que viene asignado al cliente es “S”, el auditor tiene que hacer el procedimiento de entrar a Asistencia a marcar entrar para que se puedan desplegar los módulos y seguir haciendo su trabajo con normalidad. Si el parámetro asignado al cliente es “N”, al auditor se le mostrarán todos los módulos, esto sin haber pasado por la Asistencia y sin haber marcado entrada.

**Limpieza, organización y depuración de código:** la aplicación móvil fue modificada muchas veces por dos o más desarrolladores, por lo que había mucho código “basura”. Entre ellos métodos innecesarios, bloques de código comentados, variables sin usar, entre otras cosas, todo lo que no se usaba se fue eliminando. Las cosas nuevas que se fueron añadiendo, se fueron documentando para darle una organización al código y saber qué fue lo que se hizo en ese apartado.

## 1.3 Objetivo general y específicos

### 1.3.1 Objetivo general

Desarrollar una aplicación móvil integral que optimice las actividades de mercadeo para diversos establecimientos comerciales.

### 1.3.2 Objetivos específicos

- **Optimizar funcionamiento de la base de datos:** identificar y corregir las limitaciones en la estructura y capacidad de la base de datos actual, de forma que se permita una mayor escalabilidad, flexibilidad y facilidad de acceso.
- **Actualizar funcionalidades y corregir errores:** evaluar las funcionalidades actuales de la aplicación para identificar áreas de mejora y diseñar soluciones que optimicen la experiencia del usuario. Corregir problemas reportados como la

duplicación de datos y fallos en el manejo de listas, asegurando que los módulos sean intuitivos y funcionen sin errores.

- **Desarrollar nuevos módulos:** crear un módulo de solución para gestionar productos faltantes, permitiendo a los proveedores realizar un seguimiento eficiente de estos productos. Este módulo debe incluir una interfaz intuitiva que permita asignar soluciones específicas y llevar un registro claro del proceso, además de documentar el tiempo de inicio y finalización de las tareas.
- **Depuración y documentación del código:** revisar el código existente para eliminar métodos innecesarios, bloques comentados y variables no utilizadas, manteniendo únicamente lo esencial para el funcionamiento de la aplicación. Documentar las nuevas funciones de forma clara, facilitando la colaboración y el mantenimiento por parte de otros desarrolladores.
- **Optimización de la Lógica de los WebService:** rediseñar la comunicación entre la aplicación y los módulos para asegurar que los clientes puedan acceder a los servicios sin inconvenientes, independientemente de si utilizan el módulo de asistencia. Implementar nuevas funciones que permita una lógica condicional para mostrar los módulos según las necesidades del cliente.

## 1.4 Justificación

Se ha determinado la necesidad de actualizar y mejorar la aplicación VIDO, empleada extensamente en operaciones de automercadeo. Esta exigencia surge principalmente de las limitaciones observadas en la arquitectura de la base de datos y la eficiencia general de los módulos operativos de la aplicación, que restringían la capacidad de la empresa para expandirse y escalar sus operaciones comerciales de manera efectiva.

Adicionalmente, se detectaron problemas relacionados con la gestión de datos y la duplicación de información dentro de la aplicación, lo cual no solo propiciaba ineficiencias operativas, sino que también elevaba el riesgo de errores en la entrada de datos y en la ejecución de procesos clave. Estas fallas exigían una solución que no solo abordara las

falencias inmediatas, sino que también anticipara necesidades futuras para adaptaciones.

Por lo tanto, este proyecto se justifica plenamente como una iniciativa crítica para revisar y reestructurar la aplicación VIDO. Al hacerlo, no solo se mejorará el rendimiento y la funcionalidad del sistema, sino que también se reforzará la infraestructura tecnológica de la empresa. Este fortalecimiento permitirá a la organización mejorar significativamente su eficiencia operativa, asegurando así una mayor adaptabilidad a las variaciones del mercado y mejorando la satisfacción del cliente. A largo plazo, se espera que estas mejoras consoliden la posición de la empresa en el mercado como líder en soluciones tecnológicas innovadoras y confiables, impulsando el crecimiento sostenido y la expansión en nuevos segmentos de mercado.

## **Capítulo 2. Marco teórico**

### **2.1. Fundamentos de desarrollo de aplicaciones móviles**

#### **Android Studio**

Android Studio no solo ofrece un entorno de desarrollo optimizado para Android, sino que también integra emuladores para probar aplicaciones en diferentes configuraciones de hardware y versiones de Android sin necesidad de dispositivos físicos. Esto es crucial para garantizar que una aplicación funcione bien en toda la diversidad de dispositivos Android en el mercado (Google, 2013).

**Importancia:** la capacidad de Android Studio para gestionar proyectos de manera eficiente, junto con su integración con Gradle, ofrece un sistema potente para manejar dependencias y configuraciones de compilación, lo que es esencial para proyectos complejos como VIDO.

#### **JAVA**

**Ventajas del lenguaje:** Java ofrece robustez, seguridad y portabilidad, lo que lo hace ideal para desarrollar aplicaciones que necesitan ser confiables y seguras, especialmente en un entorno móvil donde los recursos pueden ser limitados y la seguridad es una preocupación (Microsystems, Sun, 1995).

**Aplicaciones en el proyecto:** en VIDO, Java se usó para implementar lógica compleja de negocios y gestión de datos, garantizando un rendimiento óptimo y una experiencia de usuario coherente.

#### **XML**

**Roles en Android:** en Android, XML no solo se utiliza para diseñar interfaces, sino también para definir recursos como cadenas de texto, colores y dimensiones, lo que facilita la localización y adaptación de la aplicación para diferentes mercados y dispositivos (Consortium, 1998).

Implementación en VIDO: XML fue crucial para definir la estructura y el diseño de las interfaces de usuario de VIDO, asegurando que sean intuitivas y accesibles para todos los usuarios finales.

## 2.2. Desarrollo de Front-end y Back-end

### Visual Studio Code

**Facilidad de uso:** este editor es ampliamente apreciado por su interfaz ligera pero potente, su personalización y su capacidad para integrar numerosos plugins, lo que ayuda a mejorar la productividad de los desarrolladores (Microsoft, 2015).

**Uso en el proyecto:** fue utilizado para escribir y editar scripts y código de soporte, incluyendo el manejo de archivos XML y scripts de configuración del servidor.

### WampServer

**Funcionalidad en el desarrollo local:** actúa como un servidor local para el desarrollo web, proporcionando una plataforma para probar las aplicaciones antes de su implementación en un entorno de producción (Bourdon, 2005).

**Contribución al proyecto:** permitió simular cómo la aplicación interactúa con la base de datos en un entorno controlado, facilitando el testeo y ajustes antes de la implementación final.

## 2.3. Gestión de Servicios Web y API

### SoapUI

**Características y beneficios:** SoapUI no solo permite probar APIs sino que también ayuda a simular cómo las aplicaciones se comportarán en escenarios de uso real, lo que es esencial para garantizar la calidad y la fiabilidad de los servicios web en producción (Software, SmartBear, 2005).

**Utilización en el proyecto:** en VIDO, SoapUI fue fundamental para validar y asegurar la correcta comunicación entre la aplicación móvil y los servicios web, especialmente en el manejo de transacciones de datos críticos.

## **2.4. Control de Versiones y Colaboración**

### **GitHub Desktop**

**Importancia del control de versiones:** facilita la gestión de versiones de la aplicación, proporcionando un historial detallado de cambios y la capacidad de revertir a versiones anteriores si es necesario (Github, Inc., 2014).

**Impacto en VIDO:** la integración con GitHub aseguró que todos los miembros del equipo de desarrollo pudieran colaborar efectivamente, manteniendo un registro coherente de todas las modificaciones y contribuciones.

## **2.5. Teorías de Sistemas de Información**

**Teoría de Sistemas:** aplica conceptos de sistemas para analizar y diseñar sistemas complejos de información. En el contexto de VIDO, esta teoría ayudó a entender cómo diferentes componentes de software y hardware pueden integrarse para funcionar como un sistema cohesivo (Arnold, 1998).

**Modelado de procesos de negocio:** utilizando BPMN (Business Process Model and Notation) para visualizar y mejorar los procesos de negocio involucrados en el mercadeo automático que VIDO facilita.

## **Capítulo 3. Procedimiento y descripción de las actividades realizadas**

### **3.1. Introducción**

En este capítulo se describen de manera detallada las actividades y procedimientos llevados a cabo en la empresa. Se presenta una visión general de los pasos seguidos, las metodologías aplicadas y las herramientas utilizadas para el desarrollo y actualización de la aplicación móvil "VIDO".

### **3.2. Descripción de las actividades realizadas**

#### **3.2.1. Análisis y diagnóstico inicial**

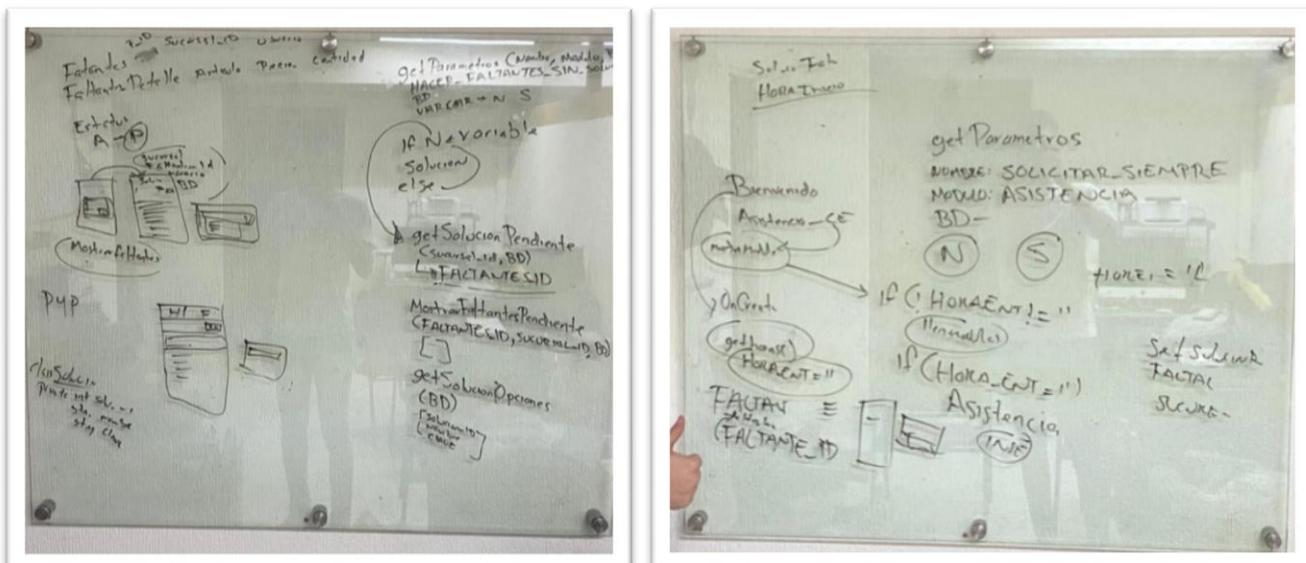
<b>Actividad</b>	<b>Descripción</b>	<b>Duración</b>
<b>Revisión del estado actual de la aplicación "VIDO".</b>	Se evaluó la funcionalidad y estructura de la aplicación "VIDO" para identificar áreas de mejora.	Dentro del periodo 06 de marzo del 2024 – 15 de marzo del 2024.
<b>Evaluación de la funcionalidad existente.</b>	Se examinó la funcionalidad actual para detectar problemas y áreas de mejora.	Dentro del periodo 06 de marzo del 2024 – 15 de marzo del 2024.
<b>Identificación de áreas de mejora y problemas reportados.</b>	Se identificaron algunos problemas reportados y se analizaron las áreas que necesitaban mejoras.	Dentro del periodo 06 de marzo del 2024 – 15 de marzo del 2024.
<b>Ánalisis de la estructura de la base de datos y la arquitectura de la aplicación.</b>	Se revisó la estructura de la base de datos y la arquitectura de la aplicación para identificar posibles mejoras.	Dentro del periodo 06 de marzo del 2024 – 15 de marzo del 2024.

Tabla 1. Análisis y diagnóstico inicial.

### 3.2.2. Planificación de actualizaciones y nuevas funcionalidades

Actividad	Descripción	Duración
<b>Definición de objetivos específicos y tareas.</b>	Se establecieron objetivos específicos y tareas a realizar por parte del asesor.	Dentro del periodo 16 de marzo del 2024 – 23 de marzo del 2024.
<b>Elaboración de un backlog con las mejoras y nuevas funcionalidades a implementar.</b>	Se establecieron mejoras y nuevas funcionalidades a implementar.	Dentro del periodo 16 de marzo del 2024 – 23 de marzo del 2024.
<b>Priorización de tareas en colaboración con el equipo de desarrollo y el asesor del proyecto.</b>	Se priorizaron las tareas en colaboración con el equipo de desarrollo y el asesor del proyecto.	Dentro del periodo 16 de marzo del 2024 – 23 de marzo del 2024.

Tabla 2. Planificación de actualizaciones y nuevas funcionalidades.



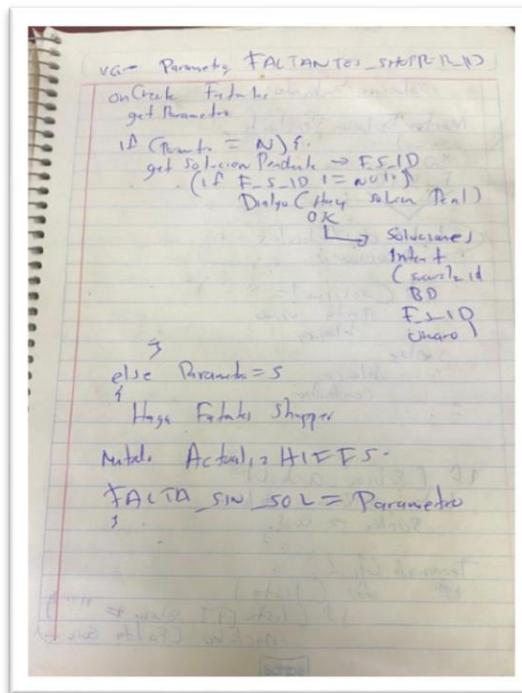
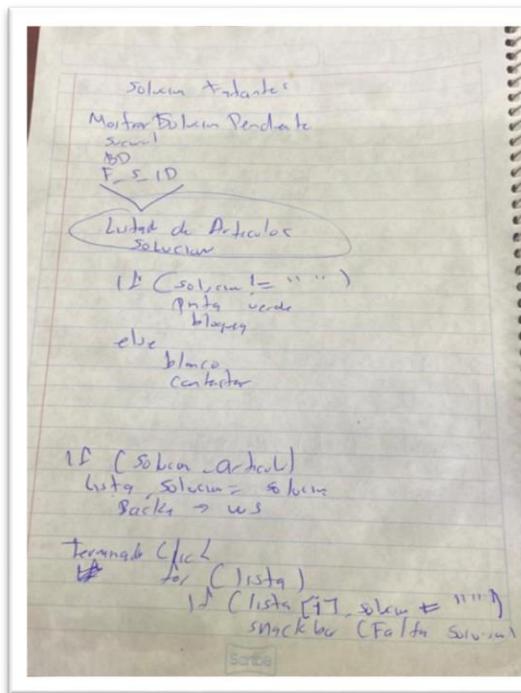
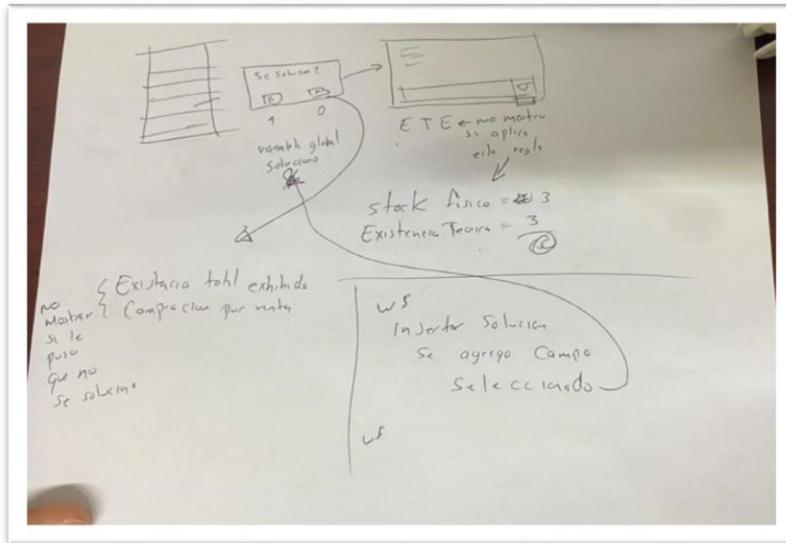


Ilustración 1. Planificación de actualizaciones y nuevas funcionalidades.

Las ideas principales de los cambios a realizar, se plasmaron primero en “físico” para tener un camino más claro sobre qué hacer exactamente y en base a lo plasmado en físico, era como se implementaba en código.

### 3.2.3. Desarrollo e implementación

Activad	Descripción	Duración
<b>Cambios en el back de la aplicación (funciones de los web service).</b>	Se hicieron cambios correspondientes en el back de la aplicación para nueva adaptación de la base de datos.	Dentro del periodo de 25 de marzo del 2024 al 17 de mayo del 2024.

Tabla 3. Desarrollo e implementación.

10 changed files	dikkat\modulos\Services17\barrido.php
dikkat\modulos\Services...\barrido.php	@@ -19,8 +19,8 @@ function BuscarBarridoID(\$FECHA,\$SUCURSAL_ID,\$conn){ 19 19 return -1; 20 20 } 21 21 } 22 22 - function BuscarBarridoDetalleID(\$BARRIDO_ID,\$ARTICULO_ID){ 23 23 - \$conn = ABRIR_CONEXION_MYSQL(FALSE); 22 22 + function BuscarBarridoDetalleID(\$BARRIDO_ID,\$ARTICULO_ID,\$BD){ 23 23 + \$conn = ABRIR_CONEXION_MYSQL(FALSE, \$BD); 24 24 \$result = 0; 25 25 if (\$conn){ 26 26 \$select = "SELECT FD.BARRIDO_DETALLE_ID FROM BARRIDO_DETALLE AS FD WHERE FD.BARRIDO_ID=\$BARRIDO_ID AND FD.ARTICULO_ID=\$ARTICULO_ID"; @@ -43,8 +43,8 @@ function BuscarBarridoDetalleID(\$BARRIDO_ID,\$ARTICULO_ID){ 43 43 return -1; 44 44 } 45 45 } 46 46 - function InsertarBarrido(\$FECHA,\$SUCURSAL_ID,\$USUARIO_CREACION,\$FECHA_HORA_CREACION,\$ARTICULO_ID,\$PRECIO/*,\$PRECIO_ARTICULO*/){ 47 47 - \$conn = ABRIR_CONEXION_MYSQL(FALSE); 46 46 + function InsertarBarrido(\$FECHA,\$SUCURSAL_ID,\$USUARIO_CREACION,\$FECHA_HORA_CREACION,\$ARTICULO_ID,\$PRECIO, \$BD/*,\$PRECIO_ARTICULO*/){ 50 50 \$BARRIDO_ID=BuscarBarridoID(\$FECHA,\$SUCURSAL_ID,\$conn); @@ -89,12 +89,15 @@ function InsertarBarrido(\$FECHA,\$SUCURSAL_ID,\$USUARIO_CREACION,\$FECHA_HORA_CREAC 89 89 )}); 90 90 //buscamos el detalle 91 91 \$BARRIDO_DETALLE_ID=0; 92 92 - \$BARRIDO_DETALLE_ID=BuscarBarridoDetalleID(\$BARRIDO_ID,\$ARTICULO_ID); 92 92 + \$BARRIDO_DETALLE_ID=BuscarBarridoDetalleID(\$BARRIDO_ID,\$ARTICULO_ID, \$BD); 93 93 //echo " Barridos detalle_id: ".\$BARRIDO_DETALLE_ID." "; 94 94 if(\$BARRIDO_DETALLE_ID==0){ 95 95 //Insertamos el detalle 96 96 \$query ="INSERT INTO BARRIDO_DETALLE (BARRIDO_ID,ARTICULO_ID,PRECIO/*,PRECIO_ARTICULO*/ )"; 97 97 - \$query .= " VALUES(\$BARRIDO_ID,\$ARTICULO_ID,\$PRECIO/*,'\$PRECIO_ARTICULO*' ); 97 97 + \$query =" VALUES(\$BARRIDO_ID,\$ARTICULO_ID,\$PRECIO)"; 98 98 + //query =" VALUES(\$BARRIDO_ID,\$ARTICULO_ID,\$PRECIO /*,\$PRECIO_ARTICULO */ )"; 99 99 + 100 100 + 98 101 //echo \$query; 99 102 if (mysql_query(\$conn, \$query)){ 100 103 \$result = \$BARRIDO_ID; @@ -120,8 +123,8 @@ function InsertarBarrido(\$FECHA,\$SUCURSAL_ID,\$USUARIO_CREACION,\$FECHA_HORA_CREAC 120 123 } 121 124 return \$result;

Ilustración 2. Cambios en el back de la aplicación (funciones de los web service).

Actividad	Descripción	Duración
<b>Actualización de módulos existentes.</b>	Se actualizaron los módulos existentes de la aplicación.	Dentro del periodo de 25 de marzo del 2024 al 17 de mayo del 2024.

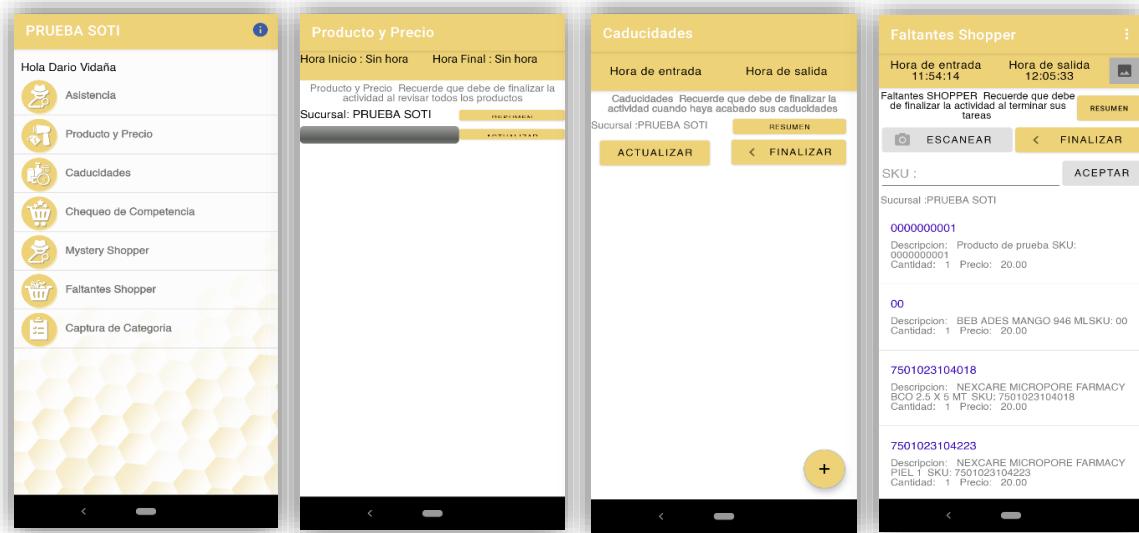


Ilustración 3. Actualización de módulos existentes.

