UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

Tema:

APLICACIÓN MÓVIL CON SENSORES DE MOVIMIENTO Y POSICIÓN PARA EL CIFRADO DE INFORMACIÓN PERSONAL EN DISPOSITIVOS MÓVILES

Trabajo de Integración Curricular Modalidad: Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Tecnologías de la Información

ÁREA: Software
LÍNEA DE INVESTIGACIÓN: Desarrollo de software
AUTOR: Samuel Alejandro Mejía Jácome
TUTOR: Ing. Franklin Oswaldo Mayorga Mayorga

Ambato - Ecuador
marzo – 2023

APROBACIÓN DEL TUTOR

En calidad de tutor del Trabajo de Integración Curricular con el tema: APLICACIÓN MÓVIL CON SENSORES DE MOVIMIENTO Y POSICIÓN PARA EL CIFRADO DE INFORMACIÓN PERSONAL EN DISPOSITIVOS MÓVILES, desarrollado bajo la modalidad Proyecto de Investigación por el señor Samuel Alejandro Mejía Jácome, estudiante de la Carrera de Tecnologías de la Información, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 de las segundas reformas al Reglamento para la ejecución de la Unidad de Integración Curricular y la obtención del título de tercer nivel, de grado en la Universidad Técnica de Ambato y el numeral 7.4 del respectivo instructivo del reglamento.

Ambato, marzo 2023.

Ing. Franklin Oswaldo Mayorga Mayorga, Mg

TUTOR

AUTORÍA

El presente trabajo de Integración Curricular titulado: APLICACIÓN MÓVIL CON SENSORES DE MOVIMIENTO Y POSICIÓN PARA EL CIFRADO DE INFORMACIÓN PERSONAL EN DISPOSITIVOS MÓVILES es absolutamente original, auténtico y personal. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, marzo 2023.

Samuel Alejandro Mejía Jácome

C.C. 1805326897

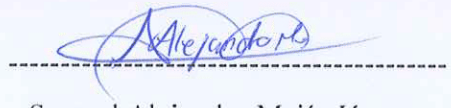
AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Integración Curricular como un documento disponible para la lectura, consulta y procesos de investigación.

Cedo los derechos de mi Trabajo de Integración Curricular en favor de la Universidad Técnica de Ambato, con fines de difusión pública. Además, autorizo su reproducción total o parcial dentro de las regulaciones de la institución.

Ambato, marzo 2023.



Samuel Alejandro Mejía Jácome

C.C. 1805326897

AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del Informe Final del Trabajo de Integración Curricular presentado por el señor Samuel Alejandro Mejía Jácome, estudiante de la Carrera de Tecnologías de la Información, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado APLICACIÓN MÓVIL CON SENSORES DE MOVIMIENTO Y POSICIÓN PARA EL CIFRADO DE INFORMACIÓN PERSONAL EN DISPOSITIVOS MÓVILES, nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 de las segundas reformas al Reglamento para la ejecución de la Unidad de Integración Curricular y la obtención del título de tercer nivel, de grado en la Universidad Técnica de Ambato y al numeral 7.6 del respectivo instructivo del reglamento. Para cuya constancia suscribimos, conjuntamente con la señora Presidenta del Tribunal.

Ambato, marzo 2023.

Firmado electrónicamente por:
ELSA PILAR URRUTIA URRUTIA

Ing. Elsa Pilar Urrutia Urrutia, Mg.
PRESIDENTE DEL TRIBUNAL

Firmado electrónicamente por:
JULIO ENRIQUE BALAREZO LOPEZ

Ing. Julio Enrique Balarezo López, PhD
PROFESOR CALIFICADOR

Firmado electrónicamente por:
CARLOS ISRAEL NUNEZ MIRANDA

Ing. Carlos Israel Nuñez Miranda, Mg.
PROFESOR CALIFICADOR

DEDICATORIA

El presente trabajo de investigación se lo dedico a Dios en primer lugar por otorgarme sabiduría, carácter y vigor a lo largo de mi trayectoria universitaria, a mi madre por ser la principal fuente de mi inspiración y fortaleza para avanzar fervientemente hasta alcanzar mis objetivos con humildad y transparencia.

A mis docentes, por el tiempo, paciencia y conocimiento impartido durante esta que ha sido mi etapa de desarrollo profesional.

Samuel Alejandro Mejía Jácome

AGRADECIMIENTO

Agradezco a Dios por darme sabiduría y salud en el transcurso de mi trayectoria universitaria, a mi madre, docentes y amigos por brindarme de su apoyo invaluable en los momentos más complicados.

A mi tutor de tesis Ing. Franklin Mayorga por la confianza e instrucción para el desarrollo del presente trabajo de investigación.

Samuel Alejandro Mejía Jácome

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN EJECUTIVO

En los últimos años, las capacidades de los dispositivos móviles han ido en aumento, junto con la aparición de problemas como el robo de datos y el fraude de identidad. Hoy en día, todos los usuarios tienen un dispositivo móvil que genera un alto nivel de interés para los atacantes. Este amplio interés se debe a que los usuarios almacenan documentos, fotos o videos sin utilizar mecanismos de protección.

El objetivo de este proyecto es implementar una aplicación móvil enfocada a encriptar datos almacenados localmente, mediante un mecanismo que se ejecuta a través de patrones de movimiento captados por sensores de los dispositivos o mediante comandos directos del usuario. El objetivo es mitigar los problemas relacionados con la exposición y el robo de datos a través de Internet o por el robo de dispositivos, lo que comprometería la integridad personal de los usuarios.

En el diseño de la aplicación móvil se utilizó la metodología MADLC y Kanban para realizar un seguimiento de las actividades realizadas durante el desarrollo, junto con la arquitectura MVVM para crear la estructura del proyecto, debido a la adaptabilidad con Android Studio y la comunicación en tiempo real. entre datos e interfaces. Además, se utilizó Cipher para cifrar o descifrar los datos, en conjunto con dos sensores integrados (acelerómetro, orientación) para reconocer el patrón de movimiento.

Palabras clave: Aplicación móvil, MVVM, Android Studio, Kotlin, sensores de Android, cifrado, cifrado.

ABSTRACT

In recent years, the capabilities of mobile devices have been increasing, along with the emergence of problems such as data theft and identity fraud. Today, all users have a mobile device that generates a high level of interest for attackers. This broad interest is due to the fact that users store documents, photos or videos without using protection mechanisms.

The objective of this project is to implement a mobile application focused on encrypting data stored locally, through a mechanism that is executed through movement patterns captured by device sensors or through direct user commands. The objective is to mitigate problems related to data exposure and theft through the internet or through device theft, which would compromise the personal integrity of users.
In designing the mobile application, the MADLC methodology and Kanban were used to track the activities carried out during development, along with the MVVM architecture to create the project structure, due to its adaptability with Android Studio and real-time communication between data and interfaces. Additionally, Cipher was used to encrypt or decrypt data, in conjunction with two integrated sensors (accelerometer, orientation) to recognize the movement pattern.

Keywords: Mobile application, MVVM, Android Studio, Kotlin, Android sensors, encryption, enciphering..

CAPÍTULO I

MARCO TEÓRICO

1.1 Tema de Investigación

APLICACIÓN MÓVIL CON SENSORES DE MOVIMIENTO Y POSICIÓN PARA EL CIFRADO DE INFORMACIÓN PERSONAL EN DISPOSITIVOS MÓVILES.

1.1.1 Planteamiento del Problema

A diario la tecnología cambia radicalmente el mundo, su constante evolución permite que se desarrollen nuevas herramientas y servicios adaptables en cualquier dispositivo electrónico. De manera que un dispositivo móvil puede almacenar una gran cantidad de información como cuentas bancarias o contenido multimedia procedente de sitios web, sin embargo, tal capacidad sugiere un problema crítico para la privacidad de los usuarios, lo que quiere decir que en caso de eventos como hackeos, pérdida o robo de dispositivos celulares los datos personales quedan totalmente expuestos [1]. Para ilustrar esto, en España el Ministerio de Interior informa que al menos 180.000 móviles fueron sustraídos a los ciudadanos, de los cuales el 89% almacenaban correos electrónicos y el 90% fotos y videos privados. Ante la situación planteada, surgen fenómenos tales como la apropiación, extorsión y suplantación de identidad [2].

En ese mismo sentido en países de América Latina y específicamente Argentina, de acuerdo con los reportajes de La Vanguardia la población afirma con un total de 1700 denuncias públicas que han sido víctimas de apropiación de identidad por pérdida de sus identificaciones personales o dispositivos móviles, y por lo tanto víctimas de fraudes bancarios, se deduce entonces que en muchos de los casos la información que almacenaban en sus teléfonos tiene amplia relación con cuentas bancarias o transacciones, que permite a individuos mal intencionados realizar fraudes sin repercusión alguna [3].

De manera semejante la filtración de datos no ocurre únicamente por robo de los dispositivos sino por ataques externos utilizando herramientas o procesos informáticos, en Ecuador, por ejemplo, expertos en ciberseguridad señalan que a causa del COVID-19 el robo de información aumento un 93 por ciento, siendo los principales blancos, los dispositivos móviles, esto a causa del tipo de datos que esta clase de dispositivos contiene y es capaz de acceder. Dichos ataques son ejecutados mediante una nueva clase de ataque denominada como SMSishing, enfocado en el engaño de los usuarios para enviarlos a sitios fraudulentos, a través de mensajes de texto [4].

1.2 Antecedentes Investigativos

Revisando la documentación digital de algunos repositorios se ha encontrado los siguientes trabajos artículos científicos o de investigación referida con el tema que se está investigando y que servirán de apoyo en el desarrollo del trabajo de investigación:

Según Dennis Denis, Daryl D. Cruz Flores, Yarelys Ferrer-Sánchez y Fermín L. Felipe Tamé [5] en su trabajo "Potencialidades de los celulares inteligentes para investigaciones biológicas. Parte 1: Sensores integrados", trabajo realizado como investigación para la Universidad de la Habana. En el año 2021 cuya finalidad es determinar la potencialidad de los celulares inteligentes actuales en áreas de la investigación concluye que:

- Los beneficios potenciales de las aplicaciones de celulares se pueden resumir en 1) su amplio alcance geográfico, 2) el incremento en eficiencia - menos tiempo de obtención, 3) datos más precisos y de más calidad, al evitarse errores humanos de transcripción, 4) mayor seguridad en la conservación de los datos, 5) la universalidad de los modelos de equipos estándares entre regiones y, 6) una amplia variedad de sensores integrados utilizables, muchos de los cuales no dependen de conexión a torres de señal telefónica o internet.
- El empleo de sensores móviles asociados a los teléfonos inteligentes aún está en una etapa que puede ser considerada como temprana, y excepto en algunas ramas científicas, aún no se utilizan con la frecuencia que pudiera esperarse dadas sus ventajas. Sin embargo, se está en un momento apasionante para la

incorporación de las nuevas tecnologías emergentes en las investigaciones de campo, sin necesidad de grandes inversiones, con el objetivo de lograr resultados innovadores e impactantes. Los teléfonos celulares, que tan intempestivamente han irrumpido la vida diaria, tienen potencialidades para ir más allá y convertirse en importantes herramientas de trabajo científico. Aprender a hacer un uso más eficiente de estas herramientas puede revolucionar la manera en que los investigadores trabajan.

- Es real e innegable que los equipos especializados comerciales para la toma de datos científicos (equipos dedicados) pueden tener, en su mayoría, mayor precisión y confiabilidad en los servicios que ofertan. No puede esperarse la misma calidad en los datos de los sensores de los teléfonos inteligentes, diseñados con otros objetivos. Sin embargo, de acuerdo con protocolos apropiados de validación y la selección cuidadosa de las aplicaciones para la toma de datos, estos sensores pueden ser alternativas razonables, de calidad suficiente para muchos trabajos de campo que no requieren de precisiones demasiado altas. Además, pudieran ser empleados por investigadores y proyectos que no cuenten con un soporte económico para sobrellevar los grandes gastos que implica la tecnología ideal.

----.

Según Martín Monteiro, Cecilia Cabeza y Arturo C. Martí [6] en su tesis "Aplicaciones de sensores vestibles y teléfonos inteligentes en el bienestar personal: Cuantificación de la actividad física y control de la práctica de mindfulness" trabajo realizado como tesis para la Universidad de Zaragoza en el año 2019, concluye que:

- A lo largo del desarrollo de esta tesis doctoral se ha comprobado la eficacia de los dispositivos vestibles como herramientas de estudio en la mejora de calidad de vida de los usuarios finales. Concretamente hemos validado estos sensores en la monitorización de la actividad física y en la meditación sedente. En ambos casos los sensores han aportado información valiosa que concluye en resultados favorables. A partir de estos resultados hemos podido aportar nuevos métodos de análisis o nuevos dispositivos con fines más específicos para la adquisición de algunas variables fisiológicas.

Según Samuel Fraile Lobato en su tesis "Uso de los teléfonos inteligentes para la realización de prácticas de laboratorio fuera del centro educativo" trabajo realizado para la Universidad de Valladolid en el año 2019 [7]. Este menciona algunos aspectos para tener en cuenta en lo referente a la importancia de los sensores integrados en los dispositivos móviles y menciona:

- "…Después hablamos de la incorporación en las aulas de los teléfonos inteligentes como herramientas pedagógicas, explicamos las funciones de los sensores que 5 incluyen, con los que podemos adquirir datos del mundo real y de las aplicaciones que controlan estos sensores y que nos permiten realizar experiencias prácticas fuera del centro educativo…"
- "…es la captación de fenómenos físicos mediante los sensores que incorporan los teléfonos. Estos componentes pueden captar sonidos, imágenes, efectos magnéticos y otros que posteriormente son procesados y pueden ser interpretados de distintas maneras."
- "…Existen multitud de aplicaciones para los teléfonos inteligentes que nos permiten tener acceso a los datos que proporcionan los sensores. Con ellas podemos obtener los valores que registran el acelerómetro y el giroscopio, el efecto del campo magnético terrestre sobre el magnetómetro, la presión atmosférica detectada por el barómetro o la ubicación del teléfono mediante los satélites GPS…"

Según Alber Montoya Benitez y Bayron Ospina en su tesis "Benchmark para determinar el sistema de cifrado con mejor rendimiento sobre dispositivos inteligentes" trabajo realizado para el Instituto Tecnológico Metropolitano en el año 2020 [8]. Estos mencionan un aspecto para tener en cuenta en lo referente a la importancia de proteger la información personal de dispositivos móviles y mencionan:

- "…Uno de los principales inconvenientes en las telecomunicaciones es la implementación de un sistema de autenticación no cifrado, o en algunos casos, el uso de un cifrado débil que puede ser fácil de atacar, los cuales son ineficientes para proteger información crítica. Por ejemplo, la aplicación WhatsApp utilizaba el sistema crypt5 o crypt7 para proteger las

conversaciones, o autenticación WEP en red inalámbrica, la cual, es muy frágil a ataques…"

Según el trabajo Presentado por Valentina Hernández Báuza de título "Tecnologías para la privacidad y la libertad de expresión: reglas sobre anonimato y cifrado Chile en el contexto latinoamericano" en el año 2017 [9]. Esta menciona algunos aspectos a tener en cuenta en lo referente a la protección de los datos a través de métodos de cifrado y menciona:

- "…El uso de herramientas que permitan el anonimato en línea y el cifrado de las comunicaciones se han identificado como técnicas efectivas para recuperar algo de control sobre mi información, minimizar los riesgos de la vida en conexión y garantizar el respeto y ejercicio de derechos como la libertad de expresión y la privacidad…"
- "…Por ello, las técnicas de cifrado, en conjunción con el anonimato, son necesarias para resguardar a un cúmulo de derechos fundamentales que están sujetos a las amenazas que han nacido de la mano del desarrollo tecnológico…"
- "…el resguardo de información como en las comunicaciones privadas, tiene como propósito disminuir los riesgos de una comunicación cuyos contenidos puedan ser interceptados, riesgos que incluyen acceso a datos sensibles asociados a la ejecución de acciones como comprar por internet, recibir datos médicos, acceder a cuentas de redes sociales, cuenta bancaria y, en general, todo tipo de actividad en línea que suponga el acceso a información y cuya extracción ilegítima pueda revelar detalles sumamente precisos de la vida privada y generar perjuicios como consecuencia..."

1.3 Fundamentación Teórica

1.3.1 Gestión de Procesos

La gestión de procesos en un sistema operativo típico implica muchas estructuras de datos y algoritmos complejos, pero no va mucho más allá del nivel de la gestión del proceso típico de estructura de datos. Android es similar en que el nivel de base de estructuras de control tiene el mismo aspecto [10].

1.3.2 Sensores Android

Los sensores están integrados en la mayoría de los dispositivos Android son capaces de proporcionar datos no procesados con exactitud y precisión, así mismo son útiles para determinar la posición y movimiento tridimensional de los dispositivos. De igual forma, los sensores pueden ser utilizados en aplicaciones meteorológicas para determinar la temperatura, humedad y dirección [11].

1.3.2.1 Acelerómetro

El acelerómetro mide la aceleración en los tres ejes físicos (x, y, z), respecto a la gravedad, siendo un sensor relativamente pequeño, sensible, y económico [12]. El uso principal de este sensor está dirigido a la detección de fuerzas dinámicas y estáticas de aceleración de automóviles [13]. Existen disponibles dos aspectos distintos de aceleración que mide el acelerómetro: la aceleración lineal y la aceleración centrífuga. En el primer caso, los datos del sensor se obtienen cuando existe un cambio de velocidad en la dirección de la señal y en el siguiente caso, los datos son recopilados por el desplazamiento de objetos en un círculo [14].

1.3.2.2 Giroscopio

El giroscopio es un sensor diseñado con la tecnología MEMS, encargada de calcular la aceleración por medio de placas de carácter capacitivo [15]. Se trata de un dispositivo capaz de detectar la velocidad angular, ángulo de rotación de un objeto, proveer estabilidad y mantener direcciones. El giroscopio obtiene los valores de la velocidad de rotación en base a los tres ejes de posicionamiento, siendo así que la unidad de medida estándar de los datos son los radianes [16]. La lectura de los datos del eje x refiere a la velocidad angular del borde más pequeño del dispositivo, el eje y expresa la velocidad angular del borde más largo, y el eje z la velocidad angular a lo largo de la pantalla [17]. Existen una variedad de giroscopios, por la que cada uno de ellos se utiliza dependiendo de la tecnología aplicada y el principio físico, sin embargo, a pesar de su gran variedad, no se encuentra disponible en los dispositivos móviles de gama baja o con una versión de Android muy antigua [16].

1.3.2.3 Magnetómetro

Este sensor obtiene la posición absoluta del dispositivo en los tres ejes físicos, lo que implica obtener los datos del sensor cuando el dispositivo se encuentra en diferentes posiciones. Primero, cuando la dirección de la pantalla está orientada al norte el valor para el eje X será 0° y cuando el dispositivo se encuentre en reposo, es decir con la pantalla hacia el cielo, el valor del eje Y es 0° [17].

1.3.2.4 Sensor de gravedad

Se encarga de medir la aceleración gravitacional de la tierra en el dispositivo móvil. De la misma forma que el acelerómetro la unidad de medición se encuentra en m/s^2 obteniendo los datos de los ejes x, y, z. En realidad, no existe un sensor de gravedad físico integrado en los dispositivos móviles, sino que se trata de un sensor de gravedad virtual generado mediante la combinación de filtros pasa bajo y sensores integrados [18].

1.3.2.5 Sensor vector de rotación

Es un sensor virtual que utiliza una combinación del acelerómetro y el giroscopio para generar una salida similar a un cuaternio que es una generalización del conjunto de número reales o una forma representativa de una rotación. El sensor indica el valor del ángulo alrededor de un determinado eje mediante un vector de rotación, el cual es transformado en una matriz para obtener un vector representativo de orientación en base a la inclinación y puntos cardinales terrestres [19].

1.3.2.6 Barómetro

El barómetro es un sensor fiable diseñado para determinar la altitud. Se trata de un dispositivo de posicionamiento que, a través de la presión de la atmósfera obtenida, determina el valor de la altitud, sin embargo, también es utilizado como un sensor de posicionamiento absoluto. Por otro lado, determinar la altitud en base a la presión atmosférica sugiere algunos problemas según investigaciones, ya que está relacionado

con factores climáticos como la localización, horario, etc. Dichos problemas implican obtener valores de medición de altitud con una desviación estándar de once metros, para lo cual es necesario aplicar correcciones barométricas que mejoren la precisión [20].

1.3.2.7 Luz Ambiental

Se trata de un sensor independiente utilizado para establecer el nivel de brillo de la pantalla de los dispositivos móviles. La mayor parte de estos sensores tienen un flujo de corriente variante, su variación depende de la intensidad de luz que se aplique [18]. Los sensores de luz ambiental son instalados en dispositivos móviles para detectar las condiciones de luz del entorno y ajustar la luz de la pantalla de manera que sea cómodo para el usuario, otros usos se le atribuyen en la detección de intensidad luminosa [21].

1.3.2.8 Sensor de Humedad

El sensor de humedad mide la cantidad de vapor de agua existente en el aire. Los datos obtenidos por este sensor son sensibles, debido a que los dispositivos móviles pueden sufrir daños al exponerse a niveles de humedad del entorno altos [22].

1.3.2.9 Sensor de Temperatura

Mide la temperatura del entorno en el que se encuentra el dispositivo, proporciona una salida que cambia en dependencia de la temperatura. Los datos obtenidos se procesan para ser datos digitales que puedan ser interpretados por microcontroladores, los cuales determinan y controlan los valores de la temperatura y umbral [22].

1.3.3 Funciones de Sensores

En la actualidad, aun con los avances tecnológicos, todavía en las aplicaciones web móviles no es directo el acceso a los datos de los sensores internos de los dispositivos para brindar información sensible al contexto. Estos sensores internos se utilizan para obtener información del dispositivo y, además, detectar cambios en el contexto, por ejemplo, la orientación, el posicionamiento, temperatura, dirección, etc. [23]

1.3.4 Reconocimiento de Actividad Humana (HAR)

El reconocimiento de actividad humana es un sistema de información en el cual sensores heterogéneos se encargan de clasificar las actividades efectuadas por efectuadas por los usuarios. El propósito de estos sensores está dirigido a predecir actividades humanas, capturar el estado del usuario, y del entorno. Siendo así que puede ser utilizado en diferentes aplicaciones de seguimiento de estado físico, detección de caídas, computación ubicua y seguridad. La estructura de su metodología compone las siguientes etapas: recopilación, procesamiento de datos, extracción de características y clasificación [12].

1.3.4.1 Bases de datos públicas HAR

Existen bases de datos que recopilan los datos de sensores integrados en dispositivos inteligentes, sin embargo, pocas de las bases de datos se encuentran disponibles, lo que interfiere en el desarrollo de la metodología HAR, para comparar los resultados obtenidos en cada fase y obtener resultados íntegros (predicciones acertadas). Debido a esto, se utilizan bases de datos públicas, en la Tabla 2.31 se detallan algunas de ellas con sus principales características [12].

1.3.4.2 WISDM

Es un almacén de datos públicos de sensores inalámbricos cuyos datos son recopilados mediante teléfonos inteligentes [24]. Comprende principalmente datos procesados y sin procesar sobre el acelerómetro. La información está relacionada con datos de 36 personas, que llevan un acelerómetro integrado en su dispositivo móvil. Para recopilar los datos se controlan seis tipos de actividades transitorias de la vida diría tales como: caminar, correr, subir escaleras, bajar escaleras, sentarse y estar de pie [12]. Los movimientos se traducen en señales de aceleración para identificar movimientos o rutinas de la vida diaria, además que puede ser utilizado para análisis corporal de ancianos y posicionamiento postural [24].

1.3.4.3 Human Activity Recognition Using Smartphones

Base de datos pública que almacena los datos procesados y sin procesar de sensores inerciales en dispositivos móviles. Contiene los datos de 30 voluntarios de las cuales se evalúan seis clases de actividades: caminar, bajar escaleras, sentarse, mantenerse quieto, subir escaleras, y acostarse. El conjunto de datos es presentado como un vector de características para cada patrón o actividad efectuada, estas comprenden: estimación de la aceleración corporal, velocidad angular triaxial del giroscopio, identificación del usuario y marca de actividad [12].

1.3.4.4 Opportunity Activity Recognition

Conjunto de datos elaborado para el reconocimiento de actividad humana por medio de 72 sensores clasificados en: ambientales, portátiles y de objetos. Está orientado a relacionar ciertos algoritmos de HAR entre los que se encuentran: fusión de sensores, extracción de propiedades, segmentación de datos y clasificación. La extracción de la información comprende la lectura de los datos que proporcionar los sensores de movimiento, mientras los participantes (4 usuarios) efectúan actividades de la vida diaria: caminar, acostarse, permanecer de pie, y sentarse. Las actividades se clasifican en diferentes etapas: clases de modos de locomoción, gestos de nivel medio, acciones de bajo nivel, y acciones de alto nivel [25].

1.3.4.5 Heterogeneity Activity Recognition Data Set (HHAR)

Es un conjunto de datos diseñado para clasificación y agrupamiento enfocado en la investigación de los impactos de la heterogeneidad de los sensores de movimiento en dispositivos inteligentes. La lectura de datos proviene de las actividades como: ciclismo permanecer de pie, sentarse, estacionar arriba y estacionar abajo, proporcionadas por 9 usuarios. Los dispositivos comunes para este conjunto de datos son los sensores de dispositivos móviles como: Samsung Galaxy S3 mini, Samsung Galaxy S3, LG Nexus 4, Samsung Galaxy S+, y relojes inteligentes como: LG y Samsung Galaxy Gears. Específicamente los datos se reúnen en base a contextos del mundo real y bajo las seis distintas orientaciones de los dispositivos [26].

1.3.4.6 Realistic Sensor Displacement (REALDISP)

Conjunto de datos dedicado al reconocimiento de la actividad humana en entornos del mundo real, recopila los datos para evaluar el desplazamiento de los sensores, causado por dos eventos: movimiento de los usuarios o por una holgura causada por el sensor. La base de datos dispone de 33 actividades físicas distintas de las cuales se puede clasificar en actividades de enfriamiento y calentamiento, y actividades de acondicionamiento [14]. Se basa en las definiciones de colocación personal, ideal y desplazamiento inducido. Esto quiere decir que se puede utilizar para emplear algoritmos de reconocimiento. La base de datos incluye la información de características propias de los sensores (aceleración, velocidad de rotación, campo magnético), actividades de movimiento y datos de usuarios [27].

1.3.4.7 MIT PlaceLab Dataset

Se trata de una de las primeras bases de datos diseñadas para el campo de la investigación. Los datos son recopilados del sensor acelerómetro y de un monitor de frecuencia. El acelerómetro se ubica en las cuatro extremidades del cuerpo y además en la cadera. Durante un lapso de cuatro horas una persona realiza ciertas actividades cotidianas, domésticas y otros tipos de tareas como recibir llamadas telefónicas y responder correos electrónicos. Esto presentar un problema ya que cada persona tiene un comportamiento diferente y realiza distintas actividades durante el día, por ello se utilizan otras bases de datos con mayor número de población [14].

1.3.4.8 Wearable action recognition database

Conjunto de datos diseñado por la Universidad de California, está estructurado por las secuencias de acciones humanas proporcionadas por dos sensores de movimiento: un giroscopio triaxial, un acelerómetro triaxial. Los sensores se colocan en las extremidades inferiores y superiores del cuerpo (muñeca, tobillos) y además en la cintura. La base de datos recopila la información de 20 personas involucradas, respecto de las acciones comunes realizadas en la vida cotidiana, estas pueden ser: sentarse, caminar, saltar, pararse [14].

1.3.4.9 USC-HAD

USC-HAD obtiene la información de un dispositivo inercial para comprender y rastrear el movimiento tridimensional. Además, el conjunto de datos recopila los datos de varios sensores combinados. Esto es, por ejemplo, la combinación de un acelerómetro y un giroscopio, con lo cual se obtiene los valores de orientación en tiempo real. Los sensores son ubicados en la cadera para que, alrededor de 14 personas involucradas efectúen 12 tipos de actividades diferentes: saltar, dormir, caminar, bajar escaleras, entre otras [28]. Cada una de las actividades son realizadas cuatro veces hasta obtener la información necesaria.

1.3.4.10 PAMAP2

Es un conjunto de datos público el cual está destinado a registrar datos de sensores portátiles. El sensor es ubicado en las manos, pecho y tobillos durante un período de diez horas, para registrar distintas actividades como: permanecer de pie, saltar la cuerda, caminar, correr, etc. Por otro lado, la composición de la base de datos contiene datos de acelerómetro, giroscopio, sensor de temperatura, magnetómetro y frecuencia cardíaca. Al procesar los datos se organizan en archivos en base a distintas características: marca de tiempo, identificador de actividad, frecuencia cardíaca, unidad de medida del brazo, cofre y tobillo [29].

1.3.4.11 SKODA

Conjunto de datos compuesto por una colección de diez actividades recopiladas por las acciones efectuadas por un trabajador de una fábrica encargada de ensamblar automóviles [30]. El acelerómetro se coloca en el brazo izquierdo, brazo derecho, para que durante tres horas el trabajador realice los mismos gestos setenta veces. Algunas de las actividades son: escribir notas. Cierre del motor, abrir puertas, verificar espacio del maletero, etc. [29]

1.3.5 Funciones de Teléfonos Inteligentes

Las funciones de teléfonos inteligentes se acercan más a "pequeños ordenadores", donde realizar y recibir llamadas de teléfono es sólo una aplicación más entre muchas otras. Físicamente los "smartphones" tienen un tamaño y peso muy parecido al de los teléfonos móviles convencionales. Desaparece el teclado numérico habitual y el espacio liberado permite equiparlos con una pantalla más grande. Sus funciones están vinculadas a las pantallas táctiles, cada vez más grandes, y al uso de teclados y controles virtuales dibujados en la pantalla [31].

1.3.6 Aplicación Móvil con Sensores de Movimiento y posición

Las aplicaciones presentes en los teléfonos móviles pueden manejar datos como las fotografías, correos o la agenda de actividades, pueden acceder a ciertos datos generados por sensores integrados en el dispositivo o conectados a él, como la localización o los signos vitales de los usuarios, y a ciertos identificadores usados por el hardware, sistema operativo, servicios y otras aplicaciones, lo que se denomina firma digital del dispositivo [32].

1.3.7 Privacidad y Seguridad de la Información

La Seguridad de la Información se puede definir como conjunto de medidas técnicas, organizativas y legales que permiten a la organización asegurar la confidencialidad, integridad y disponibilidad de un sistema de información. Consiste en la preservación de la confidencialidad, la integridad y la disponibilidad de la información; además, también pueden estar involucradas otras propiedades, como la autenticidad, responsabilidad, la confiabilidad y el no repudio [33].

1.3.8 Mecanismos de Ciberseguridad

Consiste en una serie de revisiones periódicas, algunos cambios o mejoras de diferentes aspectos que pueden ser de hardware, software o de cualquier elemento involucrado en los sistemas y procesos, por eso es por lo que las revisiones dependen de los procesos de la empresa y cada una tiene sus propios procesos. Los mecanismos

preventivos en realidad son a largo plazo y por esta razón son considerados por la mayoría como una pérdida de tiempo y dinero [34].

1.3.9 Criptografía

La criptografía es un conjunto de algoritmos que brindan seguridad, estos algoritmos cifran la información, es decir pasar un dato legible a un formato ilegible llamado datos cifrados; para pasarlo a legible se realiza el descifrado; con esto se aumenta la seguridad y se mantiene la confidencialidad, integridad y disponibilidad. Existen dos clases de algoritmos para brindar seguridad son simétricos y asimétricos; los algoritmos simétricos son DES (Data Encryption Standard), 3DES (Triple Data Encryption Standard), AES (Advanced Encryption Standard), Blowfish, Digital Signature Algorithm; los algoritmos asimétricos son curva elíptica, Elliptic Curve, Diffie-Hellman y RSA (Asymmetric Cryptography Algorithm). Para el análisis de estrategias algorítmicas orientadas a la ciberseguridad se utilizan metodologías como la de mapeo sistemático, que permite clasificar y obtener datos sobre el conocimiento adquirido por los algoritmos de seguridad y ciberseguridad [35].

1.3.10 Cifrado de Información Personal

El cifrado de información personal es un método para evitar que alguien no pueda tener acceso a información que se desea preservar. Este método consiste en alterar un mensaje antes de transmitirlo, generalmente mediante la utilización de una clave, de modo que su contenido no sea legible para los que no posea dicha clave. De esta forma, cualquier persona que tenga acceso al mensaje no podrá entender su contenido a menos que cuente con la clave para descifrarlo. No solo es necesario cifrar, sino hacerlo de manera que la información no sea inteligible ni manipulada por terceros. Sin esta última condición, el cifrado no tendría valor [36].

1.3.11 Ciclo de vida de software

El ciclo de vida de software es la base o fundamento que permite estructurar el proceso de desarrollo de software [37]. De igual manera, se define como un sistema encargado de proporcionar un conjunto de tareas que se llevarán a cabo, mientras el sistema se

encuentre en fase de desarrollo, lo cual le permite a cualquier empresa u organización controlar el producto de software. El ciclo de vida comprende las siguientes etapas: comprender el problema, establecer un plan de solución, codificar la solución, probar la solución y realizar mantenimiento del producto [38].