Actividad	Descripción	Duración
<b>Corrección de errores.</b>	Se corrigieron errores reportados, como la duplicación de datos en listas.	Dentro del periodo de 25 de marzo del 2024 al 17 de mayo del 2024.

```

 2 usages + 1 more overridden
962     private void configMostrarDialogo(View customView, int position) {
963         AlertDialog.Builder builder = new AlertDialog.Builder( context: this);
964         builder.setView(customView)
965             .setTitle( arregloProductos[position].getBarcode_number())
966             .setMessage(crearMensajeProducto(arregloProductos[position]))
967             .setPositiveButton( text: "Aceptar", (dialog, which) -> maAceptar(customView, position))
968             .setNegativeButton( text: "Cancelar", (dialog, which) -> {
969                 listSol.clear();
970                 listSolFiltrada.clear();
971             })
972             .setCancelable(false)
973             .create() AlertDialog
974             .show();
975     }

```

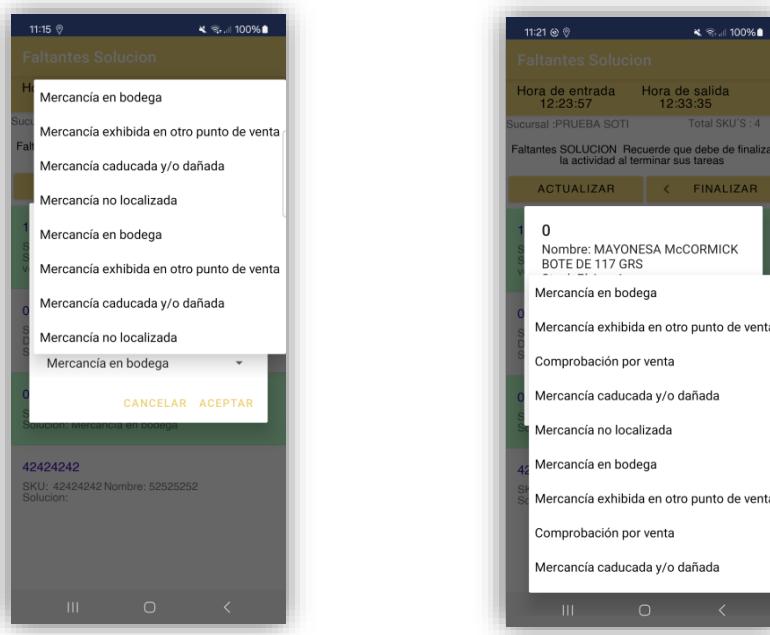


Ilustración 4. Corrección de errores.

Actividad	Descripción	Duración
<b>Optimización de la lógica de consulta y acceso a la base de datos.</b>	Se mejoró la lógica de consulta y acceso a la base de datos dentro del código de la aplicación.	Dentro del periodo de 25 de marzo del 2024 al 17 de mayo del 2024.

```

app\src\main\...\AGREGA_CADUCIDAD.java @@ -44,13 +44,14 @@ public class AGREGA_CADUCIDAD extends AppCompatActivity {
    private String usuario;
    // Variable que almacenará el id de la sucursal actual
    private String sucursal_id;
    public String URL = ""+DOMINIO;
    //public String URL = ""+DOMINIO;
    // Variable que almacena la hora y la envia por medio de una notificación
    public String nombreSucursal;
    private String userID,nombre;
    private boolean acabo = false;
    private String caducidadID = "0";
    private TextInputEditText editText;
    String NOMBRE_BD = "";
    private PermisosApp[] permisosReq = {
        new PermisosApp( android.Manifest.permission.CAMERA, "camara", true )
    };
    @@ -74,12 +75,15 @@ public class AGREGA_CADUCIDAD extends AppCompatActivity {
        nombre = extra.getStringExtra("nombre");
        userID = extra.getStringExtra("userID");
        caducidadID = extra.getStringExtra("caducidadID");
        URL = extra.getStringExtra("URL");
        URL = extra.getStringExtra("URL");
        acabo = extra.getBoolean("acabo");
        NOMBRE_BD = extra.getStringExtra("NOMBRE_BD");
        SharedPreferenceManager preferencesManager = SharedPreferenceManager.getInstance(getApplicationContext());
        URL = preferencesManager.getTxt("NOMBRE","ERROR");
        DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
        URL = preferencesManager.getTxt("DOMINIO","ERROR");
        /URL = preferencesManager.getTxt("DOMINIO","ERROR");
        /DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
        NOMBRE_BD = preferencesManager.getTxt("NOMBRE_BD","");
    }
}

```

Ilustración 5. Optimización de la lógica de consulta y acceso a la base de datos.

Actividad	Descripción	Duración
<b>Implementación e integración del módulo "Faltantes Solución".</b>	Se diseñó e implementó el nuevo módulo "Faltantes Solución" y otras funcionalidades.	Dentro del periodo de 25 de marzo del 2024 al 17 de mayo del 2024.

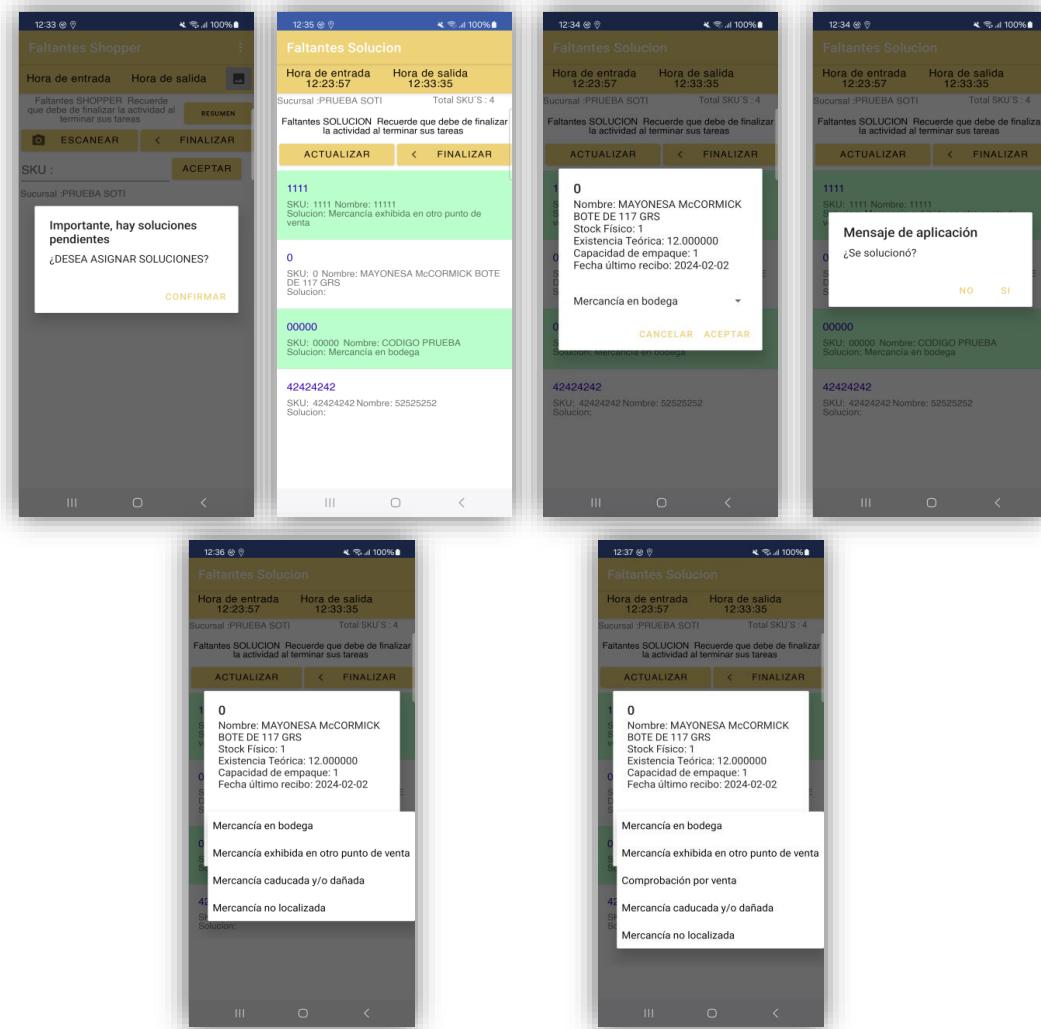


Ilustración 6. Implementación e integración del módulo "Faltantes Solución".

Actividad	Descripción	Duración
<b>Pruebas exhaustivas y ajustes basados en retroalimentación.</b>	Se realizaron pruebas exhaustivas y se ajustaron funcionalidades basadas en la retroalimentación.	Dentro del periodo de 25 de marzo del 2024 al 17 de mayo del 2024.
<b>Optimización de la aplicación.</b>	Se optimizó la aplicación, limpiando y depurando el código existente.	Dentro del periodo de 25 de marzo del 2024 al 17 de mayo del 2024.

Limpieza modulo caducidades

```
Se verifico su funcionamiento.
dario/dan1 ➜ 1fd158 ⌘ +0 -160
```

5 changed files

app\src\main\java\com\example\inmex_ver1\Caducidades\AGREGA_CADUCIDAD.java	@@ -69,18 +69,14 @@ public class AGREGA_CADUCIDAD extends AppCompatActivity { Intent intent2 = this.getIntent(); Bundle extra = intent2.getExtras(); //ID_USER = extra.getString("ID_USER"); sucursal_id = extra.getString("sucursal1"); usuario = extra.getString("usuario"); nombre = extra.getString("nombre"); userID = extra.getString("userID"); caducidad = extra.getString("caducidadID"); URL = extra.getString("URL"); acabo = extra.getBoolean("acabo"); NOMBRE_BD = extra.getString("NOMBRE_BD"); SharedPreferenceManager preferencesManager = SharedPreferenceManager.getInstance(getApplicationContext()); //URL = preferencesManager.getURL("WS","ERROR"); //DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR"); NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD","");     nombrresidual = extra.getString("sucursalsNombre"); 86 82 @@ -91,15 +87,11 @@ public class AGREGA_CADUCIDAD extends AppCompatActivity { sucursal_id = savedInstanceState.getString("sucursal1"); nombre = savedInstanceState.getString("nombre"); usuario = savedInstanceState.getString("usuario");
--	---

DEPURACION DE CODIGO

```
Se limpio el codigo de Producto precio, faltantes y captura categoria
RangelMauro ➜ 252bb09 ⌘ +16 -921
```

5 changed files

app\src\main\java\com\example\inmex_ver1\ProductoPrecio\A_PRODUCTO_PRECIO.java	@@ -102,22 +102,17 @@ public class A_PRODUCTO_PRECIO extends AppCompatActivity { * Aquí ligamos las variables antes creadas con su respectivo componente * Dev. Andres Gonzalez Cabrera */ //ID_USER = NService3.UsuarioIDGuardado; Archivo= fileList(); Intent intent = this.getIntent(); extra = intent.getExtras(); ID_USER = extra.getString("ID_USER"); sucursal_id = extra.getString("Sucursal_ID"); sucursaldNombre = extra.getString("sucursalNombre"); //Buscar(); resumen = findViewById(R.id.btnResumen); etiqueta = findViewById(R.id.tvEtiquetaPP); 116 114 //URL = extra.getString("URL"); NOMBRE_BD = extra.getString("NOMBRE_BD"); SharedPreferenceManager preferencesManager = SharedPreferenceManager.getInstance(getApplicationContext()); //URL = preferencesManager.getURL("WS","ERROR"); //DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
--	---

Ilustración 7. Optimización de la aplicación.

Actividad	Descripción	Duración
-----------	-------------	----------

<b>Documentación de nuevas funcionalidades y cambios realizados.</b>	Se documentaron las nuevas funcionalidades y los cambios realizados.	Dentro del periodo de 25 de marzo del 2024 al 17 de mayo del 2024.
--	--	--

```

52     /**
53      * Modulo A_FALTANTES_SOLUCION
54      * Esta clase contiene toda la logica del modulo A_FALTANTES_SOLUCION esta clase sirve para darles una
55      * solucion a los articulos faltantes
56      *
57      * @author Dev. Dario Vidaña ft. Dev. Mauro Rangel
58      */
1058    /**
1059     * Metodo que se ejecuta despues de que se cerro el dialog.
1060     * Lo que hace este metodo es actualizar la vista de la lista para que los cambios se vean
1061     * reflejados de inmediato.
1062     * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
1063     */
1064     * de una solucion en el sistema
1065     * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
1066     */
1067
1068 4 usages ▲ dariovidana1

```

Ilustración 8. Documentación de nuevas funcionalidades y cambios realizados.

Actividad	Descripción	Duración
<b>Cambios en la lógica de la pantalla de “bienvenido”.</b>	Se agregó nueva lógica para consulta de módulos según el parámetro que se le asigne al cliente, ya sea “parametro1” y “parametro2”.	Dentro del periodo de 25 de marzo del 2024 al 17 de mayo del 2024.

```

1045 +     }/*
1046 +
1047 +     private ArrayList<Entidad_Bienvenido> ValidarAccesos(String inte, String ext, SoapObject[] arreglo) {
1048 +
1049 +         try {
1050 +             //OcultarAccesos();
1051 +             if (inte.equals("in")) {
1052 +                 listItems.add(new Entidad_Bienvenido(R.drawable.asistencia, getString(R.string.Aistencia), ""));
1053 +                 //asistencia.setVisibility(View.VISIBLE);
1054 +                 listItems.add(new Entidad_Bienvenido(R.drawable.proveedores, getString(R.string.PlanPro), ""));
1055 +                 //planProv.setVisibility(View.VISIBLE);
1056 +                 listItems.add(new Entidad_Bienvenido(R.drawable.censo, getString(R.string.Censo), ""));
1057 +                 //censo.setVisibility(View.VISIBLE);
1058 +                 listItems.add(new Entidad_Bienvenido(R.drawable.gap, getString(R.string.GAP), ""));

```

```

1059 +
1060     //gap.setVisibility(View.VISIBLE);
1061 }
1062 +
1063 //listItems.add(new Entidad_Bienvenido(R.drawable.faltantes,getString( R.string.Faltantes),"FALTANTES"));
1064 if (ext.equals("Ee")) {
1065     listItems.add(new Entidad_Bienvenido(R.drawable.caducidad, getString(R.string.Caducidad), "CADUCIDADES"));
1066     //caducidades.setVisibility(View.VISIBLE);
1067     listItems.add(new Entidad_Bienvenido(R.drawable.guia, getString(R.string.GUIA), ""));
1068     //guia.setVisibility(View.VISIBLE);
1069     listItems.add(new Entidad_Bienvenido(R.drawable.planograma, getString(R.string.Planograma), ""));
1070 }
1071 }
1072 +
1073 //Llamada a getParametro
1074 getParametro();
1075 listItems = new ArrayList<>();
1076 if (parametro.equals("S")) {
1077     if (!HORA_ENTRADA.equals("")) {
1078         //Creamos una lista para almacenar los elementos no nulos
1079         List<SoapObject> lista = new ArrayList<>();
1080         //Iteramos sobre el arreglo y agregamos los elementos no nulos a la lista
1081         for (SoapObject elemento : arreglo) {
1082             if (elemento != null) {
1083                 lista.add(elemento);
1084             }
1085         }
1086         if (recursividad == 0) {
1087             if (lista.size() == 0) {
1088                 recursividad++;
1089                 arreglo = buscarModulos();
1090                 ValidarAccesos(inte, ext, arreglo);
1091             }
1092         }
1093         if (lista.size() > 1) {
1094             for (int i = 0; i < lista.size(); i++) {
1095                 switch (lista.get(i).getProperty("CLAVE").toString()) {
1096                     case "AA": {
1097                         listItems.add(new Entidad_Bienvenido(R.drawable.mystery, getString(R.string.asistencia), "ASISTENCIA"));
1098                         break;
1099                     }
1100                     case "PYP": {
1101                         listItems.add(new Entidad_Bienvenido(R.drawable.productoprecio, getString(R.string.PYPrecio), "PYP"));
1102                         break;
1103                     }
1104                     case "MS": {
1105                         listItems.add(new Entidad_Bienvenido(R.drawable.mystery, getString(R.string.MysteryShopper), "MS"));
1106                         break;
1107                     }
1108                     case "FS": {
1109                         listItems.add(new Entidad_Bienvenido(R.drawable.faltantes, getString(R.string.Faltantes), "FALTANTES"));
1110                         break;
1111                     }
1112                     case "CAD": {
1113                         listItems.add(new Entidad_Bienvenido(R.drawable.caducidad, getString(R.string.Caducidad), "CADUCIDADES"));
1114                         break;
1115                     }
1116                     case "CHC": {
1117                         listItems.add(new Entidad_Bienvenido(R.drawable.chequeo, getString(R.string.ChequeoCompetencia), "CHEQUEO"));
1118                         break;
1119                     }
1120                     case "CC": {
1121                         listItems.add(new Entidad_Bienvenido(R.drawable.categorias, getString(R.string.Categoría), "CATEGORIA"));
1122                         break;
1123                     }
1124                 }
1125             }
1126         }
1127     }
}

```

```

1128 +         }
1129 +         //si no tiene hora muestra solo el de asistencia
1130 +         if (HORA_ENTRADA.equals("")) {
1131 +             listItems.add(new Entidad_Bienvenido(R.drawable.mystery, getString(R.string.asistencia), "ASISTENCIA"));
1132 +         }
1133 +
1134 +     }
1135 +
1136 +     if (parametro.equals("N")) {
1137 +
1138 +         List<SoapObject> lista2 = new ArrayList<>();
1139 +         //Iteramos sobre el arreglo y agregamos los elementos no nulos a la lista
1140 +         for (SoapObject elemento : arreglo) {
1141 +             if (elemento != null) {
1142 +                 lista2.add(elemento);
1143 +             }
1144 +         }
1145 +         if (recursividad == 0) {
1146 +             if (lista2.size() == 0) {
1147 +                 recursividad++;
1148 +                 arreglo = buscarModulos();
1149 +                 ValidarAccesos(inte, ext, arreglo);
1150 +             }
1151 +         }
1152 +
1153 +         if (lista2.size() > 1) {
1154 +             for (int i = 0; i < lista2.size(); i++) {
1155 +
1156 +                 switch (lista2.get(i).getProperty("CLAVE").toString()) {
1157 +                     case "AA": {
1158 +                         listItems.add(new Entidad_Bienvenido(R.drawable.mystery, getString(R.string.asistencia), "ASISTENCIA"));
1159 +                         break;
1160 +                     }
1161 +                     case "PYP": {
1162 +                         listItems.add(new Entidad_Bienvenido(R.drawable.productoprecio, getString(R.string.PYPrecio), "PYP"));
1163 +                         break;
1164 +                     }
1165 +                     case "MS": {
1166 +                         listItems.add(new Entidad_Bienvenido(R.drawable.mystery, getString(R.string.Mysteryshopper), "MS"));
1167 +                         break;
1168 +                     }
1169 +                     case "FS": {
1170 +                         listItems.add(new Entidad_Bienvenido(R.drawable.faltantes, getString(R.string.Faltantes), "FALTANTES"));
1171 +                         break;
1172 +                     }
1173 +                     case "CAD": {
1174 +                         listItems.add(new Entidad_Bienvenido(R.drawable.caducidad, getString(R.string.Caducidad), "CADUCIDADES"));
1175 +                         break;
1176 +                     }
1177 +                     case "CHC": {
1178 +                         listItems.add(new Entidad_Bienvenido(R.drawable.chequeo, getString(R.string.ChequeoCompetencia), "CHEQUEO"));
1179 +                         break;
1180 +                     }
1181 +                     case "CC": {
1182 +                         listItems.add(new Entidad_Bienvenido(R.drawable.categorias, getString(R.string.Categoría), "CATEGORIA"));
1183 +                         break;
1184 +                     }
1185 +                 }
1186 +             }
1187 +         }
1188 +     }
1189 + } catch (Exception ex) {
1190 +     Toast.makeText(this, ex.getMessage(), Toast.LENGTH_LONG).show();
1191 + }
1192 + return listItems;
1193 +

```

*Ilustración 9. Cambios en la lógica de la pantalla de “bienvenido”.*

### 3.2.4. Pruebas y validación

Actividad	Descripción	Duración
<b>Uso de emuladores y dispositivos físicos para garantizar la compatibilidad.</b>	Se probaron emuladores y diferentes dispositivos físicos para asegurar la compatibilidad.	Dentro del periodo del 18 de mayo del 2024 al 31 de mayo del 2024.

Tabla 4. Pruebas y validación.

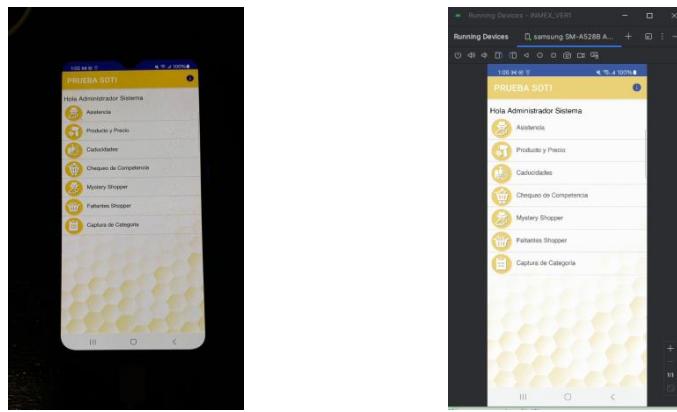


Ilustración 10. Uso de emuladores y dispositivos físicos para garantizar la compatibilidad.

Actividad	Descripción	Duración
<b>Validación de la correcta interacción entre la aplicación y los web service.</b>	Se validó la correcta interacción entre la aplicación y los servicios web haciendo múltiples pruebas.	Dentro del periodo del 18 de mayo del 2024 al 31 de mayo del 2024.

Request 1

```

<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soapenv:Header>
    <ser:MostrarFaltantesPendiente soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <@xsd:type="xsd:string">4258</xsd:type>
    </ser:MostrarFaltantesPendiente>
  </soapenv:Header>
  <soapenv:Body>
    <ser:MostrarFaltantesPendiente soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <@xsd:type="xsd:string">4258</xsd:type>
      <@xsd:type="xsd:int">33</xsd:type>
      <@xsd:type="xsd:string">soticomms_VIDO_SOTI</xsd:type>
    </ser:MostrarFaltantesPendiente>
  </soapenv:Body>
</soapenv:Envelope>

```

```

<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <@ns1:MostrarFaltantesPendienteResponse xmlns:ns1="http://webproviders.mx/soap/Servicios17">
    <@return xsi:type="SOAP-ENC:Array" SOAP-ENC:arrayType="tns:MostrarSolucion[1]">
      <item xsi:type="tns:MostrarSolucion">
        <@xsd:type="xsd:int">4258</xsd:type>
        <@xsd:type="xsd:int">278815</xsd:type>
        <@xsd:type="xsd:int">27143</xsd:type>
        <@xsd:type="xsd:string">7501023104025</xsd:type>
        <@xsd:type="xsd:string">MEXICO</xsd:type>
        <@xsd:type="xsd:string">MEXICO</xsd:type>
        <@xsd:type="xsd:string">NOMBRE</xsd:type>
        <@xsd:type="xsd:string">DESCRICION</xsd:type>
        <@xsd:type="xsd:string">STOCK_FISICO</xsd:type>
        <@xsd:type="xsd:string">PRECIO_ARTICULO</xsd:type>
        <@xsd:type="xsd:string">IMAGEN</xsd:type>
        <@xsd:type="xsd:string">soticomms_articulos/7501023104025.jpeg</xsd:type>
        <@xsd:type="xsd:string">SOLUCION</xsd:type>
        <@xsd:type="xsd:string">2024-04-22</xsd:type>
        <@xsd:type="xsd:string">EXISTENCIA_TEORICA</xsd:type>
        <@xsd:type="xsd:string">FECHA_ULT_RECEIBO</xsd:type>
        <@xsd:type="xsd:string">CAPACIDAD_EMPAQUE</xsd:type>
        <@xsd:type="xsd:string">12</xsd:type>
      </item>
      <item xsi:type="tns:MostrarSolucion">
        <@xsd:type="xsd:int">4258</xsd:type>
        <@xsd:type="xsd:int">278815</xsd:type>
        <@xsd:type="xsd:int">27143</xsd:type>
        <@xsd:type="xsd:string">7501023104025</xsd:type>
        <@xsd:type="xsd:string">MEXICO</xsd:type>
        <@xsd:type="xsd:string">MEXICO</xsd:type>
        <@xsd:type="xsd:string">NOMBRE</xsd:type>
        <@xsd:type="xsd:string">DESCRICION</xsd:type>
        <@xsd:type="xsd:string">STOCK_FISICO</xsd:type>
        <@xsd:type="xsd:string">PRECIO_ARTICULO</xsd:type>
        <@xsd:type="xsd:string">IMAGEN</xsd:type>
        <@xsd:type="xsd:string">soticomms_articulos/7501023104025.jpeg</xsd:type>
        <@xsd:type="xsd:string">SOLUCION</xsd:type>
        <@xsd:type="xsd:string">2024-04-22</xsd:type>
        <@xsd:type="xsd:string">EXISTENCIA_TEORICA</xsd:type>
        <@xsd:type="xsd:string">FECHA_ULT_RECEIBO</xsd:type>
        <@xsd:type="xsd:string">CAPACIDAD_EMPAQUE</xsd:type>
        <@xsd:type="xsd:string">12</xsd:type>
      </item>
    </return>
  </ns1:MostrarFaltantesPendienteResponse>
</SOAP-ENV:Envelope>

```

```

472  /**
473   * Metodo MostrarFaltantesPendientes
474   * Este metodo manda a llamar al webService MostrarFaltantes el cual trae toda la informacion del articulo
475   * insertado en la tabla faltantes con la fecha de hoy
476   */
477  public void MostrarFaltantesPendientes() {
478      SharedPreferencesManager preferencesManager = SharedPreferencesManager.getInstance(getApplicationContext());
479      NOMBRE_BD = preferencesManager.getNOMBRE_BD(variableKey: "NOMBRE_BD", defaultValue: "");
480      if (!NOMBRE_BD.equals("")) {
481          WService3 cs = new WService3(func: "MostrarFaltantesPendiente");
482          cs.FALTANTES_ID = faltantes_id;
483          cs.SUCURSAL_ID = sucursal_id;
484          cs.NOMBRE_BD = NOMBRE_BD;
485          cs.start();
486          try {
487              // cs.wait(2000);
488              cs.join();
489              //
490          } catch (InterruptedException e) {
491              throw new RuntimeException(e);
492          }
493          System.out.println(cs.arreglo.length);
494          if (cs.arreglo.length > 0) {
495              CantidadProductos = new int[cs.arreglo.length - 1];
496              Integerproducto = new int[cs.arreglo.length - 1];
497              ArregloProductos = new Products[cs.arreglo.length - 1];
498          }
499      }

```

Ilustración 11. Validación de la correcta interacción entre la aplicación y los web service.

Actividad	Descripción	Duración
<b>Pruebas de usuario final.</b>	Se realizaron pruebas con usuarios finales y se hicieron ajustes basados en su retroalimentación.	Dentro del periodo del 18 de mayo del 2024 al 31 de mayo del 2024.
<b>Realización de ajustes y correcciones basadas en la retroalimentación recibida.</b>		

### 3.2.5. Documentación y capacitación

Actividad	Descripción	Duración
<b>Elaboración de manuales y guías de usuario.</b>	Se elaboraron manuales y guías para los usuarios.	Dentro del periodo del 01 de junio del 2024 tentativamente al 12 de junio del 2024
<b>Documentación detallada de las nuevas funcionalidades y mejoras.</b>	Se documentaron detalladamente las nuevas funcionalidades y mejoras.	Dentro del periodo del 01 de junio del 2024 al 12 de junio del 2024.
<b>Capacitación a usuarios y equipo de soporte.</b>	Se realizaron sesiones de capacitación para asegurar el correcto uso y mantenimiento de la aplicación.	Dentro del periodo del 01 de junio del 2024 al 12 de junio del 2024.

Tabla 5. Documentación y capacitación.

## Capítulo 4. Resultados obtenidos

En este capítulo se presentan los resultados obtenidos tras la implementación de las mejoras en la aplicación "VIDO". Se detallará cómo las modificaciones han impactado la funcionalidad, la eficiencia operativa y la experiencia del usuario, con énfasis en la resolución de problemas técnicos previamente identificados y la optimización del rendimiento del sistema.

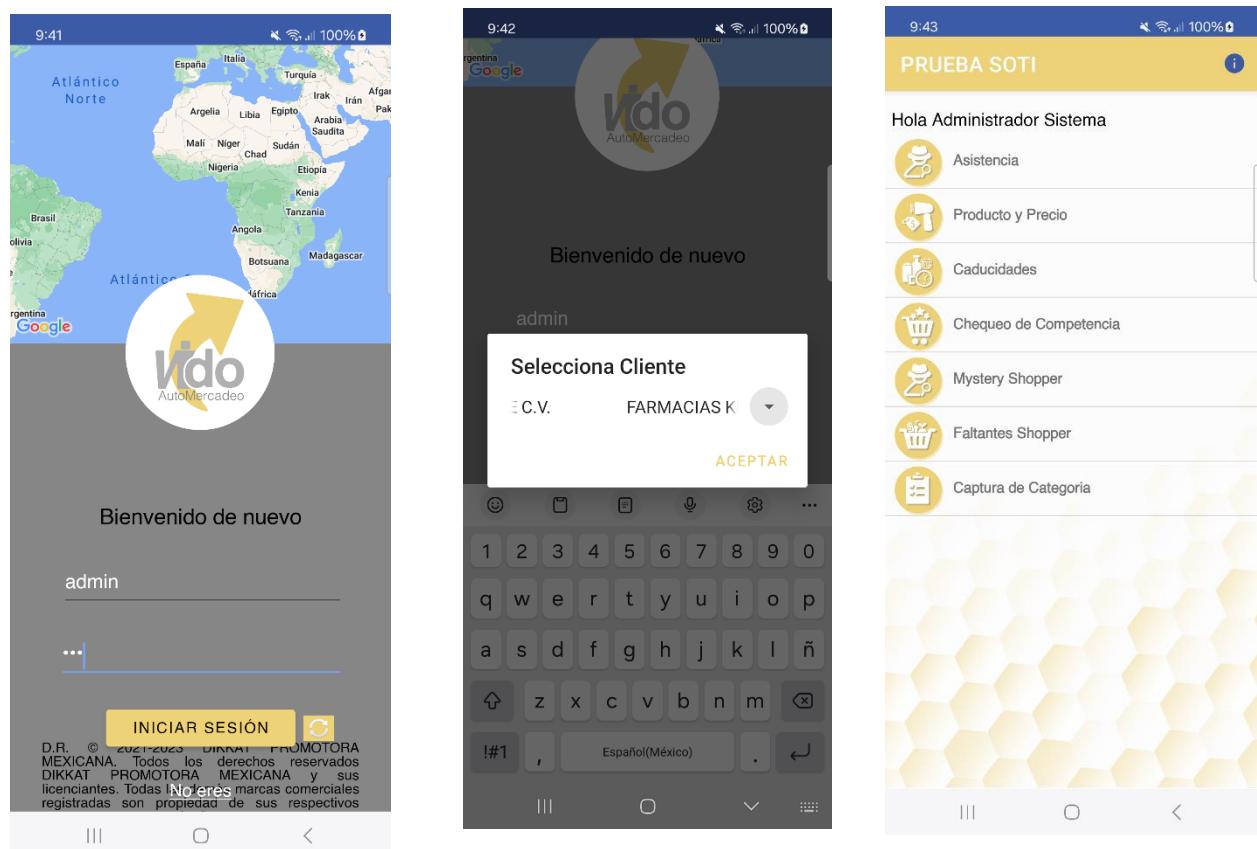


Ilustración 12. Resultados obtenidos.

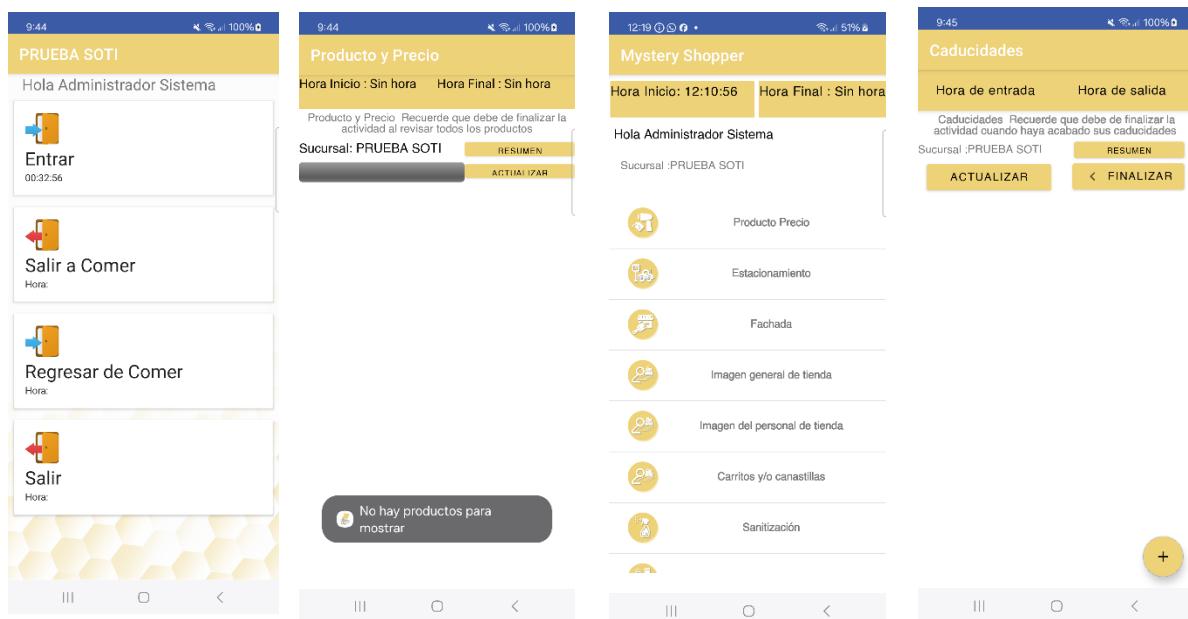


Ilustración 13. Resultados obtenidos..

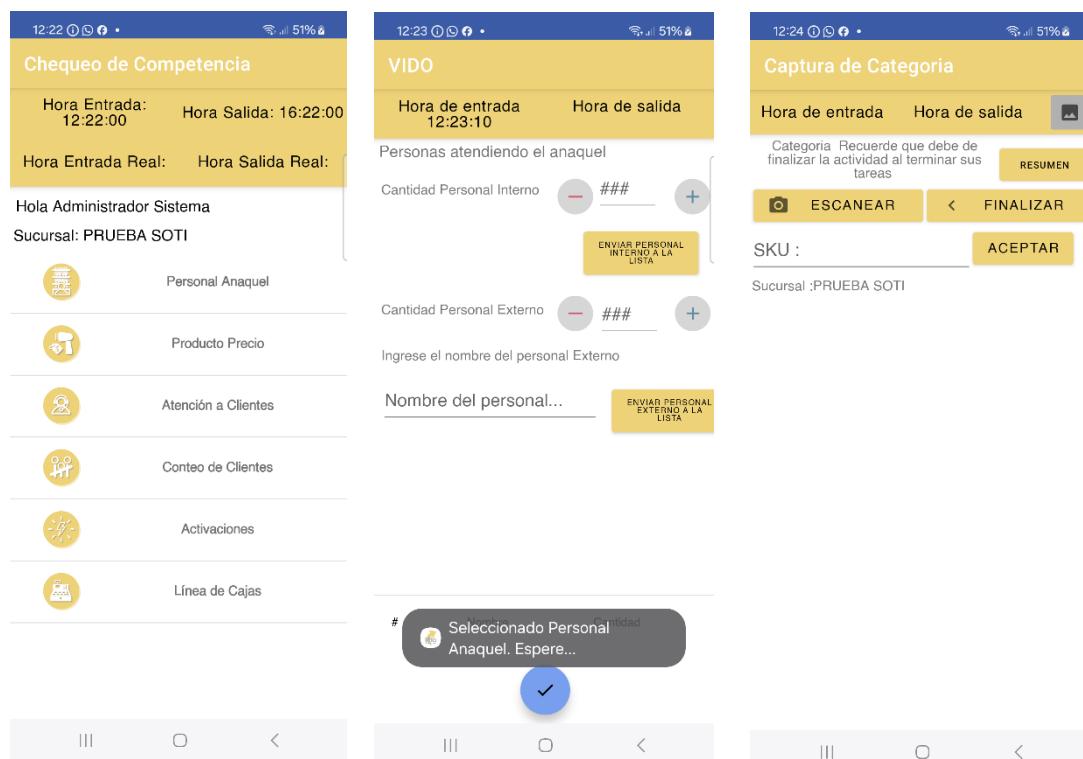


Ilustración 14. Resultados obtenidos.

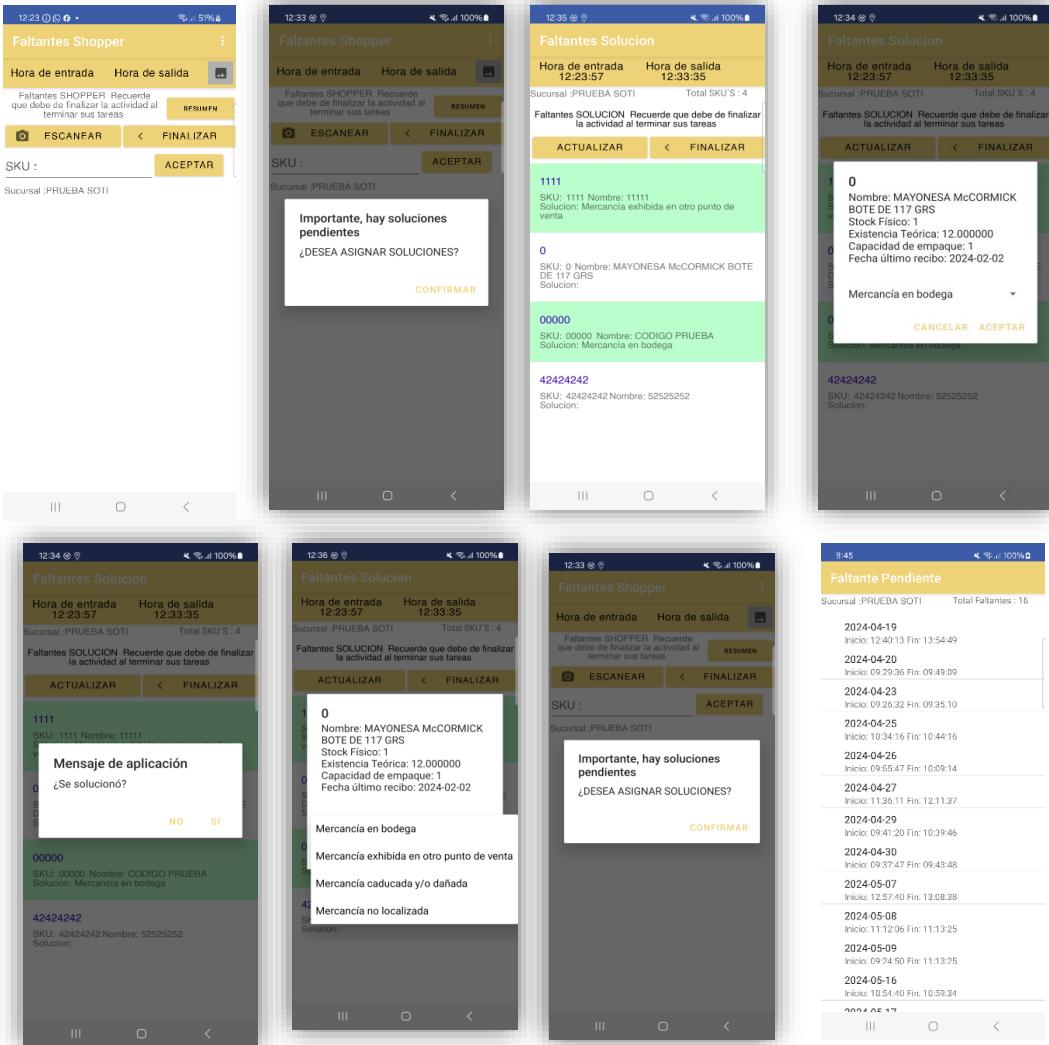


Ilustración 15. Resultados obtenidos.

## **Conclusiones**

La experiencia adquirida durante el desarrollo de VIDO en la empresa SOTI ha sido de un gran valor, tanto personal como laboralmente. A lo largo de mi estancia en esta empresa se cumplieron con cada uno de los objetivos establecidos, logrando mejorar la aplicación y darle el giro requerido y solicitado. Esta experiencia me ha permitido poner en práctica conocimientos técnicos adquiridos durante mi carrera profesional, así como poder desarrollar nueva habilidades en un entorno real.

Durante mi formación académica adquirí diversos conocimientos, los cuales fueron de gran ayuda en mi estancia en la empresa SOTI. Entre ellos destacan el uso de tecnologías y herramientas de desarrollo, tales como Android Studio y Visual Studio Code como entorno de desarrollo; Git Hub para la unión, sincronización y control de versiones; Java y XML como lenguajes de programación principales, entre otros. No obstante, durante mi estancia también me enfrenté a retos, pues se presentaron un par de tecnologías de las cuales no tenía conocimiento, entre ellas destacan: SOAP UI, para usar y administrar diferentes Web Service; Wamp Server, utilizado para simular un servicio web de forma local, entre otros. En conjunto, la utilización de estas herramientas fue sumamente importante para el desarrollo y mantenimiento de la aplicación, asegurando una correcta integración y funcionamiento de los diferentes componentes del sistema.

Dentro del proyecto VIDO, la integración del módulo de Faltantes Solución ha sido uno de los más desafiantes y gratificantes del proyecto. La implementación de esta funcionalidad no solo ha mejorado la operatividad de la aplicación, sino que también ha demostrado la capacidad de adaptarse a las necesidades del mercado y a lo que el cliente requiera.

Por otro lado, la experiencia como residente en SOTI también me fue de gran ayuda para desarrollar otro tipo de habilidades, por ejemplo: el trabajo en equipo, la comunicación, gestión del tiempo y la capacidad de resolver problemas de manera eficiente. La colaboración con el equipo de desarrollo móvil y web fue crucial para poder llevar a cabo

exitosamente el proyecto, permitiendo una retroalimentación constante y la implementación de mejoras continuas.

Finalmente, cabe mencionar que el proyecto de VIDO ha sido una experiencia enriquecedora que ha contribuido significativamente mi crecimiento personal y profesional. Los conocimientos y habilidades adquiridos durante mi residencia profesional serán de gran utilidad en futuras oportunidades laborales e incluso personales. Estoy agradecido enormemente por esta oportunidad y por la confianza para llevar a cabo este proyecto, y confío plenamente en que las mejoras implementadas en VIDO continuarán aportando valor a la empresa y a sus clientes.

## **Competencias:**

### **1. Análisis y Diagnóstico Inicial:**

- **Competencia de análisis crítico:** Se evaluó la funcionalidad y estructura de la aplicación "VIDO", identificando áreas de mejora y problemas reportados.
- **Diagnóstico técnico:** Se analizó la estructura de la base de datos y la arquitectura de la aplicación para identificar posibles mejoras.

### **2. Planificación de Actualizaciones y Nuevas Funcionalidades:**

- **Planificación estratégica:** Se definieron objetivos específicos y tareas a realizar, se elaboró un backlog con mejoras y nuevas funcionalidades, y se priorizaron tareas en colaboración con el equipo de desarrollo.
- **Gestión de proyectos:** Se estableció un camino claro y organizado para implementar cambios, asegurando una coordinación efectiva del proyecto.

### **3. Desarrollo e Implementación:**

- **Programación avanzada:** Se implementaron cambios en el backend de la aplicación, se desarrollaron e integraron nuevos módulos, y se optimizó la lógica de consulta y acceso a la base de datos.
- **Resolución de problemas:** Se corrigieron errores reportados, se mejoró la lógica de la aplicación y se realizaron pruebas exhaustivas para garantizar la funcionalidad correcta.
- **Creatividad e innovación:** Se diseñó e implementó el nuevo módulo "Faltantes Solución", mostrando una capacidad de adaptación a las necesidades del mercado y del cliente.

### **4. Pruebas y Validación:**

- **Calidad y control:** Se realizaron pruebas exhaustivas utilizando emuladores y dispositivos físicos para garantizar la compatibilidad, y se validó la correcta interacción entre la aplicación y los servicios web.
- **Mejora continua:** Se hicieron ajustes y correcciones basadas en la retroalimentación de pruebas de usuario final, asegurando una experiencia de usuario optimizada.