1.3.12 Metodología para desarrollo de aplicaciones

Es un marco empleado para planificar, estructurar y controlar las actividades pertenecientes al proceso de desarrollo de software. Seleccionar una metodología adecuada permite a los desarrolladores generar aplicaciones que cumplan con las expectativas reales de los clientes. Además, permite reducir los costos, evitar riesgos críticos y cumplir con una entrega temprana del producto final [39].

1.3.13 Metodologías tradicionales

Se trata de un proceso de diseño en el que cada fase es secuencial, es decir cada etapa tiene que ser completada y firmada por un responsable antes de avanzar al siguiente paso. Las metodologías tradicionales surgen a partir del alto costo que representa ejecutar cambios en el software y por la necesidad de adquirir un modelo que permita planificar y documentar todo antes de una fase de implementación. Es ideal para proyectos en los cuales el control de calidad es un requerimiento primordial. El éxito de esta metodología depende del análisis de requerimientos y los conocimientos adquiridos, pero sobre todo el objetivo principal es satisfacer al cliente [40].

1.3.13.1 Waterfall

Es la primera metodología de desarrollo de software en el cual cada proceso es secuencial, por lo que, cada fase debe estar completada antes de seguir con el siguiente proceso. Es adecuada para proyectos de software en los que el propietario y gerente del proyecto definen claramente los requisitos [41]. Así mismo, es reconocido por su simplicidad, ya que no es necesario contar con una certificación para implementarla. Sin embargo, algunos expertos consideran que el modelo tiene fallas, ya que en proyectos largos o complejos no cumple realmente con las expectativas. La causa

principal es que, al tratarse de una secuencia de pasos, hay poco espacio para realizar ajustes [42].

1.3.13.2 V-Model

Metodología basada en el modelo de desarrollo de procesos de software y de la metodología Waterfall. Se enfoca directamente en realizar pruebas en las que intervienen todas las fases [41]. Este es el problema principal de V-Model, ya que el cliente no se ve involucrado durante las fases iniciales, sino hasta la fase de prueba, por lo que para ese momento sería muy tarde para hacer cambios significativos. Sin embargo, hoy en día es uno de los modelos más populares implementados en sistemas industriales, debido a que cumplen con los requisitos de gestión [43].

1.3.13.3 Rational Unified Process (RUP)

Es un modelo de desarrollo para software, utiliza un marco estructurado para cumplir con cada tarea y responsabilidad de la organización. Su propósito es garantizar un producto de alta calidad y cumplir con las expectativas planteadas por los interesados del proyecto. Esto significa entregar el producto en el tiempo y el costo establecidos previamente [44]. Está constituido por cuatro fases: investigación, planificación y diseño, desarrollo y pruebas, mantenimiento, Además, proporciona una guía con todo el proceso que se llevará a cabo, esto beneficia al encargado del proyecto para efectuar cambios en caso de que el proceso no se ajuste al plan establecido [41].

1.3.14 Metodologías ágiles

Es una metodología de desarrollo amplia, la cual sirve como fundamento para el desarrollo de otras metodologías populares. Permite e los equipos o departamentos de software entregar productos de alta calidad en el tiempo estimado y bajo el presupuesto establecido. Su popularidad se debe a que no especifica instrucciones estrictas sobre cómo desarrollar las aplicaciones. Por lo tanto, la metodología ágil buscar software y usuarios que logren adaptarse a los cambios de las empresas, la tecnología y el entorno, promoviendo una mayor flexibilidad para obtener productos de alta calidad.

1.3.14.1 Scrum

Es un marco iterativo e incremental enfocado en la flexibilidad, encargado de establecer los roles para cada integrante del equipo. Generalmente se trata de un pequeño grupo jerárquico conformado por el encargado del proyecto o Scrum Máster y los desarrolladores. La tarea del Scrum Máster se encarga de instruir al resto del equipo y de monitorear la eficiencia del equipo. Los requisitos de software establecidos por el propietario del proyecto se denominan historias, las cuales son monitoreados en reuniones diarias llamadas Daily Scrum [41]. En síntesis, se trata de una metodología adecuada para cualquier organización debido al compromiso, atención, apertura y respeto y adaptabilidad que ofrece a los clientes [42].

1.3.14.2 Lean Software Development (LSD)

Modelo de desarrollo de proyectos emergente o también conocido como un conjunto de principios fundamentales los cuales son: eliminación de desperdicios, ampliar el aprendizaje, dictaminar lo más tarde posible, entregar el producto en el menor tiempo, fortalecer a los integrantes del equipo, generar integridad y obtener una visión amplia de todos los procesos. A diferencia de Scrum, Lean es una metodología más flexible, ya que ofrece un conjunto de sugerencias, las cuales favorecen a los clientes del proyecto al otorgarles valor, por medio de estrategias como la eliminación de residuos y entregas rápidas de los avances [45].

1.3.14.3 Extreme Programming (XP)

Es un marco de desarrollo considerado como la metodología de desarrollo más específica, ya que describe a detalle todos los procedimientos que se deben ejecutar para cumplir con los objetivos. Establece como valores principales a la comunicación, la sencillez del diseño del sistema y omisión de actividades innecesarias. Lo que implica que Extreme Programming no se preocupa por los requisitos conocidos del proyecto, sino por los requerimientos que aparecen a medida que el proyecto avanza. Así mismo, XP específica la jerarquía de los involucrados en: cliente, desarrollador, tracker y entrenador. El cliente se encarga de tomar las decisiones que involucran directamente al proyecto, mientras que el desarrollador observa las actividades y

trabaja en el proyecto. Por otro lado, el tracker se encarga de observar y registrar el progreso de las metas alcanzadas y de las áreas de mejora. Por último, el entrenador será el encargado de actuar como mentor en las actividades de XP, ya que es alguien totalmente experimentado en la metodología [42].

1.3.14.4 Dynamic System Development Methodology (DSDM)

La metodología de desarrollo de sistemas dinámicos se trata de un método práctico y flexible para mejorar el desarrollo exitoso del software. Utiliza cuatro filosofías: el desarrollo del proyecto es un trabajo conjunto, buena calidad requiere excelentes habilidades técnicas, reducción de la ley de rendimientos crecientes. Así mismo se basa en ciertos principios como: actividad continua de los usuarios, autoridad del equipo de software, constantemente priorizar la finalización del producto, la idoneidad con el negocio, el desarrollo es un proceso evolutivo, los cambios no deben afectar al entorno, establecer un sistema de alto nivel y pruebas de la operatividad del sistema [46]. En otras palabras, es una metodología que requiere un enfoque lento y acción prioritaria de todas las actividades. Por lo cual los propietarios del proyecto y del equipo constantemente mantiene comunicación con el fin de compartir información durante las etapas del proyecto [41].

1.3.14.5 Crystal Methods

Es una metodología adaptable desarrollada por Alistair Cockburn, la cual contiene procesos y herramientas para cada proyecto, lo que significa que todos los proyectos son tratados de la misma forma [41]. En esta metodología los aspectos que posibilitan obtener éxito en el proyecto son: comunicación osmótica, acceso total de usuarios expertos, entrega constante y mejora reflexiva [47].

1.3.14.6 Kanban

Kanban es una metodología ágil que permite visualizar, controlar y equilibrar el flujo de trabajo, dividiéndolo en subtareas que permitan representar cada paso y proceso. Las tareas son colocadas en un tablero Kanban de manera que la visualización de las tareas sea clara para que todo el equipo comprenda la situación actual. Kanban se basa

en cinco principios fundamentales los cuales buscan limitar la carga de trabajo, ayudar en la priorización de tareas y no permitir su acumulación. En conclusión, busca la mejora continua en el impacto del rendimiento general de los procedimientos [48].

1.3.14.7 MADLC

El ciclo de vida de desarrollo de aplicaciones móviles (MADLC), es considerado como una de las últimas metodologías para el desarrollo de software. Mediante procedimientos sistemáticos explica detalladamente cada fase con una descripción de las actividades realizadas durante el desarrollo del proyecto [49]. Existen siete fases que la componen: identificar el problema, diseño, prototipos, pruebas, implementación, mantenimiento. Es útil por ofrecer un enfoque sistemático del proceso, ya que la funcionalidad de aplicaciones de escritorio es distinta al de aplicaciones móviles. Por lo cual, para aplicaciones móviles implementa funciones y servicios como: telefonía, ubicación, conectividad. En el mismo sentido son las aplicaciones orientadas a la industria las que comúnmente la utilizan [50].

1.3.15 Android

Android es un sistema operativo más popular basado en una versión modificada de Linux, diseñado para distintos dispositivos con pantalla táctil como lo son: dispositivos móviles, relojes inteligentes, televisores. Ofrece un enfoque unificado para desarrollar aplicaciones, por lo tanto, los programadores se preocupan solo por desarrollar para Android, y sus aplicaciones serán compatibles con los distintos dispositivos bajo la misma plataforma [51].

1.3.15.1 Android Studio

Android Studio es el entorno de desarrollo oficial para el sistema operativo Android. Es un editor de código diseñado para implementar aplicaciones bajo el lenguaje de programación Java, con el objetivo de obtener aplicaciones de alta productividad en distintas plataformas como dispositivos móviles y tecnologías portátiles [52]. Su composición permite obtener dentro del IDE las siguientes propiedades, herramientas de análisis de código, compilador y un emulador [40].

1.3.15.2 Kotlin

Kotlin es un lenguaje de programación que combina características de la programación orientada a objetos como lenguaje oficial de Android. Es útil para que los desarrolladores puedan incluir nuevos archivos en proyectos Java existentes o para escribir aplicaciones completadas desde cero, debido a que Android Studio es el IDE oficial compatible con Kotlin brinda documentación de primera clase y una herramienta para transformar código de aplicaciones Java o Kotlin [53]. Además, Kotlin está estructurado en código de bytes Java, permitiendo que sea interoperable para invocar métodos o funciones en ambas direcciones. Según Google, más del 60% de profesionales utilizan Kotlin, y el 80% de las mejores aplicaciones Android contienen código basado en Kotlin [54].

1.3.16 Visual Studio

Visual Studio es un entorno de desarrollo integrado perteneciente a Microsoft, utilizado para desarrollar sistemas informáticos y aplicaciones móviles. Es una plataforma creativa y multiplataforma que admite muchas funciones del desarrollo de software como: IntelliSense, depurar, compilar código, perfilador de código, diseñador de código y publicador de aplicaciones. Esto ofrece la posibilidad de mejorar el proceso de desarrollo de software [40].

1.4 MVVM (Model-View-ViewModel)

El patrón arquitectónico MVVM fue implementado por Microsoft para ayudar a los programadores en la creación de aplicaciones WPF. Es resultado de agrupar la vista y el controlador, para que la lógica del negocio del controlador se convierta en un nuevo objeto conocido como vista del modelo [55]. El patrón MVVM se ha implementado para acabar con los problemas del patrón de diseño de la arquitectura MVC para gestionar el estado de la vista [56]. Por lo cual, es útil para proyectos en los cuales la interfaz de usuario y la lógica del negocio son el objetivo principal del desarrollo [57]. La arquitectura MVVM contiene tres capas:

Capa del Modelo: Se encarga de almacenar los datos y establecer comunicación bidireccional con la vista del modelo [55].

Capa de Vista: Mantiene la comunicación bidireccional con la vista del modelo y controla los eventos generados por la interacción con el usuario [55].

Capa de Vista del Modelo: Controla la lógica de presentación, dirige la comunicación con ciertos controladores que proporcionan funciones específicas como la lógica del negocio y las solicitudes de red. Además, mantiene la comunicación entre la vista y el modelo [55].

1.5 Objetivos

1.5.1 Objetivo General

Implementar una aplicación móvil para cifrado de datos personales aplicando sensores de movimiento y posición como medio de ejecución en dispositivos móviles.

1.5.2 Objetivos Específicos

- Investigar metodologías para acceso y cifrado de datos en aplicaciones.
- Identificar las funciones de los sensores de movimiento y posición en dispositivos móviles.
- Desarrollar la aplicación móvil para el cifrado de datos privados de dispositivos como medio de seguridad.

CAPÍTULO II

METODOLOGÍA

2.1 Materiales

Debido a la naturaleza del proyecto de investigación para la recolección de la información se utilizó fichas bibliográficas las cuales resumen el contenido de tesis similares al problema planteado y un cuestionario el cual recopila criterios relacionados con la exposición de datos personales. Ver Anexo A, Anexo B

2.2 Métodos

2.2.1 Modalidad de la Investigación

2.2.1.1 Bibliográfica o Documental

Debido a que se tomó como fuente de información primaria documentos, manuales, artículos, etc., para la elaboración del marco teórico sobre sensores integrados en dispositivos.

2.2.1.2 Modalidades Especiales

Porque respondió a la problemática o necesidad de tipo social con el desarrollo de una aplicación para teléfonos inteligentes.

2.2.2 Población y Muestra

El presente proyecto se desarrolló con una población específica segmentada entre niños, jóvenes y adultos de diferentes edades.

Población	Frecuencia	Porcentaje
Niños entre 10 y 13 años	5	16.67%
Jóvenes entre 15 y 26 años	15	50.00%
Adultos entre 27 y 59 años	10	33.33%
Total	30	100.00%

Tabla 2.1: Poblaciones

Elaborado por: Alejandro Mejía

Muestra

Al no contar con una población mayor a las 100 personas, no es necesario establecer una muestra representativa por lo tanto se realizó el estudio del proyecto con la población existente.

2.2.3 Recolección de Información

2.2.3.1 Validación de Instrumentos con RStudio

Para validar los instrumentos de recolección de información se utilizó RStudio, mediante el cual se calculó el Alfa de Cronbach, obteniendo el 80% de consistencia interna y confiabilidad

```
> psych::alpha(datos)
Some items ( P5 ) were negatively correlated with the total scale and
probably should be reversed.
To do this, run the function again with the 'check.keys=TRUE' option
Reliability analysis
Call: psych::alpha(x = datos)

  raw_alpha std.alpha G6(smc) average_r S/N   ase mean   sd median_r
        0.8      0.78    0.87      0.18 3.5 0.039    3 0.48     0.18

    95% confidence boundaries
          lower alpha upper
Feldt      0.71   0.8  0.88
Duhachek   0.73   0.8  0.88

 Reliability if an item is dropped:
     raw_alpha std.alpha G6(smc) average_r S/N alpha se var.r med.r
P1        0.80      0.77    0.86      0.18 3.4    0.040 0.044  0.19
P2        0.80      0.76    0.86      0.18 3.2    0.040 0.043  0.17
P3        0.81      0.79    0.87      0.20 3.8    0.037 0.039  0.20
P4        0.79      0.76    0.85      0.17 3.2    0.041 0.044  0.17
P5        0.81      0.80    0.88      0.21 4.0    0.038 0.038  0.21
P6        0.77      0.74    0.85      0.16 2.9    0.045 0.040  0.15
P7        0.80      0.77    0.86      0.18 3.3    0.039 0.042  0.18
P8_1      0.78      0.75    0.84      0.17 3.0    0.045 0.038  0.15
P8_2      0.77      0.74    0.84      0.16 2.9    0.047 0.036  0.15
P8_3      0.77      0.74    0.84      0.16 2.9    0.045 0.038  0.17
P8_4      0.77      0.74    0.83      0.16 2.9    0.046 0.034  0.17
P9        0.79      0.76    0.85      0.17 3.1    0.041 0.042  0.17
P10       0.81      0.79    0.87      0.20 3.8    0.039 0.039  0.20
P11       0.80      0.77    0.86      0.18 3.4    0.040 0.042  0.17
P12       0.81      0.79    0.87      0.20 3.7    0.038 0.039  0.20
P13       0.80      0.76    0.86      0.18 3.2    0.041 0.042  0.18
```

Figura 2.1: Alfa de Cronbach

Elaborado por: Alejandro Mejía

2.2.3.2 Análisis de Interpretación de Fichas Bibliográficas

FICHA BIBLIOGRÁFICA	
TEMA	An efficient approach to securing user data in android
TESIS	Brinda información sobre como las aplicaciones en Android tienen acceso sin precedentes a la información privada de los usuarios, debido a que dichas aplicaciones almacenan la información de forma local, por lo cual es necesario que se almacene de forma segura.
PROPÓSITO	Investigar las características de seguridad de seguridad existentes en Android que protegen la información almacenada por las aplicaciones, y los problemas conocidos asociados a estas medidas de seguridad.
IDEAS CENTRALES	- El uso de NDK aumenta significativamente la velocidad del cifrado. - Los mecanismos de seguridad existentes en Android no son suficientes para manejar las amenazas asociadas con la información del usuario almacenada localmente por las aplicaciones. - No toda la información almacenada por las aplicaciones está relacionada con el usuario y requieren un almacenamiento seguro.
CONCEPTOS CLAVES	Encriptación, Estándar de cifrado avanzado
CONCLUSIONES	Muchas aplicaciones almacenan información del usuario localmente en el dispositivo y alrededor del 87% de estas aplicaciones lo almacenaron como texto sin formato.
APORTE A TEMA ELEGIDO	Aporta información sobre como generar una aplicación que permita el cifrado selectivo de los datos, lo cual ofrece mucha más flexibilidad a las aplicaciones.
CONCLUSIÓN GENERAL: Los dispositivos Android contienen aplicaciones que en gran parte tienen acceso a los recursos o información privada de sus usuarios, debido a los permisos que las aplicaciones suelen solicitar a los usuarios para ejecutarse o realizar alguna acción específica. Es decir, cualquier persona queda totalmente expuesta a robo de información por parte de terceros utilizando dichas aplicaciones como medio de ataque. Por tal motivo, es evidente que la información privada debe ser almacenada en los dispositivos móviles de manera segura, una alternativa es la encriptación que otorga un nivel de seguridad para el usuario y a su vez, dificulta el trabajo de los atacantes para robar información.	

Tabla 2.2: Ficha Bibliográfica 1

Elaborado por: Alejandro Mejía

FICHA BIBLIOGRÁFICA	
TEMA	Design and Implementation of AES and SHA-256 Cryptography for Securing Multimedia File over Android Chat Application
TESIS	Da a conocer el proceso de diseño de una aplicación móvil en Android que utiliza un método de cifrado rápido y demuestra los resultados obtenidos tras aplicar la encriptación. Describe los módulo y funciones de Android para cifrado de archivos. Contiene un cuadro comparativo entre los archivos multimedia seleccionados, tamaño del archivo original, tamaño del archivo recuperado luego del proceso de encriptación y el tiempo transcurrido en cada caso.
PROPÓSITO	Cifrar los archivos multimedia tales como imágenes, audio y video utilizando el estándar de cifrado avanzado (AES) como método de seguridad y el algoritmo el algoritmo SHA-256 para cifrar la información.
IDEAS CENTRALES	- La seguridad es uno de los aspectos más importantes en el desarrollo de aplicaciones de comunicación, especialmente al enviar archivos confidenciales. - AES contiene un rápido proceso de encriptación y ha sido ampliamente implementado en varios campos de seguridad. - El algoritmo AES procesa los datos para convertirlos en una forma diferente conocida como texto cifrado.
CONCEPTOS CLAVES	Contiene los conceptos de los algoritmos de cifrado y descifrado más comunes, sistema de encriptación mediante el algoritmo criptográfico AES.
CONCLUSIONES	Tras el proceso de encriptación de los datos se evidencia un incremento en el tamaño de los archivos, debido a cambios en el valor del archivo de cabecera. Sin embargo, al compararlo con el archivo original utilizando funciones PSNR (Peak Signal-to-Noise Ratio) en Matlab son exactamente iguales.
APORTE A TEMA ELEGIDO	Aporta con una idea central sobre el diseño de un sistema de encriptación para archivos multimedia utilizando AES esto servirá para el diseño de la metodología necesaria para cifrar y descifrar los datos.
CONCLUSIÓN GENERAL: Cerca del 87% de dispositivos en el mercado mundial pertenecen al sistema operativo Android, por lo que a diario son millones los dispositivos que se conectan a Internet para enviar y recibir información. Los dispositivos son capaces de almacenar una gran cantidad de datos privados, por lo que es necesario utilizar mecanismos de seguridad para proteger la información de ciertas amenazas como	

robo de datos debido a vulnerabilidades en los sistemas operativos o aplicaciones de Android. AES es un algoritmo criptográfico que permite ejecutar el proceso de encriptación de forma rápida a comparación de otros algoritmos.

Tabla 2.3: Ficha Bibliográfica 2

Elaborado por: Alejandro Mejía

FICHA BIBLIOGRÁFICA	
TEMA	Comparison of implementation tiny encryption algorithm (TEA) and advanced encryption standard (AES) algorithm on android based open source cryptomator library
TESIS	Brinda información sobre la librería cryptomator que utiliza AES como algoritmo de encriptación y lo compara con TEA basado en distintos parámetros de prueba como rendimiento, implementación y tiempo de ejecución. Contiene un cuadro comparativo del tiempo de ejecución de cada algoritmo y rendimiento.
PROPÓSITO	Comparar el rendimiento de la implementación de los algoritmos TEA y AES en el proceso de encriptación utilizando como herramienta la investigación comparativa.
IDEAS CENTRALES	- El algoritmo TEA tiene una alta velocidad y poco tamaño de código, en cambio el algoritmo AES es bueno en cuanto a la velocidad del proceso de cifrado, pero requiere un gran tamaño de código. - En el caso de algoritmos que requieren el tamaño de código más bajo, los algoritmos de mejor rendimiento son TEA, XTEA, Twine y LED. - Los algoritmos que requieren menos RAM, a saber, XTEA, Lblock, TEA y MIBS - TEA y XTEA ocupan el mejor rendimiento en términos de métricas de memoria
CONCEPTOS CLAVES	Contiene conceptos de los algoritmos de encriptación TEA, AES, ciclo de vida de desarrollo de un sistema (SDLC), método de investigación y prueba de rendimiento.
CONCLUSIONES	El rendimiento de la TEA es más rápido que el de AES. Para cada tipo y tamaño de archivo, el porcentaje de comparación es diferente.

APORTE A TEMA ELEGIDO	Aporta con información específica sobre el rendimiento al implementar el algoritmo TEA y AES en el proceso de encriptación, lo que servirá como guía para la elección del algoritmo a utilizarse en el desarrollo de la aplicación.
CONCLUSIÓN GENERAL: AES y TEA son algoritmos de encriptación útiles para respaldar la información de dispositivos móviles. En la búsqueda de un algoritmo de encriptación robusto son considerables aspectos como rapidez, confiabilidad, fiabilidad e implementación. Debido a que la información de los usuarios puede comprometer su integridad personal es necesario mantener el contenido de los datos originales luego de procesar los datos.	

Tabla 2.4: Ficha Bibliográfica 3

Elaborado por: Alejandro Mejía

2.2.3.3 Análisis e Interpretación de la Encuesta aplicada a usuarios

Pregunta 1: ¿Qué tipo de información almacenada en su dispositivo móvil es más importante para usted?

Opción	Total	Porcentaje
Fotos	24	56%
Videos	1	2%
Documentos PDF/ Word/ Legales	17	40%
Música	1	2%

Tabla 2.5: Resultados Pregunta 1

Elaborado por: Alejandro Mejía

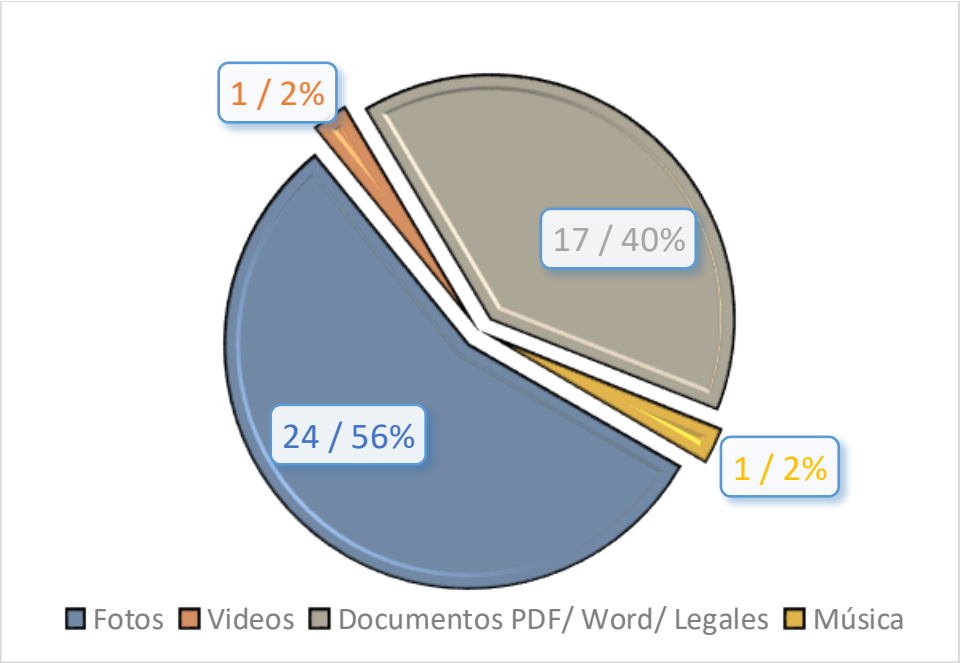


Figura 2.2: Gráfico Pregunta 1

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada el 56% manifestó que la información más importante almacenada en sus dispositivos móviles son las fotografías, el 40% respondió documentos, mientras que videos y música obtuvieron un 2% respectivamente.

Interpretación: De acuerdo con los resultados obtenidos, se puede constatar que fotos y documentos son esenciales para los usuarios, lo cual es un indicativo para determinar el tipo de información que compromete la integridad de los usuarios.

Pregunta 2: ¿Con qué constancia considera son atacados los dispositivos móviles iOS?

Opción	Total	Porcentaje
Nunca	8	19%
Diariamente	22	51%
Semanalmente	3	7%
Mensualmente	6	14%
Anualmente	4	9%

Tabla 2.6: Resultados Pregunta 2

Elaborado por: Alejandro Mejía

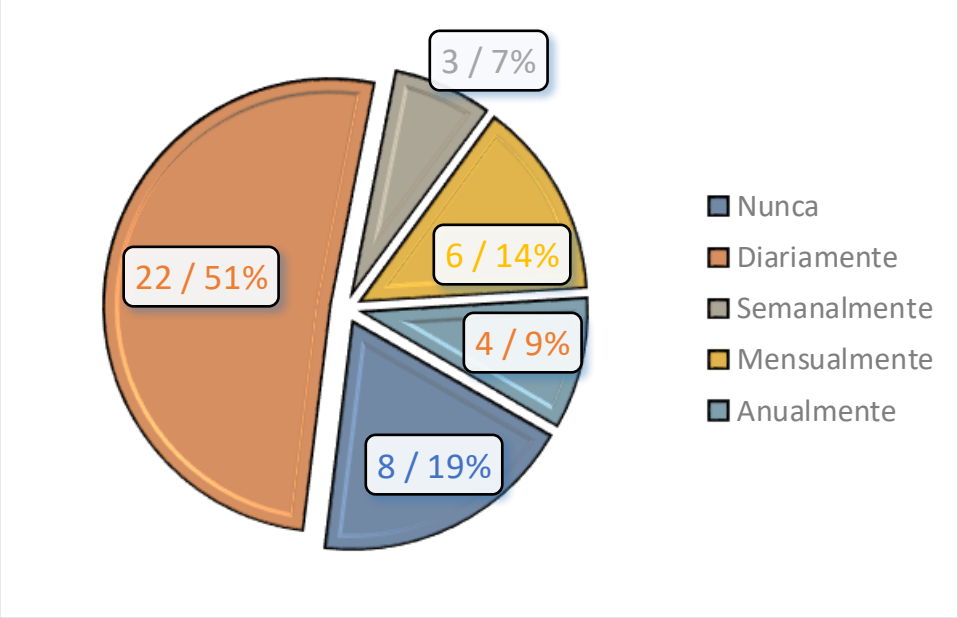


Figura 2.3: Gráfico Pregunta 2

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada el 51% respondió que los eventos relacionados con ataques a dispositivos móviles iOS suceden a diario, seguido del 19% por nunca, el 14% por mes, el 9% por año y el 14% restante que acontecen semanalmente.

Interpretación: Con este resultado se puede observar que para los usuarios los eventos relacionados con ataques a dispositivos móviles iOS suceden recurrentemente. Lo que hace necesario contar con métodos de seguridad para proteger la información que estos dispositivos almacenan.

Pregunta 3: ¿Con qué frecuencia considera ocurren ataques a dispositivos móviles Android?

Opción	Total	Porcentaje
Nunca	1	2%
Diariamente	28	65%
Semanalmente	10	23%
Mensualmente	3	7%
Anualmente	1	2%

Tabla 2.7: Resultados Pregunta 3

Elaborado por: Alejandro Mejía

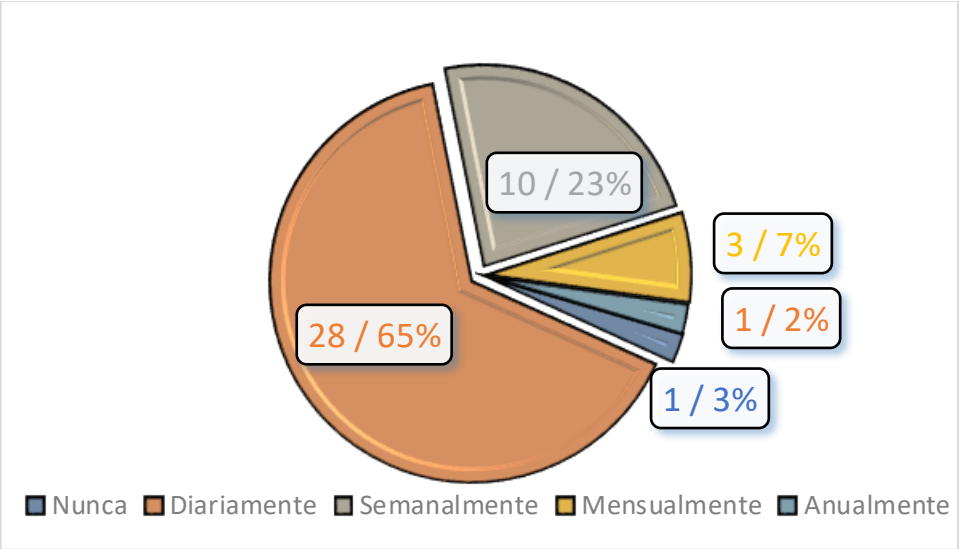


Figura 2.4: Gráfico Pregunta 3

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada el 65% respondió que los ataques a dispositivos móviles Android ocurren a diario, el 23% semanalmente, el 7% mensualmente, el 3% nunca, mientras que el 2% señaló que ocurren anualmente.

Interpretación: En comparación con la frecuencia de ataques que suceden en iOS, se puede observar que un mayor porcentaje de encuestados consideran que estos eventos ocurren en Android. Por lo que es primordial mejorar los mecanismos de seguridad de estos dispositivos, para que los datos de los usuarios no sean vulnerados.

Pregunta 4: ¿Qué dispositivo electrónico considera tiene un nivel de seguridad bajo, que lo hace vulnerable a hackeos?

Opción	Total	Porcentaje
Celular	28	65%
Computadora de Escritorio	6	14%
Laptop	5	12%
iPad	1	2%
Ninguno	3	7%

Tabla 2.8: Resultados Pregunta 4

Elaborado por: Alejandro Mejía

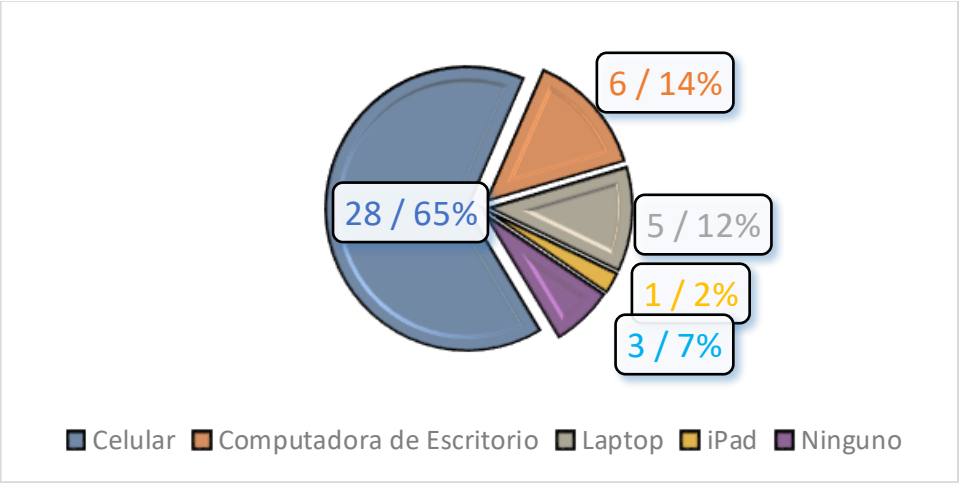


Figura 2.5: Gráfico Pregunta 4

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada el 65% respondió celular, seguido del 14% por computadora de escritorio, el 12% por laptop, el 7% por ningún dispositivo, mientras que el 2% restante respondió iPad.

Interpretación: Con este resultado se puede determinar que, para gran parte de usuarios, los dispositivos celulares a comparación de otros dispositivos electrónicos no disponen de mecanismos de protección de datos eficientes.