### **5. Documentación y Capacitación:**

- **Documentación técnica:** Se elaboraron manuales y guías para usuarios, y se documentaron detalladamente las nuevas funcionalidades y mejoras.
- **Capacitación:** Se realizaron sesiones de capacitación para asegurar el correcto uso y mantenimiento de la aplicación, facilitando la adopción de la nueva versión de la aplicación por parte de los usuarios.

Estas competencias permitieron cumplir con los objetivos establecidos para la residencia y adquirir habilidades valiosas para el desarrollo profesional en el campo de la ingeniería de sistemas computacionales.

## Referencias bibliográficas

- Amazon Web Services. (s. f.). ¿Qué es XML? <https://aws.amazon.com/es/what-is/xml/>
- Android Developers. (s. f.). Android Studio. Google. <https://developer.android.com/studio?hl=es-419>
- Android Developers. (s. f.). Diálogos | Desarrolladores de Android. <https://developer.android.com/develop/ui/views/components/dialogs?hl=es-419>
- Android Developers. (s. f.). Guías para desarrolladores. <https://developer.android.com/guide?hl=es-419>
- GitHub Docs. (s. f.). Creating a new repository - GitHub Docs. <https://docs.github.com/en/repositories/creating-and-managing-repositories/creating-a-new-repository>
- IBM. (s. f.). Ventajas de Java. <https://www.ibm.com/docs/es/aix/7.3?topic=monitoring-advantages-java>
- Microsoft. (s. f.). Visual Studio Code. [https://code.visualstudio.com/?wt.mc\\_id=DX\\_841432](https://code.visualstudio.com/?wt.mc_id=DX_841432)
- Moebio. (s. f.). Revista de epistemología de ciencias sociales. <https://www.moebio.uchile.cl/03/frprinci.html>
- Oracle. (s. f.). Java. <https://www.java.com/es/>
- SmartBear Software. (s. f.). Sample SOAP project in SoapUI. SoapUI. <https://www.soapui.org/resources/tutorials/soap-sample-project/>
- Stack Overflow. (2010). How do I display an alert dialog on Android? <https://stackoverflow.com/questions/2115758/how-do-i-display-an-alert-dialog-on-android>
- WampServer. (s. f.). WampServer, la plataforma de desarrollo web en Windows - Apache, MySQL, PHP. <https://wampserver.aviaTechno.net/>

## Anexos

### Código: módulo selección de BD.

```
@Override
public void onResume() {
    super.onResume();
    // Recuperar el valor de la variable dinámica al reanudar la actividad

    SharedPreferencesManager preferencesManager =
    SharedPreferencesManager.getInstance(getApplicationContext());
    if (nombre_bd.isEmpty())
        nombre_bd = preferencesManager.getNOMBRE_BD("NOMBRE_BD", "");

    if (es_multi.equals("true")) {
        SeleccionarBD(usuario.getText().toString().trim(),
        pass.getText().toString().trim(), true);
    } else {
        ActualizaSucursales(false);
        Noeres.setVisibility(View.VISIBLE);
        act.setVisibility(View.VISIBLE);
        try {
            nombreUsuario = getNombreUsuario();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    try {
        validarSucursalConPass(true);
        // checarSucursalesChequeoComp(true);
    } catch (InterruptedException | IOException e) {
        e.printStackTrace();
    }

    //El boton siempre se ejecutara con el metodo de lectura de datos
```

```

biometricos

    login.setOnClickListener((view) -> {
        enableUserLocation();
        if (nombre_bd.isEmpty()) {
            SeleccionarBD(usuario.getText().toString().trim(),
pass.getText().toString().trim(), true);
        } else {
            try {
                login.setEnabled(false);
                validarSucursal(false);
            } catch (InterruptedException | IOException e) {
                login.setEnabled(true);
                e.getMessage();
            }
            LoginBiometrico();
            login.setEnabled(true);
        }
    });
}

private void MostrarDialogListDB(boolean BIOMETRICO) {
    // Inflar el diseño personalizado
    View customView =
LayoutInflater.from(this).inflate(R.layout.custom_spinner_dialog, null);

    // Obtener una referencia al Spinner en el diseño personalizado
    Spinner spinner = customView.findViewById(R.id.spinner);

    // Crear un ArrayAdapter con los nombres de las ubicaciones
    ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
android.R.layout.simple_spinner_item,

listDBS.stream().map(ListDB::getNombre).collect(Collectors.toList()));
}

```

```

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)
;

// Configurar el Spinner con el ArrayAdapter
spinner.setAdapter(adapter);

// Crear el AlertDialog con el diseño personalizado
AlertDialog.Builder builder = new AlertDialog.Builder(this);
builder.setView(customView);
builder.setTitle("Selecciona Cliente");

builder.setPositiveButton("Aceptar", new DialogInterface.OnClickListener()
{
    @Override

    public void onClick(DialogInterface dialog, int which) {
        // Obtener la opción seleccionada del Spinner
        String opcionSeleccionada = (String) spinner.getSelectedItem();
        int selectedPosition = spinner.getSelectedItemPosition();

        if (selectedPosition != AdapterView.INVALID_POSITION) {
            ListDB ubicacionSeleccionada = listDBS.get(selectedPosition);
            cliente_id = ubicacionSeleccionada.getId();
            nombre_bd = ubicacionSeleccionada.getNombreBD();
            NOMBRE_SUCURSAL = ubicacionSeleccionada.getNombre();
            //new GeofenceBroadcastReceiver();
            GeofenceBroadcastReceiver.Nombre_BD = nombre_bd;

            SharedPreferencesManager preferencesManager =
            SharedPreferencesManager.getInstance(getApplicationContext());
            preferencesManager.saveNOMBRE_BD("NOMBRE_BD", nombre_bd);
            preferencesManager.saveNEsMulti("MULTI", "true");

            preferencesManager.saveNOMBRE_BD("NOMBRE_SUCURSAL", NOMBRE_SUCURSAL);
        }
    }
});

```

```

    if (!BIOMETRICO) {
        try {
            LoginConPass();
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
    } else {

        ActualizaSucursales(false);
        Noeres.setVisibility(View.VISIBLE);
        act.setVisibility(View.VISIBLE);
        try {
            nombreUsuario = getNombreUsuario();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        try {
            validarSucursalConPass(true);
        } catch (InterruptedException | IOException e) {
            e.printStackTrace();
        }

        //El boton siempre se ejecutara con el metodo de lectura
        de datos biometricos
        login.setOnClickListener((view) -> {
            enableUserLocation();
            try {
                login.setEnabled(false);
                validarSucursal(false);
                //checlarSucursalesChequeoComp(false);
            } catch (InterruptedException | IOException e) {
                login.setEnabled(true);
                e.getMessage();
            }
            LoginBiometrico();
            login.setEnabled(true);
        });
    }
}

```

```

        } ) ;

    }

    dialog.dismiss(); // Cierra el diálogo
    listDBS.clear();
    isDialogOpen = false;
}
) .setCancelable(false);

// Mostrar el AlertDialog
builder.create() .show();
}

```

## Código: módulo BIENVENIDO (nueva lógica).

```

private boolean getParametro() throws InterruptedException {
    SharedPreferencesManager preferencesManager =
    SharedPreferencesManager.getInstance(getApplicationContext());
    NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD", "");

    WService cs = new WService("getParametros");
    cs.context=context;
    cs.NOMBRE = nombre;
    cs.MODULO = modulo;
    cs.NOMBRE_BD = NOMBRE_BD;
    cs.start();
    cs.join();
    parametro = cs.RESULT;

    return false;
}

/**

```

```

* Metodo validarAccesos
* Este metodo lo que hace es verificar que tipo de usuario es el ingresado
(interno o externo)
* y dependiendo de que tipo de usuario seran las opciones que este podra
visualizar
**/

private ArrayList<Entidad_Bienvenido> ValidarAccesos(String inte, String ext,
SoapObject[] arreglo) {

    try {
        //OcultarAccesos();
        if (inte.equals("In")) {
            listItems.add(new Entidad_Bienvenido(R.drawable.asistencia,
getString(R.string.Asistencia), ""));
            listItems.add(new Entidad_Bienvenido(R.drawable.proveedores,
getString(R.string.PlanPro), ""));
            listItems.add(new Entidad_Bienvenido(R.drawable.censo,
getString(R.string.Censo), ""));
            listItems.add(new Entidad_Bienvenido(R.drawable.gap,
getString(R.string.GAP), ""));
        }

        //listItems.add(new Entidad_Bienvenido(R.drawable.faltantes,getString(
R.string.Faltantes),"FALTANTES"));
        if (ext.equals("Ee")) {
            listItems.add(new Entidad_Bienvenido(R.drawable.caducidad,
getString(R.string.Caducidad), "CADUCIDADES"));
            listItems.add(new Entidad_Bienvenido(R.drawable.guia,
getString(R.string.GUIA), ""));
            listItems.add(new Entidad_Bienvenido(R.drawable.planograma,
getString(R.string.Planograma), ""));
        }
    }
}
```

```

    }

    //Llamada a getParametro
    getParametro();
    listItems = new ArrayList<>();
    if (parametro.equals("S")) {
        if (!HORA_ENTRADA.equals("")) {
            //Creamos una lista para almacenar los elementos no nulos
            List<SoapObject> lista = new ArrayList<>();
            //Iteramos sobre el arreglo y agregamos los elementos no nulos
            a la lista
            for (SoapObject elemento : arreglo) {
                if (elemento != null) {
                    lista.add(elemento);
                }
            }
            if (recursividad == 0) {
                if (lista.size() == 0) {
                    recursividad++;
                    arreglo = buscarModulos();
                    ValidarAccesos(inte, ext, arreglo);
                }
            }
            if (lista.size() > 1) {
                for (int i = 0; i < lista.size(); i++) {

                    switch (lista.get(i).getProperty("CLAVE").toString())
                {
                    case "AA": {
                        listItems.add(new
Entidad_Bienvenido(R.drawable.mystery, getString(R.string.asistencia),
"ASISTENCIA"));
                        break;
                    }
                    case "PYP": {
                        listItems.add(new

```

```
Entidad_Bienvenido(R.drawable.productoprecio, getString(R.string.PYPrecio),  
"PYP"));  
        break;  
    }  
    case "MS": {  
        listItems.add(new  
Entidad_Bienvenido(R.drawable.mystery, getString(R.string.MysteryShopper),  
"MS"));  
        break;  
    }  
    case "FS": {  
        listItems.add(new  
Entidad_Bienvenido(R.drawable.faltantes, getString(R.string.Faltantes),  
"FALTANTES"));  
        break;  
    }  
    case "CAD": {  
        listItems.add(new  
Entidad_Bienvenido(R.drawable.caducidad, getString(R.string.Caducidad),  
"CADUCIDADES"));  
        break;  
    }  
    case "CHC": {  
        listItems.add(new  
Entidad_Bienvenido(R.drawable.chequeo, getString(R.string.ChequeoCompetencia),  
"CHEQUEO"));  
        break;  
    }  
    case "CC": {  
        listItems.add(new  
Entidad_Bienvenido(R.drawable.categorias, getString(R.string.Categoría),  
"CATEGORIA"));  
        break;  
    }  
}
```

```

        }
    }

    //si no tiene hora muestra solo el de asistencia
    if (HORA_ENTRADA.equals("")) {
        listItems.add(new Entidad_Bienvenido(R.drawable.mystery,
getString(R.string.asistencia), "ASISTENCIA"));
    }

}

if (parametro.equals("N")) {

    List<SoapObject> lista2 = new ArrayList<>();
    //Iteramos sobre el arreglo y agregamos los elementos no nulos a
la lista
    for (SoapObject elemento : arreglo) {
        if (elemento != null) {
            lista2.add(elemento);
        }
    }
    if (recursividad == 0) {
        if (lista2.size() == 0) {
            recursividad++;
            arreglo = buscarModulos();
            ValidarAccesos(inte, ext, arreglo);
        }
    }

    if (lista2.size() > 1) {
        for (int i = 0; i < lista2.size(); i++) {

            switch (lista2.get(i).getProperty("CLAVE").toString()) {
                case "AA": {
                    listItems.add(new
Entidad_Bienvenido(R.drawable.mystery, getString(R.string.asistencia),
"ASISTENCIA"));
                }
            }
        }
    }
}

```

```

        break;
    }
    case "PYP": {
        listItems.add(new
Entidad_Bienvenido(R.drawable.productoprecio, getString(R.string.PYPrecio),
"PYP"));
        break;
    }
    case "MS": {
        listItems.add(new
Entidad_Bienvenido(R.drawable.mystery, getString(R.string.MysteryShopper),
"MS"));
        break;
    }
    case "FS": {
        listItems.add(new
Entidad_Bienvenido(R.drawable.faltantes, getString(R.string.Faltantes),
"FALTANTES"));
        break;
    }
    case "CAD": {
        listItems.add(new
Entidad_Bienvenido(R.drawable.caducidad, getString(R.string.Caducidad),
"CADUCIDADES"));
        break;
    }
    case "CHC": {
        listItems.add(new
Entidad_Bienvenido(R.drawable.chequeo, getString(R.string.ChequeoCompetencia),
"CHEQUEO"));
        break;
    }
    case "CC": {
        listItems.add(new
Entidad_Bienvenido(R.drawable.categorias, getString(R.string.Categoría),
"CATEGORIA"));
    }
}

```

```

        break;
    }
}
}

}

} catch (Exception ex) {
    Toast.makeText(this, ex.getMessage(), Toast.LENGTH_LONG).show();
}

return listItems;
}

```

## Código: módulo Agrega Caducidad, cambios en acceso a la BD.

```

app\src\m...\\AGREGA_CADUCIDAD.java @@ -44,13 +44,14 @@
    private String usuario;
    // Variable que almacenará el id de la sucursal actual
    private String sucursal_id;
    public String URL = "",DOMINIO;
    //public String URL = "",DOMINIO;
    private String nombre;
    private String userID,nombre;
    private boolean acabo = false;
    private String caducidadID = "0";
    private TextInputEditText editText;
    String NOMBRE_BD = "";
    private PermisosApp[] permisosReq = {
        new PermisosApp ( android.Manifest.permission.CAMERA, "Camara", true )
    };
@@ -74,12 +75,15 @@
    public class AGREGA_CADUCIDAD extends AppCompatActivity {
        nombre = extra.getString("nombre");
        userID = extra.getString("userID");
        caducidadID = extra.getString("caducidadID");
        URL = extra.getString("URL");
        //URL = extra.getString("URL");
        acabo = extra.getBoolean("acabo");
        NOMBRE_BD = extra.getString("NOMBRE_BD");
        SharedPreferenceManager preferencesManager = SharedPreferenceManager.getInstance(getApplicationContext());
        URL = preferencesManager.getWS("ERROR");
        DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
        //preferencesManager.getWS("ERROR");
        //DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
        NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD","");
        nombresucursal = extra.getString("sucursalnombre");
    }

```

## Código: módulo Caducidades, cambios en acceso a la BD.

```
app\src\m...\\AGREGA_CADUCIDAD.java 107 @@ -107,9 +107,11 @@ public class A_CADUCIDADES extends CloseSession {  
app\src\main\...\A_CADUCIDADES.java 107 107 ContadorDelTiempo Cont ;  
108 108 Button resumen;  
...\\A_INTRODUCIR_CADUCIDADES.java 109 109 String getCaducidadID="";  
app\src\main\...\DialogoCantidad.java 110 - public String URL="";  
111 + String DOMINIO="";  
app\src\main\java\DialogoPrecio.java 110 + //Public String URL="";  
111 + //String DOMINIO="";  
app\src\...\A_FALTANTES_SHOPPER.java 112 112 Integer userID;  
app\src\m...\\PlaceholderFragment.java 113 +  
114 + String NOMBRE_BD = "";  
app\src\m...\\SectionsPagerAdapter.java 113 115 //endregion Variables  
114 116  
115 117  
116 118 @@ -140,27 +142,30 @@ public class A_CADUCIDADES extends CloseSession {  
117 119  
118 120 context = this;  
119 121 extras = getIntent().getExtras();  
120 122 this.setTitle(R.string.Caducidad);  
121 123  
122 124 nombrresuc = findViewById(R.id.textView14);  
123 125 sucursal = extras.getString("sucursal");  
124 126 nombre = extras.getString("nombre");  
125 127 usuario = extras.getString("usuario");  
126 128 nombrresucursal = extras.getString("sucursalNombre");  
127 129 userID = extras.getInt("userID");  
128 130 SharedPreferenceManager preferencesManager = SharedPreferenceManager.getInstance(getApplicationContext());  
129 131 NOMBRE_BD = extras.getString("NOMBRE_BD");  
130 132  
131 133 SharedPreferenceManager preferencesManager = SharedPreferenceManager.getInstance(getApplicationContext());  
132 134 NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD","");
133 135
```

## Código: módulo Introducir Caducidades, cambios en acceso a la BD.

```
app\src\m...\AGREGA_CADUCIDAD.java  @@ -79,8 +79,10 @@ public class A_INTRODUCIR_CADUCIDADES extends CloseSession {  
app\src\main\j...\A_CADUCIDADES.java  79  79  
 80  80      boolean acabado;  
...A_INTRODUCIR_CADUCIDADES.java  81  81      ContadorDelTiempo Cont;  
app\src\main\j...\DialogoCantidad.java  82  -  public String URL="";  
app\src\main\j...\DialogoPrecio.java  83  -  public String DOMINIO="";  
app\src\...\A_FALTANTES_SHOPPER.java  82  +  
app\src\m...\PlaceholderFragment.java  83  +  String NOMBRE_BD = "";  
app\src\m...\PlaceholderFragment.java  84  +  //public String URL="";  
app\src\m...\PlaceholderFragment.java  85  +  //public String DOMINIO="";  
app\src\m...\SectionsPagerAdapter.java  84  86      //endregion variables  
 85  87      /**  
 86  88      *  
 87  89      * @param extras  
 88  90      * @param id  
 89  91      * @param tipo  
 90  92      */  
 91  93      @Override  
 92  94      protected void onCreate(Bundle extras)  
 93  95      {  
 94  96          super.onCreate(extras);  
 95  97          setContentView(R.layout.activity_main);  
 96  98          //region Variables  
 97  99          String URL = extras.getString("URL");  
 98 100          String DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");  
 99 101          String WS = preferencesManager.getWS("WS","ERROR");  
100 102          String NOMBRE_BD = preferencesManager.getNombreBD("NOMBRE_BD","");  
101 103          String usuario = extras.getString("usuario");  
102 104          String caducidadid = extras.getString("caducidadid");  
103 105          String sucursal = extras.getString("sucursal");  
104 106          String PEP5 = findViewById(R.id.checkBoxPEP5);  
105 107          SharedpreferencesManager preferencesManager = SharedpreferencesManager.getInstance(getApplicationContext());  
106 108          preferencesManager.setNombreBD(NOMBRE_BD,"");  
107 109          //region VARIABLE URL  
108 110          URL = extras.getString("URL");  
109 111          //URL = extras.getString("URL");  
110 112          //endregion  
111 113          URL = preferencesManager.getWS("WS","ERROR");  
112 114          DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");  
113 115          WS = preferencesManager.getWS("WS","ERROR");  
114 116          DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
```

## Código: módulo Dialogo Cantidad, cambios en acceso a la BD.

```
app\src\m...\\AGREGA_CADUCIDAD.java @@ -81,8 +81,9 @@ public class DialogoCantidad {  
    // Variable que nos sirve para saber si el producto escaneado existe en la base de datos o no  
    private boolean existe;  
...\\A_INTRODUCIR_CADUCIDADES.java  
app\src\main\j...\\DialogoCantidad.java @@ - 84 + 84,6 @@ public String URL;  
    public String DOMINIO;  
app\src\main\java..\\DialogoPrecio.java @@ 84 + //public String URL;  
    85 + //public String DOMINIO;  
    86 + String NOMBRE_BD = "";  
app\src\m...\\PlaceholderFragment.java @@ 86 87 //endregion Variables  
app\src\m...\\SectionsPagerAdapter.java @@ 87 88  
    89 /* Metodo DialogoCantidad  
     * @param sku es el SKU o BarCode del producto  
     * @author Ing. Osvaldo Esperza  
     */  
    105 - public DialogoCantidad(Context context, FinalizarCantidadDialogo finalizarCantidadDialogo, String sku, String URL) {  
    106 107     interfaz = finalizarCantidadDialogo;  
    108 109     SKU = sku;  
    109 110     this.context = context;  
    126 127     imageView = dialog.findViewById(R.id.imageView);  
    127 128  
    128 129     //region VARIABLE URL  
    129 -     this.URL = URL;  
    130 +     //this.URL = URL;  
    131 +     this.NOMBRE_BD = NOMBRE_BD;  
    132 +  
    133 //endregion  
    134     inputFilter filter = new InputFilter() {
```

## Código: módulo Dialogo Precio, cambios en acceso a la BD.

```
app\src\m...\\AGREGA_CADUCIDAD.java @@ -28,6 +28,7 @@ import java.util.Date;  
app\src\main\j...\\A_CADUCIDADES.java @@ 20 20  
    21 21     public class DialogoPrecio {  
...\\A_INTRODUCIR_CADUCIDADES.java @@ 22 22  
app\src\main\j...\\DialogoCantidad.java @@ 23 +  
    24 24     /**  
     * Interfaz FinalizarCantidadDialogo  
     * Esta interfaz se utiliza como medio de comunicacion entre este dialogo y la clase que lo llama  
app\src\main\java..\\DialogoPrecio.java @@ 25 26 @@ -42,8 +43,9 @@ public class DialogoPrecio {  
    42 43     private String caducidadid, sucursal, articuloID;  
    43 44  
    44 45     public boolean aceptarPrecio = false;  
    45 -     public String URL;  
    46 +     //public String URL;  
    46 47     private String usuario;  
    47 48     String NOMBRE_BD ;  
    48 49     //endregion Variables  
    49 51  
    51 52     /** Metodo DialogoCantidad  
     * @param -61,7 +63,7 @@ public class DialogoPrecio {  
    61 63     * @param context es el contexto de la clase  
    62 64     * @author Ing. Osvaldo Esperza  
    63 65     */  
    64 -     public DialogoPrecio(Context context, String caducidadid, String sucursal, String articuloID, String usuario, String URL) {  
    65 66     +     public DialogoPrecio(Context context, String caducidadid, String sucursal, String articuloID, String usuario, String NOMBRE_BD) {  
    66 67         this.context = context;  
    67 68         dialog = new Dialog(this.context);  
    68 69  
    69 70     @@ -75,7 +77,8 @@ public class DialogoPrecio {  
    75 77         this.caducidadid = caducidadid;  
    76 78         this.sucursal = sucursal;
```