Pregunta 5: ¿Conoce personas que hayan sido víctima de robo de información privada de sus dispositivos móviles?

Opción	Total	Porcentaje
Si	29	67%
No	14	33%

Tabla 2.9: Resultados Pregunta 5

Elaborado por: Alejandro Mejía

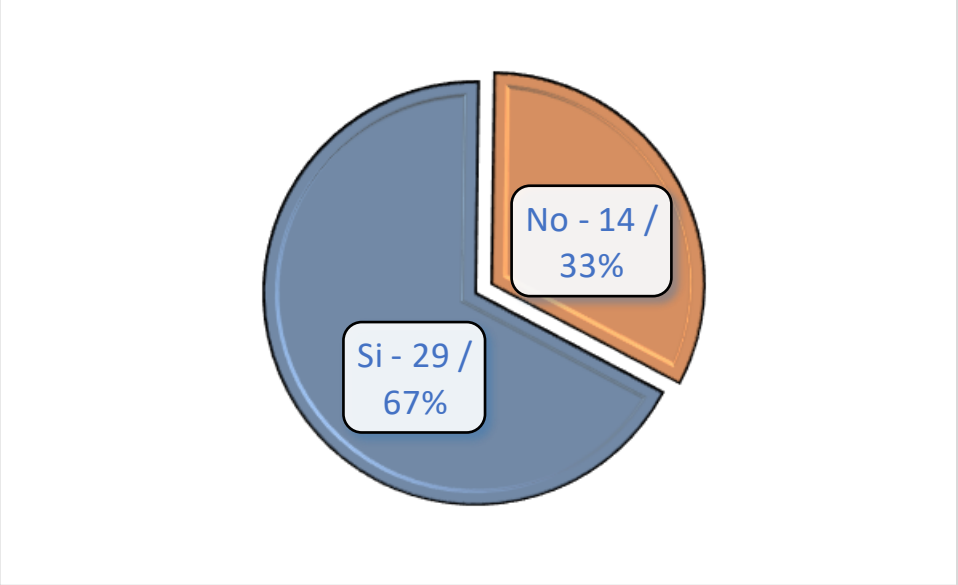


Figura 2.6: Gráfico Pregunta 5

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada el 67% afirmó que conocen personas que fueron víctima de robo de información privada de sus dispositivos móviles, mientras que el 33% respondieron que no.

Interpretación: Con el resultado obtenido, se evidencia que el 67% ha sido víctima o conoce personas cuya información privada fue vulnerada de sus dispositivos, lo cual permite constatar la información recopilada en la pregunta 2 y 3.

Pregunta 6: ¿Qué grupo de personas consideraría es más probable que roben su información privada?

Opción	Total	Porcentaje
Jóvenes	14	33%
Adultos Mayores	12	28%
Niños	7	16%
Adolescentes	10	23%
Ninguno	0	0%

Tabla 2.10: Resultados Pregunta 6

Elaborado por: Alejandro Mejía

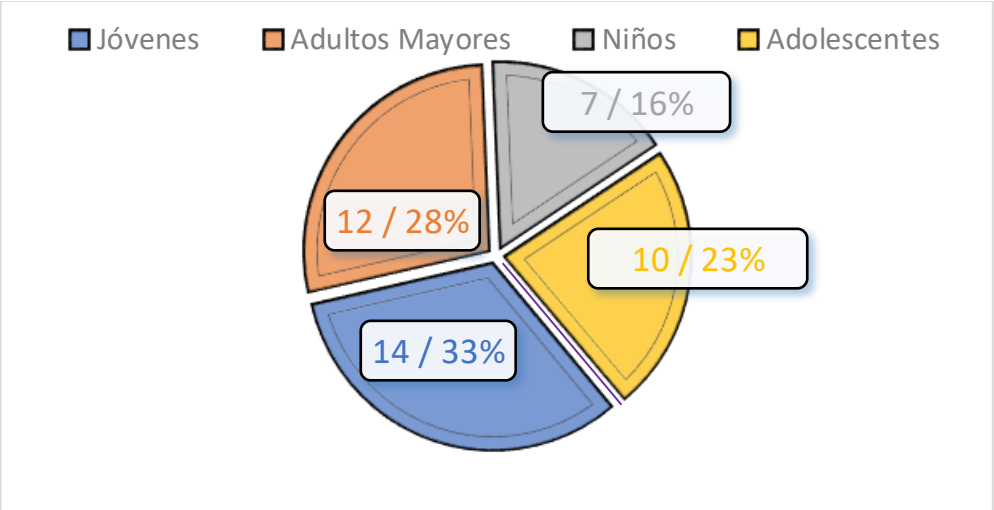


Figura 2.7: Gráfico Pregunta 6

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada el 33% respondió Jóvenes como el sector de la población más probable a ser víctima de robo de información, seguido del 28% por adultos mayores. Luego un 23% por adolescentes, mientras que el 16% restante consideraron a los niños.

Interpretación: Se puede observar que los jóvenes son el sector de la población al que los hackers dirigen la mayor parte de los ataques.

Pregunta 7: ¿Quién se ve más afectado en caso de que su información personal sea expuesta en Internet?

Opción	Total	Porcentaje
Jóvenes	14	33%
Adolescentes	12	28%
Niños	5	12%
Adultos Mayores	11	26%
Ninguno	1	2%

Tabla 2.11: Resultados Pregunta 7

Elaborado por: Alejandro Mejía

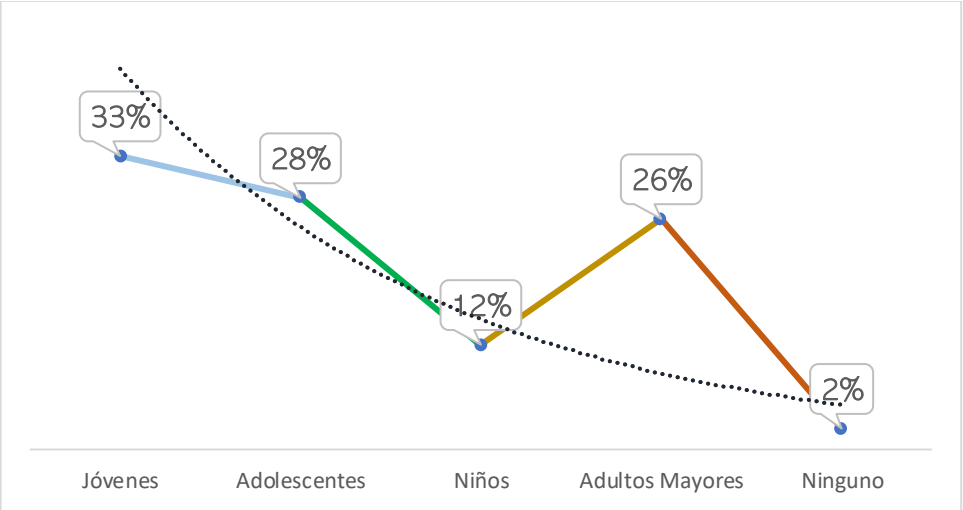


Figura 2.8: Gráfico Pregunta 7

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada se encontró que el 33% respondió jóvenes, seguido del 28% por adolescentes, luego un 26% por adultos mayores, el 12% niños, mientras que el 2% seleccionó que ningún sector.

Interpretación: La información que almacenan los dispositivos móviles de jóvenes y adolescentes es sensible y comprometedora, mientras que la información de adultos mayores o niños no implica un riesgo a su integridad personal.

Pregunta 8: ¿Califique el tipo de información que comprometería su integridad personal en caso de ser hackeada o vulnerada de su dispositivo móvil?

Opción	Total	Porcentaje
Nada Comprometedora	13	30%
Poco Comprometedora	16	37%
Comprometedora	5	12%
Muy Comprometedora	9	21%

Tabla 2.12: Resultados Pregunta 8 Contactos Telefónicos

Elaborado por: Alejandro Mejía

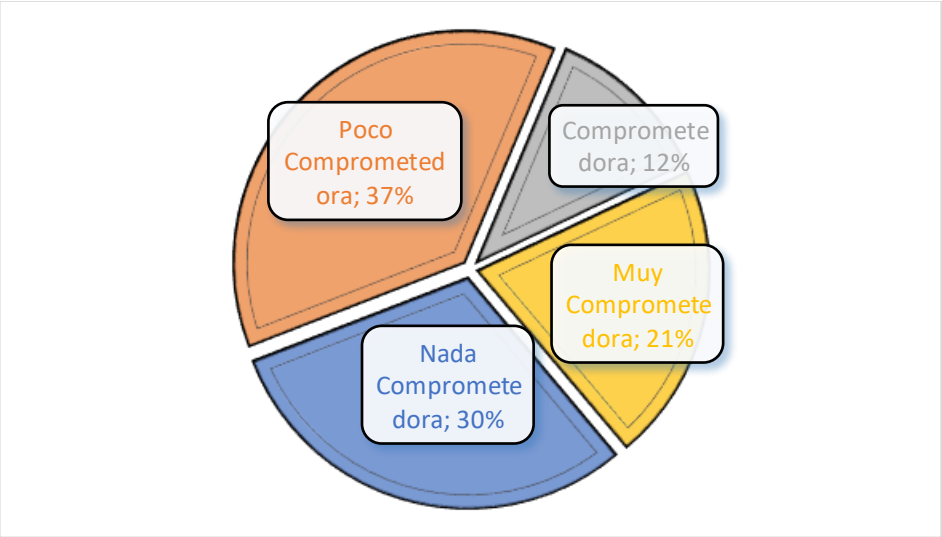


Figura 2.9: Gráfico Pregunta 8 Contactos Telefónicos

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada en el caso de contactos telefónicos, el 37% respondió que esta información es poco comprometedora, el 30% nada comprometedora, luego el 21% muy comprometedora y el 12% restante consideró es comprometedora.

Opción	Total	Porcentaje
Nada Comprometedora	14	33%
Poco Comprometedora	6	14%
Comprometedora	13	30%
Muy Comprometedora	10	23%

Tabla 2.13: Resultados Pregunta 8 Videos

Elaborado por: Alejandro Mejía

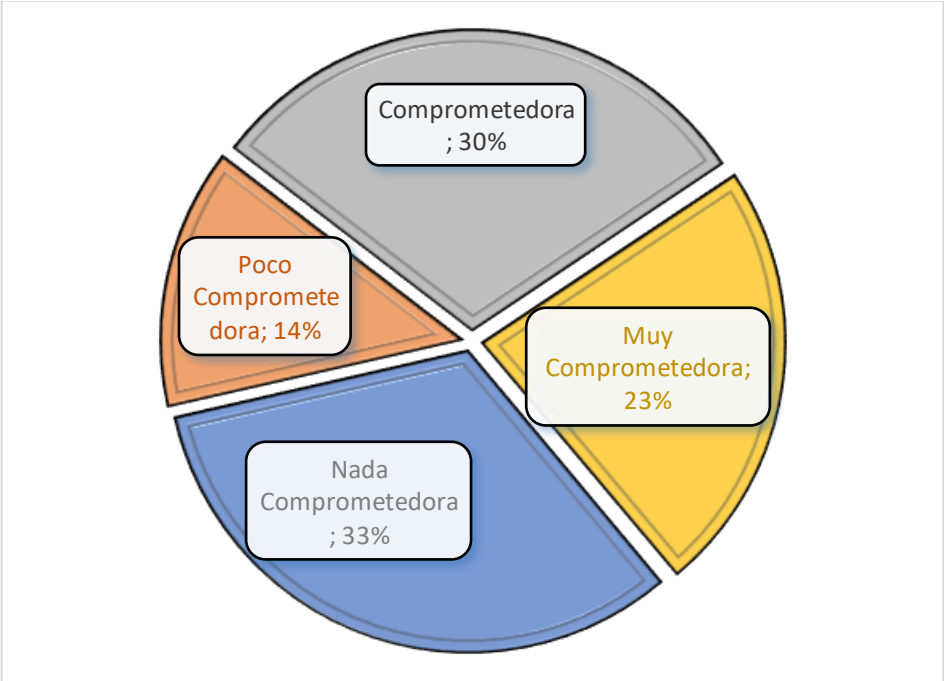


Figura 2.10: Gráfico Pregunta 8 Videos

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada en el caso de videos, el 33% seleccionó que no es comprometedora, el 30% comprometedora, seguido del 23% que considera es muy comprometedora, mientras que un 14% consideró es poco comprometedora.

Opción	Total	Porcentaje
Nada Comprometedora	4	9%
Poco Comprometedora	12	28%
Comprometedora	7	16%
Muy Comprometedora	20	47%

Tabla 2.14: Resultados Pregunta 8 Documentos

Elaborado por: Alejandro Mejía

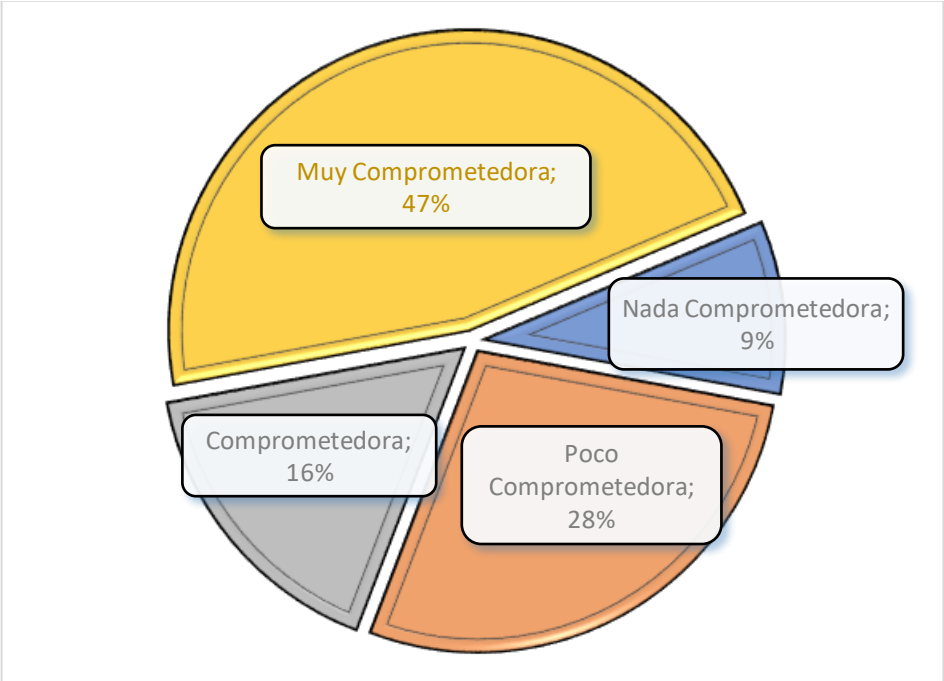


Figura 2.11: Gráfico Pregunta 8 Documentos

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada en el caso de documentos, el 47% respondió que es muy poco comprometedora, seguido del 28% por poco comprometedora, el 16% comprometedora, mientras que el 9% señaló no es comprometedora.

Opción	Total	Porcentaje
Nada Comprometedora	9	21%
Poco Comprometedora	9	21%
Comprometedora	8	19%
Muy Comprometedora	17	40%

Tabla 2.15: Resultados Pregunta 8 Fotos

Elaborado por: Alejandro Mejía

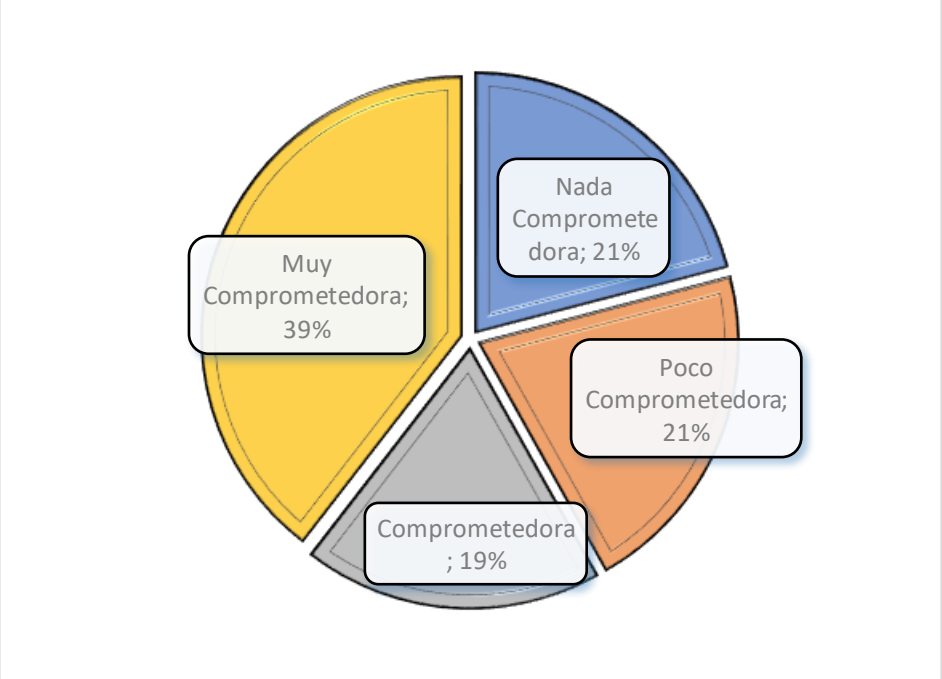


Figura 2.12: Gráfico Pregunta 8 Fotos

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada en el caso de fotos, el 39% respondió que es muy poco comprometedora, nada y poco comprometedora obtuvieron un valor de 21% respectivamente y el 19% restante consideró es comprometedora.

Interpretación: Se puede observar que el contenido de fotos y documentos compromete la seguridad e integridad de los usuarios. Por lo que hackers pueden robar y utilizar esa información para efectuar extorsiones.

Pregunta 9: ¿Qué método de seguridad utiliza para seguridad de su dispositivo móvil?

Opción	Total	Porcentaje
Contraseña	7	16%
Huella Digital	26	60%
Reconocimiento Facial	2	5%
PIN	8	19%
Ninguno	0	0%

Tabla 2.16: Resultados Pregunta 9

Elaborado por: Alejandro Mejía

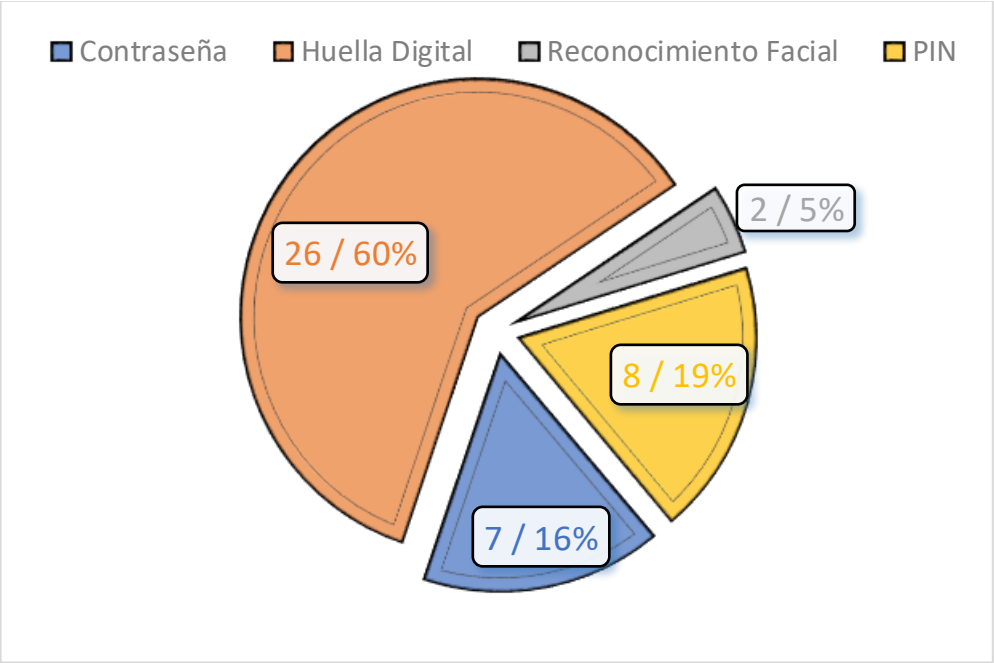


Figura 2.13: Gráfico Pregunta 9

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada el 60% manifestó que utiliza la huella digital como método de seguridad en sus dispositivos, seguido del 19% que respondió PIN, el 16% por contraseña, mientras que el 5% utiliza reconocimiento facial.

Interpretación: Se puede constatar que todos los usuarios utilizan algún método de seguridad para proteger su información privada, pero cabe mencionar que el método más común es la huella digital, mientras que el menos aplicado es el reconocimiento facial. Lo cual permite considerarlo como el mecanismo menos confiable.

Pregunta 10: ¿Con que frecuencia descarga e instala aplicaciones en su dispositivo móvil?

Opción	Total	Porcentaje
Frecuentemente	25	58%
Casi Nunca	16	37%
Siempre	2	5%
Nunca	0	0%

Tabla 2.17: Resultados Pregunta 10

Elaborado por: Alejandro Mejía

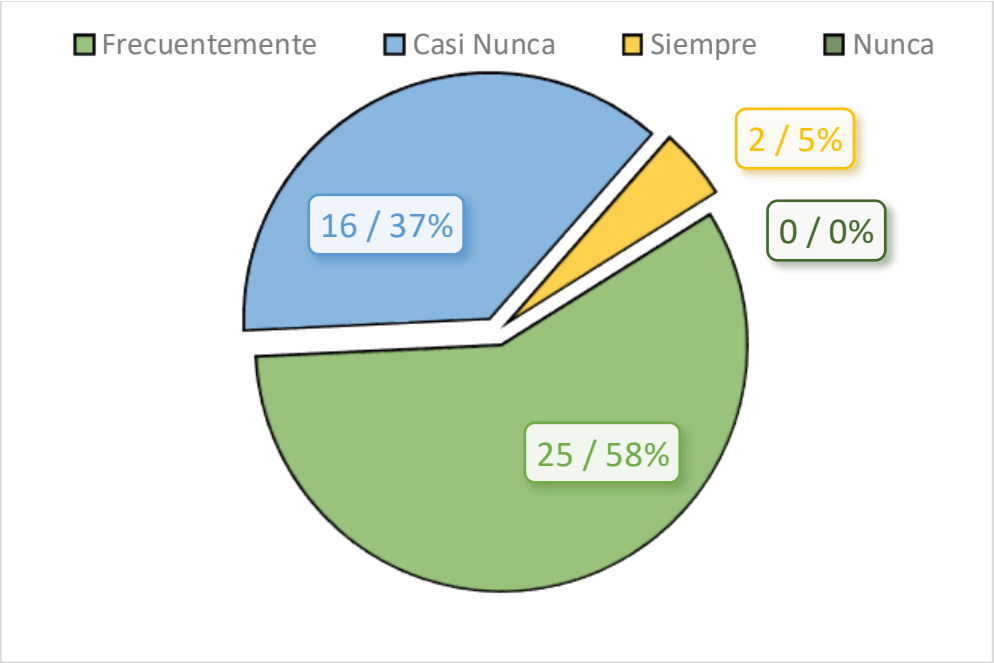


Figura 2.14: Gráfico Pregunta 10

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada el 58% señaló frecuentemente, seguido del 37% que respondió casi nunca, y el 5% restante por siempre.

Interpretación: Se puede señalar que los usuarios frecuentemente descargan e instalan aplicaciones en sus dispositivos móviles, aunque cabe mencionar que un 37% casi nunca lo hace.

Pregunta 11: ¿Considera que las apps de tiendas de aplicaciones oficiales como PlayStore, Aptoide, Uptodown son seguras y confiables?

Opción	Total	Porcentaje
La mayor parte son seguras	29	67%
Ninguna aplicación es segura	4	9%
Pocas aplicaciones son seguras	7	16%
Todas las aplicaciones son seguras	3	7%

Tabla 2.18: Resultados Pregunta 11

Elaborado por: Alejandro Mejía

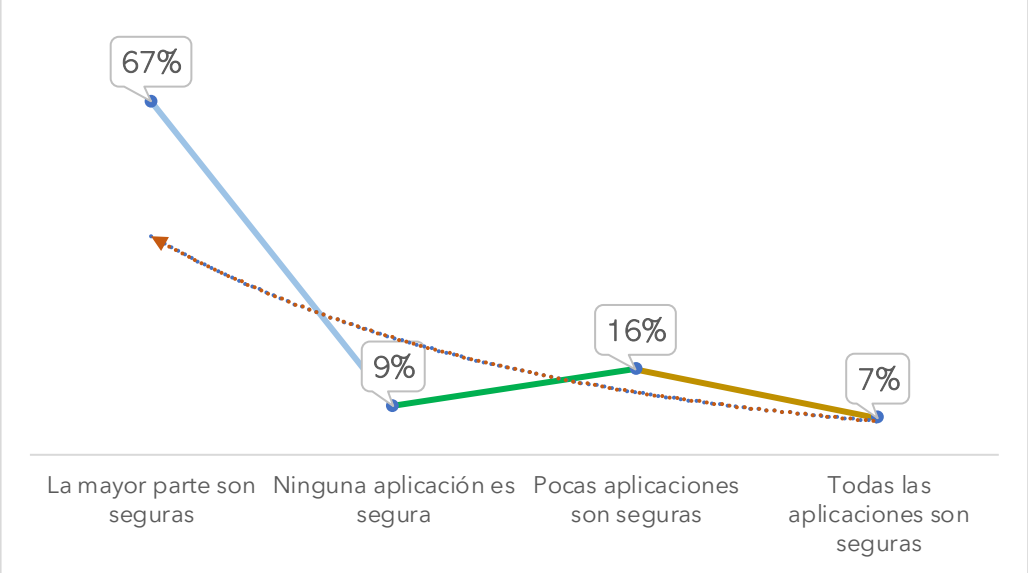


Figura 2.15: Gráfico Pregunta 11

Elaborado por: Alejandro Mejia

Análisis: Del total de la población encuestada el 67% respondió que la mayor parte de aplicaciones son seguras, el 16% consideró que pocas aplicaciones son seguras, el 9% ninguna aplicación es segura, mientras que por último el 7% restante manifestó que todas las aplicaciones son seguras.

Interpretación: Únicamente el 16% de usuarios considera que pocas aplicaciones de tiendas de virtuales son confiables, lo cual permite observar que son pocas las personas que conocen de los riesgos a los que se ven expuestos al instalar cualquier aplicación que aparenta ser segura por estar en sitios oficiales como PlayStore.

Pregunta 12: ¿Cuál considera es el factor principal para que ocurra el robo o hackeo de información privada de los dispositivos móviles?

Opción	Total	Porcentaje
Desinformación de los usuarios	12	28%
Aplicaciones inseguras	26	60%
Beneficio Económico	4	9%
No se utilizan contraseñas	1	2%
Ninguno	0	0%

Tabla 2.19: Resultados Pregunta 12

Elaborado por: Alejandro Mejía

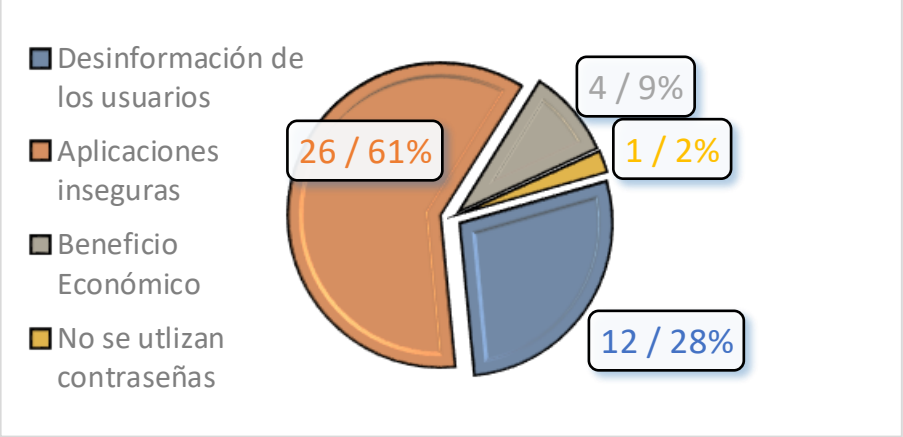


Figura 2.16: Gráfico Pregunta 12

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada el 61% respondió aplicaciones inseguras, el 28% desinformación de los usuarios, el 9% por beneficio económico, y el 2% restante señaló que el factor principal es que no se utilizan contraseñas.

Interpretación: Se puede constatar que los usuarios consideran que los hackeos suceden principalmente por: desinformación de las personas, beneficio económico y por aplicaciones inseguras.

Pregunta 13: ¿Consideraría útil contar con una aplicación móvil para proteger los datos privados almacenados en los dispositivos móviles?

Opción	Total	Porcentaje
Poco útil	3	7%
Muy útil	25	58%
Útil	15	35%

Tabla 2.20: Resultados Pregunta 13

Elaborado por: Alejandro Mejía

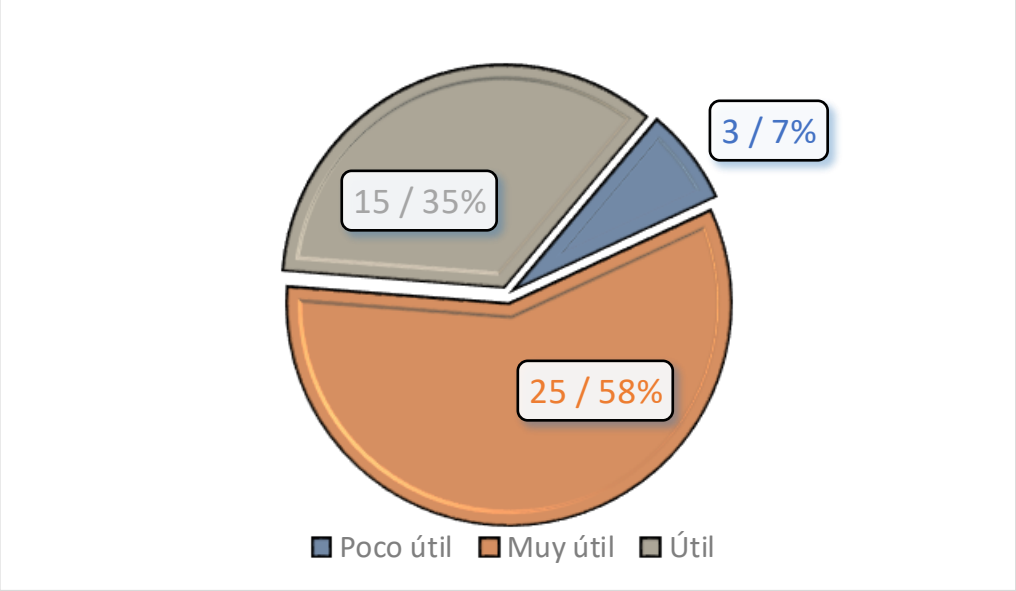


Figura 2.17: Gráfico Pregunta 13

Elaborado por: Alejandro Mejía

Análisis: Del total de la población encuestada el 58% respondió muy útil, seguido del 35% por útil, y el 7% restante que consideró poco útil.

Interpretación: Se puede observar que la mayor parte de usuarios consideran factible contar una aplicación móvil para proteger la información privada de sus dispositivos móviles.

2.2.4 Procesamiento y Análisis de Datos

Con la información recopilada de fichas bibliográficas se determinó los siguientes criterios:

- Los dispositivos Android contienen aplicaciones que al ser instaladas requieren de ciertos permisos para ejecutarse, por lo que, en caso de tratarse de una aplicación insegura la información almacenada queda totalmente expuesta.
- Para garantizar la seguridad y protección de los datos es necesario utilizar algoritmos de encriptación que procesen los datos de forma rápida y conserven el formato original.

De acuerdo con los resultados obtenidos del cuestionario aplicado a niños, adolescentes, jóvenes y adultos mayores, se determinaron los siguientes aspectos:

- En general, los encuestados están conscientes de la ocurrencia de los eventos relacionados con el robo de datos. Así mismo, comprenden que son los dispositivos móviles los que diariamente reciben gran parte de los ataques; debido a que emplean mecanismos de protección de datos vulnerables.
- Las aplicaciones inseguras son el medio principal por el cual los atacantes acceden a los dispositivos, evaden la seguridad y roban la información.
- Jóvenes y adolescentes son el sector de la población que evidencia una gran probabilidad de recibir ataques en sus teléfonos celulares y que esta información sea expuesta en Internet.
- Los dispositivos móviles son capaces de almacenar cualquier tipo de dato de forma local, sin embargo, para los usuarios el contenido de fotos y documentos es fundamental. Debido a que, en caso de ser vulnerado, comprometería a su integridad personal. Por lo tanto, esto puede ocasionar que sean víctimas de extorsiones a cambio de recuperar esa información.

Todo lo anteriormente mencionado, permite obtener los requerimientos y justificación necesaria para el desarrollo de la aplicación móvil con sensores de movimiento y posición para el cifrado de datos personales.