## Código: módulo Faltantes Shopper, cambios en acceso a la BD.

```
app\src\m...\\AGREGA_CADUCIDAD.java @@ -400,7 +400,7 @@ public class A_FALTANTES_SHOPPER extends CloseSession implements DialogoCantidad
    i.putExtra("SKU", SKU);
    i.putExtra("usuario", userID);
    //i.putExtra("URL", URL);
    i.putExtra("NOMBRE_DB", NOMBRE_DB);
    i.putExtra("NOMBRE_DB", NOMBRE_DB);
    startActivity(i);
    //dialogoCantidad = new DialogoCantidad(context, A_FALTANTES_SHOPPER.this, SKU);
} else {
app\src\m...\\PlaceholderFragment.java @@ -415,7 +415,7 @@ public class A_FALTANTES_SHOPPER extends CloseSession implements DialogoCantidad
    i.putExtra("SKU", SKU);
    i.putExtra("usuario", userID);
    //i.putExtra("URL", URL);
    i.putExtra("NOMBRE_DB", NOMBRE_DB);
    i.putExtra("NOMBRE_DB", NOMBRE_DB);
    startActivity(i);
    //dialogoCantidad = new DialogoCantidad(context, A_FALTANTES_SHOPPER.this, SKU);
}
....
```

## Código: módulo Conteo Clientes, cambios en acceso a la BD.

```
app\src\main\java\A_CONTEO_CLIENTES.java 47 @@ -47,8 +47,9 @@ public class A_CONTEO_CLIENTES extends AppCompatActivity {  
    context context;  
    boolean band;  
    ContadorDeTiempo Cont ;  
    public String URL = "";  
    public String DOMINIO="";  
    //Public String URL = "";  
    //Public String DOMINIO="";  
    String NOMBRE_BD;  
    @SuppressLint("WrongViewCast")  
    @Override  
    @+ 75,23 -76,23 @@ public class A_CONTEO_CLIENTES extends AppCompatActivity {  
        SharedPreferenceManager preferencesManager = SharedPreferenceManager.getInstance(getApplicationContext());  
        usuario2 = getIntent().getExtras().getString("usuario");  
        competencia = getIntent().getExtras().getString("CHEQUEO_COMPETENCIA_DETALLE_ID");  
        URL = getIntent().getExtras().getString("URL");  
        //URL = getIntent().getExtras().getString("URL");  
        NOMBRE_BD = getIntent().getExtras().getString("NOMBRE_BD");  
        //URL = preferencesManager.getUIS("UIS","ERROR");  
        //DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");  
        NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD","");
    }  
    - URL = preferencesManager.getUIS("UIS","ERROR");  
    - DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");  
    if (savedInstanceState != null){  
        usuario2 = savedInstanceState.getString("usuario");  
    }  
}
```

## Código: módulo Línea Cajas, cambios en acceso a la BD.

```

app\src\m...\\A_CONTEO_CLIENTES.java □
app\src\main\ja...\\A_LINEA_CAJAS.java □
app\src\\A_PERSONAL_ANAQUEL.java □
app\src\main\jav...\\COMPETENCIAS.java □
app\\DialogoBuscarProveedores.java □
app\src\main\ja...\\A PROCESO_GAP.java □
    ↑
    @@ -44,8 +44,9 @@ @ public class A_LINEA_CAJAS extends AppCompatActivity {
44   44     ContadorDeTiempo Cont ;
45   45     WService2 wServiceHoras;
46   46     String s ;
47   47     -     public String URL = "";
48   48     -     public String DOMINIO="";
49   49     +     String NOMBRE_BD;
48   48     +     //public String URL = "";
49   49     +     //public String DOMINIO="";
49   50     @Override
50   51     protected void onCreate(Bundle savedInstanceState) {
51   52       setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
    ↑
    @@ -86,18 +87,22 @@ public class A_LINEA_CAJAS extends AppCompatActivity {
86   87       SharedPreferenceManager preferencesManager = SharedPreferenceManager.getInstance(getApplicationContext());
87   88       competencia = getIntent().getExtras().getString("CHEQUEO_COMPETENCIA_DETALLE_ID");
88   89       usuario2 = getIntent().getExtras().getString("usuario");
89   90     -     URL = getIntent().getExtras().getString("URL");
90   91     +     //URL = getIntent().getExtras().getString("URL");
91   92     +     NOMBRE_BD = getIntent().getExtras().getString("NOMBRE_BD");
92   93     -     URL = preferencesManager.getWS("WS","ERROR");
93   94     -     DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
94   95     +     //URL = preferencesManager.getWS("WS","ERROR");
94   96     +     //DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
95   96     +     NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD","");
96   97     if (savedInstanceState != null) {
97       competencia = savedInstanceState.getString("CHEQUEO_COMPETENCIA_DETALLE_ID");
    ↑

```

## Código: módulo Personal Anaquel, cambios en acceso a la BD.

```

app\src\m...\\A_CONTEO_CLIENTES.java □
app\src\main\ja...\\A_LINEA_CAJAS.java □
app\\A_PERSONAL_ANAQUEL.java □
app\src\main\jav...\\COMPETENCIAS.java □
app\\DialogoBuscarProveedores.java □
app\src\main\ja...\\A PROCESO_GAP.java □
    ↑
    @@ -72,8 +72,9 @@ public class A_PERSONAL_ANAQUEL extends CloseSession implements DialogoBuscarPro
72   72     private String Hora, PERSONAL_ANAQUEL_ID;
73   73     private TextView tvHoraEntrada, tvHorasSalida;
74   74     ContadorDeTiempo Cont ;
75   75     -     public String URL;
76   76     -     public String DOMINIO="";
75   75     +     //public String URL;
76   76     +     //public String DOMINIO="";
77   77     +     String NOMBRE_BD;
77   78     //private ArrayList<Proveedor> proveedoresArray;
78   79     @Override
79   80     protected void onCreate(Bundle savedInstanceState) {
    ↑
    @@ -116,11 +117,13 @@ public class A_PERSONAL_ANAQUEL extends CloseSession implements DialogoBuscarPro
116  117     Intent intent2 = this.getIntent();
117  118     Bundle extra2 = intent2.getExtras();
118  119     SharedPreferenceManager preferencesManager = SharedPreferenceManager.getInstance(getApplicationContext());
119  120     -     URL = extra2.getStringExtra("URL");
120  121     +     //URL = extra2.getStringExtra("URL");
121  122     +     NOMBRE_BD = extra2.getStringExtra("NOMBRE_BD");
120  123     //endregion
122  122     -     URL = preferencesManager.getWS("WS","ERROR");
123  123     -     DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
124  124     +     //URL = preferencesManager.getWS("WS","ERROR");
125  125     +     //DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
126  126     +     NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD","");
124  127     PNL2.addTextChangedListener(new TextWatcher() {
    ↑

```

## Código: módulo Competencias, cambios en acceso a la BD.

```
app\src\m...\\A_CONTEO_CLIENTES.java  @@ -47,8 +47,9 @@ @@ public class A_VALIDACION_CHEQUEO_COMPETENCIAS extends AppCompatActivity {
    47 |    47 |        String horaEnt, horasAl;
    48 |    48 |        public boolean res = false;
    49 |    49 |        ContadorDeTiempo cont ;
  50 |+ 50 |        public String URL = "";
  51 |+ 51 |        //public String DOMINIO="";
  52 |+ 52 |        //public String URL = "";
  53 |+ 53 |        String NOMBRE_BD;
  54 |+ 54 |        @Override
  55 |+ 55 |        /**
  56 |+ 56 |        * Este activity lo que hace es mostrar una lista de las sucursales a las que el usuario fue asignado para hacer las tareas de
  57 |+ 57 |        */
  58 |+ 58 |        @@ -65,20 +66,24 @@ public class A_VALIDACION_CHEQUEO_COMPETENCIAS extends AppCompatActivity {
  59 |+ 59 |            nombreusuario = extra.getStringExtra("usuario");
  60 |+ 60 |            SoloNombreDeMuestra = extra.getStringExtra("solonombre");
  61 |+ 61 |            SharedPreferencesManager preferencesManager = SharedPreferencesManager.getInstance(getApplicationContext());
  62 |+ 62 |            URL = extra.getString("URL");
  63 |+ 63 |            //URL = extra.getString("URL");
  64 |+ 64 |            NOMBRE_BD = extra.getString("URL");
  65 |+ 65 |            URL = preferencesManager.getWS("ws","ERROR");
  66 |+ 66 |            DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
  67 |+ 67 |            //preferencesManager.getWS("ws","ERROR");
  68 |+ 68 |            //DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
  69 |+ 69 |            NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD","");
  70 |+ 70 |            if ( savedInstanceState != null ) {
  71 |+ 71 |                usuario = savedInstanceState.getString("userID");
  72 |+ 72 |                nombreusuario = savedInstanceState.getString("usuario");
  73 |+ 73 |            }
  74 |+ 74 |        }
  75 |+ 75 |        if ( savedInstanceState != null ) {
  76 |+ 76 |            usuario = savedInstanceState.getString("userID");
  77 |+ 77 |            nombreusuario = savedInstanceState.getString("usuario");

```

## Código: módulo Dialogo Buscar Proveedores, cambios en acceso a la BD.

```
app\src\m...\\A_CONTEO_CLIENTES.java  @@ -59,7 +59,8 @@ @@ public class DialogoBuscarProveedores {
    59 |    59 |
    60 |    60 |        public boolean aceptarProveedor = false;
    61 |    61 |
  62 |+ 62 |        public String URL;
  63 |+ 63 |        String NOMBRE_BD;
  64 |+ 64 |        //endregion Variables
  65 |+ 65 |
  66 |+ 66 |        /**
  67 |+ 67 |        * Metodo DialogoCantidad
  68 |+ 68 |        */
  69 |+ 69 |        @@ -77,9 +78,10 @@ public class DialogoBuscarProveedores {
  70 |+ 70 |            * @param context es el contexto de la clase
  71 |+ 71 |            * @author Ing. Osvaldo Esparza
  72 |+ 72 |            */
  73 |+ 73 |            public DialogoBuscarProveedores(Context context,FinalizarProveedoresDialogo finalizarProveedoresDialogo,String URL) {
  74 |+ 74 |                this.context = context;
  75 |+ 75 |                this.URL=URL;
  76 |+ 76 |                this.NOMBRE_BD = NOMBRE_BD;
  77 |+ 77 |                interfaz = finalizarProveedoresDialogo;
  78 |+ 78 |                dialog = new Dialog(this.context);
  79 |+ 79 |                dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
  80 |+ 80 |
  81 |+ 81 |            private void buscarProveedores() throws InterruptedException {
  82 |+ 82 |                this.context = context;
  83 |+ 83 |                this.URL=URL;
  84 |+ 84 |                this.NOMBRE_BD = NOMBRE_BD;
  85 |+ 85 |                interfaz = finalizarProveedoresDialogo;
  86 |+ 86 |                dialog = new Dialog(this.context);
  87 |+ 87 |                dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
  88 |+ 88 |
  89 |+ 89 |                /**
  90 |+ 90 |                * Metodo que se ejecuta cuando el usuario presiona el botónceptar
  91 |+ 91 |                */
  92 |+ 92 |                dialog.setContentView(R.layout.dialogo_buscar_proveedores);
  93 |+ 93 |                dialog.setCancelable(true);
  94 |+ 94 |                dialog.show();
  95 |+ 95 |            }
  96 |+ 96 |        }
  97 |+ 97 |        /**
  98 |+ 98 |        * Metodo que se ejecuta cuando el usuario presiona el botóncancelar
  99 |+ 99 |        */
 100 |+100 |        dialog.dismiss();
 101 |+101 |    }
 102 |+102 |}
 103 |+103 |private void buscarProveedores() throws InterruptedException {
 104 |+104 |
 105 |+105 |    /**
 106 |+106 |    * Metodo que se ejecuta cuando el usuario presiona el botónceptar
 107 |+107 |    */
 108 |+108 |    dialog.setContentView(R.layout.dialogo_buscar_proveedores);
 109 |+109 |    dialog.setCancelable(true);
 110 |+110 |    dialog.show();
 111 |+111 |    /**
 112 |+112 |    * Metodo que se ejecuta cuando el usuario presiona el botóncancelar
 113 |+113 |    */
 114 |+114 |    dialog.dismiss();
 115 |+115 |}
```

## Código: módulo Proceso GAP, cambios en acceso a la BD.

```
app\src\main\java\A_CONTEO_CLIENTES.java  @@ -73,7 +73,8 @@ public class A_PROCESO_GAP extends AppCompatActivity {
 73 73     public String FECHA_HORA_CREACION;
 74 74     public boolean Band;
 75 75     ContadorDeTiempo Cont ;
 76 76     - public String URL;
 77 77     + //public String URL;
 78 78     + String NOMBRE_BD;
 79 79     @Override
 80 80     protected void onCreate(Bundle savedInstanceState) {
 81 81         super.onCreate(savedInstanceState);
 82 82     }
 83 83
 84 84     @@ -117,7 +121,8 @@ public class A_PROCESO_GAP extends AppCompatActivity {
 85 85     //region VARIABLE URL
 86 86     Intent intentz = this.getIntent();
 87 87     Bundle extra2 = intentz.getExtras();
 88 88     - URL = extra2.getStringExtra("URL");
 89 89     + //URL = extra2.getStringExtra("URL");
 90 90     125 + NOMBRE_BD = extra2.getString("NOMBRE_BD");
 91 91     //endregion
 92 92
 93 93     @@ -230,10 +232,11 @@ public class A_PROCESO_GAP extends AppCompatActivity {
 94 94     //Consultamos la fecha de la ultima visita en el WebService mediante el ID del proveedor y de la sucursal
 95 95     private void FechaUltVista() throws InterruptedException {
 96 96     WService3 cs = new WService3("FechaUltimaVisita",URL);
 97 97     WService3 cs = new WService3("FechaUltimaVisita");
```

## Código: módulo Mistery Shopper, cambios en acceso a la BD.

```
app\src\main\java\A_MISTERY_SHOPPER.java  @@ -62,7 +62,8 @@ public class A_MISTERY_SHOPPER extends AppCompatActivity {
 62 62     ArrayList<String> Listamodulos = new ArrayList<>();
 63 63     private Adaptador_Misteryshopper Adaptador_misteryshopper;
 64 64     ContadorDeTiempo Cont ;
 65 65     - public String URL = "",DOMINIO = "";
 66 66     + //public String URL = "",DOMINIO = "";
 67 67     + String NOMBRE_BD;
 68 68     @Override
 69 69     protected void onCreate(Bundle savedInstanceState) {
 70 70     @@ -89,18 +90,22 @@ public class A_MISTERY_SHOPPER extends AppCompatActivity {
 71 71     usuario = extra.getStringExtra("nombre");
 72 72     SoloNombreDeNuevostra = extra.getStringExtra("soloNombre");
 73 73     nombresucursal = extra.getStringExtra("nombresucursal");
 74 74     - URL = extra.getStringExtra("URL");
 75 75     + //URL = extra.getStringExtra("URL");
 76 76     SharedPreferencesManager preferencesManager = SharedPreferencesManager.getInstance(getApplicationContext());
 77 77     - URL = preferencesManager.getW5("WS","ERROR");
 78 78     DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
 79 79     - //DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
 80 80     + //URL = preferencesManager.getW5("WS","ERROR");
 81 81     //DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
 82 82     NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD","");
 83 83     if (savedInstanceState != null) {
 84 84         sucursal_id = savedInstanceState.getString("Sucursal_ID");
 85 85         UsuarioIDGuardado = savedInstanceState.getString("UsuarioIDGuardado");
 86 86         MSModulo = savedInstanceState.getString("MSmodulo");
 87 87         SoloNombreDeNuevostra = savedInstanceState.getString("soloNombre");
 88 88         URL = savedInstanceState.getString("URL");
 89 89         URL = preferencesManager.getW5("WS","ERROR");
 90 90         DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");
 91 91 }
```

## Código: módulo MS Producto Precio, cambios en acceso a la BD.

```
app\src\m...IA_MISTERY_SHOPPER.java [ ] 81 82 @@ -82,7 +82,8 @@ public class A_MS_PRODUCTO_PRECIO extends AppCompatActivity {  
 82 | 82 |     int Aumento;  
 83 | 83 |     int Valor = 0;  
 84 | 84 |     ContadorDeTiempo Cont;  
 85 | 85 |     public String URL = "",DOMINIO = "";  
 85 | 85 |     //public String URL = "",DOMINIO = "";  
 86 | 86 |     String NOMBRE_BD;  
 86 87  
 87 88     @Override  
 88 89     protected void onCreate(Bundle savedInstanceState) {  
 89 | 89 |     @@ -99,10 +100,12 @@ public class A_MS_PRODUCTO_PRECIO extends AppCompatActivity {  
 99 | 99 |         ID_USER = extra.getString("usuario");  
 100| 100|         sucursal_Id = extra.getString("sucursal_id");  
 101| 101|         sucursalNombre = extra.getString("sucursalNombre");  
 102| 102|         URL = extra.getString("URL");  
 103| 103|         //URL = extra.getString("URL");  
 104| 104|         NOMBRE_BD = extra.getString("NOMBRE_BD");  
 104| 104|         SharedPreferenceManager preferencesManager = SharedPreferenceManager.getInstance(getApplicationContext());  
 105| 105|         URL = preferencesManager.getWS("WS","ERROR");  
 105| 105|         DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");  
 106| 106|         //URL = preferencesManager.getWS("WS","ERROR");  
 107| 107|         //DOMINIO = preferencesManager.getDominio("DOMINIO","ERROR");  
 108| 108|         NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD","");
 108 109  
 109 110     Buscar();  
 108 111     /**  
 108 | 108 |     @@ -116,9 +119,11 @@ public class A_MS_PRODUCTO_PRECIO extends AppCompatActivity {  
 116| 119|         nombre = savedInstanceState.getString("nombre");  
 117| 120|         ID_USER = savedInstanceState.getString("ID_USER");  
 118| 121|         GeofenceBroadcastReceiver.sucursalActual = savedInstanceState.getString("sucursalActual");
```

## Código: implementación de módulo Faltantes Solución.

```
package com.example.inmex_ver1.Faltantes;  
  
import android.app.AlertDialog;  
import android.content.Context;  
import android.content.DialogInterface;  
import android.content.Intent;  
import android.content.pm.ActivityInfo;  
import android.graphics.Color;  
import android.os.AsyncTask;  
import android.os.Bundle;  
import android.os.Handler;  
import android.text.format.DateFormat;  
import android.util.Log;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.widget.AdapterView;
```

```
import android.widget.ArrayAdapter;
import android.widget.BaseExpandableListAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ExpandableListAdapter;
import android.widget.ExpandableListView;
import android.widget.ImageButton;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;
import androidx.annotation.NonNull;
import
com.example.inmex_ver1.Adaptadores.CustomExpandableListAdapterProductosSolucion;
import com.example.inmex_ver1.CloseSession;
import com.example.inmex_ver1.Dialogos.DialogoImagen;
import com.example.inmex_ver1.Dialogos.DialogoSKU;
import com.example.inmex_ver1.GeofenceBroadcastReceiver;
import com.example.inmex_ver1.ListSoluciones;
import com.example.inmex_ver1.MainActivity;
import com.example.inmex_ver1.NotificationHelper;
import com.example.inmex_ver1.ProductoPrecio.Products;
import com.example.inmex_ver1.R;
import com.example.inmex_ver1.ServiciosWeb.WService;
import com.example.inmex_ver1.ServiciosWeb.WService2;
import com.example.inmex_ver1.ServiciosWeb.WService3;
import com.example.inmex_ver1.SharedPreferencesManager;
import com.google.android.material.snackbar.Snackbar;
import org.ksoap2.serialization.SoapObject;
import java.util.ArrayList;
import java.util.Date;
import java.util.HashMap;
import java.util.List;
import java.util.TreeMap;
import java.util.stream.Collectors;
```

```

/**
 * Modulo A_FALTANTES_SOLUCION
 * Esta clase contiene toda la logica del modulo A_FALTANTES_SOLUCION esta
clase sirve para darles una
 * solucion a los articulos faltantes
*
* @author Dev. Dario Vidaña ft. Dev. Mauro Rangel
*/
public class A_FALTANTES_SOLUCION extends CloseSession  {

    /**STRING*/
    // Variable que almacena la hora y la envia por medio de una notificacion
    public String Hora, nombresucursal;

    // Variable que almacena el id de la sucursal
    private String sucursal_id;

    // Variable que almacena el id del articulo
    private String articulo_id;

    // Variable que almacena el usuario
    private String usuario;

    // Variable que almacenará la hora
    private String hora;

    // Variable que almacenará el id de las faltantes
    private String faltantes_id;

    // Variable que recibe un URL para mostrar una imagen
    private String URLWhats;

    // Variable que recibe una hora
    private String hor1;

    // Variable que almacena el Nombre de la Base de datos
}

```

```
String NOMBRE_BD = "";

// Variable que almacena el nombre de la solucion
private String nombre_sol;

// Variable que almacena la clave de la solucion
private String clave_sol;

// Variable que almacena el ID del usuario
private String userID;

// Variable que almacenaba la Solucion
private String solucion;

public String Archivo[];

/**ENTERO*/
// Variable que indicará si el item está desplegado o no
private int lasExpandedPosition = -1;

//Variable para denotar la recursividad en una funcion
int Recursividad;

// Variable que almacena el id de da solucion
int solucion_id;

// Variable que almacena el Faltante ID
int faltante_id;

int solucionado;

//Variable que almacena el ID de la solucion
int buscarSolucionId;

/**BOOLEAN*/
```

```

// Variable que servirá para indicar si ya hay alguna faltante insertada
en ese dia
private boolean isSomeInserted = false;

// Variable que indica si la actividad ya acabo o no
private boolean acabo;

//Variable que indica si la hora ya se subio.
private boolean subioI = true;

/**ARRAY*/
// Array para extarer la lista de los productos del WebService
private final ArrayList<Products> listaProductossinInteger = new
ArrayList<>();
private final HashMap<Integer, Products> listaProductos = new HashMap<>();

// LISTA SOLUCIONES
List<ListSoluciones> listSol = new ArrayList<>();
List<ListSoluciones> listSolFiltrada = new ArrayList<>();
Products ArregloProductos[];
int CantidadProductos[];
int Integerproducto[];

/**VARIABLES QUE SIRVEN COMO REFERENCIA HACIA LOS TEXTOS DONDE SE MOSTRARÁ
LA HORA DE INICIO Y FIN DE LA ACTIVIDAD*/
// Variable que sirve para hacer referencia a los TextView de la Hora de
entrada, salida y de la sucursal.
private TextView tvHoraEntrada, tvHoraSalida, tvSucursal, tvTotalSKU;
// VAriable que sirve para indicar al usuario que debe de finalizar la
tarea
private TextView etiqueta;
// Variable que almacenará la referencia del contexto de la aplicacion
private Context context;
// Variable que almacenará la referencia hacia la lista desplegable en la
pantalla
private ExpandableListView expandableListView;

```

```

// Variable que contendrá la referencia hacia el adaptador de la lista
desplegable
private ExpandableListAdapter expandibleListAdapter;

// Variable que nos hace referencia a un botón donde muestra una imagen
private ImageButton imagenWhats;

/**OTRAS VARIABLES*/
// Variable que servirá para mandar notificaciones al usuario
private NotificationHelper notificationHelper;

private static final long MIN_CLICK_INTERVAL = 2000;
private long lastClickTime = 0;

// Variable extra
Bundle extra;

/** AQUI TERMINAN LAS VARIABLES DE LA CLASE */

/** METODO ON CREATE */
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
    setContentView(R.layout.activity_faltantes_solucion);
    this.setTitle(R.string.FaltantesSolucion);

    Archivo = fileList();

    try {
        Intent intent2 = this.getIntent();
        extra = intent2.getExtras();
        sucursal_id = extra.getString("sucursal");
        usuario = extra.getString("usuario");
    }
}

```

```

        userID = extra.getString("userID");
        NOMBRE_BD = extra.getString("NOMBRE_BD");
        faltantes_id = extra.getString("FALTANTES_ID");
        SharedpreferencesManager preferencesManager =
        SharedpreferencesManager.getInstance(getApplicationContext());
        NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD", "");

        nombresucursal = extra.getString("sucursalnombre");
    } catch (Exception ex) {
        Toast.makeText(this,
        getString(R.string.FaltantesShopperErrorDeRedNoCargoSuc),
        Toast.LENGTH_LONG).show();
    }

    Button btnActualizar = findViewById(R.id.btnActualizarSol);
    Button btnFinish = findViewById(R.id.btnFinish);

    context = this;
    Bundle extra = getIntent().getExtras();
    usuario = extra.getString("usuario");
    if (savedInstanceState != null) {
        usuario = savedInstanceState.getString("usuario");
        sucursal_id = savedInstanceState.getString("sucursal_id");
        nombresucursal = savedInstanceState.getString("sucursalnombre");
        SharedpreferencesManager preferencesManager =
        SharedpreferencesManager.getInstance(getApplicationContext());
        NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD", "");
        NOMBRE_BD = extra.getString("NOMBRE_BD");
    }

    notificationHelper = new NotificationHelper(this);
    tvHoraEntrada = findViewById(R.id.tvHoraEntrada);
    tvHoraSalida = findViewById(R.id.tvHoraSalida);
    etiqueta = findViewById(R.id.txvEtiqueta);
    imagenWhats = findViewById(R.id.imagenWhats);

```

```

expandableListView = findViewById(R.id.elvFaltantes);

if (GeofenceBroadcastReceiver.isOnGeofence)
    sucursal_id = GeofenceBroadcastReceiver.sucursalActual;

imagenWhats.setOnClickListener((View) -> {
    DialogoImagen dialogoImagen = new DialogoImagen(this, URLWhats);
});

//BOTON ACTUALIZAR
btnActualizar.setOnClickListener((View v) -> {
    //recreate();
    finish();
    startActivity(getIntent());
    overridePendingTransition(0, 0);
});

// BOTON FINALIZAR
btnFinish.setOnClickListener((View v) -> {
    if( ArregloProductos == null){
        finish();
    }

    if (revisarSolucion()) {
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("Mensaje de aplicación");
        builder.setMessage("¿Seguro que quieres finalizar la actividad?");
        builder.setPositiveButton("Aceptar", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                try {
                    Hora = getHora();
                    if (subirHoraFaltantes("F")) {

```

```

        if (setSolucionFaltante() == 1) {
            etiqueta.setBackgroundColor(Color.WHITE);
            etiqueta.setTextColor(Color.BLACK);

notificationHelper.sendHighPriorityNotification(getString(R.string.NotifyPushA
ctividadFaltantesFinalizada),

getString(R.string.NotifyPushHoraFinFaltantes) + " \t" + Hora,
MainActivity.class);
                finish();
            }
        }
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}
} );
}

builder.setNegativeButton("Cancelar", new
DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        return;
    }
});
AlertDialog dialog = builder.create();
dialog.show();

} else {
    Snackbar.make(btnFinish, "AUN QUEDAN SOLUCIONES PENDIENTES",
Snackbar.LENGTH_SHORT)
        .setTextColor(Color.WHITE)
        .setBackgroundTint(Color.RED).show();
}

```

```

        }

    });

    tvSucursal = findViewById(R.id.textSucursal);
    tvTotalSKU = findViewById(R.id.totalSKU);
    PonerNombreSucursal();
    MostrarFaltantesPendientes();
    PonerTotalSKU();

}

/***
 * Método para verificar si todos los renglones de la ExpandableListView
están en verde.
 * @return true si todos los renglones están en verde, false de lo
contrario.
 */
private boolean revisarSolucion() {
    for (Products producto : ArregloProductos) {
        if (producto.getSolucion() == null ||
producto.getSolucion().isEmpty()) {
            return false;
        }
    }
    return true; //Solo retorna verdadero si todos los productos tienen
solución
}

/***
 * Método PonerNombreSucursal
 * se encarga de consultar la variable del WebService para añadir el
nombre de la sucursal
 * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
 */
public void PonerNombreSucursal() {
    if (nombresucursal != null) {

```

```

        tvSucursal.setText("Sucursal :" + nombresucursal);
    } else {
        if (Recursividad < 3) {
            Recursividad++;
            PonerNombreSucursal();
        }
    }
}

public void PonerTotalsSKU(){
    tvTotalSKU.setText("Total SKU'S : " + listaProductos.size());
}

/**
 * Metodo init
 * Este metodo manda a llamar al metodo MostrarFaltantes
 *
 * @author Ing. Osvaldo Esparza
 */
private void init() {
    try {
        MostrarFaltantesPendientes();
    } catch (Exception ex) {
        Log.i("error", ex.getMessage());
    }
}

@Override
public void onResume() {
    AutoTimeIsOn2();
    SharedPreferencesManager preferencesManager =
    SharedPreferencesManager.getInstance(getApplicationContext());
    NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD", "");
    Recursividad = 0;
}

```

```

        PonerNombreSucursal();

        ProductsAsyncTask productsAsyncTask = new ProductsAsyncTask();
        productsAsyncTask.execute();
        super.onResume();
    }

    public boolean res = false;
    AlertDialog.Builder builder;
    TextView Mensaje;
    public void AutoTimeIsOn2() {
        res = android.provider.Settings.Global.getInt(getApplicationContext(),
        android.provider.Settings.Global.AUTO_TIME, 0) == 0;
        if (res) {
            MensajeG(this);
            builder.show();
        }
    }

    /**
     * Mensaje de error el cual sale cuando hay un problema con la hora.
     * @param context
     * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
     */
    public void MensajeG(Context context) {
        builder = new AlertDialog.Builder(context);
        builder.setTitle("Error de Hora");
        Mensaje = new TextView(context);
        builder.setCancelable(false);
        Mensaje.setText("\n\t Tiene la Hora Automatica DesHabilitada, la
aplicacion no le permitira entrar. Vaya a las opciones y active de nuevo la
Hora Automatica.");
        Mensaje.setPadding(20, 20, 20, 20);
        builder.setView(Mensaje);
        builder.setPositiveButton("OK", new DialogInterface.OnClickListener()
{
    @Override

```

```

        public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
            finishAffinity();
            System.exit(0);
        }
    });

}

/***
 * Metodo llenarHoras
 * Este metodo manda a llamar al webService MuestraHoraInicioFinCaducidad
el cual nos dará las horas
 * de inicio y fin de la actividad para así poder indicarle a los usuarios
su hora de inicio y de fin
 * de la actividad, ademas en este metodo es en donde se desvanece la
progressbar
 *
 * @author Ing. Osvaldo Esparza
 */
public void llenarHoras() throws InterruptedException {
    buscarSolucionIDs();
    if (!NOMBRE_BD.equals("")) {
        StringBuilder horaEnt = new StringBuilder("Hora de entrada ");
        StringBuilder horaSal = new StringBuilder("Hora de salida ");
        if (String.valueOf(buscarSolucionId).equals("")) {
            tvHoraEntrada.setText(horaEnt.toString());
            tvHoraSalida.setText(horaSal.toString());
        } else if (!String.valueOf(buscarSolucionId).equals("")) {
            WService cs = new WService("MuestraHoraInicioFinSolucion");
            cs.context = context;
            cs.SOLUCION_ID = String.valueOf(buscarSolucionId);
            cs.NOMBRE_BD = NOMBRE_BD;
            cs.start();
            cs.join();
            for (int i = 0; i < cs.arreglo.length - 1; i++) {
                horaEnt.append(cs.arreglo[i].getProperty("HORA_INICIO") !=

```

```

null ? (String) cs.arreglo[i].getProperty("HORA_INICIO") : "");
                horI = cs.arreglo[i].getProperty("HORA_INICIO") != null ?
(String) cs.arreglo[i].getProperty("HORA_INICIO") : "";
                subioI = cs.arreglo[i].getProperty("HORA_INICIO") != null;
                horaSal.append(cs.arreglo[i].getProperty("HORA_FIN") != null ?
(String) cs.arreglo[i].getProperty("HORA_FIN") : "");
                acabo = cs.arreglo[i].getProperty("HORA_INICIO") != null
&& cs.arreglo[i].getProperty("HORA_FIN") != null;
            }
            tvHoraEntrada.setText(horaEnt.toString());
            tvHoraSalida.setText(horaSal.toString());
        }

    }
}

/**
 * "Ordenamiento" que se ejecuta despues de haber finalizado la tarea y en
donde se le da formato
 * a la lista.
 * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
 */
private void Orden() {
    Ordenamiento(CantidadProductos);
    expandibleListAdapter = new
CustomExpandibleListAdapterProductosSinIntegerSolucion(context,
listaProductossinInteger);
    expandableListView.setAdapter(expandibleListAdapter);
    expandableListView.setOnGroupExpandListener(groupPosition -> {
        if (lasExpandedPosition != -1 && groupPosition != lasExpandedPosition) {
            expandableListView.collapseGroup(lasExpandedPosition);
            // Despues de haber encontrado la referencia a tu
ExpandableListView
            expandableListView.setOnGroupClickListener(new

```

```

ExpandableListView.OnGroupClickListener() {
    @Override
    public boolean onGroupClick(ExpandableListView parent,
View v, int groupPosition, long id) {
        // Retornar true para evitar que el grupo se expanda o
contraiga
        return true;
    }
} ;
}
lasExpandedPosition = groupPosition;
}) ;
}

/**
 * Metodo getHora
 * Este metodo regresa la fecha y la hora actual del sistema en formato
2021-12-31 23:59:59
 * @author Ing. Osvaldo Esparza
 */
public String getFechaHora() {
    String fecha;
    Date d = new Date();
    fecha = (String) DateFormat.format("yyyy-MM-dd HH:mm:ss",
d.getTime());
    return fecha;
}

public String getFecha() {
    String fecha;
    Date d = new Date();
    fecha = (String) DateFormat.format("yyyy-MM-dd", d.getTime());
    return fecha;
}

/**

```

```

* Metodo MostrarFaltantesPendientes
* Este metodo manda a llamar al webService MostrarFaltantes el cual trae
toda la informacion del articulo
* insertado en la tabla faltantes con la fecha de hoy
*/

public void MostrarFaltantesPendientes() {
    SharedPreferencesManager preferencesManager =
    SharedPreferencesManager.getInstance(getApplicationContext());
    NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD", "");
    if (!NOMBRE_BD.equals("")) {
        WService3 cs = new WService3("MostrarFaltantesPendiente");
        cs.FALTANTES_ID = faltantes_id;
        cs.SUCURSAL_ID = sucursal_id;
        cs.NOMBRE_BD = NOMBRE_BD;
        cs.start();
        try {
            // cs.wait(2000);
            cs.join();
            //
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
    }

    System.out.println(cs.arreglo.length);
    if (cs.arreglo.length > 0) {
        CantidadProductos = new int[cs.arreglo.length - 1];
        Integerproducto = new int[cs.arreglo.length - 1];
        ArregloProductos = new Products[cs.arreglo.length - 1];
    }

    if (cs.arreglo.length > 0) {
        listaProductos.clear();
        isSomeInserted = true;
        String protocolo = "https://";
        etiqueta.setBackgroundColor(getResources().getColor(R.color.txvEtiquetaFaltantesShopperRojo));
    }
}

```

```

;

etiqueta.setTextColor(Color.WHITE);

for (int i = 0; i < cs.arreglo.length - 1; i++) {
    faltantes_id = cs.arreglo[i].getProperty("FALTANTES_ID")
!= null ? Integer.toString((int) cs.arreglo[i].getProperty("FALTANTES_ID")) :
"";

        int FALTANTES_DETALLE_ID =
cs.arreglo[i].getProperty("FALTANTES_DETALLE_ID") != null ? (int)
cs.arreglo[i].getProperty("FALTANTES_DETALLE_ID") : 0;

        int ARTICULO_ID = cs.arreglo[i].getProperty("ARTICULO_ID")
!= null ? (int) cs.arreglo[i].getProperty("ARTICULO_ID") : 0;

        String SKU = cs.arreglo[i].getProperty("SKU") != null ?
(String) cs.arreglo[i].getProperty("SKU") : "";

        String DESCRIPCION =
cs.arreglo[i].getProperty("DESCRIPCION") != null ? (String)
cs.arreglo[i].getProperty("DESCRIPCION") : "";

        String STOCK_FISICO =
cs.arreglo[i].getProperty("STOCK_FISICO") != null ? (String)
cs.arreglo[i].getProperty("STOCK_FISICO") : "";

        String PRECIO_ARTICULO =
cs.arreglo[i].getProperty("PRECIO_ARTICULO") != null ? (String)
cs.arreglo[i].getProperty("PRECIO_ARTICULO") : "";

        String IMAGEN = cs.arreglo[i].getProperty("IMAGEN") !=
null ? (String) cs.arreglo[i].getProperty("IMAGEN") : "";

        String SOLUCION = cs.arreglo[i].getProperty("SOLUCION") !=
null ? (String) cs.arreglo[i].getProperty("SOLUCION") : "";

        boolean SOLUCIONADO =
cs.arreglo[i].getProperty("SOLUCION") != "" ? true:false;

        String FECHA_ULT_RECIBO =
cs.arreglo[i].getProperty("FECHA_ULT_RECIBO") != null ? (String)
cs.arreglo[i].getProperty("FECHA_ULT_RECIBO") : "";

        String CAPACIDAD_EMPAQUE =
cs.arreglo[i].getProperty("CAPACIDAD_EMPAQUE") != null ? (String)
cs.arreglo[i].getProperty("CAPACIDAD_EMPAQUE") : "";

        String FECHA = cs.arreglo[i].getProperty("FECHA") != null
? (String) cs.arreglo[i].getProperty("FECHA") : "";

```

```

        String EXISTENCIA_TEORICA =
cs.arreglo[i].getProperty("EXISTENCIA_TEORICA") != null ? (String)
cs.arreglo[i].getProperty("EXISTENCIA_TEORICA") : "";
        IMAGEN = IMAGEN.contains(protocolo) ? IMAGEN : protocolo +
IMAGEN;
        Products product = new Products(SKU, new String[]{IMAGEN},
DESCRIPCION, PRECIO_ARTICULO, SOLUCION, FECHA, ARTICULO_ID,
EXISTENCIA_TEORICA,FECHA_ULT_RECIBO,CAPACIDAD_EMPAQUE);
        product.setCantidad(STOCK_FISICO);
        product.setPrecio(PRECIO_ARTICULO);
        listaProductos.put(FALTANTES_DETALLE_ID, product);
        Integerproducto[i] = FALTANTES_DETALLE_ID;
        product.contestado = SOLUCIONADO;
        ArregloProductos[i] = product;
        CantidadProductos[i] =
ConvertidorStringAInteger(STOCK_FISICO);
    }
    TreeMap<Integer, Products> tm = new TreeMap<>(listaProductos);
    listaProductos.putAll(tm);
}
}

/**
 * Convertidor de Strings A intener que nos sirve mas adelante para el
ordenamiento de los productos escaneados
*/
public int ConvertidorStringAInteger(String Numero) {
    int NumeroConvertido;
    try {
        NumeroConvertido = Integer.parseInt(Numero);
    } catch (Exception e) {
        NumeroConvertido = 0;
    }
    return NumeroConvertido;
}

```

```

/**
 * Esta es la clase de ordenamiento a modalidad de burbuja, ordena los
articulos
 * escaneados pero solo hasta que el usuario preciona el boton de
finalizar
 */
public void Ordenamiento(int[] A) {
    int i, j;
    int CantidadProductosAux;
    int IntegerproductoAux;
    Products ArregloProductosAux;
    if (A == null)
        return;
    if (A.length > 0) {

        for (i = 0; i < A.length - 1; i++) {
            for (j = 0; j < A.length - i - 1; j++) {
                if (A[j + 1] < A[j]) {
                    CantidadProductosAux = A[j + 1];
                    IntegerproductoAux = Integerproducto[j + 1];
                    ArregloProductosAux = ArregloProductos[j + 1];
                    A[j + 1] = A[j];
                    Integerproducto[j + 1] = Integerproducto[j];
                    ArregloProductos[j + 1] = ArregloProductos[j];
                    A[j] = CantidadProductosAux;
                    Integerproducto[j] = IntegerproductoAux;
                    ArregloProductos[j] = ArregloProductosAux;
                }
            }
        }
        for (i = 0; i < A.length; i++) {
            listaProductossinInteger.add(ArregloProductos[i]);
        }
    }
}

```

```

    /**
     * Metodo getHora
     * Este metodo regresa la hora actual del sistema en formato 23:59:59
     *
     * @author Ing. Osvaldo Esparza
     */
    public String getHora() {
        String fecha;
        Date d = new Date();
        fecha = (String) DateFormat.format("HH:mm:ss", d.getTime());
        hora = fecha;
        return fecha;
    }

    /**
     * Metodo subirHoraCaducidad
     * Este metodo tiene como funcionamiento subir la hora en la cual
     inicio/finalizo la actividad
     * mandando a llamar el webservice ActualizarHoraInicioFinFaltantes, el
     cual necesita el tipo de hora inicio(I)
     * Fin (F) , la hora y la caducidadID
     *
     * @author Ing. Osvaldo Esparza
     */
    public boolean subirHoraFaltantes(String tipo) throws InterruptedException
    {
        if (Hora == null)
            Hora = getHora();
        SharedPreferencesManager preferencesManager =
        SharedPreferencesManager.getInstance(getApplicationContext());
        NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD", "");
        WService cs = new WService("ActualizarHoraInicioFinSolucion");
        cs.context = context;
        cs.TIPO = tipo;
    }
}

```

```

        cs.HORA = getHora();
        cs.FALTANTES_ID = faltantes_id;
        cs.SOLUCION_ID = String.valueOf(buscarSolucionId);
        cs.USUARIO_MODIFICACION = userID;
        cs.FECHA_HORA_MODIFICACION = getFechaHora();
        cs.NOMBRE_BD = NOMBRE_BD;
        System.out.println(tipo);
        System.out.println(hora);
        System.out.println(faltantes_id);
        System.out.println(sucursal_id);
        cs.start();
        cs.join();
        if(cs.RESBOL)
            tvHoraEntrada.setText(getHora());
        return cs.RESBOL;
    }

    /**
     * Clase ProductsAsyncTask
     * Esta clase permite crear un hilo para ejecutarse a la par con el metodo
     * onCreate al iniciar la actividad
     *
     * @author Ing. Osvaldo Esparza
     */
    public class ProductsAsyncTask extends AsyncTask {

        /**
         * Metodo onPreExecute
         * Este metodo se ejecuta antes de que el hilo se empiece a ejecutar y
         * hace una preparacion para cuando
         * el hilo se esté ejecutando
         */
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
        }
    }
}

```

```

    /**
     * Metodo doInBackground
     * Este es el hilo que se ejecuta en segundo plano a la par de que la actividad se crea
    */
    @Override
    protected Object doInBackground(Object[] objects) {
        runOnUiThread(() -> {
            SharedPreferencesManager preferencesManager =
            SharedPreferencesManager.getInstance(getApplicationContext());
            NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD", "");
            try {
                init();
                llenarHoras();
                if (!subioI) {
                    subirHoraFaltantes("I");
                    llenarHoras();
                }

            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        });
        return null;
    }

    /**
     * Metodo onPostExecute
     * Este metodo se ejecuta despues de que termino de ejecutarse el metodo doInBackground
     * |           * aprovechamos el metodo onPostExecute para poder asignar el listener al momento de darle
     * click al grupo deseado del expandibleLastView, cuando se selecciona una opcion, este
     * se pintara de verde

```

```

* @author Dev. Dario Vidaña ft. Dev Mauro Rangel
*/
@Override
protected void onPostExecute(Object o) {
    TreeMap<Integer, Products> tm = new TreeMap<>(listaProductos);
    List<Integer> expandibleLisNombre = new
ArrayList<>(tm.descendingKeySet());

    expandibleListAdapter = new
CustomExpandableListAdapterProductosSolucion(context, expandibleLisNombre,
listaProductos);
    expandableView.setAdapter(expandibleListAdapter);
    expandableView.setOnGroupExpandListener(groupPosition -> {
        if (lasExpandedPosition != -1 && groupPosition != lasExpandedPosition) {
            expandableView.collapseGroup(lasExpandedPosition);
        }
        lasExpandedPosition = groupPosition;
    });

    if (acabo) {
        etiqueta.setBackgroundColor(Color.WHITE);
        etiqueta.setTextColor(Color.BLACK);
        Orden();
    }
    expandableView.setOnGroupClickListener(new
ExpandableListView.OnGroupClickListener() {

    @Override
    public boolean onGroupClick(ExpandableListView parent, View v,
int groupPosition, long id) {
        long currentTime = System.currentTimeMillis();

        //Evitar clics rápidos
        if (currentTime - lastClickTime < MIN_CLICK_INTERVAL) {

```

```

        return true;
    }

    lastClickTime = currentTime;

    //Deshabilitar temporalmente los clics
    parent.setEnabled(false);

    Handler handler = new Handler();
    handler.postDelayed(() -> parent.setEnabled(true),
MIN_CLICK_INTERVAL);

    Products producto = (Products)
expandibleListAdapter.getChildAt(groupPosition, 0);

    if (producto.isContestado() && !producto.setEditable()) {
        Toast.makeText(context, "La solución ya no se puede
modificar", Toast.LENGTH_LONG).show();
        return true;
    }

    WService ws = new WService("getSolucionOpciones");
    ws.context = context;
    ws.NOMBRE_BD = NOMBRE_BD;
    ws.start();
    try {
        ws.join();
    } catch (InterruptedException e) {
        throw new RuntimeException(e);
    }

    String cantidad = producto.getCantidad();
    String existencia = producto.getExistencia_teorica();

    //LISTA DE SOLUCIONES
    List<SoapObject> lista = new ArrayList<>();
    for (SoapObject elemento : ws.arreglo) {