CAPÍTULO III

RESULTADOS Y DISCUSIÓN

3.1 Análisis y Discusión

3.1.1 Análisis de Sensores

En esta sección se efectuó un benchmarking genérico con el objetivo de analizar los tipos, funciones, aplicaciones y capacidades de los sensores integrados en dispositivos móviles Android. El análisis se llevó a cabo obteniendo información de documentos, manuales y proyectos de investigación, para posteriormente procesar los datos sintetizando sus características fundamentales. Por último, se realizó una tabla comparativa que evidencia las capacidades, eficiencia y alcance de los sensores Android en el uso de aplicaciones de seguridad o de otras áreas de aplicación. La Tabla 3.1 resume los tipos de plataforma de ejecución de sensores Android.

Plataforma de Ejecución	
Plataforma	Función
Hardware	Obtener información de los sensores a través de componentes físicos incorporados en dispositivos móviles
Software	Recoger los datos provenientes de los componentes físicos de varios sensores basados en hardware

Tabla 3.1: Sensores basados en software y hardware

Elaborado por: Alejandro Mejía

3.1.1.1 Sensores basados en hardware

La Tabla 3.2 muestra la disponibilidad de sensores basados en hardware en Android.

Disponibilidad de sensores basados en hardware en versiones Android			
Sensor	Android 4.0	Android 2.3	Android 1.0
Acelerómetro	Disponible	Disponible	Disponible
Ambiental o Temperatura	Disponible	No	No
Gravedad	Disponible	Disponible	No
Giroscopio	Disponible	Disponible	No
Luz	Disponible	Disponible	Disponible
Aceleración	Disponible	Disponible	Disponible

Campo Magnético	Disponible	Disponible	Disponible
Presión	Disponible	Disponible	No
Proximidad	Disponible	Disponible	Disponible
Humedad	Disponible	Disponible	No
Rotación	Disponible	Disponible	No

Tabla 3.2: Sensores hardware disponibles en distintas versiones Android

Elaborado por: Alejandro Mejía

3.1.1.2 Sensores basados en software

La Tabla 3.3 muestra la disponibilidad de varios sensores basados en software en distintas versiones Android.

Disponibilidad de sensores basados en software en versiones Android			
Sensor	Versión		
	Android 4.0	Android 2.3	Android 1.0
Gravedad	Disponible	Disponible	No
Aceleración	Disponible	Disponible	Disponible
Orientación	Disponible	Disponible	Disponible
Rotación	Disponible	Disponible	No

Tabla 3.3: Sensores software disponibles en distintas versiones Android

Elaborado por: Alejandro Mejía

3.1.1.3 Categorías de Sensores

La Tabla 3.4 explica los objetivos principales de los sensores de movimiento, posición, ambientales y otros tipos de sensores.

Categoría de Sensores Android	
Categorías	Objetivo
Sensores de Movimiento	Mediante cálculos de medición determinan el valor de las fuerzas de rotación y aceleración en diversas direcciones
Sensores de Posición	Establecer la posición física de los dispositivos, en relación con el entorno en el que se encuentran.
Sensores Ambientales	Medir y obtener el valor de los atributos físicos del ambiente como: sonido, temperatura, presión del aire, humedad e iluminación del entorno.
Otros sensores	Generar datos personales a las que las aplicaciones pueden acceder y generar información

Tabla 3.4: Distintas categorías de sensores Android

Elaborado por: Alejandro Mejía

A. Ejemplos de Sensores de Movimiento

La Tabla 3.5 resume distintos ejemplos de sensores de movimiento integrados en dispositivos móviles Android.

Ejemplos de Sensores de Movimiento	
Categoría	Ejemplo
Sensores de Movimiento	Giroscopio
	Sensor de Gravedad
	Acelerómetro
	Sensor del vector de rotación

Tabla 3.5: Ejemplos de sensores de movimiento

Elaborado por: Alejandro Mejía

B. Ejemplos de Sensores Ambientales

La Tabla 3.6 resume ejemplos de sensores ambientales en dispositivos Android.

Ejemplos de Sensores Ambientales	
Categoría	Ejemplo
Sensores Ambientales	Barómetro
	Termómetro
	Luz Ambiental
	Humedad

Tabla 3.6: Ejemplos de sensores ambientales

Elaborado por: Alejandro Mejía

C. Ejemplos de Sensores de Posición

La Tabla 3.7 resume distintos ejemplos de sensores de posición integrados en dispositivos móviles Android.

Ejemplos de Sensores de Posición	
Categoría	Ejemplo
Sensores de Posición	Sensor de Orientación
	Magnetómetro
	Proximidad

Tabla 3.7: Ejemplos de sensores de posición

Elaborado por: Alejandro Mejía

D. Ejemplos de otros Sensores

La Tabla 3.8 resume distintos ejemplos de otros sensores integrados en dispositivos móviles Android.

Ejemplos de otros Sensores	
Categoría	Ejemplo
Otros Sensores	Cámara
	Micrófono

Tabla 3.8: Ejemplos de otros tipos de sensores Android

Elaborado por: Alejandro Mejía

3.1.1.4 Ejemplos de ataques de ciberseguridad basados en sensores

La Tabla 3.9 describe ejemplos ataques de ciberseguridad basados en sensores de movimiento, posición y ambientales.

Función de sensores Android		
Categoría del Sensor	Sensor	Descripción del Ataque
Sensores de Movimiento	Acelerómetro	Olfatear (sniffing) contraseñas de relojes inteligentes
		Inferir texto
		Reconocimiento del área de la pantalla digital de los dispositivos
		Decodificar vibraciones de teclados adyacentes

		Ataque del canal posterior del sensor acelerómetro
		Deducción de pulsaciones en entornos virtuales
		Inferir pulsaciones de teclas en relojes inteligentes
		Predecir contraseñas de teléfonos inteligentes
	Giroscopio	Inferir pulsaciones de teclas basado en la actividad
		Inspección de teclas independientemente del idioma
		Deducir contraseñas de cerraduras mecánicas
		Inferir información privada
		Valoración de riesgos de sensores de movimiento
		Descubrimiento del habla
Sensores Ambientales	Luz	Espionaje óptico de pantallas digitales
Sensores de Posición	Magnético	Difusiones electromagnéticas comprometedoras
		Conocimiento de impresiones
		Detectar ubicaciones

Tabla 3.9: Ataques de ciberseguridad basados en sensores Android

Elaborado por: Alejandro Mejía

3.1.1.5 Sensores aceptados por distintas plataformas móviles

En la Tabla 3.10 se detallan los sensores disponibles en los dispositivos de plataformas móviles como: Android, iOS y Windows Phone.

Sensores	Android	iOS	Windows Phone
Acelerómetro	Si	Si	Si
Campo Magnético	Si	No	No
Orientación	Si	No	Si
Presión	Si	No	No
Temperatura	Si	No	No
Acelerómetro Lineal	Si	No	No
Vector de rotación	Si	No	No
Humedad Relativa	Si	No	No
Temperatura	Si	No	No
Vector de rotación del juego	Si	No	No
Movimiento significativo	Si	No	No

Detector de pasos	Si	No	No
Contador de pasos	Si	No	No
Campo Magnético sin Calibración	Si	No	No
Proximidad	Si	No	Si
Vector de rotación geomagnética	Si	No	No
Giroscopio	Si	Si	Si
Gravedad	Si	No	No
Ritmo cardiaco	Si	No	No
Latido cardíaco	Si	No	No
Giroscopio	Si	No	No
Gravedad	Si	No	No
Giroscopio sin calibrar	Si	No	No
Detección de movimiento	Si	No	No
Luz	Si	No	Si
Detección Estacionaria	Si	No	No
Magnetómetro	Si	Si	Si
Altímetro	No	Si	Si
Podómetro	No	Si	Si
Sensor de Actitud	No	Si	No
Barómetro	Si	No	Si
Inclinómetro	No	No	Si
Actividad	Si	No	Si
Orientación Simple	No	No	Si
Brújula	Si	Si	Si

Tabla 3.10: Sensores aceptados por plataformas móviles

Elaborado por: Alejandro Mejía

3.1.1.6 Identificación y determinación de capacidad de sensores

La Tabla 3.11 explica la función de cada clase utilizada para interactuar con los sensores integrados en los dispositivos móviles dentro del área de desarrollo.

Métodos para determinar las capacidades de los sensores	
Método	Función
getSystemService()	Generar una referencia a los servicios de los sensores
getSensorList()	Obtener la lista completa de sensores integrados en el dispositivo
getDefaultSensor()	Obtener información de un sensor específico
getResolution()	Obtener la resolución y rango máximo de medición
getMaximumRange()	
getPower()	Información relacionada con los requisitos de energía
getVendor()	Optimizar el rendimiento de la aplicación, verificando la existencia de sensores y aplicando reglas sobre ellos.
getVersion()	
getMinDelay()	Mostrar referencias sobre el tiempo que tarda el sensor para detectar información

Tabla 3.11: Métodos para determinar la capacidad de los sensores Android

Elaborado por: Alejandro Mejía

3.1.1.7 Supervisión de eventos del sensor

La Tabla 3.12 resume los métodos para controlar los datos sin procesar del sensor.

Supervisión de eventos de sensores en Android		
Método	Condición de Ejecución	Objetivo
onAcurracyChanged()	Siempre que; el valor de la exactitud del sensor cambió	Controlar cuando una de las cuatro constantes: low, médium, high, unreliable, proporciona un valor distinto e informar sobre el nuevo valor de exactitud establecido.
onSensorChanged()	Siempre que; el sensor muestra un valor nuevo	Informar acerca del valor de los nuevos datos obtenidos por el sensor

Tabla 3.12: Métodos para supervisar eventos del sensor

Elaborado por: Alejandro Mejía

3.1.1.8 Funcionalidad de Sensores

La Tabla 3.13 explica las funciones de los sensores comunes integrados en los dispositivos Android.

Función de sensores Android		
Categoría	Sensor	Función
Sensores de Movimiento	Giroscopio	Medir la velocidad de rotación en torno a los ejes físicos (x, y, z) del entorno en el que se encuentra el dispositivo
	Sensor de Gravedad	Proporcionar el efecto de la fuerza de gravedad ejercida sobre el dispositivo indicando su valor de dirección y magnitud
	Acelerómetro	Medir la fuerza de aceleración en m/s, la aceleración estática, aceleración dinámica, y el campo gravitatorio de la Tierra.
Sensores Ambientales	Barómetro	Indicar la altitud y la presión atmosférica
	Temperatura	Medir la temperatura del dispositivo
	Luz Ambiental	Mide el nivel de luminosidad del ambiente
	Humedad	Obtener el valor de la humedad relativa del ambiente
Sensores de Posición	Sensor de Orientación	Medir el grado de orientación en base a los tres ejes físicos del ambiente
	Magnetómetro	Determinar en base a mediciones los valores de dirección y fuerza del campo magnético
	Proximidad	Detectar la distancia o cercanía de objetos respecto a la posición de la pantalla dispositivo.
	Sensor del vector de rotación	Obtener el valor de orientación del dispositivo proporcionado por los ejes físicos
Oros sensores	Cámara	Determinar la distancia entre la cámara del

		dispositivo y un objeto cercano, capturando la luz magnificada de varios lentes y generando una imagen 3D
	Micrófono	Capturar las ondas acústicas que producen los sonidos externos, y transmitirlo a través de aplicaciones

Tabla 3.13: Función de distintos sensores

Elaborado por: Alejandro Mejía

3.1.1.9 Análisis de sensores integrados en dispositivos Android

Para la determinación de los sensores a utilizar se optó por comparar los sensores comunes integrados en los dispositivos Android, según el área de aplicación, usos comunes, ventajas y desventajas. La Tabla 3.14 muestra los resultados obtenidos.

Sensor	Áreas de aplicación	Usos comunes	Ventajas	Desventajas
Giroscopio	• Juegos en primera persona • Sistemas de navegación • Sistema de referencia de rumbos • Realidad aumentada y virtual • Videos en 360 • Multimedia	• Detección de rotación • Eliminar aceleración de los datos cielo de Google • Sismos • Arcore de Google	Datos en tiempo real, mayor precisión, reconoce mejor las actividades como subir y bajar escaleras que el acelerómetro.	Error en las mediciones, los datos necesitan procesamiento, no se encuentra disponible en dispositivos de gama baja.
Sensor de Gravedad	• Autenticación • Seguridad de los dispositivos • Robótica • Seguridad de los usuarios • Juegos • Detección de maniobras laterales • Procesos de orientación	• Proporcionar datos vectores tridimensionales • Autenticación de usuarios • Describir la postura del dispositivo técnicas de autenticación • Control de gestos • Control de movimiento de extremidades de un robot inteligente • Detección de caídas	Utilizando autenticación G-key, es mejor que autenticaciones biométricas o por contraseña, ya que no requiere el ingreso de credenciales de usuarios y reduce la carga de memoria	Los datos recopilados son muy precisos cuando el dispositivo o el usuario están en reposo, sin embargo, cuando existe un cambio de posición, se presenta una disminución en la precisión e incluso un fallo en la localización
Acelerómetro	• Mapa • Salud • Deporte • Juegos en primera persona • Capacitación • Educación • Seguridad • Cuidado de la salud	• Detectar inactividad y actividad vigorosa • Autocorrelación • Determinar la fuerza de gravedad • Detección de movimiento • Autenticación biométrica • Diagnóstico de enfermedades	Provee datos en tiempo real, mayor precisión en datos de reconocimiento, más lineal y preciso que un giroscopio, reconoce mejor	Antes de utilizarlo es necesario reorientarlo en los ejes correctos para compensar posibles desviamientos y obtener mediciones incorrectas.

	• Industria • Automatización • Vigilancia • Tecnología informática • Actividad humana • Protección de privacidad	• Monitoreo del sueño • Detección de gestos • Implementar aplicaciones de realidad aumentada • Airbags de automóviles • Sistemas de detección de multitudes • Cuantificar actividades	las actividades como estar de pie, caminar, andar en bicicleta y trotar, bajo consumo de batería	
Sensor del vector de rotación	• Juegos en primera persona • Sistemas inteligentes y computacionales • Modelos de aprendizaje profundo • Detección de gestos • Realidad aumentada	• Orientación del dispositivo móvil • Supervisión del movimiento • Detección de rotación • Sistemas de control • Google Glas • Ángulo de rotación del dispositivo • Control remoto • Clasificar características • Aplicaciones de estabilización de cámara	El potencial es elevado, ya que puede combinarse con los sensores comunes para desarrollar sistemas de control, precisión en modelos de clasificación.	Si el dispositivo no tiene un sensor giroscopio, el sensor vector de rotación no se podrá visualizar y por lo tanto no podrá ser utilizado.
Barómetro	• Servicio Meteorológico • Aplicaciones de Ubicación	• Presión del aire • Cambios del clima • Medir altitud relativa • Detectar caídas • Cambios de presión • Posicionamiento absoluto	Ayuda a obtener localizaciones más precisas del GPS.	En presencia de gradientes de presión la exactitud de la medición de la altitud está limitada, se limitan a mediciones de altitud.

Temperatura	• Industria • seguridad de los dispositivos	• Temperatura corporal • Temperatura ambiental • Control de la temperatura de los componentes del dispositivo • Apagar el dispositivo en caso de sobrecalentamiento • Informe de condiciones climatológicas	Evitar temperaturas altas en el dispositivo móvil, detener carga del dispositivo cuando la batería exceda el límite.	La temperatura de la batería, el uso de la CPU y la temperatura de superficie de contacto influyen en los datos recopilados por el sensor, lo que provoca mediciones inexactas.
Luz Ambiental	• Control de la salud • Entretenimiento personalizado	• Control del nivel de brillo de pantalla en dispositivos electrónicos • Ejecución de acciones en respuesta a cambios • Ajuste de brillo automático	Todos los dispositivos móviles de gama alta disponen de este sensor e incluso algunos de gama baja.	El ajuste de brillo automático puede variar en ciertas aplicaciones, con un porcentaje de batería bajo el sensor no permite ajustar el brillo a niveles altos.
Humedad	• Agrícola • Cuidado de la salud • Ambiente • Bienestar del dispositivo	• Humedad ambiental • Humedad absoluta y relativa monitoreo del punto de roció • Medir niveles de humedad de la piel	Detecta si la avería del teléfono fue provocada por contacto con el agua.	Los dispositivos pueden corroerse en presencia de niveles altos de humedad.
Sensor de Orientación	• Multimedia • Ubicación	• Grado de orientación del dispositivo • Reproductor de video • Navegadores • Análisis de posición	Pueden ser utilizados por aplicaciones interesadas en el análisis de ubicación	En sistemas de detección de caídas el sensor de orientación en comparación con el sensor de gravedad es más probable a detectar caídas falsas.

		• Sistemas de sugerencia de lugares populares.	geográfica de las personas.	
Magnetómetro	• Ubicación • Búsqueda de objetos • Procesos de orientación	• Elaboración y funcionamiento de una brújula • Eliminar aceleración de los datos • Identificar posición de ubicación respecto a la tierra • Detector de metales • Google Maps • Apple Maps	Datos en tiempo real, mayor precisión, puede ser utilizado en aplicaciones de detección de metales.	Las mediciones del campo magnético pueden ser distorsionadas por campos variables provocados por dispositivos cercanos, aparatos electrónicos y objetos metálicos.
Proximidad	• Seguridad personal • Comunicación	• Detectar objetos y cambios de orientación • Apagar la pantalla durante llamadas telefónicas	Cuando está en ejecución desactiva otras funciones para ahorrar energía del dispositivo.	Al desactivarlo no se podrá utilizar las funciones de brújula, acelerómetro, cámara y sensor de proximidad.
Cámara	• Medicina • Escáner de código • Redes sociales • Multimedia	• Enlaces web inmediatos • Búsqueda de imágenes • Monitor de pulso cardíaco • Medir altura y distancia • Grabación de videos • Tomar fotografías	Recoge más datos en órdenes de magnitud, en combinación con otros sensores da paso al desarrollo de las aplicaciones como redes sociales, los datos recogidos se efectúan en tiempo real.	Oclusiones, Intrusiva, Condiciones de adquisición, vulnerable a hackeos, la calidad de los datos (imágenes) dependerá de la marca, modelo, sistema operativo del dispositivo móvil.

Micrófono	• Multimedia • Redes sociales • Transporte • Grabación de sonido • Seguridad personal • Realidad virtual.	• Procesamiento del lenguaje • Reconocimiento de identidad por voz • Identificar canciones • Grabar sonido • Detección del modo de transporte • Asistente de voz • Método de entrada de datos para gafas inteligentes.	Obtener sonidos de alta calidad en estéreo, reducción de ruido mientras el dispositivo integre varios micrófonos, es posible controlar el dispositivo mediante comandos de voz	Presencia de ruido, la distancia que la persona debe mantener con el dispositivo, intrusivo, irrumpe la privacidad, espionaje cibernético, la calidad y claridad del sonido capturado varia en dependencia de la cantidad de micrófonos existentes

Tabla 3.14: Comparativa de Sensores Android

Elaborado por: Alejandro Mejía

Con base al análisis realizado en la Tabla 3.14 respecto a los sensores Android, se ha determinado utilizar el acelerómetro y sensor de orientación debido a que permiten recopilar datos en tiempo real con mayor precisión. Además, con frecuencia son utilizados en HAR por ser más lineales

3.1.2 Análisis de bases de datos públicas para HAR

3.1.2.1 Características de bases de datos públicas para Reconocimiento de Actividad Humana

La Tabla 3.15 resume las principales características de las bases de datos para (HAR).

Base de datos	Sensor	N° de actividades	N° de usuarios	Datos recopilados	Herramienta
WISDM	Acelerómetro	6	36	Procesados, sin procesar	Dispositivos móviles Android
REALDISP	Acelerómetro, giroscopio, orientación	33	17	Sin procesar	Dispositivos móviles, relojes inteligentes
Human Activity Recognition Using Smartphones	Acelerómetro, giroscopio	6	30	Procesados, sin procesar	Dispositivo con sensor inercial
OPPORTUNITY	Acelerómetro, inercial	4	4	Sin procesar	Sensores integrados
HHAR	Acelerómetro, giroscopio	6	9	Sin procesar	Relojes inteligentes, dispositivo móvil
CHEST	Acelerómetro	7	15	Sin procesar	Disp. Google Nexus One

Tabla 3.15: Resumen de bases de datos públicas con datos obtenidos del acelerómetro para HAR

Elaborado por: Alejandro Mejía

3.1.2.2 Características de bases de datos unimodales para Reconocimiento de Actividad Humana

La Tabla 3.16 resume el comportamiento, opiniones, cantidad y ubicación del acelerómetro, de las bases de datos unimodales para (HAR).

Base de datos	N° de personas voluntarias	Actividades registradas	N° de acelerómetros	Ubicaciones del sensor	Opiniones
MIT Place Lab	1	Preparar recetas, buscar artículos	5	• Piernas • brazos • cadera	Los resultados obtenidos por los sensores sobre las actividades no son precisos, debido a que solo se evalúa el comportamiento de una persona.
Ward	20	Ponerse de pie, caminar, sentarse	5	• Extremidades superiores e inferiores • Cintura • Tobillos	La pérdida de datos es un problema en la recolección de información
Usc-Had	14	Caminar, subir y bajar escaleras	1	• Cadera	La información recolectada proviene de un acelerómetro.
Realdisp	17	Correr, saltar, caminar, rotar, trotar	9	• Muslos • Pantorrillas • Brazo • Antebrazo	Contiene gran cantidad de datos eficientes para HAR, al recopilar una gran cantidad de tipos de actividades y sensores.

Tabla 3.16: Resumen de bases de datos unimodales públicas que utilizan datos del acelerómetro

Elaborado por: Alejandro Mejía

3.1.2.3 Áreas de aplicación de los datos obtenidos por sensores para HAR

La Tabla 3.17 resume las principales áreas o ámbitos de aplicación de los datos obtenidos por sensores en el Reconocimiento de Actividad Humana.

Área de Aplicación	Ejemplo de Aplicación
Educativa	Enseñanza por medio de ambientes virtuales
Realidad virtual	Capacitación
Seguridad	Aplicación móvil para reconocimiento de identidad biométrica, adherencia médica
Medicina	Diagnósticos de enfermedades comunes: Parkinson, Alzheimer, Epilepsia. Otra aplicación es la detección de accidentes provocados por caídas.
Bienestar de la salud	Seguimiento de un plan alimenticio, evaluación del estado de salud de personas de la tercera edad, monitoreo del sueño
Vigilancia	Monitoreo del comportamiento infantil
Automatización	Control del entorno diario
Juegos	En juegos de azar: descubrir acciones y gestos para jugar
Industrial	Monitoreo de un robot industrial: brazo mecánico
Deportiva	Detección de ejercicios para gimnasia

Tabla 3.17: Áreas de aplicación de los datos obtenidos de los sensores para HAR

Elaborado por: Alejandro Mejía

3.1.2.4 Actividades recolectas por bases de datos HAR

La Tabla 3.18 resume las principales características: sensores utilizados, frecuencia de los sensores, actividades y cantidad de usuarios.

Base de datos	Sensores	Frecuencia	Actividades	Usuarios
IM-WSHA	3 sensores IMU	100 Hz	Beber, lectura, caminar, cocinar	10
PAMAP-2	3 sensores IMU	9 Hz	Caminar, planchar, ciclismo, sentarse	9
MobiAct	Dispositivo móvil	20 Hz	Estar de pie, acostarse, trotar	19
UCI-HAR	Acelerómetro, giroscopio	50 Hz	Caminar, bajar escaleras	30
MOTIONSENSE	Dispositivo móvil	50 Hz	Bajar o subir escaleras, correr	24

Tabla 3.18: Actividades recolectadas por bases de datos HAR

Elaborado por: Alejandro Mejía

3.1.3 Análisis de metodologías para gestión de proyectos

La Tabla 3.19 resume las principales características de las metodologías tradicionales y ágiles enfocados en la gestión de los proyectos de desarrollo de software.

Áreas	Tradicional	Ágil
Organización	Grande	Pequeña
Comunicación	Comunicación Formal	Comunicación Informal
Criticidad	Extrema	Baja
Tamaño del equipo	Grande	Pequeño
Requisitos del proyecto	Los requisitos y especificaciones se mantienen estables	Los requisitos cambian durante el proceso
Requerimientos	El requisito inicial tiene que ser claro y conciso	No es claro por lo cual este sujeto a cambios
Involucrados del proyecto	Los clientes tienen una participación baja dentro del proceso	Los clientes están altamente involucrados en el proyecto
Escala del proyecto	Grande	Pequeña
Características del equipo de desarrollo	Grande, totalmente hábil y capaz	Pequeño, cooperativo con las tareas a realizarse
Modelo de desarrollo	Sigue la estructura del modelo de ciclo de vida	Iterativo y evolutivo

Control de calidad del producto	Planificación, pruebas tardías	Evaluación constante de los requerimientos y pruebas
Disposición de la empresa	Cumple a cabalidad las demandas	Permite incluir cambios

Tabla 3.19: Comparativa entre metodologías tradicionales y ágiles

Elaborado por: Alejandro Mejía

Con base al análisis realizado en la Tabla 3.19 se ha determinado utilizar metodologías ágiles, debido a que comprometen directamente a los interesados durante el desarrollo del ciclo de vida del proyecto, convirtiéndola en una metodología flexible y fácil de implementar en cualquier aplicación, Además, es iterativo, ofreciendo la posibilidad de realizar cambios significativos hasta obtener un producto de alta calidad.

3.1.4 Análisis de metodologías para desarrollo de software

Para la determinación de la metodología a utilizar se efectuó un benchmarking genérico entre las metodologías más comunes en el desarrollo de software. El análisis se llevó a cabo empleando como factores de análisis las cualidades, flexibilidad, costo de implementación y cualidades. La información se obtuvo de documentos, manuales y proyectos de investigación, para posteriormente procesar los datos sintetizando sus principales características. Por último, se realizó una tabla comparativa que evidencia la escalabilidad de las metodologías en la aplicación de proyectos de investigación. La Tabla 3.20 muestra los resultados obtenidos.

Método	Cualidades	Flexibilidad	Costo	Fortalezas	Debilidades
Scrum	• Equipó de desarrollo organizado • Incremental e iterativa • Marco adaptativo • Define roles específicos para el equipo • Inspección • Transparencia	Flexible	Alto	• El cliente interviene en el proyecto • Establece pautas para mejorar la comunicación • Efectúa reuniones diarias para no perder de vista los requerimientos establecidos • Entrega inmediata del producto final • Los requisitos se adaptan a las necesidades • Comunicación frecuente del equipo • Respeto entre los involucrados	Incomodidad de adaptación del equipo. El gerente del proyecto no interviene en el procedimiento. Se enfoca demasiado en la planificación lo que puede retrasar los procesos. La documentación no es un requerimiento. El costo no es un factor medible al inicio de cualquier proyecto.
LSD	• Iterativo • Mejorar el producto es un requerimiento • Prioriza la eficiencia • Reduce las interrupciones • Intervención activa de los clientes • Controla el desarrollo del software • Evitar la producción innecesaria • Equipo motivado • Reduce las pérdidas • Optimización	Flexible	Medio	• Basado en pruebas continuas • Reducir los costos de implementación • Los componentes de software son reutilizables • Entrega inmediata del código del proyecto • Producto de alta calidad • Motivación del gerente de proyectos. • Fomenta la eficiencia y productividad de los procesos • Los logros alcanzados son reconocidos	La documentación es explícita, de manera que incumplir con ello ocasiona retrasos. Los requisitos no son claramente definidos, lo que puede exceder el tiempo de implementación del proyecto. El proyecto depende en su totalidad por el equipo de desarrollo.

XP	• Participación de los clientes durante el desarrollo • Pruebas continuas en cada etapa mantenimiento del producto • La comunicación es un valor • Sencillez • Prioriza tareas necesarias • Basado en los requisitos del proyecto • Retroalimentación • Coraje • Diseño incremental • Roles adecuadamente establecidos	Flexible	Alto	• Los requisitos de los usuarios se detallan en historias, las cuales describen exactamente las metas • Integración continua • Promueve el desarrollo basado en pruebas • El cliente toma decisiones sobre el proyecto • Comunicación abierta • Desarrollo inmediato	Las reuniones frecuentes son necesarias. Frecuentes cambios afectan la calidad del producto y por lo tanto no existe una documentación. Es difícil cumplir con el ciclo de vida de XP, cuando el cliente no puede estar siempre presente con el equipo. Es costoso reunir el cliente y al equipo en el mismo sitio.
DSDM	• Metodología basada en pruebas continuas • Iterativo • Participación de los usuarios • Flexibilidad • Calidad • Ciclo de vida completo • Escalabilidad • Colaborativa • Comunicación continua • Control	Poco flexible	Medio	• El equipo de desarrollo toma las decisiones • La calidad de los productos • Independiente del proveedor • Ofrece sugerencias para entregar el proyecto a tiempo • Adaptable para cualquier sector comercial • Intervención recurrente de los usuarios • La entrega de los proyectos cumple con el presupuesto establecido	Generar la documentación es un proceso demasiado complejo. Tanto el equipo de desarrollo como los gerentes del proyecto tienen que ser altamente capacitados. Es necesario contar con un gran equipo, para abarcar con todas las tareas por cumplir.

	• Rapidez				
Crystal Methods	• Adaptabilidad • Flexibilidad • Comunicación efectiva • Implementación sencilla • Colaborativa • Desarrollo incremental	Flexible	Bajo	• La comunicación eficaz es la clave de éxito del proyecto • Proporciona control de eventualidades • Existen diferentes sub-métodos que pueden adaptarse según la naturaleza del producto • Prioriza la entrega inmediata del producto	Los principios cambian según el tiempo, tamaño del equipo y tipo del proyecto, por lo cual entenderlo resulta complejo. No es aplicable en proyectos con múltiples tareas. La toma de decisiones críticas no involucra a todo el equipo.
Kanban	• Flexibilidad • Optimización • Control • Control de calidad • Aumento de la producción • Reducción de costos • Rendimiento • Adaptación • Control de problemas • Producción • Optimización • Administración			• Los productos o componentes son fabricados solo cuando sea necesario • Control y eliminación de la sobreproducción • Prioriza la reducción de costos, tiempo y desperdicio • Control del inventario evitando un exceso innecesario • En presencia de problemas el flujo de los procesos se detiene para corregirlos, por lo cual, los problemas son examinados con claridad	La aparición de problemas repentinos puede ocasionar períodos de inactividad impredecibles que interrumpan la actividad del sistema. Es adecuado para proyectos en los cuales las metas están claramente definidas y las actividades son consistentes
MADLC	• Evolutiva • Versatilidad • Eficiente	Flexible	Medio	• Está diseñado para cumplir con los requisitos del usuario, por lo cual su participación es activa	La metodología está diseñada para ambientes con equipos

	• Adaptabilidad • Escalable • Multifuncional • Comunicación • Calidad • Productividad • Estabilidad • Efectividad • Continuidad			• Comentarios recurrentes de los usuarios • Se adapta fácilmente en aplicaciones orientadas a la industria • Permite abordar aplicaciones móviles con numerosas funciones de manera eficaz y acertada • En la fase de prototipado permite realizar correcciones recursivas hasta conseguir un prototipo ideal • Es una metodología actual que permite adaptarse fácilmente a las nuevas tecnologías, demandas de organizaciones y requerimientos de los clientes • Permite crear a los desarrolladores aplicaciones de alta calidad • El mantenimiento es un proceso continuo	de software bien capacitados. El cumplimiento de cada actividad requiere extrema dedicación del equipo.
Waterfall	• Sencillez • Implementación • Simplicidad • Lineal • Gestión • Comprensión • Jerárquico • Secuencial • Previsibilidad • Comunicación • Independiente	No flexible	Bajo	• Adecuado para proyectos sencillos, en los cuales el objetivo no está completamente definido • Las etapas están claramente definidas • El costo de implementación es bajo. • Cada proceso tiene sus propios objetivos y resultados • Documentación completa y sencillamente descrita	Cada etapa tiene que estar completada en su totalidad, por lo cual no se puede seguir a la siguiente actividad, provocando retrasos en el desarrollo. No existe retroalimentación, por lo cual resultado difícil corregir errores.

	• Documentación clara • Rigurosidad • Inmutable • Presupuesto cerrado • Análisis de necesidades			• Fácil gestión y comprensión por el equipo del proyecto, incluso en proyectos de alta demanda • Las fases son completadas y procesadas por separado • Adecuado para implementar en proyectos pequeños	Presencia de errores por implementar nuevos requisitos, lo cual provoca problemas para las fases posteriores, ya que todas las fases son continuas.
V-Model	• Sencillez • Diseño directo • Compresible • Arquitectura eficiente • Versatilidad • Optimización • Integridad • Calidad	No flexible	Medio	• Realización de pruebas continuas para todas las etapas • La implementación del proyecto consume menos tiempo • La metodología permite cambiar los requerimientos durante la etapa de desarrollo • Promueve la importancia del mantenimiento del producto • Minimización de errores • Incluye medidas integradas para asegurar la calidad	Carece de prototipos de software en la fase de requerimientos. No se puede implementar en proyectos pequeños. Es una metodología completamente rígida. No es adecuada para proyectos orientados a objetos. Inestabilidad.
RUP	• Calidad • Proactivo • Integración continua • Organización • Documentación precisa • Iterativo • Control de actividades • Previsibilidad • Aborda los riesgos • Cumple las expectativas	Poco flexible	Alto	• Prioriza entregar un producto que satisfaga las necesidades del cliente. • Generar productos de alta calidad y con el presupuesto adecuado • Disminución de riesgos durante el desarrollo • Tareas claramente definidas y asignadas • Resuelve las eventualidades con facilidad	El equipo debe ser altamente calificados para el desarrollo de software. La etapa de desarrollo es compleja, por lo cual, para proyectos grandes genera confusión y problemas Costo de producción elevado.

	• Compatibilidad • Eficiencia			• La integración requiere menos tiempo • Documentación adecuada, que permite a los involucrados comprender y utilizar el producto. • Capacitación fácilmente disponible para los usuarios mediante tutoriales paso a paso de los procesos • Mayor nivel de reutilización de código • Resuelve proactivamente los riesgos	No se puede aplicar en proyectos pequeños. Desorganización. No es posible reutilizar los componentes. Requiere un cambio de mentalidad en los involucrados, para enfocarse en los riesgos, y tolerar incertidumbres.

Tabla 3.20: Comparativa de metodologías de desarrollo de software
Elaborado por: Alejandro Mejía

Con base al análisis realizado en la Tabla 3.20 se opta por utilizar la metodología MADLC, debido a que es una metodología actual que permite desarrollar aplicaciones móviles con numerosos módulos y funciones, obteniendo resultados eficaces y de alta calidad. De igual forma se utilizará la metodología Kanban debido a que permite gestionar las actividades, optimizar el tiempo y mejorar el flujo de las actividades.

3.1.5 Análisis de herramientas para el desarrollo móvil

Para determinar la herramienta de desarrollo de aplicaciones móviles se efectuó un benchmarking genérico entre Android Studio y Visual Studio. El análisis se llevó a cabo realizando una búsqueda entre documentos, manuales y proyectos de investigación que contengan información fundamental que aporte valor a la investigación, para posteriormente procesar los datos sintetizando sus principales características. Una vez que los datos fueron procesados, se realizó una tabla comparativa que expone sus ventajas y desventajas. La Tabla 3.21 muestra los resultados obtenidos.

Plataforma	Ventajas	Desventajas
Android Studio	• Ejecución de código inmediata • Sistema de construcción sencillo • Permite crear aplicaciones compatibles con los distintos dispositivos Android • Posee un editor de diseño eficaz y confiable • Construir aplicaciones altamente competitivas • Posee un sistema inteligente que identifica errores de código y muestra posibles soluciones • Incluye un emulador que ayuda a probar las aplicaciones antes de implementarlas • El emulador Android permite comprobar la compatibilidad de hardware y software de las aplicaciones generadas • Ayuda a determinar la integridad de la aplicación • Sistema de construcción altamente flexible que permite personalizar las interfaces con plena libertad • Permite la creación de nuevos módulos sin la necesidad de cambiar de espacio de trabajo • Enfatiza en sus desarrolladores la reutilización de código y el trabajo colaborativo • Permite ejecutar aplicaciones desde la línea de comandos, ahorrando consumo de memoria • Las nuevas versiones de Android Studio incorporan un nuevo formateador que facilita a los desarrolladores dar seguimiento a los registros, es decir distinguir los tipos de registros entre mensajes de error o advertencias • El código generado es público y compartido	• Probablemente su principal problema es el consumo de memoria, los computadores dependen de ciertos requisitos mínimos para ejecutar aplicaciones, sobre todo aquellas que integran numerosas funciones, capas, módulos, etc. • El proceso de instalación es lento, debido a la cantidad de paquetes que debe descargar. • La opción de virtualización necesariamente debe estar activa para usar el emulador de Android Studio, lo que puede ocasionar desconcierto en usuarios que desconocen la herramienta.

	• No es necesario pagar cuotas para desarrollar aplicaciones de alto nivel en Android • Rapidez para abrir proyectos utilizados recientemente	
Visual Studio	• Compatibilidad con varios lenguajes de programación: CSS, C++ y Python • Posee una arquitectura sólida con herramientas y funciones que mejoran la experiencia con el usuario • Es personalizable • Permite crear aplicaciones nativas compatibles con IOS, Android y Windows • Depuración rápida • Puede ser utilizado como depurador de código • El soporte en línea lo hace escalable • Incorpora un administrador de funciones que localiza complementos similares y los mantiene actualizados • Las aplicaciones desarrolladas pueden ser implementadas en diferentes dispositivos • Ofrece soporte para carba balanceada, uno de los métodos que permiten a múltiples servidores ejecutarse de forma simultánea • Permite agregar varios módulos en una misma aplicación sin la necesidad de modificarla • Integra soporte para ejecutar código no seguro • Apto para lenguajes de programación que requieren compilación • Es posible ejecutar los programas desde el símbolo del sistema de Windows	• Las actualizaciones de paquetes o complementos, e incluso la instalación resulta exhaustiva en muchos de los casos • Para ciertos complementos suele ser necesario instalarlos y personalizarlos, por lo que puede ser difícil para usuarios novatos • A diferencia de otros entornos, las aplicaciones de Visual Studio están desarrolladas solo para ejecutarse sobre plataformas Windows

Tabla 3.21: Comparativa de herramientas de desarrollo de aplicaciones móviles

Elaborado por: Alejandro Mejía

Con base al análisis realizado en la Tabla 3.21 se decidió utilizar Android Studio como herramienta para desarrollar la aplicación móvil ya que permite crear aplicaciones con alto nivel de calidad, compatibles con distintas plataformas y dispositivos inteligentes Android.

3.2 Desarrollo de la Propuesta

3.2.1 Aplicación de la metodología MADLC y Kanban

El presente proyecto se desarrolló combinando las cinco fases de la metodología MADLC (Identificación, Diseño, Desarrollo, Prototipado, Prueba) y la metodología Kanban que consta de seis fases. De la metodología Kanban se utilizará la primera fase (visualizar el flujo de trabajo) ya que permite llevar un control y gestionar el flujo de los procesos para el desarrollo de la aplicación. Para la metodología MADLC solo se requerirá el uso de las cinco fases iniciales que se ajustan con los objetivos del proyecto.

3.2.2 Fase I: Visualizar el trabajo en Kanban

3.2.2.1 Visualizar el flujo de trabajo

Para visualizar el flujo de trabajo se utilizó el software de administración de proyectos Trello, ya que es una herramienta útil para gestionar las actividades pertenecientes a cada etapa de la metodología MADLC. A través de un tablero Kanban se representa el flujo de los procesos, para lo cual se establecen tres etapas: pendiente, en proceso, finalizado. La primera etapa contiene la lista de todas las actividades planificadas, la segunda etapa por el contrario especifica las tareas que se encuentran en ejecución, pero que aún no se completan, y la etapa "Finalizado" que contiene las actividades realizadas. La Figura 3.18, Figura 3.19 y Figura 3.20 representan lo expuesto.

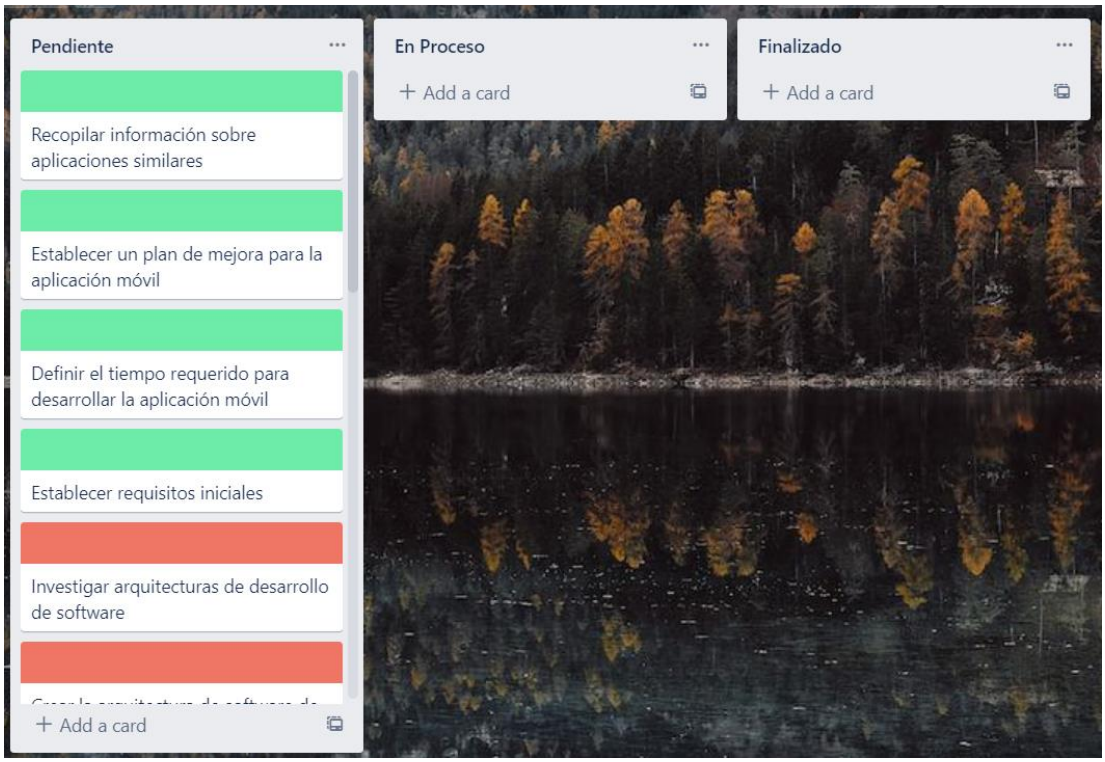


Figura 3.18: Tablero Kanban I

Elaborado por: Alejandro Mejía

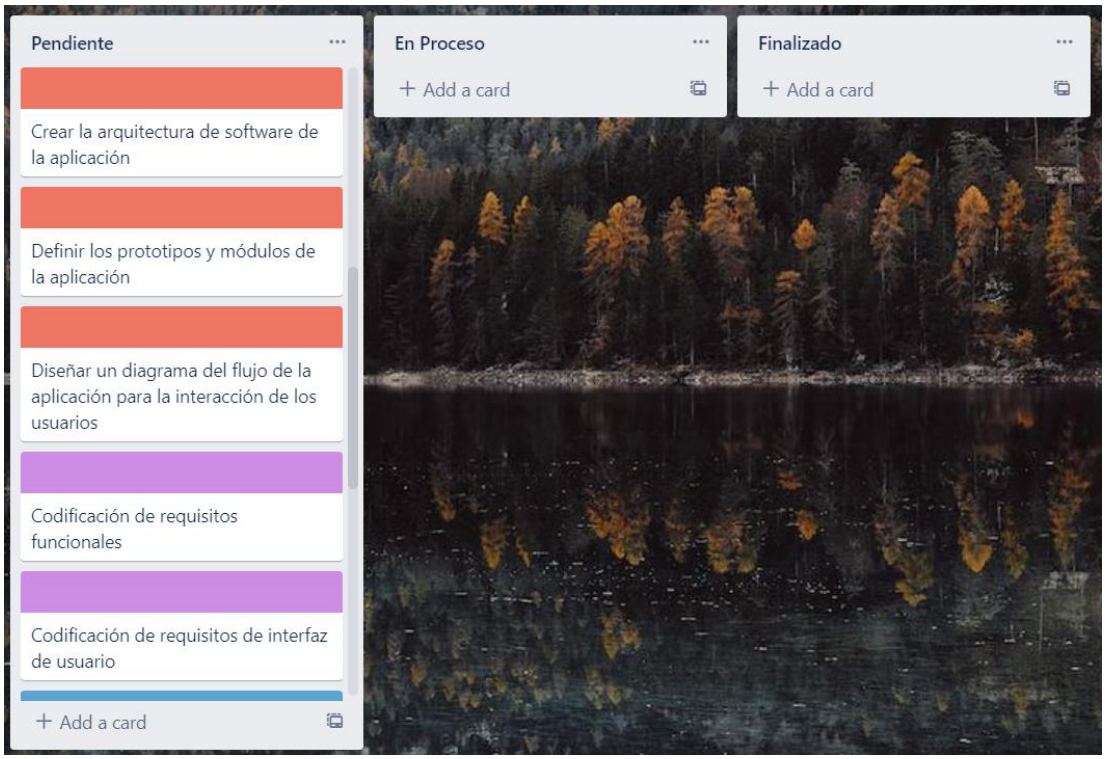


Figura 3.19: Tablero Kanban II

Elaborado por: Alejandro Mejía

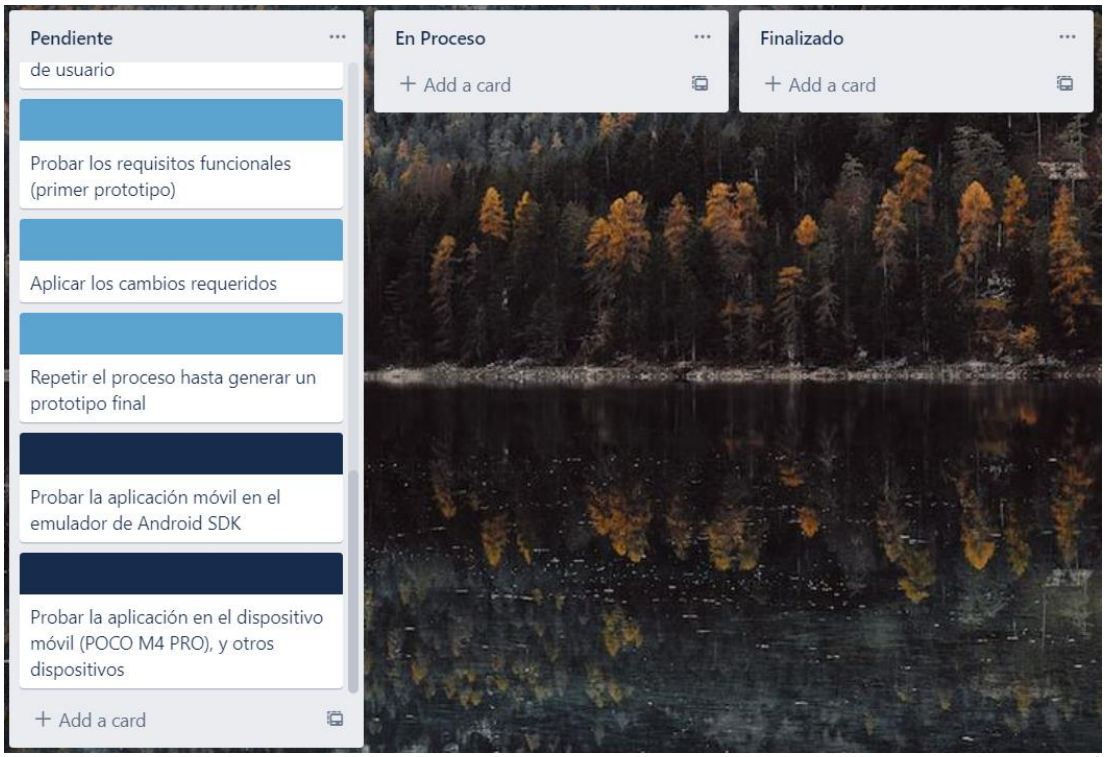


Figura 3.20: Tablero Kanban III

Elaborado por: Alejandro Mejía

Se especificó detalles adicionales en los procesos, con la finalidad de llevar un registro claro y estructurado de las actividades. Los procesos contienen un campo personalizado ubicado en la parte superior que representa la fase de la metodología a la que pertenece, mientras que la prioridad del proceso está representada por una etiqueta. La Tabla 3.22 detalla los colores asignados en las fases de la metodología MADLC.

Fases	Color
Fase de Identificación	
Fase de Diseño	
Fase de Desarrollo	
Fase de Prototipado	
Fase de Prueba	

Tabla 3.22: Fases de la metodología MADLC para procesos Kanban

Elaborado por: Alejandro Mejía

La Tabla 3.23 detalla los colores asignados en la prioridad de los procesos

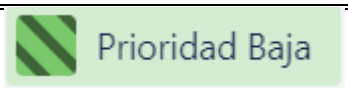
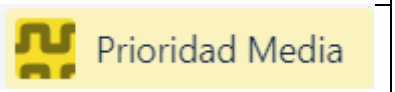
Prioridad	Color
Media	
Baja	
Alta	

Tabla 3.23: Prioridad de procesos de Kanban

Elaborado por: Alejandro Mejía

Luego de definir la gama de colores necesaria para representar las fases y la prioridad, se añadieron los detalles a la lista de actividades del tablero Kanban.

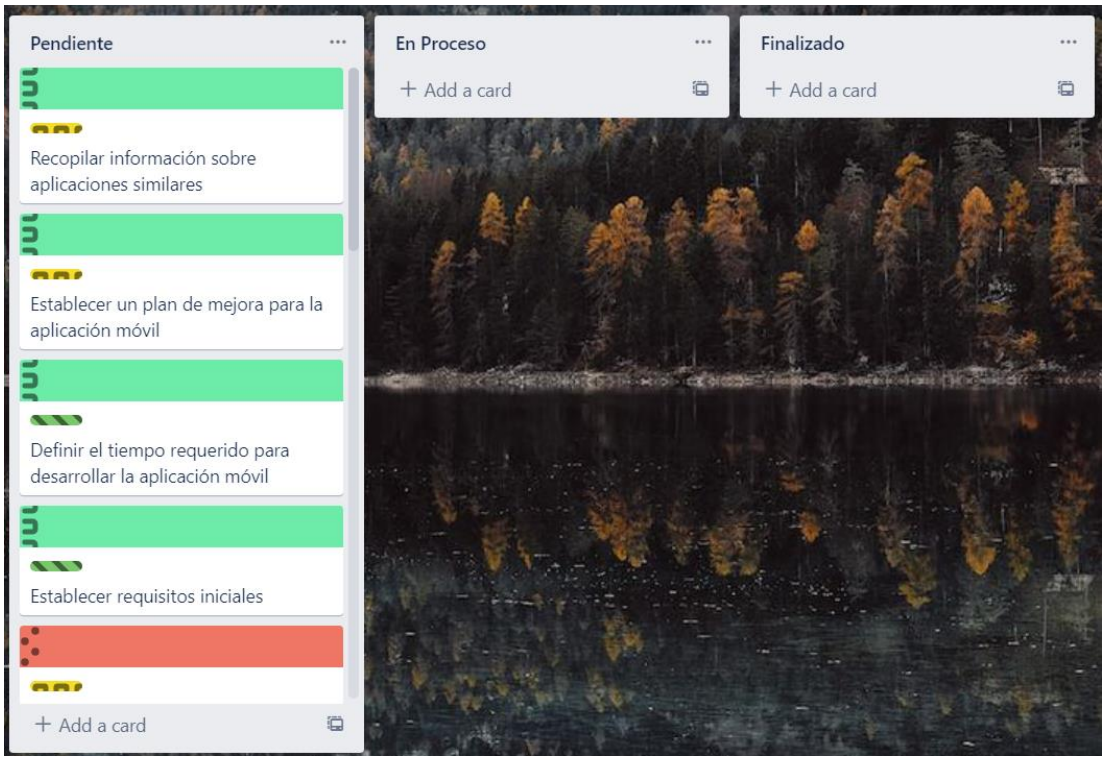


Figura 3.21: Tablero Kanban IV

Elaborado por: Alejandro Mejía

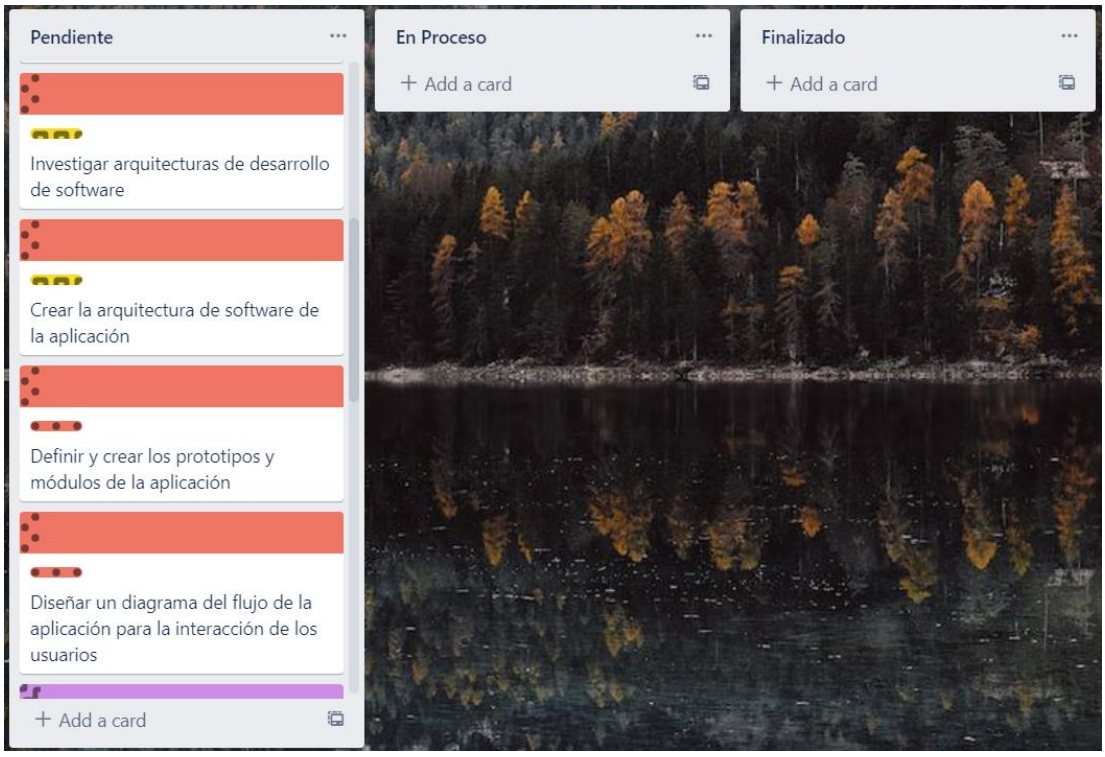


Figura 3.22: Tablero Kanban V

Elaborado por: Alejandro Mejía

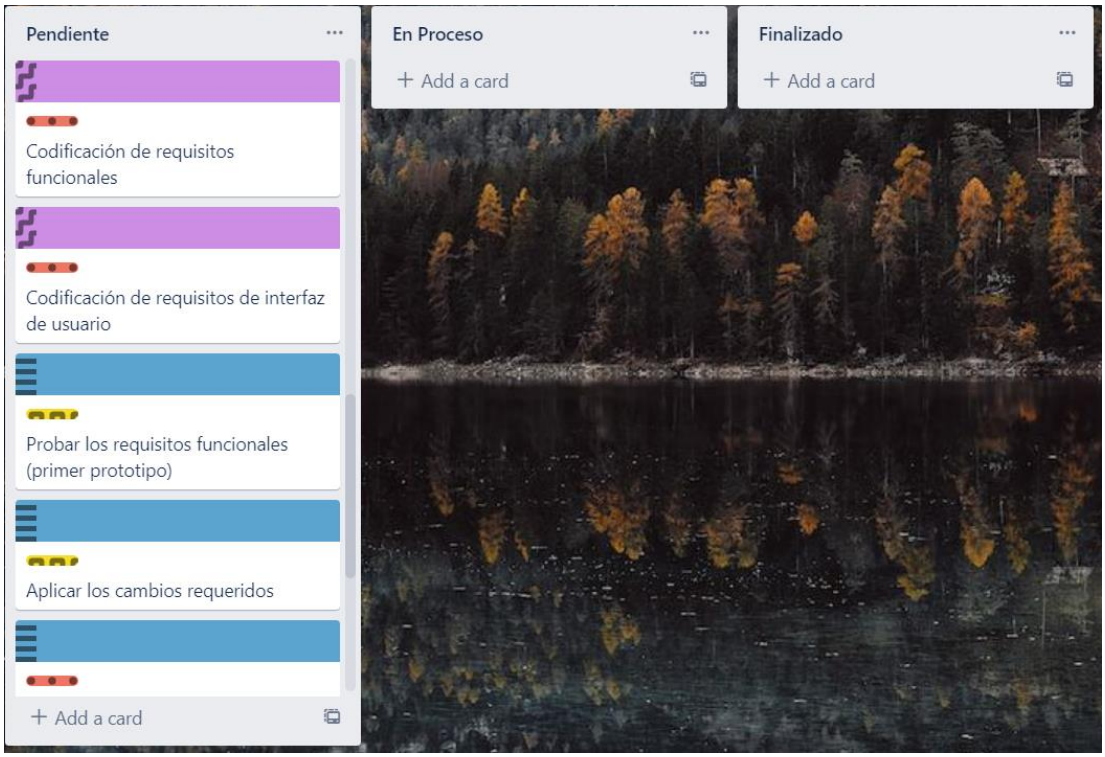


Figura 3.23: Tablero Kanban VI

Elaborado por: Alejandro Mejía

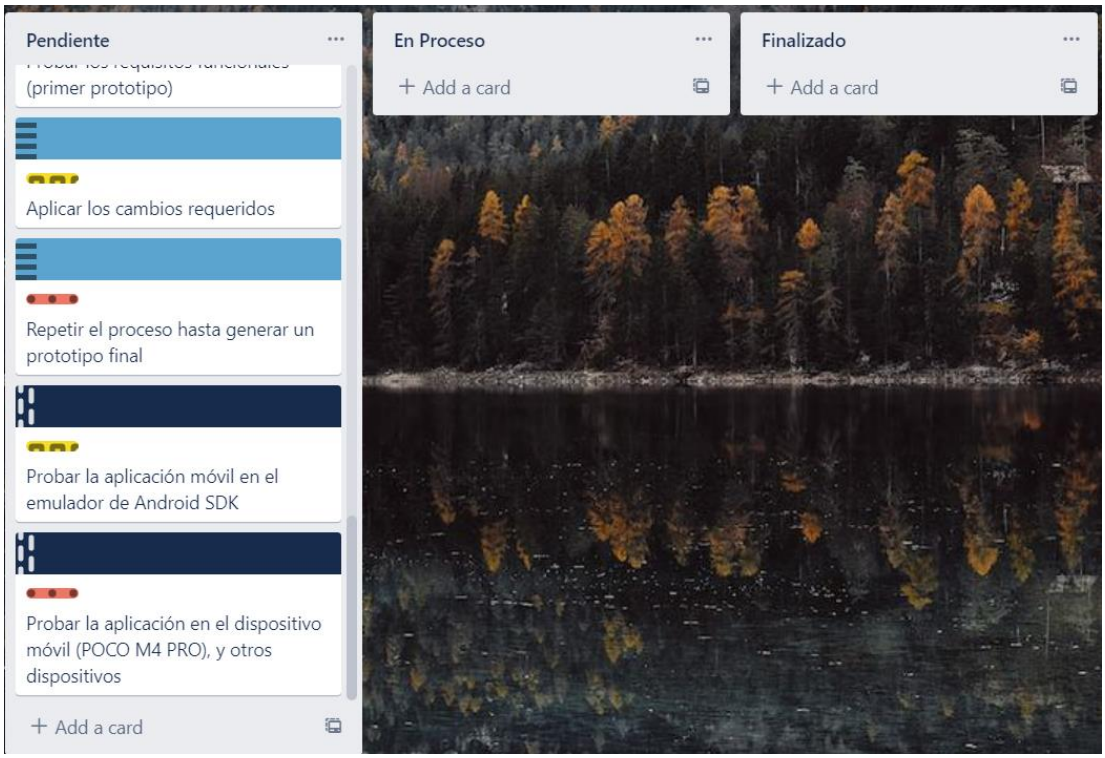


Figura 3.24: Tablero Kanban VII

Elaborado por: Alejandro Mejía

3.2.3 Fase de Identificación

3.2.3.1 Recopilar información sobre aplicaciones similares

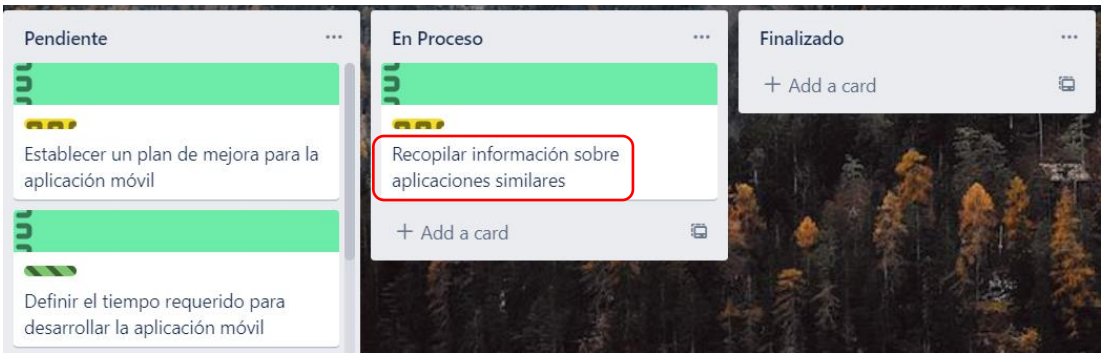


Figura 3.25: Desarrollo de Fase de Identificación Proceso I

Elaborado por: Alejandro Mejía

En la presente etapa, se recaba información en distintos repositorios institucionales sobre aplicaciones móviles similares, con la finalidad de generar ideas novedosas que mejoren la calidad del proyecto. La Tabla 3.24 detalla los resultados obtenidos.

Tesis referente	Análisis	Similitudes
Mobile Application Design for Protecting the Data in Cloud Using Enhanced Technique of Encryption	En este documento los autores desarrollaron una aplicación móvil para cifrado y descifrado de datos utilizando el algoritmo AES como algoritmo principal para cifrado de datos y el algoritmo MD5 para cifrar las claves de cifrado. El procedimiento de cifrado lo efectúa el mismo usuario al generar sus propias claves, las cuales se comparan con las claves almacenadas en un servidor para ejecutar el procedimiento, generando un mecanismo de seguridad robusto [58].	1. Utilización del algoritmo AES 2. Creación de cuenta de usuario 3. Cifrar y descifrar los datos almacenados en el dispositivo
Secure Storage of Data on Android Based Devices	Los investigadores desarrollan un medio de almacenamiento seguro, flexible y fácil de usar para dispositivos Android. Mediante una aplicación encriptan los datos, textos o contraseñas del dispositivo. La aplicación permite a los usuarios realizar un cifrado selectivo, ya que pueden ver, restaurar, encriptar y	1. La clave maestra para descifrar los datos es proporcionada por el usuario 2. Incluye módulo de restauración y selección de

	eliminar datos especificados a través de los módulos o interfaces que dispone la aplicación móvil [59].	archivos para encriptar

Tabla 3.24: Análisis de aplicaciones móviles similares

Elaborado por: Alejandro Mejía

3.2.3.2 Establecer un plan de mejora para la aplicación móvil

Figura 3.26: Desarrollo de Fase de Identificación Proceso II

Elaborado por: Alejandro Mejía

Para el plan de mejora de la aplicación móvil se generan nuevas ideas que aporten innovación en el proyecto, para establecer diferencias claras entre los proyectos similares expuestos con anterioridad. Las ideas son resultado de la opinión de los usuarios encuestados respecto a la seguridad de los dispositivos, y de aportaciones de los involucrados en el desarrollo. La Tabla 3.25 resume los resultados obtenidos.

Ideas	Explicación
Incluir un módulo de activación de cifrado	El usuario podrá activar el mecanismo de cifrado a partir de dos medios: activación por sensores, activación manual. La activación por sensores se trata de cifrar los datos cuando el usuario presione el botón de apagado del dispositivo y realice un patrón o movimiento específico del dispositivo móvil. Los sensores (acelerómetro y giroscopio), reconocen el patrón y ejecutan el mecanismo. Mientras que la activación manual empieza cuando el usuario accede al módulo de activación y presiona el botón “Empezar Cifrado”.
Mecanismos de protección de acceso	Para evitar que otros usuarios ingresen a la aplicación y puedan observar el contenido de los datos y el proceso interno. El propietario del dispositivo puede activar un mecanismo de seguridad, esto quiere decir que, puede usar la huella digital,

	patrón o una contraseña para que cuando se ejecute, la aplicación requiera el mecanismo especificado.
Cifrado de aplicaciones de redes sociales	El usuario puede especificar si necesita que en el proceso de cifrado se incluyan redes sociales. La aplicación eliminara los datos de almacenamiento o caché de las principales aplicaciones: Facebook, Instagram, WhatsApp. Con el objetivo, que cuando dichas aplicaciones se ejecuten, usuarios mal intencionados no puedan acceder a su contenido.
Módulo de Respaldo	En este módulo el usuario puede generar un respaldo de archivos encriptados, para que pueda visualizar su contenido en otros dispositivos móviles que dispongan de la aplicación móvil.
Módulo de seguridad	Si el usuario olvida la contraseña master, puede restaurarla a través de los siguientes métodos de seguridad como la Huella Digital.