```

```

        if (elemento != null) {
            String clave =
elemento.getProperty("CLAVE").toString();
            if(!cantidad.equals(existencia)) {
                if(!clave.equals("CFE")) {
                    lista.add(elemento);
                }
            } else {
                lista.add(elemento);
            }
        }

        if (lista.size() > 1) {
            for (int i = 0; i < lista.size(); i++) {
                listSol.add(
                    new ListSoluciones(
Integer.parseInt(lista.get(i).getProperty("SOLUCION_OPCIONES_ID").toString()),

lista.get(i).getProperty("NOMBRE").toString(),

lista.get(i).getProperty("CLAVE").toString()
                )
            );
        }
    } else if (lista.size() == 1) {
        solucion_id =
Integer.parseInt(lista.get(0).getProperty("SOLUCION_OPCIONES_ID").toString());
        nombre_sol =
lista.get(0).getProperty("NOMBRE").toString();
        clave_sol =
lista.get(0).getProperty("CLAVE").toString();
        SharedPreferencesManager preferencesManager =
SharedPreferencesManager.getInstance(getApplicationContext());
        preferencesManager.saveNOMBRE_BD("NOMBRE_BD",

```

```

NOMBRE_BD) ;

        new GeofenceBroadcastReceiver();
        GeofenceBroadcastReceiver.Nombre_BD = NOMBRE_BD;
    }

    // LISTA DE SOLUCIONES FILTRADA

    List<SoapObject> lista2 = new ArrayList<>();
    for (SoapObject elemento : ws.arreglo) {
        if (elemento != null) {
            String clave =
elemento.getProperty("CLAVE").toString();
            if (!clave.equals("CFE") && !clave.equals("EOL"))
{
                lista2.add(elemento);
            }
        }
    }

    if (lista2.size() > 1) {
        for (int i = 0; i < lista2.size(); i++) {
            listSolFiltrada.add(
                new ListSoluciones(
Integer.parseInt(lista2.get(i).getProperty("SOLUCION_OPCIONES_ID")).toString())
,
                lista2.get(i).getProperty("NOMBRE").toString(),
                lista2.get(i).getProperty("CLAVE").toString()
            );
        }
    } else if (lista2.size() == 1) {
        solucion_id =
Integer.parseInt(lista2.get(0).getProperty("SOLUCION_OPCIONES_ID").toString())
    }
}

```

```

;

        nombre_sol =
lista2.get(0).getProperty("NOMBRE").toString();
        clave_sol =
lista2.get(0).getProperty("CLAVE").toString();
        SharedPreferencesManager preferencesManager =
SharedPreferencesManager.getInstance(getApplicationContext());
        preferencesManager.saveNOMBRE_BD("NOMBRE_BD",
NOMBRE_BD);
        new GeofenceBroadcastReceiver();
        GeofenceBroadcastReceiver.Nombre_BD = NOMBRE_BD;
    }

    AlertDialog.Builder builder = new
AlertDialog.Builder(context);
    builder.setTitle("Mensaje de aplicación");
    builder.setMessage("¿Se solucionó?");

    builder.setPositiveButton("Si", new
DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which)
{
            solucionado = 1;
            MostrarDialogListSoluciones(v, groupPosition);
        }
    });

    builder.setNegativeButton("No", new
DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which)
{
            solucionado = 0;
            MostrarDialogListSolucionesFiltrado(v,
groupPosition);
        }
    });
}

```

```

        }

    });

    AlertDialog dialog = builder.create();
    dialog.show();
    dialog.setCancelable(false);

    return true;
}

} );
super.onPostExecute(o);
}

}

@Override
protected void onSaveInstanceState(@NonNull Bundle outState) {
    super.onSaveInstanceState(outState);
    outState.putString("usuario", usuario);
    outState.putString("sucursal_id", sucursal_id);
    outState.putString("sucursalnombre", nombresucursal);
    outState.putString("NOMBRE_BD", NOMBRE_BD);
}

@Override
protected void onRestoreInstanceState(@NonNull Bundle savedInstanceState)
{
    super.onRestoreInstanceState(savedInstanceState);
    usuario = savedInstanceState.getString("usuario");
    sucursal_id = savedInstanceState.getString("sucursal_id");
    NOMBRE_BD = savedInstanceState.getString("NOMBRE_BD");
}

/**
 * Metodo buscarSolucionIDs
 * Este metodo consulta un WebService que lo que hace es que te busca
nuevamente el ID de solucion, por que

```

```

* este se pierde y de esta manera se recupera.
* @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
*/
private void buscarSolucionIDs() {
    WService2 WS = new WService2("BuscarSolucionIDs");
    WS.context = context;
    WS.FECHA = "0";
    WS.SUCURSAL_ID = sucursal_id;
    WS.FALTANTES_ID = faltantes_id;
    WS.NOMBRE_BD = NOMBRE_BD;
    WS.start();
    try {
        WS.join();
    } catch (InterruptedException e) {
        throw new RuntimeException(e);
    }
    buscarSolucionId = Integer.parseInt(WS.RESULT);
}

/**
 * Metodo setSolucionFaltante
 * Metodo que finaliza la solucion del faltante indicado
 * @return
 * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
*/
private int setSolucionFaltante() {
    SharedpreferencesManager preferencesManager =
SharedpreferencesManager.getInstance(getApplicationContext());
    NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD", "");
    WService2 WS = new WService2("setSolucionFaltante");
    WS.context = context;
    WS.FALTANTES_ID = faltantes_id;
    WS.NOMBRE_BD = NOMBRE_BD;
    WS.start();
    try {
        WS.join();
    }

```

```

        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
        solucion = WS.RESULT;
        return Integer.parseInt(solucion);
    }

    /**
     * Fragmento de codigo que en conjunto hace que se ejecute el dialog que
     muestra las soluciones
     * y en donde se pueden insertar las soluciones.
     * @param v
     * @param position
     * @author Dev. Dario Vidaña ft. Dev. Mauro Rangel
     */
    private void MostrarDialogListSoluciones(View v, int position) {
        LayoutInflater inflater = LayoutInflater.from(this);
        View customView = inflater.inflate(R.layout.custom_spinner_dialog,
null);
        Spinner spinner = customView.findViewById(R.id.spinner);

        configSpinner(spinner, position);
        configMostrarDialogo(customView, position);
    }

    private void MostrarDialogListSolucionesFiltrado(View v, int position) {
        LayoutInflater inflater = LayoutInflater.from(this);
        View customView = inflater.inflate(R.layout.custom_spinner_dialog,
null);
        Spinner spinner = customView.findViewById(R.id.spinner);

        configSpinnerFiltrado(spinner, position);
        configMostrarDialogo(customView, position);
    }

```

```

    /**
     * Metodo en el cual se configura el spinner dentro del DialogSolucion
     * @param spinner
     * @param position
     * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
     */
    private void configSpinner(Spinner spinner, int position) {
        List<String> nombressoluciones = listSol.stream()
            .map(ListSoluciones::getNombre)
            .collect(Collectors.toList());
        ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
            android.R.layout.simple_spinner_item, nombressoluciones);

        adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        spinner.setAdapter(adapter);
    }

    private void configSpinnerFiltrado(Spinner spinner, int position) {
        List<String> nombressoluciones = listSolFiltrada.stream()
            .map(ListSoluciones::getNombre)
            .collect(Collectors.toList());
        ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
            android.R.layout.simple_spinner_item, nombressoluciones);

        adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        spinner.setAdapter(adapter);
    }

    /**
     * Metodo en el cual se configura la vista del DialogSolucion y en donde

```

```

esta el boton "aceptar"
 * y "cancelar" junto con sus funciones
 * @param customView
 * @param position
 * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
 */
private void configMostrarDialogo(View customView, int position) {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setView(customView)
        .setTitle(ArregloProductos[position].getBarcode_number())
        .setMessage(crearMensajeProducto(ArregloProductos[position]))
        .setPositiveButton("Aceptar", (dialog, which) ->
maAceptar(customView, position))
        .setNegativeButton("Cancelar", (dialog, which) -> {
            listSol.clear();
            listSolFiltrada.clear();
        })
        .setCancelable(false)
        .create()
        .show();
}

/**
 * Metodo en el cual esta configurada la vista del Dialog de soluciones
 * @param producto
 * @return
 * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
 */
private String crearMensajeProducto(Products producto) {
    return "Nombre: " + producto.getDescription() + "\n"
        + "Stock Físico: " + producto.getCantidad() + "\n"
        + "Existencia Teórica: " + producto.getExistencia_teorica()+
"\n"
        + "Capacidad de empaque: " + producto.getCantidad() + "\n"
        + "Fecha último recibo: " + producto.getFecha_ult() + "\n";
}

```

```

    }

    /**
     * Este metodo se ejecuta al momento de pulsar sobre el boton Aceptar del
     dialog.
     * @param v
     * @param position
     * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
     */
    private void maAceptar(View v, int position) {
        insertarSolucion miHilo = new insertarSolucion();
        Spinner spinner = v.findViewById(R.id.spinner);
        String opcionSeleccionada = (String) spinner.getSelectedItem();
        int selectedPosition = spinner.getSelectedItemPosition();
        actualizarProductoSeleccionado(opcionSeleccionada, position,
selectedPosition);
        actualizarInterfaz();
        miHilo.start();
        dismissDialog();
        listSol.clear();
        listSolFiltrada.clear();
    }

    /**
     * Este metodo se ejecuta para poder actualizar correctamente el producto
     seleccionando, o al que
     * se le quiere dar solucion
     * @param opcion
     * @param position
     * @param selectedPosition
     * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
     */
    private void actualizarProductoSeleccionado(String opcion, int position,
int selectedPosition) {
        ArregloProductos[position].setSolucion(opcion);
        if (selectedPosition != AdapterView.INVALID_POSITION) {

```

```

        ListSoluciones solucionSeleccionada =
listSol.get(selectedPosition);
        solucion_id = solucionSeleccionada.getId();
        nombre_sol = solucionSeleccionada.getNombre();
        clave_sol = solucionSeleccionada.getClave();

SharedPreferencesManager.getInstance(getApplicationContext()).saveNOMBRE_BD("N
OMBRE_BD", NOMBRE_BD);
    }

articulo_id = ArregloProductos[position].getArticulo_Id();
}

/**
 * Metodo el cual actualiza la interfaz completa de FaltantesSolucion (si
es la primera se inserta
 * la hora de inicio)
 * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
 */
private void actualizarInterfaz() {
    insertarSolucion miHilo = new insertarSolucion();
    try {
        llenarHoras();
        if (hor1 == null || hor1.isEmpty()) {
            miHilo.start();
            subirHoraFaltantes("I");
            llenarHoras();

etiqueta.setBackgroundColor(getColor(R.color.txvEtiquetaFaltantesShopperRojo))
;
        etiqueta.setTextColor(Color.WHITE);

notificationHelpR.sendHighPriorityNotification(getString(R.string.NotifyPushA
ctividadFaltantesIniciada),
                getString(R.string.NotifyPushHoraInicio) + " \t" +
Hora, MainActivity.class);
        // recreate();
    }
}

```

```

        finish();
        startActivity(getIntent());
        overridePendingTransition(0, 0);
    }
} catch (InterruptedException e) {
    e.printStackTrace();
}

}

/**
 * Metodo que se ejecuta despues de que se cerro el dialog.
 * Lo que hace este metodo es actualizar la vista de la lista para que los
cambios se vean
 * reflejados de inmediato.
 * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
 */
private void dismissDialog() {
    ((BaseExpandableListAdapter)
expandableListView.getExpandableListAdapter()).notifyDataSetChanged();

}

/**
 * Clase en donde se usa hilos para optimizar la aplicacion a la hora de
insertar una solucion
 * en este caso consulta el WebService insertarSolucion que lo que hace es
que inserta un articulo
 * de una solucion en el sistema
 * @author: Dev. Dario Vidaña ft. Dev. Mauro Rangel
 */
private class insertarSolucion extends Thread {
    @Override
    public void run() {
        WService2 ws = new WService2("InsertarSolucion");
        ws.context = context;
        ws.FECHA_SOLUCION = getFecha();
    }
}

```

```

        ws.SUCURSAL_ID = sucursal_id;
        ws.FALTANTES_ID = faltantes_id;
        ws.USUARIO_CREACION = usuario;
        ws.FECHA_HORA_CREACION = getFechaHora();
        ws.ARTICULO_ID = articulo_id;
        ws.SOLUCION_OPCIONES_ID = Integer.toString(solucion_id);
        ws.SOLUCIONADO = solucionado;
        ws.NOMBRE_BD = NOMBRE_BD;

        ws.start();
        try {
            ws.join();
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
        if (!ws.RESULT.equals(""))
            faltante_id = Integer.parseInt(ws.RESULT);
    }
}

}

```

## Código: Adaptador de la lista expandible de Faltantes Solucion.

```

package com.example.inmex_ver1.Adaptadores;

import android.annotation.SuppressLint;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.BaseExpandableListAdapter;
import android.widget.TextView;

```

```

import androidx.core.content.ContextCompat;

import com.example.inmex_ver1.ProductoPrecio.Products;
import com.example.inmex_ver1.R;
import com.squareup.picasso.Picasso;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

import de.hdodenhof.circleimageview.CircleImageView;

/**
 * Esta clase tiene como principal funcionalidad crear un adaptador
personalizado para los datos
 * que se necesitan visualizar en la pantalla de FALTANTESSHOPPER
 */
public class CustomExpandableListAdapterProductosSolucion extends
BaseExpandableListAdapter {
    private final Context context;
    private final List<Integer> list;
    private final HashMap<Integer, Products> expandibleList;

    /**
     * Este es el constructor de la clase la cual pide tres parametros, los
cuales son:
     * Context: el contexto que necesita la clase para mostrar los datos en la
actividad
     * list: es una lista de cadenas que contienen identificadores para los
elementos a mostrar
     * expandibleList: Es el arreglo de objetos de productos que contiene los
datos que visualizarán en la vista
    */
    public CustomExpandableListAdapterProductosSolucion(Context context,
List<Integer> list, HashMap<Integer, Products> expandibleList) {

```

```
    this.context = context;
    this.list = list;
    this.expandibleList = expandibleList;
}

public CustomExpandibleListAdapterProductosSolucion(Context context) {
    this.context = context;
    this.list = new ArrayList<>();
    this.expandibleList = new HashMap<>();
}

@Override
public int getGroupCount() {
    return this.list.size();
}

@Override
public int getChildrenCount(int groupPosition) {
    return 1;
}

@Override
public Object getGroup(int groupPosition) {
    return this.list.get(groupPosition);
}

@Override
public Object getChild(int groupPosition, int childPosition) {
    return this.expandibleList.get(this.list.get(groupPosition));
}

@Override
public long getGroupId(int groupPosition) {
    return groupPosition;
}
```

```

@Override
public long getChildId(int groupPosition, int childPosition) {
    return childPosition;
}

@Override
public boolean hasStableIds() {
    return false;
}

@Override
public void notifyDataSetChanged() {
    super.notifyDataSetChanged();
}

/**
 * En este metodo se agregan los diferentes elementos para cada uno de los
objetos
 * Osea se agrega un renglon o item de la vista por cada objeto dentro del
arreglo
 * que contiene los datos
 */
@SuppressLint("InflateParams")
@Override
public View getGroupView(int groupPosition, boolean isExpanded, View
convertView,
                           ViewGroup parent) {
    Products proveedor = (Products) getChild(groupPosition, 0);
    if(convertView == null){
        LayoutInflater layoutInflater =
(LayoutInflater)this.context.getSystemService(
                Context.LAYOUT_INFLATER_SERVICE);
        convertView = layoutInflater.inflate(R.layout.list_group2,null);
    }
}

```

```

        TextView txtNombre = convertView.findViewById(R.id.nomGroupProveedor);
        TextView txtHourEnt1 = convertView.findViewById(R.id.Horas);
        String texto;

        txtNombre.setText(proveedor.getBarcode_number());
        texto = "SKU: \t"+proveedor.getBarcode_number() +"\tNombre:"+
" "+proveedor.getDescription()+"\n"+
" Solucion:\t "+proveedor.getSolucion()+"\t";
        txtHourEnt1.setText(texto);

        boolean hasSolucion = proveedor.getSolucion() != null &&
!proveedor.getSolucion().isEmpty();
        convertView.setEnabled(!hasSolucion);

        if (hasSolucion) {
            convertView.setBackgroundColor(ContextCompat.getColor(context,
R.color.contestadoVerde));
        } else {
            convertView.setBackgroundColor(ContextCompat.getColor(context,
R.color.white));
        }

        return convertView;
    }

    /**
     * Este metodo es el que obtiene la vista para cuando algun elemento del
     expandable list es clickeado
     * y muestre en este caso, el nombre del articulo y su SKU junto con la
     cantidad y el precio
     */
}

@SuppressWarnings("InflateParams")
@Override

```

```

    public View getChildView(int groupPosition, int childPosition, boolean
isLastChild,
                           View convertView, ViewGroup parent) {
        final Products proveedor = (Products)
getChild(groupPosition,childPosition);
        if(convertView == null){
            LayoutInflater layoutInflater =
(LayoutInflater)this.context.getSystemService(
                Context.LAYOUT_INFLATER_SERVICE);
            convertView = layoutInflater.inflate(R.layout.list_item2,null);
        }
        CircleImageView imageView = convertView.findViewById(R.id.circleImg);
        TextView tvTitle = convertView.findViewById(R.id.titulo);
        TextView tvCantidad = convertView.findViewById(R.id.cantidad);
        TextView tvPrecio = convertView.findViewById(R.id.precio);
        Picasso.get().load(proveedor.getImages()[0]).into(imageView);
        String title = "Titulo: \t" + proveedor.getTitle();
        String cantidad = "Cantidad: \t" + proveedor.getCantidad();
        String precio = "Precio: \t" + proveedor.getPrecio();
        tvTitle.setText(title);
        tvCantidad.setText(cantidad);
        tvPrecio.setText(precio);
        Animation animation = AnimationUtils.loadAnimation(context,
android.R.anim.fade_in);
        convertView.startAnimation(animation);
        return convertView;
    }

    @Override
    public boolean isChildSelectable(int groupPosition, int childPosition) {
        return true;
    }
}

```

## Código: Adaptador de la lista expandible de Faltantes Solucion después de finaliza la actividad.

```
package com.example.inmex_ver1.Faltantes;

import android.annotation.SuppressLint;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.BaseExpandableListAdapter;
import android.widget.TextView;

import androidx.core.content.ContextCompat;

import com.example.inmex_ver1.ProductoPrecio.Products;
import com.example.inmex_ver1.R;
import com.squareup.picasso.Picasso;

import java.util.ArrayList;
import java.util.List;

import de.hdodenhof.circleimageview.CircleImageView;

public class CustomExpandableListAdapterProductosSinIntegerSolucion extends BaseExpandableListAdapter {
    private final Context context;
    private final List<Products> list;

    /**
     * Este es el constructor de la clase la cual pide tres parametros, los cuales son:
     * Context: el contexto que necesita la clase para mostrar los datos en la actividad
     * list: es una lista de cadenas que contienen identificadores para los

```

```
elementos a mostrar

    * expandibleList: Es el arreglo de objetos de productos que contiene los
datos que visualizarán en la vista

    **/


    /**
     * Clase creada únicamente para mostrar la lista de objetos en Faltantes
shopper después de haber necesitado el ordenamiento
     */
    public CustomExpandibleListAdapterProductosSinIntegerSolucion(Context
context, List<Products> list) {
        this.context = context;
        this.list = list;
    }

    public CustomExpandibleListAdapterProductosSinIntegerSolucion(Context
context) {
        this.context = context;
        this.list = new ArrayList<>();
    }

    @Override
    public int getGroupCount() {
        return this.list.size();
    }

    @Override
    public int getChildrenCount(int groupPosition) {
        return 1;
    }

    @Override
    public Object getGroup(int groupPosition) {
        return this.list.get(groupPosition);
    }
```

```

@Override
public Object getChild(int groupPosition, int childPosition) {
    return this.list.get(groupPosition);
}

@Override
public long getGroupId(int groupPosition) {
    return groupPosition;
}

@Override
public long getChildId(int groupPosition, int childPosition) {
    return childPosition;
}

@Override
public boolean hasStableIds() {
    return false;
}

@Override
public void notifyDataSetChanged() {
    super.notifyDataSetChanged();
}

/**
 * En este metodo se agregan los diferentes elementos para cada uno de los
objetos
 * Osea se agrega un renglon o item de la vista por cada objeto dentro del
arreglo
 * que contiene los datos
 */
@SuppressWarnings("InflateParams")
@Override
public View getGroupView(int groupPosition, boolean isExpanded, View

```

```

convertView,
                ViewGroup parent) {
    Products proveedor = (Products) getChild(groupPosition, 0);
    if(convertView == null){
        LayoutInflater layoutInflater =
(LayoutInflater)this.context.getSystemService(
            Context.LAYOUT_INFLATER_SERVICE);
        convertView = layoutInflater.inflate(R.layout.list_group2,null);
    }
    TextView txtNombre = convertView.findViewById(R.id.nomGroupProveedor);
    TextView txtHourEnt1 = convertView.findViewById(R.id.Horas);
    String texto;

    txtNombre.setText(proveedor.getBarcode_number());
    texto = "SKU: \t"+proveedor.getBarcode_number() +"\tNombre:
"+proveedor.getDescription()+"\n"
            +"Solucion:\t "+proveedor.getSolucion()+"\t";
    txtHourEnt1.setText(texto);

    boolean hasSolucion = proveedor.getSolucion() != null &&
!proveedor.getSolucion().isEmpty();
    convertView.setEnabled(!hasSolucion);

    if (hasSolucion) {
        convertView.setBackgroundColor(ContextCompat.getColor(context,
R.color.contestadoVerde));
    } else {
        convertView.setBackgroundColor(ContextCompat.getColor(context,
R.color.white));
    }

    return convertView;
}

/**
 * Este metodo es el que obtiene la vista para cuando algun elemento del

```

```

expandable list es clickeado
    * y muestre en este caso, el nombre del articulo y su SKU junto con la
cantidad y el precio
    **/


    @SuppressLint("InflateParams")
    @Override
    public View getChildView(int groupPosition, int childPosition, boolean
isLastChild,
                           View convertView, ViewGroup parent) {
        final Products proveedor = (Products)
getChild(groupPosition,childPosition);
        if(convertView == null){
            LayoutInflater layoutInflater =
(LayoutInflater) this.context.getSystemService(
                Context.LAYOUT_INFLATER_SERVICE);
            convertView = layoutInflater.inflate(R.layout.list_item2,null);
        }
        CircleImageView imageView = convertView.findViewById(R.id.circleImg);
        TextView tvTitle = convertView.findViewById(R.id.titulo);
        TextView tvCantidad = convertView.findViewById(R.id.cantidad);
        TextView tvPrecio = convertView.findViewById(R.id.precio);
        Picasso.get().load(proveedor.getImages()[0]).into(imageView);
        String title = "Titulo: \t" + proveedor.getTitle();
        String cantidad = "Cantidad: \t" + proveedor.getCantidad();
        String precio = "Precio: \t" + proveedor.getPrecio();
        tvTitle.setText(title);
        tvCantidad.setText(cantidad);
        tvPrecio.setText(precio);
        Animation animation = AnimationUtils.loadAnimation(context,
android.R.anim.fade_in);
        convertView.startAnimation(animation);
        return convertView;
    }

```

```
    @Override
    public boolean isChildSelectable(int groupPosition, int childPosition) {
        return false;
    }
}
```

## Código: Módulo Reporte, clase en la que se visualiza grafica del módulo Faltantes Solucion.

```
package com.example.inmex_ver1.Reportes;

import static java.security.AccessController.getContext;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.drawable.Drawable;
import android.net.Uri;
import android.os.Bundle;
import android.os.Handler;
import android.text.format.DateFormat;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.LinearLayout;
import android.widget.RelativeLayout;
import android.widget ScrollView;
import android.widget.TextView;
```

```
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.content.FileProvider;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

import com.example.inmex_ver1.DetalleFaltantes;
import com.example.inmex_ver1.R;
import com.example.inmex_ver1.ReporteFaltantes;
import com.example.inmex_ver1.ServiciosWeb.WService;
import com.github.mikephil.charting.charts.PieChart;
import com.github.mikephil.charting.data.PieData;
import com.github.mikephil.charting.data.PieDataSet;
import com.github.mikephil.charting.data.PieEntry;
import com.github.mikephil.charting.formatter.PercentFormatter;
import com.google.android.material.snackbar.Snackbar;
import com.google.gson.Gson;
import com.google.gson.JsonArray;
import com.google.gson.JsonElement;
import com.google.gson.JsonObject;
import com.google.gson.JsonParser;

import org.w3c.dom.Text;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.sql.Array;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Random;
```

```
public class Reporte extends AppCompatActivity {

    private PieChart pieChart;
    private String response;
    private TextView descripcionConExistencia,descripcionConExistenciaN;
    private TextView tituloSin;
    private TextView tituloCon,tituloConN;
    private TextView descripcionSinExistencia;
    Bundle extra;
    private String sucursal_id;
    private String usuario;
    private String userID;
    private String NOMBRE_BD;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_reporte);

        this.setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
        try {
            Intent intent2 = this.getIntent();
            extra = intent2.getExtras();
            sucursal_id = extra.getString("sucursal");
            usuario = extra.getString("usuario");
            NOMBRE_BD = extra.getString("NOMBRE_BD");
        }
        catch(Exception ex) {
            Log.i("error",ex.getMessage());
        }

        pieChart = (PieChart) findViewById(R.id.pieChart);
        descripcionSinExistencia = findViewById(R.id.sinExistencia);
        descripcionConExistencia = findViewById(R.id.conExistencia);
        descripcionConExistenciaN = findViewById(R.id.conExistenciaN);
        tituloSin = findViewById(R.id.tituloSinExistencia);
    }
}
```

```

        tituloCon = findViewById(R.id.tituloConExistencia);
        tituloConN = findViewById(R.id.tituloConExistenciaN);

        try {
            obtenerRespuesta();
        } catch (InterruptedException e) {
            Snackbar.make(pieChart, e.getMessage(), Snackbar.LENGTH_SHORT)
                .setBackgroundTint(Color.RED)
                .setTextColor(Color.BLACK)
                .show();
        }
        reportGraphic();
    }

private void obtenerRespuesta() throws InterruptedException {
    WService ws = new WService("ReporteFaltantesSolucion");
    ws.FALTANTES_ID = "-1";
    ws.SUCURSAL_ID = sucursal_id;
    ws.ARTICULO_ID = "0";
    ws.FECHA_INI_FAL = "";
    ws.FECHA_FIN_FAL = "";
    ws.NOMBRE_BD = NOMBRE_BD;
    ws.start();
    ws.join();
    response = ws.RESULT;
}

private void reportGraphic() {

    Gson gson = new Gson();
    List<ReporteFaltantes> reporteFaltantes = ReadJson(response);
    String ColorSinExistencia = generarColorHexFijo("Sin existencia");
}

```

```

        String ColorConExistencia = generarColorHexFijo("Con existencia");
        String ColorconExistenciaN = generarColorHexFijo("Con existencia
negativa");

        int[] colores = {
            Color.parseColor(ColorSinExistencia),
            Color.parseColor(ColorConExistencia),
            Color.parseColor(ColorconExistenciaN)
        };

        int sinExis = 0;
        int conExis = 0;
        int conExisN = 0;
        int totalExist = 0;
        String productoSinExistencia = "";
        String productoConExistencia = "";
        String productoConExistenciaN = "";

        ArrayList<PieEntry> pieEntries = new ArrayList<>();
        for (int j = 0; j < reporteFaltantes.size(); j++) {
            for (int i = 0; i < reporteFaltantes.get(j).getDetalles().size();
i++) {
                if
                ((reporteFaltantes.get(j).detalles.get(i).getExistenciaTeorica() -
reporteFaltantes.get(j).detalles.get(i).getStockFisico()) == 0
                ) {
                    productoSinExistencia += "•" +
reporteFaltantes.get(j).detalles.get(i).getDescripcion() + ",\n\n";
                    sinExis++;
                } else
                if((reporteFaltantes.get(j).detalles.get(i).getExistenciaTeorica() -
reporteFaltantes.get(j).detalles.get(i).getStockFisico()) < 0) {
                    //existencia negativa
                    productoConExistenciaN += "•" +
reporteFaltantes.get(j).detalles.get(i).getDescripcion() + ",\n\n";
                    conExisN++;
                }
            }
        }
    }
}

```

```

        }

        else {
            productoConExistencia += "•" +
reporteFaltantes.get(j).detalles.get(i).getDescripcion() + ",\n\n";
            conExis++;
        }
    }

}

totalExist = sinExis + conExis + conExisN;

if (totalExist > 0) {
    float porcentageSinExis = (sinExis * 100f) / totalExist;
    float porcentageConExis = (conExis * 100f) / totalExist;
    float porcentageConExisN = (conExisN * 100f) / totalExist;

    if (sinExis > 0)
        pieEntries.add(new PieEntry(porcentageSinExis , "% \n
Productos sin Existencia"));
    if (conExis > 0)
        pieEntries.add(new PieEntry(porcentageConExis, "% \n Productos
con Existencia"));
    if(conExisN >0)
        pieEntries.add(new PieEntry(porcentageConExisN, "% \n
Productos con Existencia Negativa"));
}

// Crear el conjunto de datos del gráfico
PieDataSet dataSet = new PieDataSet(pieEntries, "");
dataSet.setColors(colores);

// Configurar el conjunto de datos del gráfico
PieData data = new PieData(dataSet);
data.setValueTextSize(12f);
data.setValueFormatter(new PercentFormatter());

```

```

// Establecer los datos en el gráfico
pieChart.setData(data);

pieChart.getDescription().setEnabled(true);
pieChart.getDescription().setText("Solución Faltantes");

pieChart.getDescription().setTextSize(12f);
pieChart.getDescription().setTextColor(Color.GRAY);
pieChart.getDescription().setEnabled(true);
pieChart.setEntryLabelTextSize(10f);
pieChart.setEntryLabelColor(Color.BLACK);

pieChart.setDrawHoleEnabled(true);
pieChart.setTransparentCircleRadius(20f);
pieChart.setHoleRadius(20f);
pieChart.invalidate(); // Actualizar el gráfico

descripcionSinExistencia.append(productoSinExistencia);
descripcionConExistencia.append(productoConExistencia);
descripcionConExistenciaN.append(productoConExistenciaN);

tituloSin.setText("Sin existencias (" + sinExis + ")");
tituloCon.setText("Con existencias (" + conExis + ")");
tituloConN.setText("Con existencias negativas (" + conExisN + ")");

}

private List<ReporteFaltantes> ReadJson(String json) {
    // Parseamos el JSON
    JsonParser parser = new JsonParser();
    JsonObject jsonObject = parser.parse(json).getAsJsonObject();

    // Obtenemos el array de resultados
    JSONArray resultsArray = jsonObject.getAsJSONArray("results");

```

```

// Creamos un objeto ReporteFaltantes
List<ReporteFaltantes> lista = new ArrayList<>();

// Iteramos sobre los resultados
for (JsonElement resultElement : resultsArray) {
    JsonObject resultObject = resultElement.getAsJsonObject();
    ReporteFaltantes reporteFaltantes = new ReporteFaltantes();
    // Obtenemos el FALTANTES_ID
    String faltantesId =
resultObject.get("FALTANTES_ID").getAsString();
    reporteFaltantes.setFaltantesId(faltantesId);

    // Obtenemos el array de detalles
    JsonArray detallesArray = resultObject.getAsJsonArray("detalles");

    // Creamos una lista de DetalleFaltantes
    List<DetalleFaltantes> detallesList = new ArrayList<>();

    // Iteramos sobre los detalles
    for (JsonElement detalleElement : detallesArray) {
        JsonObject detalleObject = detalleElement.getAsJsonObject();

        // Creamos un objeto DetalleFaltantes
        DetalleFaltantes detalleFaltantes = new DetalleFaltantes();

        // Asignamos los valores del detalle

        detalleFaltantes.setFaltantesDetalleId(getStringOrDefault(detalleObject,
"DETALLE_ID", ""));
        detalleFaltantes.setArticuloId(getStringOrDefault(detalleObject,
"ARTICULO_ID", ""));
        detalleFaltantes.setSku(getStringOrDefault(detalleObject,
"SKU", ""));
        detalleFaltantes.setNombre(getStringOrDefault(detalleObject,
"NOMBRE", ""));

```

```

detalleFaltantes.setDescripcion(getStringOrDefault(detalleObject,
"DESCRIPCION", ""));

detalleFaltantes.setStockFisico(getIntOrDefault(detalleObject,
"STOCK_FISICO", 0));

detalleFaltantes.setPrecioArticulo(getStringOrDefault(detalleObject,
"PRECIO_ARTICULO", ""));
        detalleFaltantes.setImagen(getStringOrDefault(detalleObject,
"IMAGEN", ""));

detalleFaltantes.setSolucion(getStringOrDefault(detalleObject, "SOLUCION",
""));
        detalleFaltantes.setFecha(getStringOrDefault(detalleObject,
"FECHA", ""));

detalleFaltantes.setExistenciaTeorica(getDoubleOrDefault(detalleObject,
"EXISTENCIA_TEORICA", 0));

        // Agregamos el detalle a la lista
        detallesList.add(detalleFaltantes);
    }

        // Asignamos la lista de detalles al reporte
        reporteFaltantes.setDetalles(detallesList);
        lista.add(reporteFaltantes);
    }

    return lista;
}

private static String getStringOrDefault(JSONObject jsonObject, String
key, String defaultValue) {

```

```

        JsonElement element = jsonObject.get(key);
        return (element != null && !element.isJsonNull()) ?
element.getAsString() : defaultValue;
    }

    private static int getAsIntOrDefault(JsonObject jsonObject, String key,
int defaultValue) {
    JsonElement element = jsonObject.get(key);
    return (element != null && !element.isJsonNull()) ? element.getAsInt()
: defaultValue;
}

private static double getAsDoubleOrDefault(JsonObject jsonObject, String
key, double defaultValue) {
    JsonElement element = jsonObject.get(key);
    return (element != null && !element.isJsonNull()) ?
element.getAsDouble() : defaultValue;
}

public static String generarColorHexFijo(String tipo) {
    String colorHex = "";
    switch (tipo) {
        case "Sin existencia":
            colorHex = "#A5A5A5"; // R:165 G:165 B:165
            break;
        case "Con existencia":
            colorHex = "#D1560B"; // R:209 G:86 B:11
            break;
        case "Con existencia negativa":
            colorHex = "#F69240";
            break;
        default:
            // Si se especifica un tipo de color incorrecto, se devuelve
            // un color negro
            colorHex = "#000000";
            break;
    }
}

```

```
        }

        return colorHex;
    }

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.share, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {

    if (item.getItemId() == R.id.share) {
        // Convertir la vista raíz en una imagen

        Bitmap bitmap;// = viewToBitmap(findViewById(R.id.sv));

        bitmap = captureFullScrollView(findViewById(R.id.sv),
            findViewById(R.id.relativeLayout));

        // Texto que deseas incluir en el mensaje
        String texto = "Reporte Faltantes por " + usuario + " " +
getFechaHora();

        // Compartir la imagen
        compartirImagen(bitmap, texto);

    }

    return super.onOptionsItemSelected(item);
}

public Bitmap viewToBitmap(View view) {
```

```

        Bitmap bitmap = Bitmap.createBitmap(view.getWidth(), view.getHeight(),
Bitmap.Config.ARGB_8888);
        if (bitmap != null) {
            Canvas canvas = new Canvas(bitmap);
            view.draw(canvas);
        } else {
            // El bitmap no se creó correctamente, maneja la situación aquí
            Log.e("osvaldo", "No se pudo crear el Bitmap");
        }
        return bitmap;
    }

    //version funcional
    private void compartirImagen(Bitmap bitmap, String texto) {
        try {
            // Guardar la imagen en la caché interna
            File cachePath = new File(getCacheDir(), "images");
            cachePath.mkdirs(); // Asegúrate de que la carpeta exista
            File imagePath = new File(cachePath, "image.png");
            FileOutputStream stream = new FileOutputStream(imagePath);
            bitmap.compress(Bitmap.CompressFormat.PNG, 100, stream);
            stream.close();

            // Obtener el URI del archivo utilizando FileProvider
            Uri contentUri = FileProvider.getUriForFile(this,
"com.example.inmex_ver1.provider", imagePath);

            // Crear el intent para compartir
            Intent shareIntent = new Intent(Intent.ACTION_SEND);
            shareIntent.setType("image/*");
            shareIntent.putExtra(Intent.EXTRA_STREAM, contentUri);
            shareIntent.putExtra(Intent.EXTRA_TEXT, texto);
            shareIntent.addFlags(Intent.FLAG_GRANT_READ_URI_PERMISSION);

            // Iniciar la actividad de compartir
            startActivity(Intent.createChooser(shareIntent, "Compartir"))
        }
    }
}

```

```

reporte" ));

} catch (IOException e) {
    e.printStackTrace();
}

}

private Bitmap scrollViewToBitmap(ScrollView scrollView, int y, int
height) {
    Bitmap bitmap = Bitmap.createBitmap(scrollView.getWidth(), height,
Bitmap.Config.ARGB_8888);
    Canvas canvas = new Canvas(bitmap);
    canvas.translate(0, -y);
    scrollView.draw(canvas);
    return bitmap;
}

public Bitmap captureFullScrollView(ScrollView scrollView, RelativeLayout
relativeLayout) {
    int totalHeight = scrollView.getChildAt(0).getHeight();
    int visibleHeight = scrollView.getHeight();

    // Captura el contenido del RelativeLayout como un bitmap
    Bitmap relativeBitmap = viewToBitmap(relativeLayout);

    // Revisa si el contenido del ScrollView cabe completamente en la
    pantalla
    if (totalHeight <= visibleHeight) {
        return relativeBitmap;
    }

    Bitmap fullBitmap = Bitmap.createBitmap(scrollView.getWidth(),
totalHeight, Bitmap.Config.ARGB_8888);

```

```

Canvas fullCanvas = new Canvas(fullBitmap);

// Guardar la posición actual de scroll
int savedScrollY = scrollView.getScrollY();

// Captura cada segmento del ScrollView
for (int scrollY = 0; scrollY < totalHeight; scrollY += visibleHeight)
{
    int height = Math.min(visibleHeight, totalHeight - scrollY);
    scrollView.scrollTo(0, scrollY);
    Bitmap segmentBitmap = scrollViewToBitmap(scrollView, scrollY,
height);
    fullCanvas.drawBitmap(segmentBitmap, 0, scrollY, null);
}

// Restaurar la posición de scroll original
scrollView.scrollTo(0, savedScrollY);

// Combinar el bitmap del RelativeLayout con el bitmap del ScrollView
completo
return combineBitmaps(relativeBitmap, fullBitmap);
}

private Bitmap combineBitmaps(Bitmap bitmap1, Bitmap bitmap2) {
    int width = Math.max(bitmap1.getWidth(), bitmap2.getWidth());
    int totalHeight = bitmap1.getHeight() + bitmap2.getHeight();
    Bitmap combinedBitmap = Bitmap.createBitmap(width, totalHeight,
Bitmap.Config.ARGB_8888);
    Canvas canvas = new Canvas(combinedBitmap);
    canvas.drawBitmap(bitmap1, 0, 0, null);
    canvas.drawBitmap(bitmap2, 0, bitmap1.getHeight(), null);
    return combinedBitmap;
}

public String getFechaHora() {

```

```

        String fecha;

        Date d = new Date();
        fecha = (String) DateFormat.format("yyyy-MM-dd HH:mm:ss",
d.getTime());
        return fecha;
    }

}

```

## Código: Solucion a Valores nulos de Mistery Shopper.

```

app\src\main\java...\A_BIENVENIDO.java 768 769         clase = P_MS.class;
app\src\...\Adaptador_Bienvenido.java 769 770         break;
app\src\main...\Adaptador_Chequeo.java 770 771
+.. @@ -774,6 +775,24 @@ public class A_MISTERY_SHOPPER extends AppCompatActivity {
app\...\Adaptador_MysteryShopper.java 774 775         }
app\src\main...\AdapterCaducidad.java 775 776     }
776 777
+    private void BuscarIDusuario(){
778 |    SharedPreferencesManager preferencesManager = SharedPreferencesManager.getInstance(getApplicationContext());
779 |    NOMBRE_BD = preferencesManager.getNOMBRE_BD("NOMBRE_BD",NOMBRE_BD);
780 |    WService3 ws = new WService3("ObtenerUsuarioIDClave");
781 |    ws.USUARIO_CLAVE = usuario ;
782 |    ws.NOMBRE_BD = NOMBRE_BD;
783 |    ws.start();
784 |    try {
785 |        ws.join();
786 |    } catch (InterruptedException e) {
787 |        throw new RuntimeException(e);
788 |    }
789 |
790 |
791 |    userID = ws.RESULT;
792 |
793 |
794 |}
795 |
777 796    private void insertarIntents(Intent intent, String destino, int C) {
778 797        System.out.println("....."+C);
779 798        Toast.makeText(this, getString(R.string.MSSeleccionadoVariable1)+" "+ destino + getString(R.string.MSSeleccionadoVariable2), Toast.LENGTH_SHORT).sh
ow();

```

## Código: Solucion a Valores nulos de Producto Precio.

app\src\main\java\..\A_BIENVENIDO.java	77	77	@@ -77,7 +77,7 @@ public class A_PROCESO_PRODUCTO_PRECIO extends AppCompatActivity { Context context;
app\src\main\java\..\Adaptador_Bienvenido.java	78	78	ExtendedFloatingActionButton botonflotante;
app\src\main\java\..\Adaptador_Chequeo.java	79	79	AlertDialog.Builder builder;
app\src\main\java\..\Adaptador_MysteryShopper.java	80	-	String ID_user, IDProducto, NombreProducto, SKU, Descripcion, Estatus, SeRealizado, ID_Sucursal, NombreUsuario, check2, URLlink, ASIGNACION_DETALLE_ID, NombreImagen, NombreImagenV, NombreImageny, NombreImagenP, SoloNombreImagenS, SoloNombreImagenE, SoloNombreImageny, SoloNombreImagenP, SoloNombreImagen;
app\src\main\java\..\AdapterCaducidad.java	80	+	String ID_user, IDProducto, NombreProducto, SKU, Descripcion, Estatus, SeRealizado, ID_Sucursal, NombreUsuario, check2, URLlink, ASIGNACION_DETALLE_ID, NombreImagen, NombreImagenV, NombreImageny, NombreImagenP, SoloNombreImagenS, SoloNombreImagenE, SoloNombreImageny, SoloNombreImagenP, SoloNombreImagenP, user_ID;
a...\CustomExpandibleListAdapter.java	81	81	String encondeImagenV, encondeImagen, encondeImagenS, encondeImagenP, chIso = "";
app\src\main\java\..\ASISTENCIA.java	82	82	String valorGuardado = "", Opcionseleccionada = "", cbCoincide = "", check = "";
...\\A_INTRODUCIR_CADUCIDADES.java	83	83	TextView text_SKU, text_descripcion, textSeRealizacion, textEtiqueta, textVerificador, Mensaje, Mensajej, OP1, OP2, OP3, txtUbicacion;
app\src\main\java\..\A_CENSO_PROV.java	118	118	@@ -118,7 +118,8 @@ public class A_PROCESO_PRODUCTO_PRECIO extends AppCompatActivity { .....
app\src\main\java\..\A_MISTERY_SHOPPER.java	119	119	SeRealizado = extra.getString("SeRealizado");
app\src\main\java\..\A_MS_PRODUCTO_Precio.java	120	120	ID_Sucursal = extra.getString("ID_Sucursal"); NombreUsuario = extra.getString("NombreUsuario");
app\src\main\java\..\CTO_Precio.java	121	-	ID_user = extra.getString("ID_user");
	121	+	ID_user = extra.getString("USER_ID");
	122	+	user_ID = extra.getString("USER_ID");
	122	123	llamado = extra.getString("llamado");
	123	124	NOMBRE_BO = extra.getString("NOMBRE_BO");
	124	125	Bitmap image;
	125	.....	.....

## Código: Implementacion de botón Actualizar en Faltantes Solucion.

```
app\src\main\java\com\luisl.manejador\A_FALTANTES_SOLUCION.java 187 187
188 188     ) {
189 189     }
190 190 }
191 191
192 192 - 
192 192 +     Button btnActualizar = findViewById(R.id.btnActualizarSol);
193 193     Button btnFinish = findViewById(R.id.btnExit);
194 194 +
195 195 +
196 196         context = this;
197 197         Bundle extra = getIntent().getExtras();
198 198         usuario = extra.getString("usuario");
199 199 +
200 200 * @@@ -217,6 +219,11 @@ public class A_FALTANTES_SOLUCION extends CloseSession {
201 201             DialogoImagen dialogoImagen = new DialogoImagen(this, URLwhats);
202 202         });
203 203
204 204 +
205 205         //BOTON ACTUALIZAR
206 206         btnActualizar.setOnClickListener((View v) -> {
207 207             recreate();
208 208         });
209 209 +
210 210         // BOTON FINALIZAR
211 211         btnFinish.setOnClickListener((View v) -> {
212 212             if( ArregloProductos == null){
213 213 +
214 214 * @@@ -308,6 +315,7 @@ public class A_FALTANTES_SOLUCION extends CloseSession {
215 215             tvTotalSKU.setText("Total SKU's : " + listaProductos.size());
216 216         }
217 217
218 218 +
219 219         /**
220 220             * Metodo init
221 221             * Este metodo manda a llamar al metodo MostrarFaltantes
222 222 }
```