Tabla 3.25: Plan de mejora

Elaborado por: Alejandro Mejía

3.2.3.3 Estimación de tiempo requerido para desarrollar la aplicación móvil

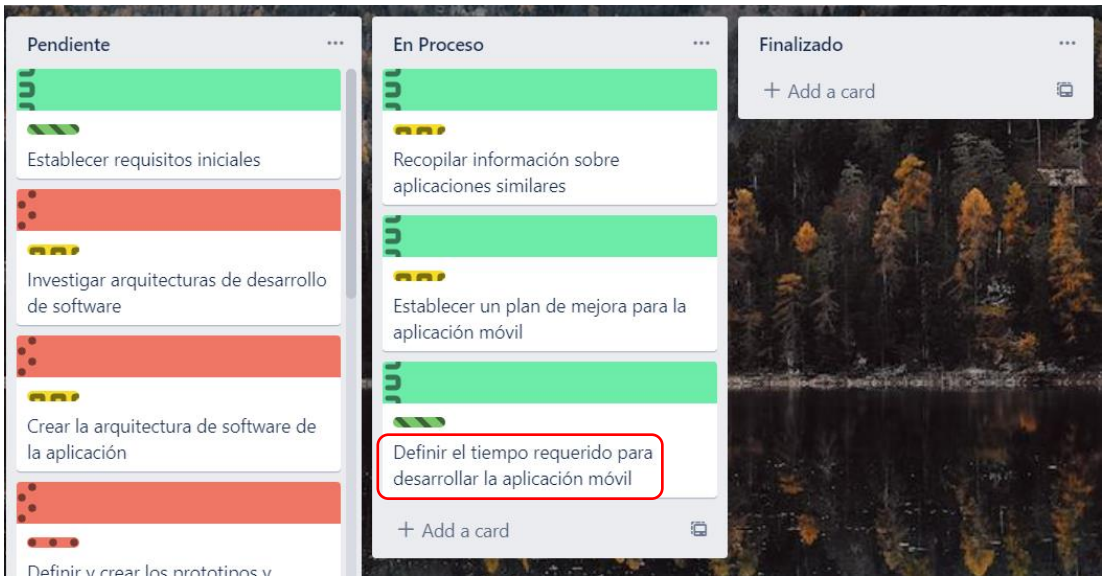


Figura 3.27: Desarrollo de Fase de Identificación Proceso III

Elaborado por: Alejandro Mejía

Para estimar el tiempo de desarrollo de la aplicación, se utilizó la herramienta Project para establecer la duración de las actividades a realizarse en cada fase del proceso. La Figura 3.28 resume el tiempo individual de cada fase y el tiempo estimado total.

	Modo de	Nombre de tarea	Duración	Comienzo	Fin
		Desarrollo de la Propuesta	**76,5 días?**	**jue 3/11/22**	**mié 8/2/23**
		Inicio	0 días	jue 3/11/22	jue 3/11/22
		Fase de Identificación	**5 días?**	**jue 3/11/22**	**jue 10/11/22**
		Fase de Diseño	**8 días?**	**jue 10/11/22**	**lun 21/11/22**
		Fase de Desarrollo	**28 días**	**lun 12/12/22**	**vie 13/1/23**
		Fase de Prototipado	**11 días**	**lun 16/1/23**	**vie 27/1/23**
		Fase de Prueba	**6 días**	**lun 30/1/23**	**lun 6/2/23**
		Fin	0 días	mié 8/2/23	mié 8/2/23

Figura 3.28: Tiempo estimado para desarrollar la aplicación móvil

Elaborado por: Alejandro Mejía

3.2.3.4 Establecimiento de requisitos iniciales

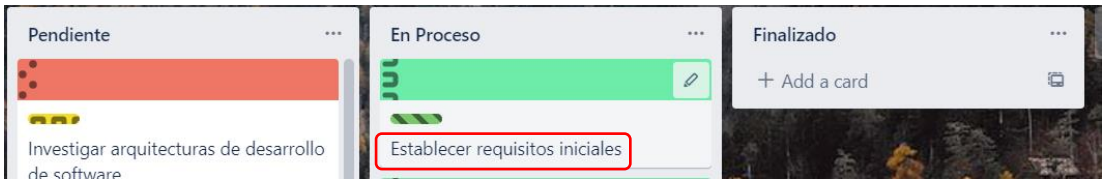


Figura 3.29: Desarrollo de Fase de Identificación Proceso IV

Elaborado por: Alejandro Mejía

Para el desarrollo de la aplicación móvil se requirió de las siguientes herramientas y componentes, detalladas a continuación:

- Entorno de Desarrollo Integrado: Android Studio
- Lenguaje de Programación: Kotlin
- Sensores: Acelerómetro, Sensor de orientación (integrados en el dispositivo móvil)
- Dispositivo móvil: POCO M4 PRO

3.2.4 Fase de Diseño

3.2.4.1 Arquitectura de software de la aplicación

Figura 3.30: Desarrollo de Fase de Diseño Proceso I

Elaborado por: Alejandro Mejía

Para la arquitectura del proyecto se aplicó la arquitectura MVVM (Modelo Vista Vista del Modelo), que al combinarse con Kotlin, ayuda a optimizar el código de las aplicaciones y mantiene la vista de los datos actualizada. En la arquitectura propuesta, el usuario realiza un patrón específico para activar el mecanismo de seguridad. La capa "Vista del Modelo" se encarga de solicitar los datos (fotos, videos, documentos) a la capa de "Modelo" para efectuar las operaciones (cifrar los datos) de lectura/escritura y retornar la información al usuario en la capa "Vista". La capa vista y modelo no tiene relación por lo cual, la capa vista del modelo es el enlace de comunicación entre ellas.

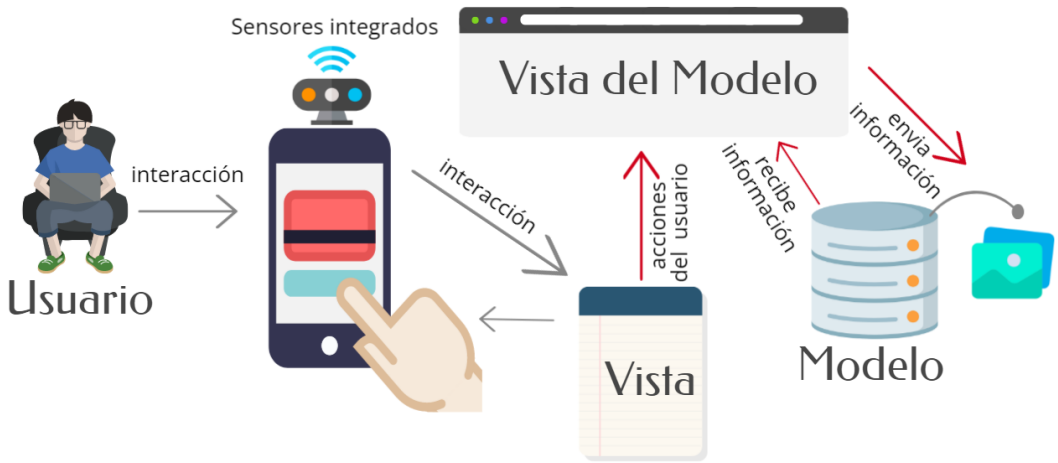


Figura 3.31: Arquitectura de la aplicación

Elaborado por: Alejandro Mejía

3.2.4.2 Prototipos y módulos de la aplicación

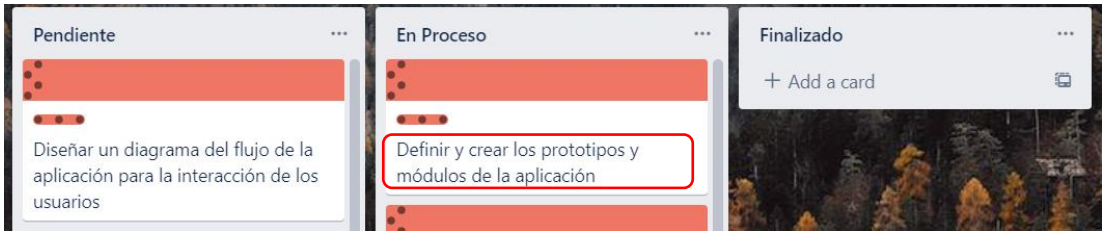


Figura 3.32: Desarrollo de Fase de Diseño Proceso I

Elaborado por: Alejandro Mejía

Para el proyecto, se utilizó la herramienta Balsamiq Wireframes para diseñar el prototipo inicial de la aplicación, con el objetivo de brindar al usuario una perspectiva general del funcionamiento operativo de los módulos e interfaces.

A. Diseño del Módulo de Bienvenida

El módulo de bienvenida contiene una interfaz de inicio con el nombre de la aplicación y un "spinner" de carga. Además, una vez culmine el ciclo de carga de la pantalla anterior se genera una transición para mostrar otra interfaz con un mensaje de bienvenida, con información general respecto al uso de la aplicación.

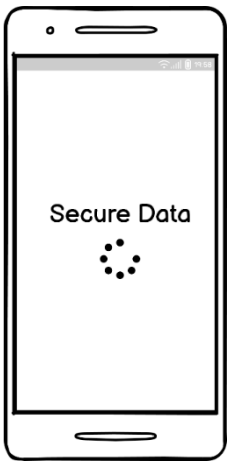


Figura 3.33: Prototipo Pantalla de Carga

Elaborado por: Alejandro Mejía

Figura 3.34: Prototipo Pantalla de Bienvenida

Elaborado por: Alejandro Mejía

B. Diseño del Módulo de Ingreso y Registro de Usuario

Para el módulo de ingreso y registro como requisito inicial el usuario debe crear una cuenta local para que sea únicamente el propietario del dispositivo quien tenga acceso a las funciones e información de la aplicación. La Figura 3.35 ejemplifica la pantalla presentada al usuario.

Figura 3.35: Prototipo Pantalla de Ingreso

Elaborado por: Alejandro Mejía

A continuación, el usuario ingresa los datos requeridos y señala la opción "guardar inicio de sesión" (opcional), para que al abrir nuevamente la aplicación no se presente la pantalla de inicio de sesión. La Figura 3.36 resume lo señalado. Una vez realizado

el procedimiento el usuario tiene acceso a los distintos módulos de la aplicación. En caso de que el usuario disponga de una cuenta local creada solo debe ingresar su nombre de usuario y contraseña.

Figura 3.36: Prototipo Pantalla de Registro de Usuario Local

Elaborado por: Alejandro Mejía

C. Módulo Principal

En el módulo principal, el usuario observará un cuadro de texto que indica al usuario añadir un mecanismo de seguridad adicional. El mecanismo le permitirá al usuario recuperar su cuenta local en caso de olvidar la contraseña de inicio. La Figura 3.37 resume el mensaje de aviso para el usuario.

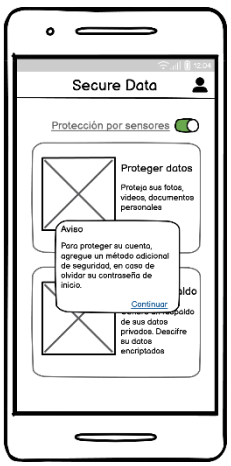


Figura 3.37: Prototipo Pantalla Principal de la aplicación I

Elaborado por: Alejandro Mejía

Una vez que, el usuario añadió el mecanismo de seguridad de su preferencia el usuario podrá visualizar la pantalla principal con información de los módulos encargados de los procesos para proteger los datos privados. Además, en la parte superior derecha el usuario podrá activar/desactivar un switch para que el mecanismo de seguridad de la aplicación (encriptación de los datos) entre en funcionamiento con los sensores integrados del dispositivo (acelerómetro, sensor de orientación).

Figura 3.38: Prototipo Pantalla Principal de la aplicación II

Elaborado por: Alejandro Mejía

D. Módulo de Recuperación y restauración de Contraseña

La pantalla de este módulo mostrará la interfaz ejemplificada en la Figura 3.39. El usuario deberá seleccionar el mecanismo de seguridad de respaldo que eligió durante el proceso de creación de su cuenta local.

Figura 3.39: Prototipo Pantalla de Recuperación de Contraseña

Elaborado por: Alejandro Mejía

La Figura 3.40 muestra la interfaz que el usuario observará luego de confirmar correctamente el mecanismo de seguridad de respaldo. A continuación, debe ingresar una nueva contraseña para culminar con el proceso de actualización o restauración.

Figura 3.40: Prototipo Pantalla de Actualización de Contraseña

Elaborado por: Alejandro Mejía

E. Módulo de Mecanismos de Seguridad

Este módulo se conecta directamente con el Módulo Principal. El módulo de mecanismos de seguridad mostrará al usuario tres alternativas: Huella digital, PIN, patrón. En la Figura 3.41 se resume la interfaz que el usuario observará.

Figura 3.41: Prototipo Pantalla de Menú de Mecanismos de Seguridad

Elaborado por: Alejandro Mejía

La Figura 3.42 demuestra el diseño de la interfaz que se mostrará si el usuario optó por la seguridad de la Huella digital.

Figura 3.42: Prototipo Pantalla de Desbloqueo por Huella Digital I

Elaborado por: Alejandro Mejía

Al presionar el botón Empezar la aplicación requerirá al usuario colocar su huella hasta completar con el procedimiento.

Figura 3.43: Prototipo Pantalla de Desbloqueo por Huella Digital II

Elaborado por: Alejandro Mejía

Una vez que el proceso culmine exitosamente el usuario visualizará el siguiente contenido resumido en la Figura 3.44.

Figura 3.44: Prototipo Pantalla de Desbloqueo por Huella Digital III

Elaborado por: Alejandro Mejía

La Figura 3.45 demuestra el diseño de la interfaz que se mostrará al usuario en caso de elegir el mecanismo de seguridad de respaldo por patrón.

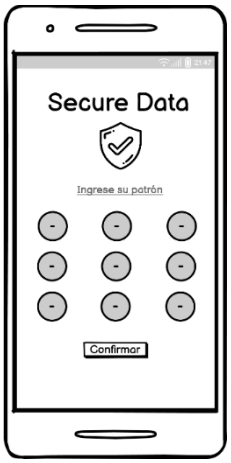


Figura 3.45: Prototipo Pantalla de Desbloqueo por patrón I

Elaborado por: Alejandro Mejía

El procedimiento requerirá al usuario volver a ingresar el patrón elegido para confirmar que sea el correcto. En el caso de que la aplicación detecte que los patrones ingresados no coinciden mostrará al usuario un mensaje de aviso, indicando que debe ingresarlo nuevamente.

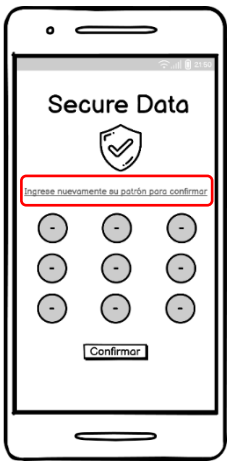


Figura 3.46: Prototipo Pantalla de Desbloqueo por patrón II

Elaborado por: Alejandro Mejía

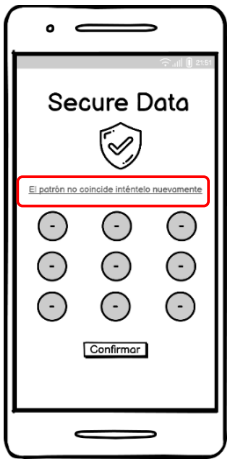


Figura 3.47: Prototipo Pantalla de Desbloqueo por patrón III

Elaborado por: Alejandro Mejía

La Figura 3.48 demuestra la interfaz que se mostrará al usuario en caso de elegir el mecanismo de seguridad de respaldo por PIN.

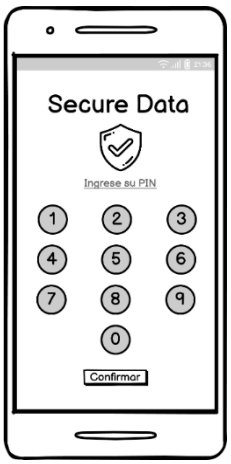


Figura 3.48: Prototipo Pantalla de Desbloqueo por PIN I

Elaborado por: Alejandro Mejía

El procedimiento requerirá al usuario volver a ingresar el PIN ingresado para culminar. En el caso de que la aplicación detecte que el PIN ingresado no coinciden mostrará al usuario un mensaje de aviso, indicando que debe ingresarlo nuevamente.

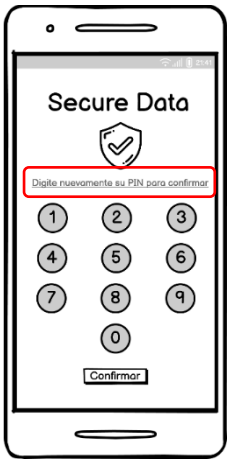


Figura 3.49: Prototipo Pantalla de Desbloqueo por PIN II

Elaborado por: Alejandro Mejía

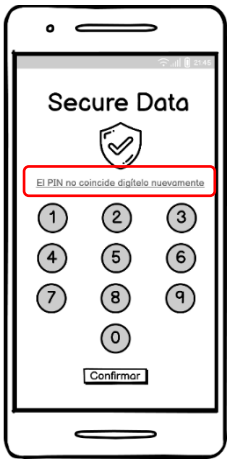


Figura 3.50: Prototipo Pantalla de Desbloqueo por PIN III

Elaborado por: Alejandro Mejía

F. Módulo de Cifrado de Datos

El Módulo Principal está vinculado con el Módulo de Cifrado de datos, en el momento que el usuario presione en la sección Respaldo de datos la aplicación mostrará una pantalla con el contenido señalado en la Figura 3.51. El usuario podrá filtrar el contenido de los datos que desea visualizar, por ejemplo: fotos, videos y documentos mediante un panel ubicado en la parte superior de la aplicación. Además, podrá ordenar y actualizar el contenido filtrado.

Figura 3.51: Prototipo Pantalla Módulo de Cifrado

Elaborado por: Alejandro Mejía

La Figura 3.52 ejemplifica la interacción del usuario con la aplicación, al señalar los datos que desea proteger. Inmediatamente después, deberá presionar el botón OK para comenzar con el proceso de encriptación de los datos.

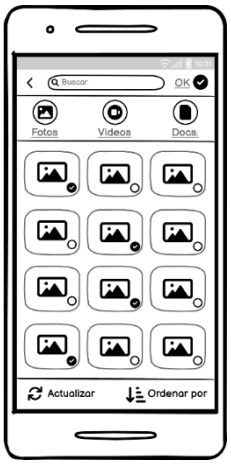


Figura 3.52: Prototipo Pantalla Módulo de Cifrado Selección de Archivos

Elaborado por: Alejandro Mejía

Si el proceso se ha llevado a cabo sin inconvenientes, la aplicación mostrará al usuario un mensaje de confirmación para informar al usuario que los datos seleccionados se encuentran protegidos. La Figura 3.53 resume el contenido del mensaje.

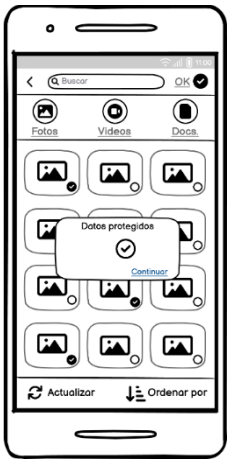


Figura 3.53: Prototipo Pantalla Módulo de Cifrado Mensaje de Confirmación

Elaborado por: Alejandro Mejía

G. Módulo de Respaldo y restauración de datos

El módulo de respaldo y restauración de datos permitirá al usuario descifrar los datos que han sido cifrados en el Módulo de Cifrado. Se visualizará una pantalla en la cual los datos no son visibles como lo muestra la Figura 3.54, ya que se encuentran cifrados. El usuario seleccionará los datos que necesite descifrar o respaldar (opcional), es decir crear un archivo de respaldo con ciertos datos encriptados para visualizarlo en otros dispositivos con la misma aplicación móvil.

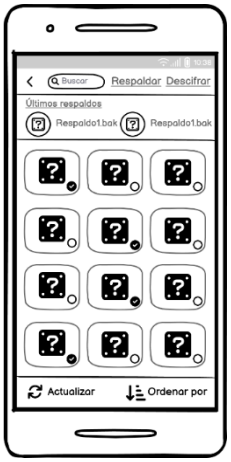


Figura 3.54: Prototipo Pantalla Módulo de Respaldo y restauración

Elaborado por: Alejandro Mejía

La Figura 3.55 demuestra el diseño de la interfaz que se mostrará al usuario en caso de respaldar los datos seleccionados. Mientras que, la Figura 3.55 demuestra el diseño de la interfaz en caso de descifrar los datos.

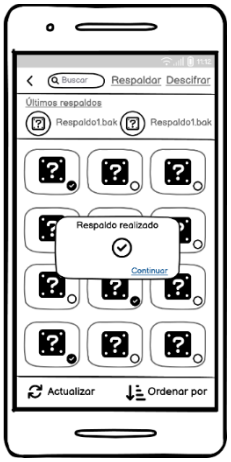


Figura 3.55: Prototipo Pantalla de confirmación respaldo realizado

Elaborado por: Alejandro Mejía

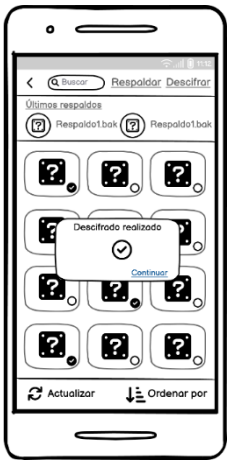


Figura 3.56: Prototipo Pantalla de confirmación descifrado realizado

Elaborado por: Alejandro Mejía

Además, si el usuario desea visualizar la lista de los respaldos efectuados deberá ingresar en la opción Últimos respaldos, como lo demuestra la Figura 3.57 y la Figura 3.58

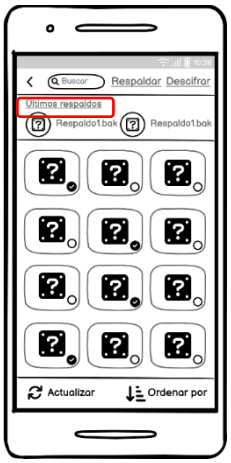


Figura 3.57: Prototipo Pantalla de selección de respaldos

Elaborado por: Alejandro Mejía

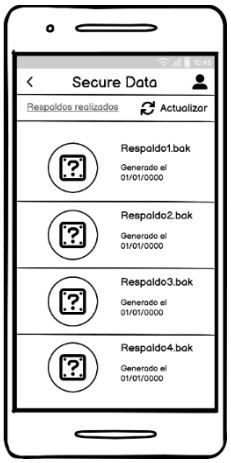


Figura 3.58: Prototipo Pantalla de respaldos efectuados

Elaborado por: Alejandro Mejía

H. Módulo de Usuario

En este módulo el usuario podrá visualizar su información personal, cambiar su contraseña de inicio y cerrar la sesión local. La sección de cambio de contraseña se conectará con el Módulo de Recuperación y restauración de Contraseña para realizar el procedimiento respectivo. La Figura 3.59 demuestra el diseño de la interfaz establecida para este módulo.

Figura 3.59: Prototipo Pantalla de Usuario

Elaborado por: Alejandro Mejía

I. Cuadro resumen de los Módulos de la Aplicación

Módulo	Explicación
Módulo de Inicio	Contiene las interfaces de apertura al ejecutarse la aplicación
Módulo de Ingreso y Registro de Usuario	Registra los datos del usuario y la contraseña máster que servirá para ejecutar el proceso de cifrado. Aquí además se solicita la selección de uno de los mecanismos de seguridad de respaldo disponibles en caso de perder la contraseña. La contraseña máster sirve para descifrar los datos.
Módulo Principal	Permite la interacción con los distintos módulos de la aplicación (Módulo de Cifrado de Datos – Módulo de Respaldo y Restauración de datos)
Módulo de Respaldo y restauración de datos	Contiene las copias de seguridad (archivo que contiene datos cifrados) realizados por el usuario y la lista de los datos encriptados por el usuario.
Módulo de Cifrado de Datos	El usuario puede empezar con el proceso de cifrado eligiendo los datos (fotos, videos, documentos) que desea cifrar y eliminar (en caso de redes sociales)
Módulo de Recuperación y Restauración de Contraseña	Recuperar la contraseña máster. Si el usuario pierde su contraseña, la aplicación requiere uno de los mecanismos seleccionados en la etapa de creación de usuario.
Módulo de mecanismos de seguridad	El usuario selecciona el mecanismo de seguridad que desee usar, cuando la aplicación se ejecute. Los mecanismos son: huella digital, PIN o un patrón.
Módulo de Usuario	Contiene información personal del usuario y opciones adicionales de gestión de la cuenta.

Tabla 3.26: Módulos de la aplicación

Elaborado por: Alejandro Mejía

3.2.5 Fase de Desarrollo

3.2.5.1 Configuración de Android Studio

Figura 3.60: Desarrollo de Fase de Desarrollo Proceso I

Elaborado por: Alejandro Mejía

A. Instalación de Android Studio

Desde la página oficial de Android se descargó e instaló el (IDE) Android Studio

Figura 3.61: Instalación de Android Studio I

Elaborado por: Alejandro Mejía

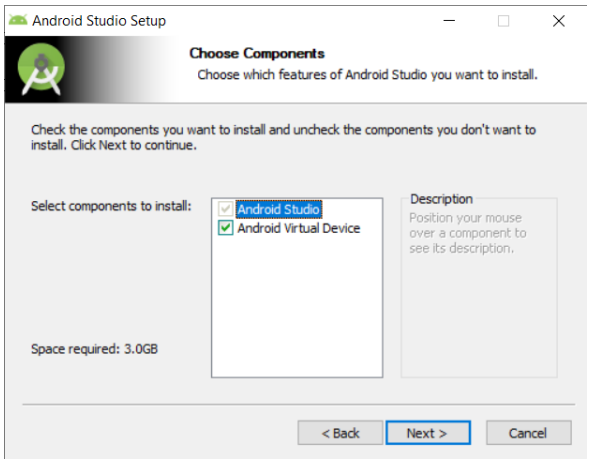


Figura 3.62: Instalación de Android Studio II

Elaborado por: Alejandro Mejía

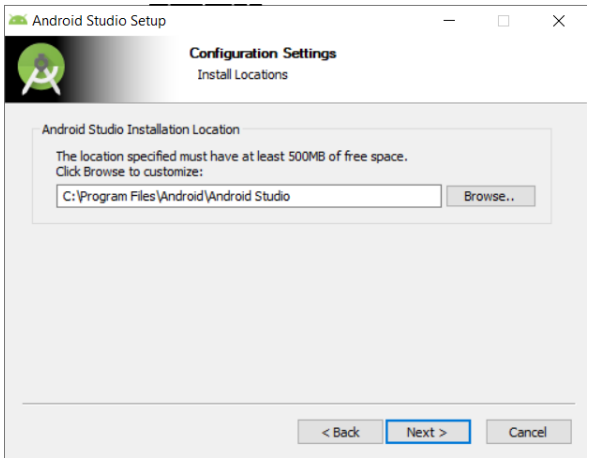


Figura 3.63: Instalación de Android Studio III

Elaborado por: Alejandro Mejía

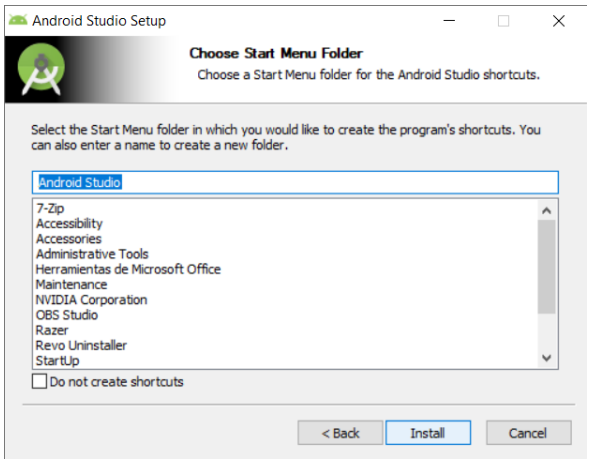


Figura 3.64: Instalación de Android Studio IV

Elaborado por: Alejandro Mejía

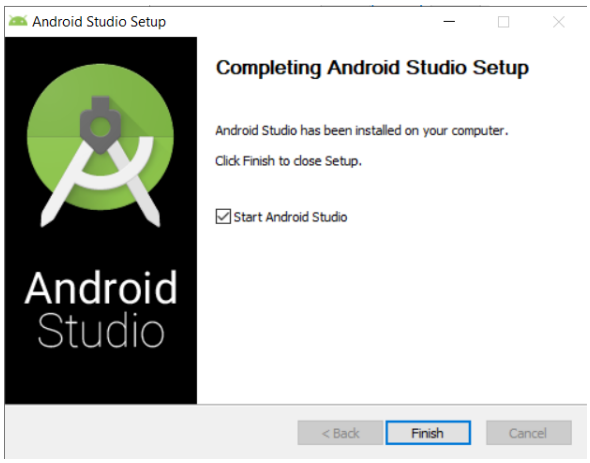


Figura 3.65: Instalación de Android Studio V

Elaborado por: Alejandro Mejía

B. Configuración del Entorno de Desarrollo

Se descargaron e instalaron los requisitos funcionales a través de una configuración estándar proporcionada por Android Studio:

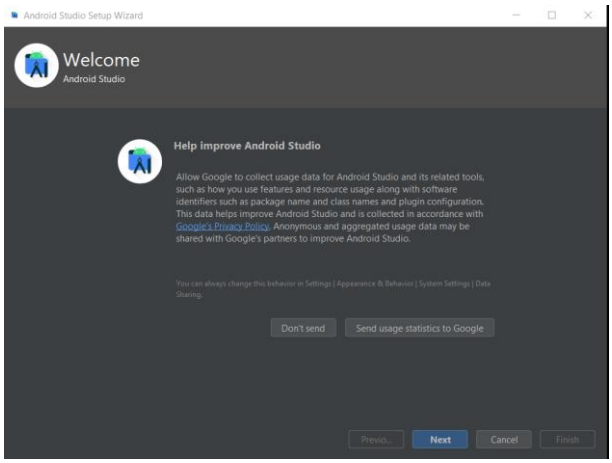


Figura 3.66: Configuración de Entorno de Desarrollo I

Elaborado por: Alejandro Mejía

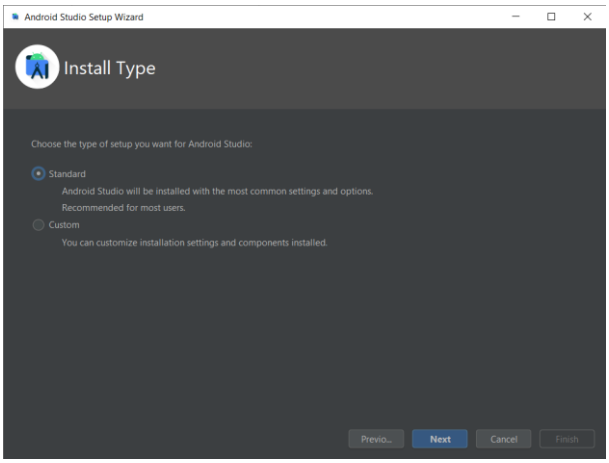


Figura 3.67: Configuración de Entorno de Desarrollo II

Elaborado por: Alejandro Mejía

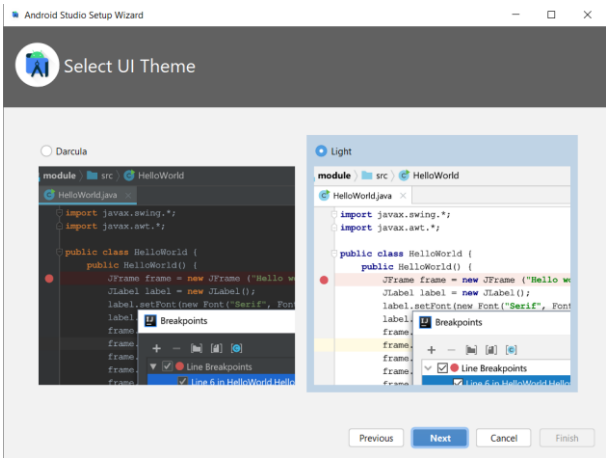


Figura 3.68: Configuración de Entorno de Desarrollo III

Elaborado por: Alejandro Mejía

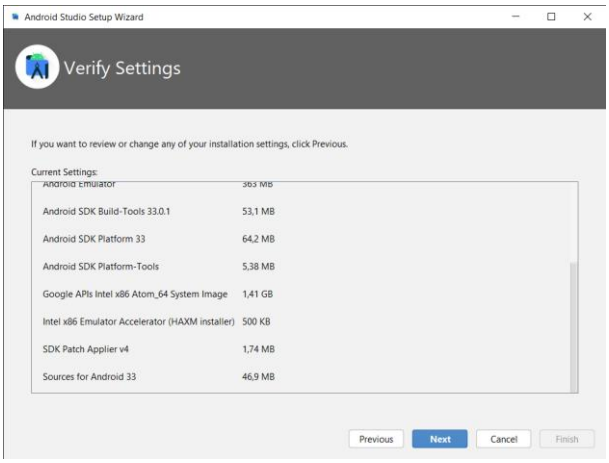


Figura 3.69: Configuración de Entorno de Desarrollo IV

Elaborado por: Alejandro Mejía

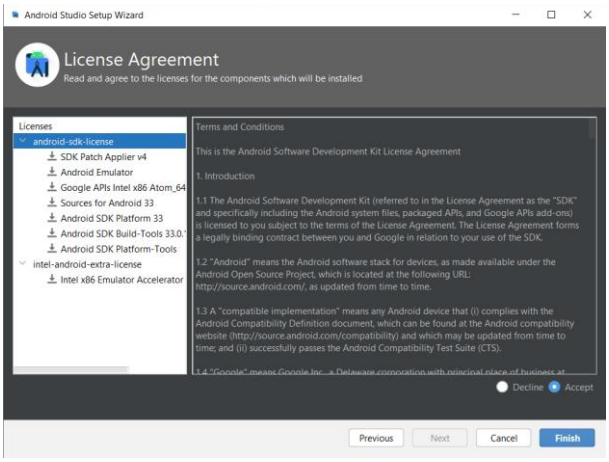


Figura 3.70: Configuración de Entorno de Desarrollo V

Elaborado por: Alejandro Mejía

Luego, se añadió los componentes necesarios para ejecutar aplicaciones en diferentes versiones Android como: Oreo, Pie, R, Q o S. Además, se incorporó varias herramientas para el funcionamiento del emulador de dispositivos. La Figura 3.73 resume algunos de los componentes añadidos para ser instalados.

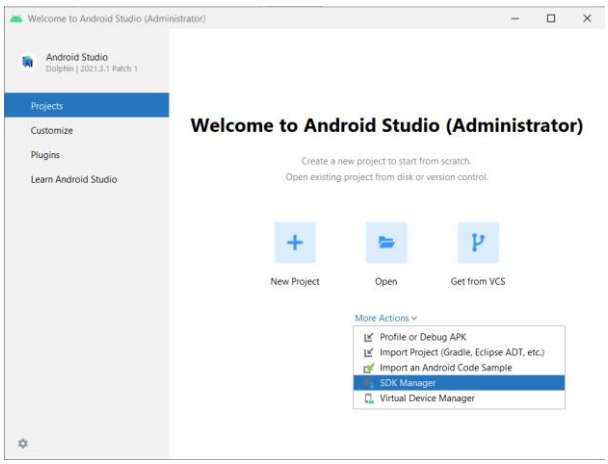


Figura 3.71: Instalación de paquetes Android I

Elaborado por: Alejandro Mejía

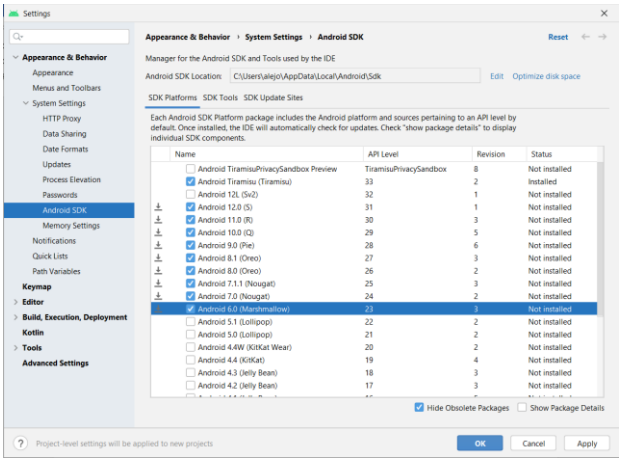


Figura 3.72: Instalación de paquetes Android II

Elaborado por: Alejandro Mejía

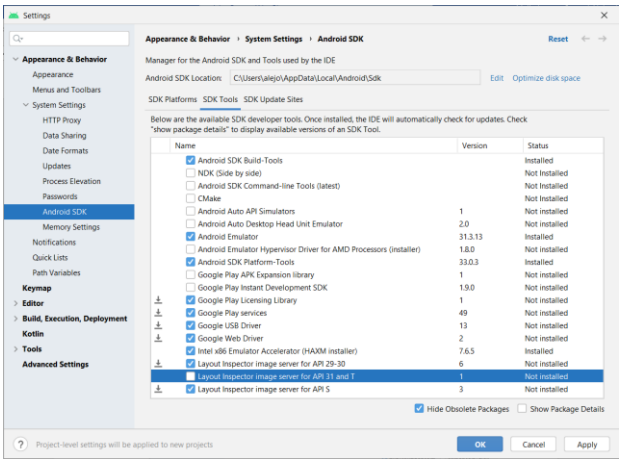


Figura 3.73: Instalación de paquetes Android III

Elaborado por: Alejandro Mejía

C. Configuración del Emulador de Dispositivos

Se configuró un nuevo emulador de dispositivos, para lo cual se tomó en consideración el tipo de dispositivo, configuraciones de memoria, configuraciones de espacio en disco y la versión de Android. Se seleccionó la versión R o Android 11, al tratarse de una versión estándar. Es necesario tomar en cuenta la versión de Android seleccionada durante esta etapa ya que las dependencias utilizadas serán compatibles únicamente con esta versión, por lo que si se utilizan plugin en desuso Android no podrá instalar o ejecutar aplicaciones.

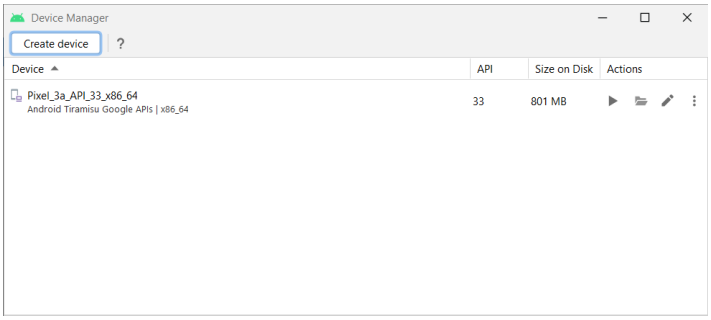


Figura 3.74: Configuración de Emulador de Dispositivos I

Elaborado por: Alejandro Mejía

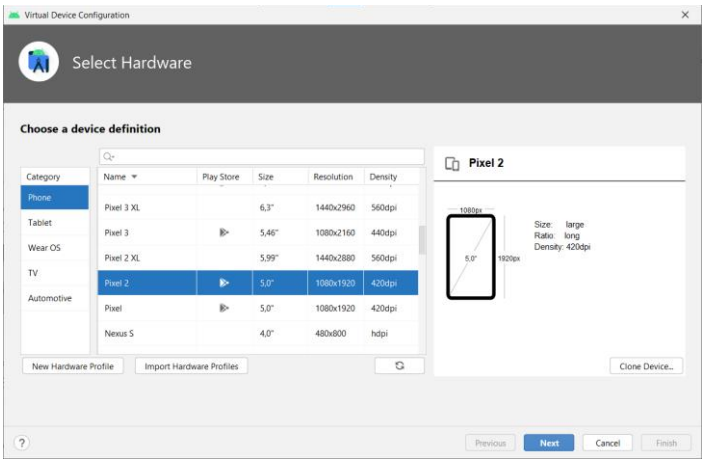


Figura 3.75: Configuración de Emulador de Dispositivos II

Elaborado por: Alejandro Mejía

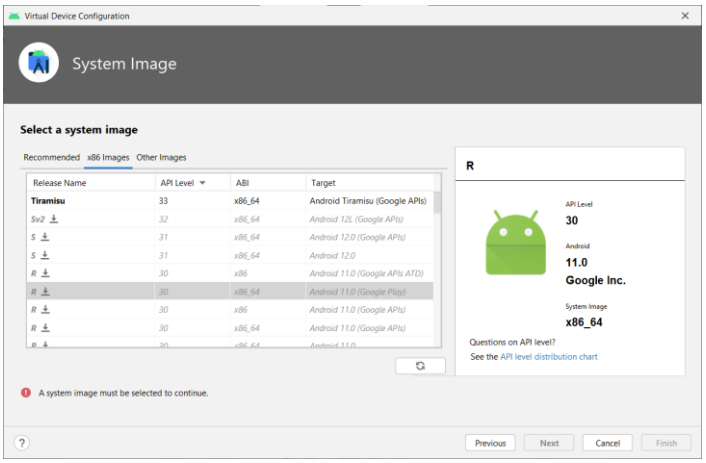


Figura 3.76: Configuración de Emulador de Dispositivos III

Elaborado por: Alejandro Mejía

Figura 3.77: Configuración del Dispositivo Virtual Android

Elaborado por: Alejandro Mejía

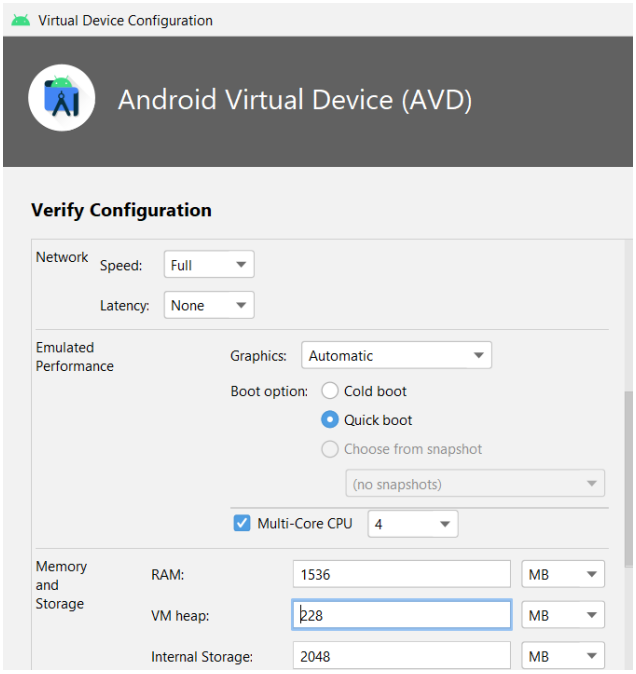


Figura 3.78: Configuración de Memoria y Almacenamiento

Elaborado por: Alejandro Mejía

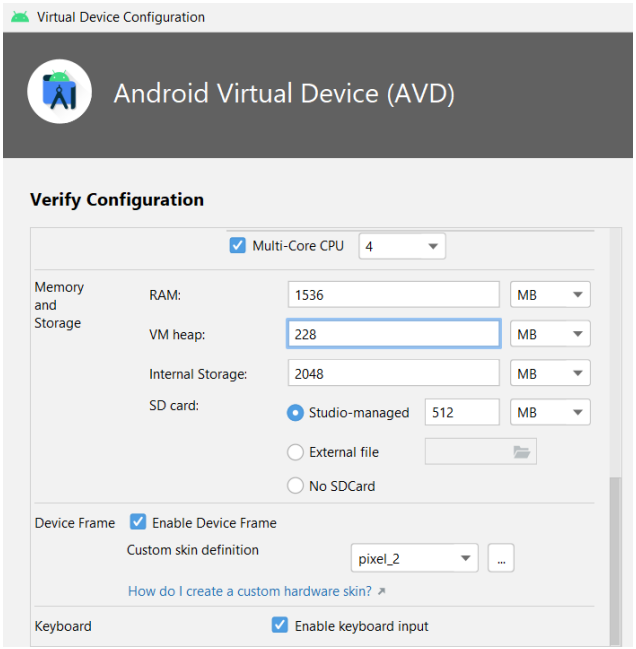


Figura 3.79: Configuración de Memoria y Almacenamiento II

Elaborado por: Alejandro Mejía

Figura 3.80: Verificación de Instalación de Dispositivo Virtual I

Elaborado por: Alejandro Mejía

D. Creación del Proyecto

Android Studio proporciona plantillas que facilitan el proceso de desarrollo de aplicaciones a usuarios novatos, no obstante, para el presente proyecto se seleccionó una estructura vacía.

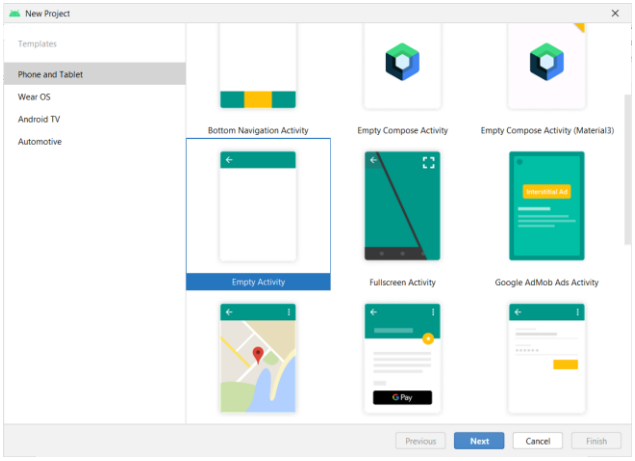


Figura 3.81: Selección de esquema para el proyecto

Elaborado por: Alejandro Mejía

Se realizó la configuración inicial del proyecto, designando el lenguaje de programación Kotlin y la versión mínima de SDK. Minium SDK define la versión mínima aceptada por Android para ejecutar las aplicaciones, por lo cual es fundamental seleccionar una versión estable para la mayor parte de dispositivos móviles.

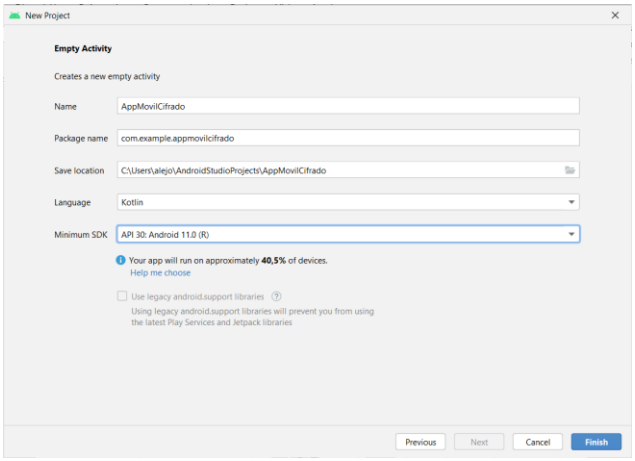


Figura 3.82: Configuración general del proyecto

Elaborado por: Alejandro Mejía

E. Esquema de estructura del proyecto

El diseño de la estructura se realizó mediante la metodología MVVM que define:

- Dependencias: Contiene módulos encargados de proporcionar un contenedor para el: código generado, configuraciones, preferencias o servicios.
- Modelo: Contiene las referencias a los datos requeridos, entre los que se pueden considerar: base de datos, preferencias compartidas o contenido almacenado localmente.
- Modelo - Vista: Se encarga de mantener la comunicación entre el Modelo y la Vista, por lo que no dispone de una interfaz gráfica para su control. Es el encargado de efectuar las operaciones de lectura/escritura.
- Vista: Almacena por lo general actividades, fragmentos y adaptadores. En esta etapa los datos presentados en tiempo real no se pueden alterar.

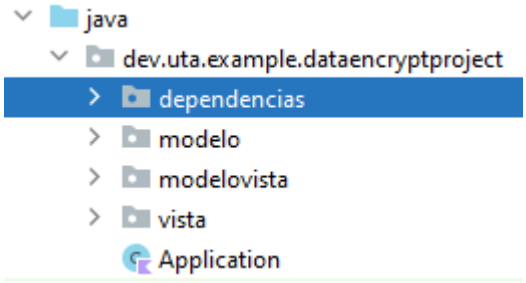


Figura 3.83: Creación de estructura del proyecto basado en la metodología (MVVM)

Elaborado por: Alejandro Mejía

3.2.5.2 Implementación de Entidades

Se realizaron las distintas clases para almacenar la información recolectada por las clases e interfaces.

A. Esquema Clase Usuario

La clase Usuarios contiene los datos personales, los cuales validan el Ingreso al Sistema mediante el nombre de usuario y la contraseña, mientras que para realizar operaciones relacionados con cifrado de datos se utiliza la clave de encriptación. El atributo pin es un valor que permite acceder a la etapa de registro de usuario, para recuperar el inicio de sesión o actualizar la información.

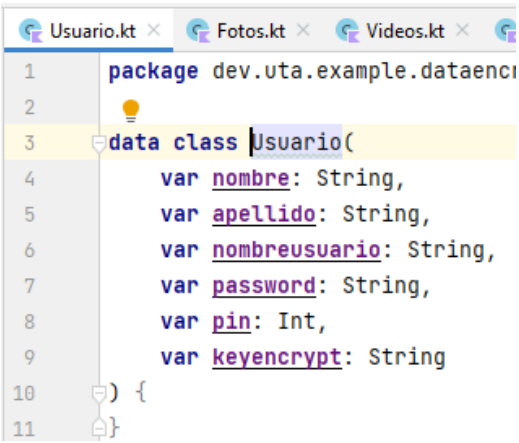


Figura 3.84: Entidad Usuario

Elaborado por: Alejandro Mejía

B. Esquema Clase Fotos, Videos, Documentos

Las clases Fotos, Videos, Documentos comparten la misma estructura de la cual se puede detallar:

- Uri: Valor utilizado por componentes de visualización como Image View
- Seleccionado: Identifica si el usuario selecciona el dato y lo agrega en una lista de espera, caso contrario cambia el valor de la propiedad a false.
- extFig: Almacena el identificador de una imagen incluida dentro del proyecto. Determina si la extensión del archivo pertenece a una foto, video, documento o una imagen encriptada.
- Tam: Indica el tamaño del archivo en bytes, MB, o GB. El valor original pertenece al tipo de Long, por lo que es procesado para visualizarlo en un formato legible.

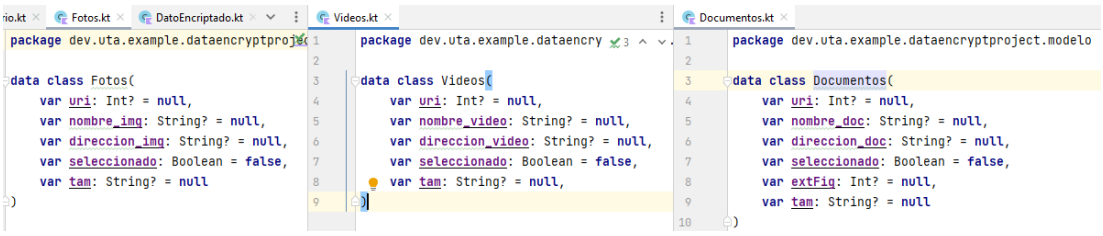


Figura 3.85: Entidades Fotos, Videos, Documentos

Elaborado por: Alejandro Mejía

C. Esquema Datos Encriptados

Se encarga de guardar la información de los archivos encriptados

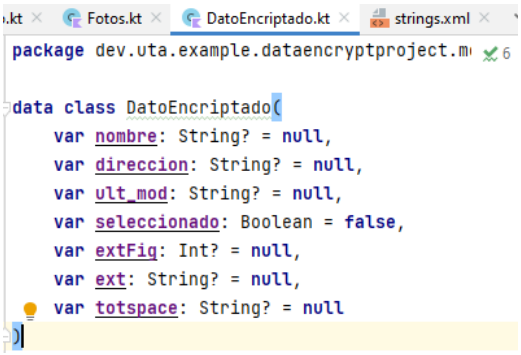


Figura 3.86: Entidad Dato Encriptado

Elaborado por: Alejandro Mejía

3.2.5.3 Preferencias de Usuario

Una vez generado el esquema de cada entidad, se implementó la estructura para almacenar los datos en un archivo de preferencias de la aplicación:

- KEY_USUARIO_DATA_STORE: Es el nombre del archivo al cual se hace referencia para obtener sus datos.
- Context.dataStore: Es la biblioteca encargada de proporcionar los métodos para crear los archivos y parámetros de las preferencias compartidas.

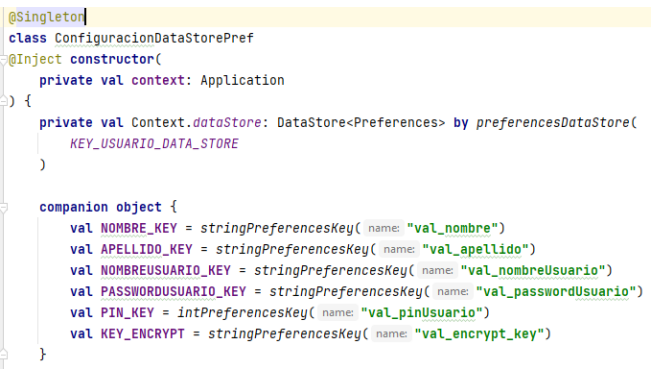


Figura 3.87: Preferencias de Usuario

Elaborado por: Alejandro Mejía

Se creó la clase Application encargada de obtener el contexto del proyecto y utilizarlo en la interacción con las preferencias del usuario.

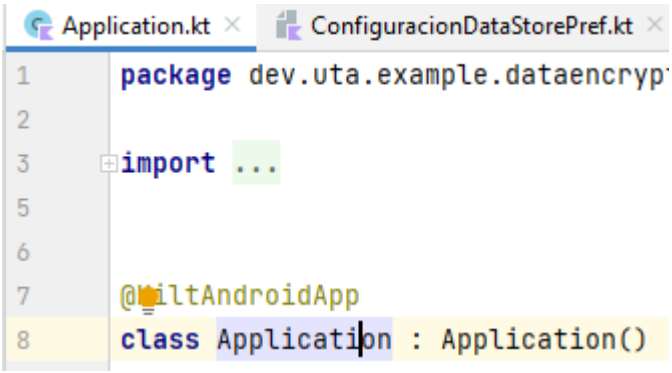


Figura 3.88: Contexto del Proyecto

Elaborado por: Alejandro Mejía

El siguiente método permite crear un nuevo objeto al cual se le asignan los datos propios de la clase Usuarios y se almacenan en el archivo de preferencias:

- Context.dataStore.edit: Cambiar el valor de la columna seleccionado por el dato ingresado por el usuario, en caso de que exista un usuario creado permite sobrescribir los datos.

```
suspend fun crearUsuario(usuario: Usuario) {
    context.dataStore.edit { caracteristicas ->
        caracteristicas[NOMBRE_KEY] = usuario.nombre
        caracteristicas[APELLIDO_KEY] = usuario.apellido
        caracteristicas[NOMBREUSUARIO_KEY] = usuario.nombreusuario
        caracteristicas[PASSWORDUSUARIO_KEY] = usuario.password
        caracteristicas[PIN_KEY] = usuario.pin
        caracteristicas[KEY_ENCRYPT] = usuario.keyencrypt
    }
}
```

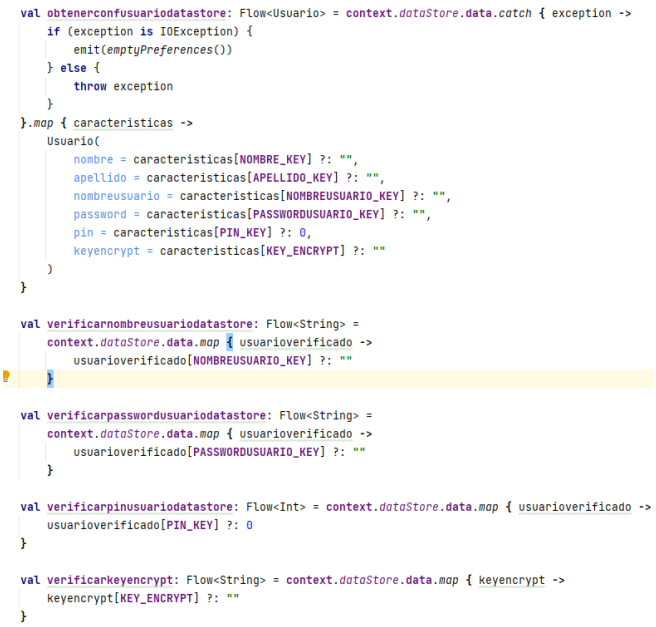
Figura 3.89: Crear usuario por medio de DataStore

Elaborado por: Alejandro Mejía

Para obtener los datos almacenados en la biblioteca Preferences se utilizaron dos métodos que permiten optimizar el consumo de memoria:

- Context.dataStore.data.catch.map: Obtiene el objeto almacenado en el archivo de preferencias luego de verificar su existencia.

- Context.dataStore.data.map: Obtiene el valor de una columna especificada



```
val obtenerconfusuariodatastore: Flow<Usuario> = context.dataStore.data.catch { exception ->
    if (exception is IOException) {
        emit(emptyPreferences())
    } else {
        throw exception
    }
}.map { caracteristicas ->
    Usuario(
        nombre = caracteristicas[NOMBRE_KEY] ?: "",
        apellido = caracteristicas[APELLIDO_KEY] ?: "",
        nombreusuario = caracteristicas[NOMBREUSUARIO_KEY] ?: "",
        password = caracteristicas[PASSWORDUSUARIO_KEY] ?: "",
        pin = caracteristicas[PIN_KEY] ?: 0,
        keyencrypt = caracteristicas[KEY_ENCRYPT] ?: ""
    )
}

val verificarnombreusuariodatastore: Flow<String> =
    context.dataStore.data.map { usuarioverificado ->
        usuarioverificado[NOMBREUSUARIO_KEY] ?: ""
    }

val verificarpasswordusuariodatastore: Flow<String> =
    context.dataStore.data.map { usuarioverificado ->
        usuarioverificado[PASSWORDUSUARIO_KEY] ?: ""
    }

val verificarpinusuariodatastore: Flow<Int> = context.dataStore.data.map { usuarioverificado ->
    usuarioverificado[PIN_KEY] ?: 0
}

val verificarkeyencrypt: Flow<String> = context.dataStore.data.map { keyencrypt ->
    keyencrypt[KEY_ENCRYPT] ?: ""
}
```

Figura 3.90: Métodos para obtener preferencias de usuario

Elaborado por: Alejandro Mejía

3.2.5.4 Preferencias de Inicio

Se implementó la Clase DataStoreSession destinada a controlar la sesión del usuario, si el valor obtenido es false la aplicación inicia en la sección de Login, mientras que cuando es valor es true, accede directamente al Menú Principal.

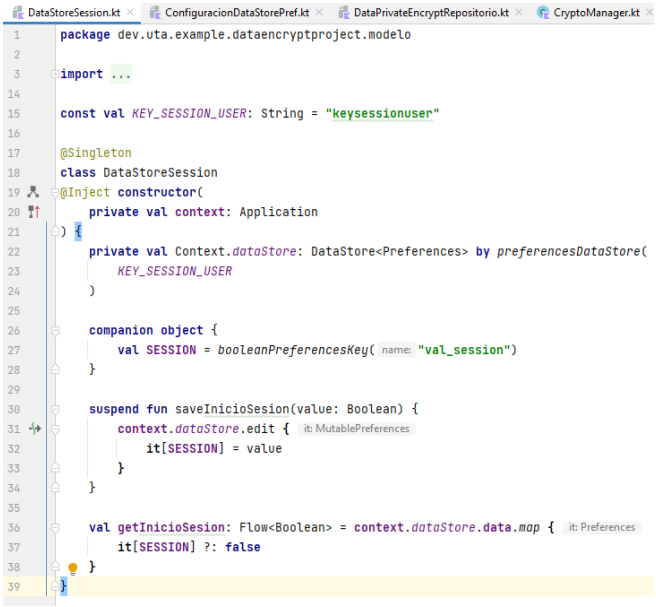


Figura 3.91: Preferencias de inicio de sesión

Elaborado por: Alejandro Mejía

3.2.5.5 Datos del Dispositivo

Se implementaron los siguientes métodos encargados de obtener los archivos almacenadas en el dispositivo, a través de ContentResolver y MediaStore, el método interactúa con el contenido multimedia, para especificar los parámetros de búsqueda de un volumen de almacenamiento específico y almacenarlo en una lista temporal. Los parámetros se explican a continuación:

- MediaStore.MediaColumns._ID: Uri del archivo
- MediaStore.MediaColumns.DATA: Ruta de acceso local
- MediaStore.MediaColums.DISPLAY_NAME: Nombre original
- MediaStore.MediaColumns.SIZE: Tamaño del archivo en formato .Long
- MediaStore.MediaColumns.DATE_ADDED: Última fecha de modificación
- Cursor: Encargado de recorrer la lista de archivos encontrado mientras sea posible.
- getColumnIndexorThrow(): Obtiene la posición de un determinado valor para almacenarlo en la lista temporal
- getString(): Mediante una posición determinada obtiene su contenido

- MediaStore.Media.EXTERNAL_CONTENT_URI: Ruta en la cual buscar archivos

Para esta etapa se verificó que todos los mecanismos de búsqueda o modificación de archivos finalicen, caso contrario el consumo de memoria del dispositivo aumentará notablemente, ya que el dispositivo intentará realizar tareas incompletas en segundo plano hasta que sean completadas, incrementando el consumo de memoria.

A. Fotos Dispositivo

```
fun obtenerImagenesDispositivo(context: Context): ArrayList<Fotos>? {
    val foto = ArrayList<Fotos>()
    val retCol = arrayOf(
        MediaStore.Images.Media._ID,
        MediaStore.Images.Media.DISPLAY_NAME,
        MediaStore.Images.Media.DATA,
        MediaStore.Images.Media.SIZE
    )
    val cur: Cursor? = context.contentResolver.query(
        MediaStore.Images.Media.EXTERNAL_CONTENT_URI,
        retCol,
         selection: null,
         selectionArgs: null,
         sortOrder: "${MediaStore.Images.Media.DATE_ADDED} DESC"
    )
    if (cur != null) {
        while (cur.moveToNext()) {
            if (!cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.DISPLAY_NAME))
                    .contains( other: "crypt_")
            ) {
                val id = cur.getInt(cur.getColumnIndexOrThrow(MediaStore.MediaColumns._ID))
                val nombre =
                    cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.DISPLAY_NAME))
                val direccion =
                    cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.DATA))
                        .replace(nombre,  newValue: "")
                val tam = cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.SIZE))
                foto.add(
                    Fotos(
                        uri = id,
                        nombre_img = nombre,
                        direccion_img = direccion,
                        seleccionado = false,
                        tam = convertirTamFileP(tam.toLong())
                    )
                )
            }
        }
        cur.close()
    }
    return foto
}
```

Figura 3.92: Método de acceso a fotos del dispositivo

Elaborado por: Alejandro Mejía

B. Videos Dispositivo

El siguiente método se encarga de obtener la lista de videos disponibles en el dispositivo.

```
fun obtenerVideosDispositivo(context: Context): ArrayList<Videos>? {
    val video = ArrayList<Videos>()
    val retCol = arrayOf(
        MediaStore.Video.Media._ID,
        MediaStore.Video.Media.DISPLAY_NAME,
        MediaStore.Video.Media.DATA,
        MediaStore.Video.Media.SIZE
    )
    val cur: Cursor? = context.contentResolver.query(
        MediaStore.Video.Media.EXTERNAL_CONTENT_URI,
        retCol,
        selection: null,
        selectionArgs: null,
        sortOrder: "${MediaStore.Video.Media.DATE_ADDED} DESC"
    )
    if (cur != null) {
        while (cur.moveToNext()) {
            if (!cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.DISPLAY_NAME))
                    .contains( other: "crypt_")
            ) {
                val id = cur.getInt(cur.getColumnIndexOrThrow(MediaStore.MediaColumns._ID))
                val nombre =
                    cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.DISPLAY_NAME))
                val direccion =
                    cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.DATA))
                        .replace(nombre, newValue: "")
                val tam = cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.SIZE))
                video.add(
                    Videos(
                        uri = id,
                        nombre_video = nombre,
                        direccion_video = direccion,
                        seleccionado = false,
                        tam = convertirTamFileP(tam.toLong())
                    )
                )
            }
        }
        cur.close()
    }
    return video
```

Figura 3.93: Método de acceso a videos del dispositivo

Elaborado por: Alejandro Mejía

C. Documentos Dispositivo

El siguiente método se encarga de obtener la lista de los principales formatos de documentos disponibles en el dispositivo. A diferencia de los métodos anteriores, se agregan otros filtros para realizar una búsqueda más específica, los valores describen archivos cuya extensión pertenezca a “pdf”, “doc”, “docx”, “xls”, “xlsx”,ppt, “pptx”, “txt”

```
fun obtenerDocumentosDispositivo(context: Context): List<Documentos> {
    val documento = ArrayList<Documentos>()
    val arc_pdf = MimeTypeMap.getSingleton().getMimeTypeFromExtension( extension: "pdf")
    val arc_doc = MimeTypeMap.getSingleton().getMimeTypeFromExtension( extension: "doc")
    val arc_docx = MimeTypeMap.getSingleton().getMimeTypeFromExtension( extension: "docx")
    val arc_xls = MimeTypeMap.getSingleton().getMimeTypeFromExtension( extension: "xls")
    val arc_xlsx = MimeTypeMap.getSingleton().getMimeTypeFromExtension( extension: "xlsx")
    val arc_ppt = MimeTypeMap.getSingleton().getMimeTypeFromExtension( extension: "ppt")
    val arc_pptx = MimeTypeMap.getSingleton().getMimeTypeFromExtension( extension: "pptx")
    val atc_txt = MimeTypeMap.getSingleton().getMimeTypeFromExtension( extension: "txt")
    val uri = MediaStore.Files.getContentUri(MediaStore.VOLUME_EXTERNAL)
    val columnas = arrayOf(
        MediaStore.Files.FileColumns._ID,
        MediaStore.Files.FileColumns.DISPLAY_NAME,
        MediaStore.Files.FileColumns.DATA,
        MediaStore.Files.FileColumns.SIZE
    )
    val param_busqueda =
        (MediaStore.Files.FileColumns.MIME_TYPE + "=?" + " OR " +
                MediaStore.Files.FileColumns.MIME_TYPE + "=?" +
                " OR " + MediaStore.Files.FileColumns.MIME_TYPE +
                "=?" + " OR " + MediaStore.Files.FileColumns.MIME_TYPE +
                "=?" + " OR " + MediaStore.Files.FileColumns.MIME_TYPE +
                "=?" + " OR " + MediaStore.Files.FileColumns.MIME_TYPE +
                "=?" + " OR " + MediaStore.Files.FileColumns.MIME_TYPE + "=?")
    val argumentos =
        arrayOf(arc_pdf, arc_doc, arc_docx, arc_xls, arc_xlsx, arc_ppt, arc_pptx, atc_txt)
    context.contentResolver.query(
        uri,
        columnas,
        param_busqueda,
        argumentos,
        sortOrder: "${MediaStore.Files.FileColumns.DATE_ADDED} DESC"
    ).use { cur ->
        if (cur != null && cur.count > 0) {
            while (cur.moveToNext()) {
                if (!cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.DISPLAY_NAME))
                        .contains( other: "crypt_")
                ) {
```

Figura 3.94: Método de acceso a documentos del dispositivo

Elaborado por: Alejandro Mejía

```
                ) {
                    val id = cur.getInt(cur.getColumnIndexOrThrow(MediaStore.MediaColumns._ID))
                    val nombre =
                        cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.DISPLAY_NAME))
                    val direccion =
                        cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.DATA))
                            .replace(nombre, newValue: "")
                    val tam =
                        cur.getString(cur.getColumnIndexOrThrow(MediaStore.MediaColumns.SIZE))
                    documento.add(
                        Documentos(
                            uri = id,
                            nombre_doc = nombre,
                            direccion_doc = direccion,
                            seleccionado = false,
                            extFig = extArcDoc(
                                nombre.substring(nombre.lastIndexOf( string: ".")).replace( oldValue: ".", newValue: "")
                            ),
                            tam = convertirTamFileP(tam.toLong())
                        )
                    )
                }
            }
            cur.close()  ^use
        } else if (cur == null) Log.d( tag: "myTag",  msg: "No hay documentos")  ^use
        else Log.d( tag: "myTag",  msg: "Lista vacia")  ^use
    }
    return documento
}
```

Figura 3.95: Método de acceso a documentos del dispositivo II

Elaborado por: Alejandro Mejía

3.2.5.6 Encriptación

Para el cifrado y descifrado de archivos se utilizó la clase Cipher, la cual se encarga de encriptar los archivos seleccionados en base a criterios asignados. Algunos de los

criterios son: ENCRYPT_MODE, DECRYPT_MODE, WRAP_MODE y UNWRAP_MODE, entre otros.

```
fun genClave(password: String): SecretKey? {
    val factory = SecretKeyFactory.getInstance( algorithm: "PBKDF2WithHmacSHA256")
    val spec: KeySpec =
        PBEKeySpec(password.toCharArray(), "A8768CC5BEAA6093".toByteArray(), iterationCount: 10000, keyLength: 256)
    val originalKey: SecretKey = SecretKeySpec(factory.generateSecret(spec).encoded, algorithm: "AES")
    return originalKey
}
```

Figura 3.96: Generar clave de encriptación para Cipher
Elaborado por: Alejandro Mejía

Se generó el método que permite generar la clave de encriptación para que Cipher sea ejecutado. SecretKeyFactory obtiene una clave secreta asignado por el proveedor de seguridad luego de asignar el algoritmo que será utilizado para encriptar dicha clave.

PBEKeySpec es el encargado de convertir la clave secreta proporcionada por SecretKeyFactory, recibe como parámetro: alias, cantidad de saltos, número de iteraciones y tamaño de la clave.

El alias es el valor proporcionado por el usuario para cifrar y descifrar los datos durante la etapa de registro, de modo que sin la clave indicada no es posible restaurar archivos encriptados. La cantidad de saltos y el número de operaciones son los valores complementarios que ayudan a que las claves no sean obtenidas con facilidad.

A. Cifrado de Archivos

El siguiente método detallado en la Figura 3.100 se encarga de cifrar los archivos, eliminar los archivos originales y actualizar el sistema de archivos. Los archivos encriptaos son visibles únicamente en la aplicación, de modo que otras aplicaciones no pueden acceder a su contenido.

- Recibe como parámetros la ruta del archivo y el contexto de la Aplicación, Fragment o Activity
- Process.THREAD_PRIORITY_FOREGROUND: proporciona la prioridad de los procesos en ejecución, si se requiere tener un mayor control para no saturar el espacio de memoria del dispositivo, se requiere utilizar

Process.THREAD_PRIORITY_BACKGROUND, caso contrario Android utilizará la prioridad estándar

- Se utilizó AES como algoritmo para transformar los datos, debido a su velocidad para transformar los datos y estandarización con Cipher
- Cipher.getInstance(): Recibe como parámetro el algoritmo de transformación de los datos. El algoritmo provee modos para transformar los datos, para AES existen disponibles el modo ECB y CBC. El modo ECB a pesar de ser rápido, produce textos de bloque similares lo que significa que cualquier usuario mal intencionado podría adivinar el texto encriptado. A pesar de ser un caso extremo no está aislado, para lo cual el modo CBC cifra cada bloque de manera independiente. Para la última sección se designó el padding PKCS5Padding encargado de generar bits aleatorios que se agregan al inicio o al final para otorgar mayor seguridad al archivo encriptado.
- Luego se inicializó Cipher con el modo de ejecución, la clave de encriptación generado por el alias del usuario y el bytearray de saltos aleatorios, en caso de utilizar el modo ECB este campo no se requiere.
- CipherOutputStream se encarga de generar el archivo de salida especificado con los datos encriptados. Esta etapa es fundamental, ya que los dispositivos móviles tienen una capacidad limitada de memoria para ser usada, cuando es excedida produce errores y por lo tanto podría producir interrupciones durante el proceso, dañando el archivo general. Para ello, se designó un tamaño de buffer significativo que permita realizar las operaciones sin inconvenientes
- Se finalizaron las entradas de archivos abiertas FileInputStream, FileOutputStream, CipherOutputStream para evitar posibles fugas de memoria
- MediaStore se encarga de proveer el contenido multimedia sin embargo para versiones anteriores a Android 11, los cambios efectuados en ciertos archivos no son visibles inmediatamente ya que MediaStore tiene que leer nuevamente el sistema de archivos y encontrar nuevos medios. Por lo cual se implementó un método que permite actualizar los cambios realizados en un archivo y notificarlo a MediaStore para que todas las aplicaciones y el sistema de archivos identifique las actualizaciones.

A continuación, la Figura 3.97 resume lo expuesto:

```
private fun delMediaStore(id: Long, context: Context) {
    val uri = ContentUris.withAppendedId(
        MediaStore.Audio.Media.EXTERNAL_CONTENT_URI, id
    )
    context.contentResolver.delete(uri,  where: null,  selectionArgs: null)
}
```

Figura 3.97: Eliminar archivo de MediaStore

Elaborado por: Alejandro Mejía

- La siguiente etapa se encarga de eliminar directamente del sistema de archivos el archivo original, puesto que se conservará el archivo encriptado.

```
private fun deleteSisArc(direccion: String, nombre: String) {
    val file = File(direccion.substringBefore(nombre), nombre)
    if (file.exists()) {
        file.delete()
    }
}
```

Figura 3.98: Eliminar archivo del sistema de archivos

Elaborado por: Alejandro Mejía

- Se notifican las actualizaciones del archivo encriptado al sistema de archivos, de manera que los cambios sean visibles por otras aplicaciones

```
private fun actualizar(path: File, activity: Context) {
    MediaScannerConnection.scanFile(
        activity, arrayOf(path.path),  mimeTypes: null
    ) { path, uri ->

    }
}
```

Figura 3.99: Actualizar modificaciones en archivos

Elaborado por: Alejandro Mejía

```
fun encryptData(password: String, row: TipoDato, context: Context) {
    android.os.Process.setThreadPriority(android.os.Process.THREAD_PRIORITY_FOREGROUND)
    try {

        val fis = FileInputStream(File(row.direccion, row.nombre))
        val aes = Cipher.getInstance( transformation: "AES/CBC/PKCS5Padding")
        aes.init(
            Cipher.ENCRYPT_MODE,
            genClave(password),
            IvParameterSpec("A8768CC5BEAA6093".toByteArray())
        )
        val fos = FileOutputStream(
            File(
                Environment.getExternalStoragePublicDirectory( type: ".PrivateSecurity"),
                child: ".crypt_" + row.nombre
            )
        )
        val out = CipherOutputStream(fos, aes)
        out.use {  it: CipherOutputStream
            var read = -1
            val buf = ByteArray( size: 1024)
            while (fis.read(buf).also { read = it } != -1) {
                it.write(buf,  off: 0, read)
            }
        }
        out.close()
        fos.close()
        fis.close()
        row.uri?.let { delMediaStore(it.toLong(), context) }
        row.direccion?.let { row.nombre?.let { it1 -> deleteSisArc(it, it1) } }
        actualizar(File(row.direccion, row.nombre), context)

    } catch (e: Exception) {
        Log.d( tag: "Error Enc", e.toString())
    }
}
```

Figura 3.100: Método para encriptar archivos con Cipher

Elaborado por: Alejandro Mejía

B. Restauración de Archivos

La aplicación escanea el sistema de archivos en busca de archivos encriptados y los almacenada en una lista temporal. Los archivos deben cumplir con determinadas condiciones dispuestas por la aplicación para ser almacenadas. A continuación, se detalla una sección del código original para obtener los parámetros de cada archivo:

```
dat_enc.add(
    DatoEncriptado(
        file.name,
        file.absolutePath.replace(file.name,  newValue: ""),
        format.format(Date(file.lastModified())),
         seleccionado: false,
        extensionArchivo(file.extension),
        idArchivo(file.extension),
        convertirTamFile(file)
    )
)
```

Figura 3.101: Método para obtener archivos cifrados del sistema de archivos

Elaborado por: Alejandro Mejía

Se diseñó el método por el cual se restaura el estado original de los archivos. De manera similar que el mecanismo para cifrar los datos se modificó los valores del archivo de salida, todos los archivos recuperados serán escritos en el directorio público /Files. El modo de operación de Cipher se cambió por DECRYPT_MODE. La actualización de archivos con MediaStore no es necesaria ya que los archivos encriptados no son visibles.

```
fun decryptData(password: String, row: DatoEncriptado, activity: Context) {
    android.os.Process.setThreadPriority(android.os.Process.THREAD_PRIORITY_FOREGROUND)
    try {
        val aes = Cipher.getInstance( transformation: "AES/CBC/PKCS5Padding")
        aes.init(
            Cipher.DECRYPT_MODE,
            genClave(password),
            IvParameterSpec("A8768CC5BEAA6093".toByteArray())
        )
        val fis = FileInputStream(
            File(
                row.direccion, row.nombre
            )
        )
        val fos = FileOutputStream(
           ruta
        )
        val opis = CipherInputStream(fis, aes)
        opis.use { inps ->
            var read = -1
            val buf = ByteArray( size: 1024)
            while (inps.read(buf).also { read = it } != -1) {
                fos.write(buf, 0, read)
            }
        }
        opis.close()
        fos.close()
        fis.close()
        row.direccion?.let { row.nombre?.let { it1 -> deleteSisArc(it, it1) } }
        actualizar(
            ruta2, activity
        )
    } catch (e: Exception) { Log.d( tag: "Error", e.toString()) }
}
```

Figura 3.102: Método para restaurar archivos con Cipher

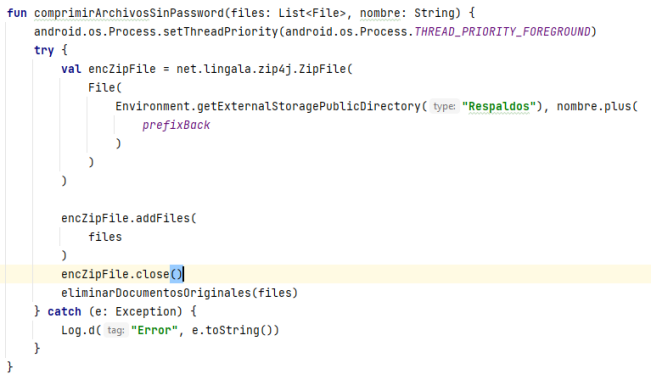
Elaborado por: Alejandro Mejía

C. Encriptación con Respaldos

Los datos seleccionados serán encriptados, añadidos en un archivo .zip y trasladados a un directorio público generado por la aplicación: /Respaldos. El usuario debe seleccionar si desea generar el archivo .zip con o sin el uso de una contraseña para extraer su contenido. Se implementó el método que permite obtener una lista de archivos y asignar un nombre de identificación.

- Se utilizó la clase ZipFile la cual se encarga de obtener la lista de archivos encriptados y los agrega en un archivo de salida .zip

- Cuando el proceso se haya realizado, el contenido original se elimina y se conservan los datos encriptados.



```
fun comprimirArchivosSinPassword(files: List<File>, nombre: String) {
    android.os.Process.setThreadPriority(android.os.Process.THREAD_PRIORITY_FOREGROUND)
    try {
        val encZipFile = net.lingala.zip4j.ZipFile(
            File(
                Environment.getExternalStoragePublicDirectory( type: "Respaldos"), nombre.plus(
                    prefixBack
                )
            )
        )

        encZipFile.addFiles(
            files
        )
        encZipFile.close()
        eliminarDocumentosOriginales(files)
    } catch (e: Exception) {
        Log.d( tag: "Error", e.toString())
    }
}
```

Figura 3.103: Método para cifrar archivos con respaldo

Elaborado por: Alejandro Mejía

Para los usuarios que opten por crear un respaldo con contraseña se utilizó el método de la Figura 3.104, el cual emplea adicionalmente la clase Zip Parameters para establecer que los archivos serán encriptados con un método de encriptación estándar AES y una contraseña para acceder a su contenido

```
fun comprimirArchivos(lista: List<File>, password: String, nombre: String) {
    android.os.Process.setThreadPriority(android.os.Process.THREAD_PRIORITY_FOREGROUND)
    try {
        val encZipFile = net.lingala.zip4j.ZipFile(
            File(
                Environment.getExternalStoragePublicDirectory( type: "Respaldos"), nombre.plus(
                    prefixBack
                )
            ), password.toCharArray()
        )
        val zipParameters = ZipParameters()
        zipParameters.isEncryptFiles = true
        zipParameters.encryptionMethod = EncryptionMethod.ZIP_STANDARD

        encZipFile.addFiles(
            lista, zipParameters
        )
        encZipFile.close()
        eliminarDocumentosOriginales(lista)

    } catch (e: Exception) {
        Log.d( tag: "Error", e.toString())
    }
}
```

Figura 3.104: Método para cifrar archivos con respaldo y contraseña

Elaborado por: Alejandro Mejía

3.2.5.7 Clases View Model

Se generó la clase diseñado para obtener las preferencias de Usuario de la capa de Datos y almacenarlos en datos de tipo Live Data. La característica de estos datos es

obtener los datos en tiempo real, de manera que si existen actualizaciones los cambios serán visibles sin necesidad de ejecutar métodos recurrentemente. Para ello se instanció las clases que contiene los métodos o datos requeridos y transforma su contenido a Live Data. Además, se emplea el uso de corrutinas que permiten ejecutar bloques de código de manera asíncrona respetando los subprocesos principales y secundarios que se asignen.

```
@HiltViewModel
class LoginViewModel
@Inject constructor(
    private val confdatastorepref: ConfiguracionDataStorePref,
    private val repo: UsuarioInfoEncryptRepositorio,
    private val sessionuser: DataStoreSession
) : ViewModel() {
```

Figura 3.105: Declaración de estructura View Model

Elaborado por: Alejandro Mejía

- obtconfdslogvm, se encarga de obtener todos los datos de usuario, registrado en las preferencias de la aplicación
- obtnomusulogvm, obtiene el nombre de usuario
- obtpassusulogvm, obtiene la contraseña
- obtpinusulogvm, obtiene el pin de seguridad
- obtenersession, obtiene el estado de inicio de sesión
- obtenerkeyencrypt, obtiene la clave máster para cifrar y descifrar los datos

```
val obtconfdslogvm = confdatastorepref.obtenerconfusuariodatastore.asLiveData()
val obtnomusulogvm = confdatastorepref.verificarnombreusuariodatastore.asLiveData()
val obtpassusulogvm = confdatastorepref.verificarpasswordusuariodatastore.asLiveData()
val obtpinusulogvm = confdatastorepref.verificarpinusuariodatastore.asLiveData()
val obtenersession = sessionuser.getInicioSesion.asLiveData()
val obtenerkeyencrypt = confdatastorepref.verificarkeyencrypt.asLiveData()
```

Figura 3.106: Métodos de acceso a preferencias compartidas

Elaborado por: Alejandro Mejía

- GuardarDatosUsuario, invoca al proveedor de preferencias y añade el nuevo Usuario creado.
- CerrarSesion, actualiza el valor de la aplicación a false, de manera que, al abrir nuevamente la aplicación, el usuario será dirigido a la interfaz de Login

- GenerarDirectorio, genera en el directorio público del dispositivo un directorio que almacenará los respaldos y archivos encriptados
- EstadoSesion, obtiene el estado de inicio de sesión true-false

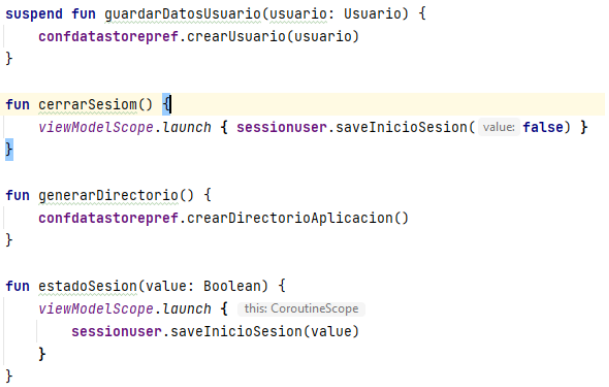


Figura 3.107: Métodos de acceso a preferencias compartidas II

Elaborado por: Alejandro Mejía

A. Datos Teléfono View Model

- Define la clase DatosTelefonoViewModel, encargada de obtener fotos, videos, documentos y archivos encriptados del sistema
- El tipo de dato Mutable Live Data se declaró como var y Live Data como un tipo de dato val, de manera que Live Data serán los datos que se utilizarán para la comunicación con el usuario en los Fragmentos. Los datos no serán modificados ya que la capa de Vista no se encarga de realizar operaciones de escritura, solo de lectura
- postValue: Agrega la lista de datos proporcionada por la capa de datos

```
ewModel.kt × DatosTelefonoViewModel.kt ×

import ...

class DatosTelefonoViewModel : ViewModel() {

    private val datosprivados = ConfiguracionDataPhone()
    private val dataPrivateEncrypt = DataPrivateEncryptRepositorio()
    private val huellarepositorio = HuellaDigitalDataStore()

    //FOTOS
    private var live_fotos: MutableLiveData<ArrayList<Fotos>> = MutableLiveData()
    val fotos_a: LiveData<ArrayList<Fotos>> get() = live_fotos

    private var live_dat_enc: MutableLiveData<ArrayList<DatoEncriptado>> = MutableLiveDat
    val dato_enc: LiveData<ArrayList<DatoEncriptado>> get() = live_dat_enc

    var live_key_error: MutableLiveData<Boolean> = MutableLiveData()

    fun setFoto(context: Context) {
        live_fotos.postValue(datosprivados.obtenerImagenesDispositivo(context)!!)
    }

    fun setEncriptados(context: Context) {
        live_dat_enc.postValue(datosprivados.datosEncriptados(context))
    }

    //VIDEOS
    private var live_videos: MutableLiveData<List<Videos>> = MutableLiveData()
    val videos_a: LiveData<List<Videos>> get() = live_videos

    fun setVideo(context: Context) {
        live_videos.postValue(datosprivados.obtenerVideosDispositivo(context)!!)
    }

    //DOCUMENTOS
    private var live_documentos: MutableLiveData<List<Documentos>> = MutableLiveData()
    val documentos_a: LiveData<List<Documentos>> get() = live_documentos

    fun setDocumentos(context: Context) {
        live_documentos.postValue(datosprivados.obtenerDocumentosDispositivo(context))
    }
```

Figura 3.108: Métodos de acceso a preferencias compartidas III

Elaborado por: Alejandro Mejía

3.2.5.8 Adaptadores

A. Adaptador Fotos

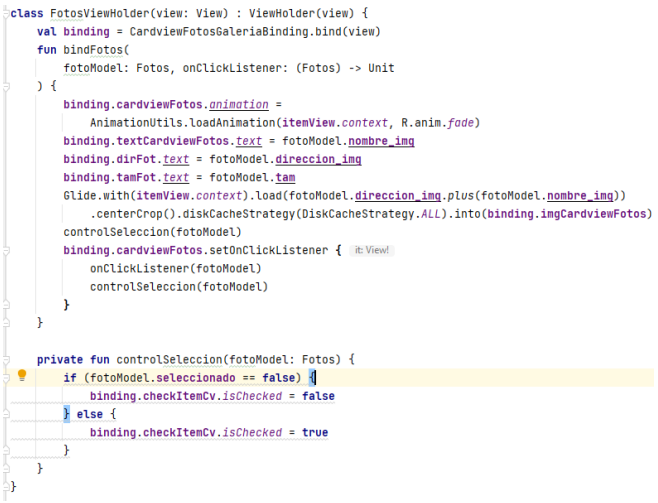
Para visualizar los archivos en la actividad o fragmento se añade un adaptador que recoge los valores de una lista temporal y se distribuye al View Holder

```
class AdaptadorFotos(
    private var fotos: List<Fotos>,
    private val onClickListener: (Fotos) -> Unit
) : RecyclerView.Adapter<FotosViewHolder>() {
    override fun onCreateViewHolder(
        viewGroup: ViewGroup, viewType: Int
    ): FotosViewHolder {
        val layoutInflater = LayoutInflater.from(viewGroup.context)
        return FotosViewHolder(
            layoutInflater.inflate(
                R.layout.cardview_fotos_galeria, viewGroup, attachToRoot: false
            )
        )
    }

    override fun onBindViewHolder(holder: FotosViewHolder, position: Int) {
        val item = fotos[position]
        holder.bindFotos(item, onClickListener)
    }

    override fun getItemCount(): Int = fotos.size

    fun fotosFiltradas(coin: List<Fotos>){
        fotos = coin
        notifyDataSetChanged()
    }
}
```

Figura 3.109: Diseño de adaptador para fotos

Elaborado por: Alejandro Mejía

El View Holder recoge la información de la lista temporal, la distribuye para cada uno de los componentes de la vista y es representado a través de un RecyclerView. La librería Glide permitió representar la ruta de los archivos seleccionados, mediante la propiedad DiskCacheStrategy.ALL se optimiza el tiempo de carga de los archivos, así cuando la aplicación sea ejecutada nuevamente la información se encuentre disponible. El proceso se realizó para configurar los adaptadores de Videos, Documentos y Archivos Encriptados.



```
class FotosViewHolder(view: View) : ViewHolder(view) {
    val binding = CardviewFotosGaleriaBinding.bind(view)
    fun bindFotos(
        fotoModel: Fotos, onClickListener: (Fotos) -> Unit
    ) {
        binding.cardviewFotos.animation =
            AnimationUtils.loadAnimation(itemView.context, R.anim.fade)
        binding.textCardviewFotos.text = fotoModel.nombre_img
        binding.dirFot.text = fotoModel.direccion_img
        binding.tamFot.text = fotoModel.tam
        Glide.with(itemView.context).load(fotoModel.direccion_img.plus(fotoModel.nombre_img))
            .centerCrop().diskCacheStrategy(DiskCacheStrategy.ALL).into(binding.imgCardviewFotos)
        controlSeleccion(fotoModel)
        binding.cardviewFotos.setOnClickListener {  it: View!
            onClickListener(fotoModel)
            controlSeleccion(fotoModel)
        }
    }

    private fun controlSeleccion(fotoModel: Fotos) {
        if (fotoModel.seleccionado == false) {
            binding.checkItemCv.isChecked = false
        } else {
            binding.checkItemCv.isChecked = true
        }
    }
}
```

Figura 3.110: View Holder para adaptador de Fotos

Elaborado por: Alejandro Mejía

B. Adaptador Videos

```
class AdaptadorVideos(
    private var lista_videos: List<Videos>, private val onClickListener: (Videos) -> Unit
) : RecyclerView.Adapter<VideosViewHolder>() {

    override fun onCreateViewHolder(viewGroup: ViewGroup, viewType: Int): VideosViewHolder {
        val layoutInflater = LayoutInflater.from(viewGroup.context)
        return VideosViewHolder(
            layoutInflater.inflate(
                R.layout.cardview_videos_galeria, viewGroup,  attachToRoot: false
            )
        )
    }

    override fun onBindViewHolder(holder: VideosViewHolder, position: Int) {
        val item = lista_videos[position]
        holder.render(item, onClickListener)
    }

    override fun getItemCount(): Int {
        return lista_videos.size
    }

    fun videosFiltradas(coin: List<Videos>) {
        lista_videos = coin
        notifyDataSetChanged()
    }

}
```

Figura 3.111: Diseño de adaptador para fotos

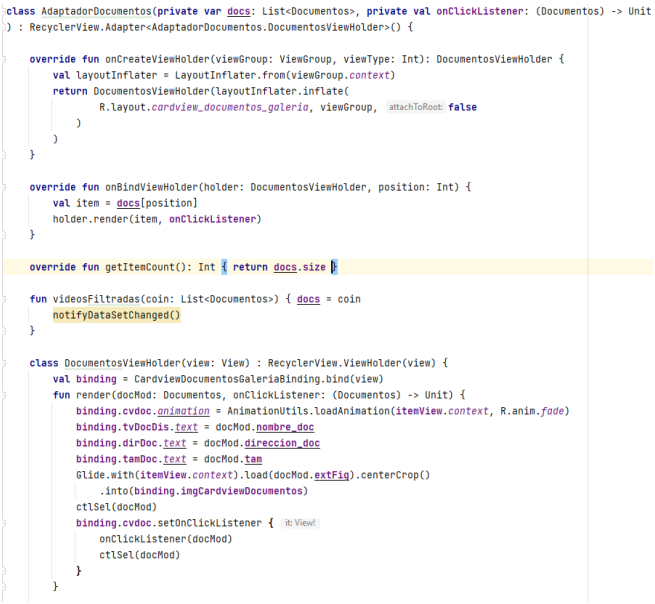
Elaborado por: Alejandro Mejía

```
class VideosViewHolder(view: View) : RecyclerView.ViewHolder(view) {
    val binding = CardviewVideosGaleriaBinding.bind(view)
    fun render(videoModel: Videos, onClickListener: (Videos) -> Unit) {
        binding.cardviewVideos.animation =
            AnimationUtils.loadAnimation(itemView.context, R.anim.fade)
        binding.textCardviewVideos.text = videoModel.nombre_video
        binding.dirVid.text = videoModel.direccion_video
        binding.tamVid.text = videoModel.tam
        Glide.with(itemView.context).load(videoModel.direccion_video.plus(videoModel.nombre_video))
            .centerCrop().diskCacheStrategy(DiskCacheStrategy.ALL).into(binding.videos)
        controlSeleccion(videoModel)
        binding.cardviewVideos.setOnClickListener {  it: View!
            onClickListener(videoModel)
            controlSeleccion(videoModel)
        }
    }

    private fun controlSeleccion(videoModel: Videos) {
        if (videoModel.seleccionado == false) {
            binding.checkVItemCv.isChecked = false
        } else {
            binding.checkVItemCv.isChecked = true
        }
    }
}
```

Figura 3.112: View Holder para adaptador de Videos

Elaborado por: Alejandro Mejía

C. Adaptador Documentos



```
class AdaptadorDocumentos(private var docs: List<Documentos>, private val onClickListener: (Documentos) -> Unit
) : RecyclerView.Adapter<AdaptadorDocumentos.DocumentosViewHolder>() {

    override fun onCreateViewHolder(viewGroup: ViewGroup, viewType: Int): DocumentosViewHolder {
        val layoutInflater = LayoutInflater.from(viewGroup.context)
        return DocumentosViewHolder(layoutInflater.inflate(
                R.layout.cardview_documentos_galeria, viewGroup,  attachToRoot: false
            )
        )
    }

    override fun onBindViewHolder(holder: DocumentosViewHolder, position: Int) {
        val item = docs[position]
        holder.render(item, onClickListener)
    }

    override fun getItemCount(): Int { return docs.size }

    fun videosFiltradas(coin: List<Documentos>) { docs = coin
        notifyDataSetChanged()
    }

    class DocumentosViewHolder(view: View) : RecyclerView.ViewHolder(view) {
        val binding = CardviewDocumentosGaleriaBinding.bind(view)
        fun render(docMod: Documentos, onClickListener: (Documentos) -> Unit) {
            binding.cvdoc.animation = AnimationUtils.loadAnimation(itemView.context, R.anim.fade)
            binding.tvDocDis.text = docMod.nombre_doc
            binding.dirDoc.text = docMod.direccion_doc
            binding.tamDoc.text = docMod.tam
            Glide.with(itemView.context).load(docMod.extFig).centerCrop()
                .into(binding.imgCardviewDocumentos)
            ctlSel(docMod)
            binding.cvdoc.setOnClickListener {  it: View!
                onClickListener(docMod)
                ctlSel(docMod)
            }
        }
```

Figura 3.113: Diseño de adaptador para documentos

Elaborado por: Alejandro Mejía

D. Adaptador Archivos Encriptados

```
class AdaptadorEncriptados(
    private var enc: List<DatoEncriptado>, private val onClickListener: (DatoEncriptado) -> Unit
) : RecyclerView.Adapter<EncriptadosViewHolder>() {
    override fun onCreateViewHolder(
        viewGroup: ViewGroup, viewType: Int
    ): EncriptadosViewHolder {
        val layoutInflater = LayoutInflater.from(viewGroup.context)
        return EncriptadosViewHolder(
            layoutInflater.inflate(
                R.layout.cardview_encriptados, viewGroup,  attachToRoot: false
            )
        )
    }

    override fun onBindViewHolder(holder: EncriptadosViewHolder, position: Int) {
        val item = enc[position]
        holder.bindEncriptados(ite
    }

    override fun getItemCount(): Int = enc.size

    fun datosEnc(fl: ArrayList<DatoEncriptado>) {
        enc = fl
        notifyDataSetChanged()
    }
}
```

value-parameter holder: EncriptadosViewHolder
dataencryptproject.app.main

Figura 3.114: Diseño de adaptador para fotos

Elaborado por: Alejandro Mejía

```
package dev.uta.example.dataencryptproject.vista.adaptador

import ...

class EncriptadosViewHolder(view: View) : ViewHolder(view) {
    val binding = CardviewEncriptadosBinding.bind(view)
    fun bindEncriptados(
        encript: DatoEncriptado, onClickListener: (DatoEncriptado) -> Unit
    ) {
        binding.idLine.animation = AnimationUtils.loadAnimation(itemView.context, R.anim.fade)
        binding.textCardviewEnc.text = encript.nombre?.replace( oldValue: ".crypt_",  newValue: "")?.replace( oldValue: "crypt_", newValue: "")
            ?: ""
        binding.modEnc.text = encript.ult_mod
        binding.tamEnc.text = encript.totspace.toString()
        binding.extEnc.text = encript.ext.toString()

        Glide.with(itemView.context).load(encript.extFig).centerCrop().into(binding.imgCardviewEnc)
        controlSeleccion(encript)
        binding.idLine.setOnClickListener {
            onClickListener(encript)
            controlSeleccion(encript)
        }
    }

    private fun controlSeleccion(encript: DatoEncriptado) {
        if (encript.seleccionado == false) {
            binding.checkEncriptados.isChecked = false
        } else {
            binding.checkEncriptados.isChecked = true
        }
    }
}
```

Figura 3.115: View Holder para adaptador de Datos Encriptados

Elaborado por: Alejandro Mejía

3.2.5.9 Servicio Sensor

Para la ejecución del Servicio Sensor se generó la clase Módulo mediante la cual se genera la notificación de inicio del servicio, debido a que Android solicita para determinadas versiones de Android que para ejecutar servicios en primer plano es necesario notificar al usuario a través de Notification Manager. Se verificó la versión de Android para agregar las configuraciones a la notificación; cuando se presiona en la notificación no se debe crear una nueva actividad o fragmento sino retornar a la

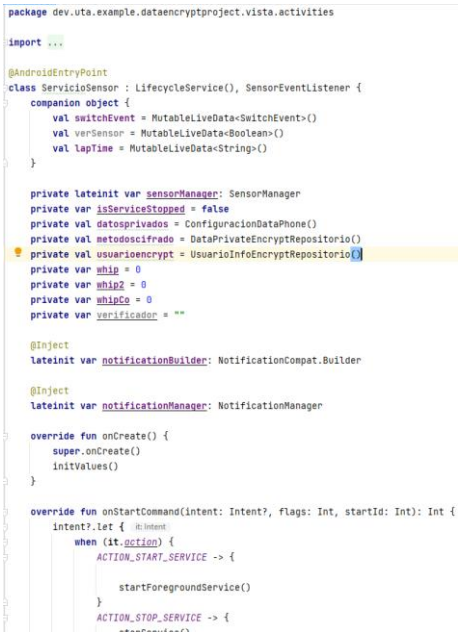
actividad inicial, por lo cual se establece la propiedad FLAG_INMUTABLE para versiones de Android 23 posteriores o caso contrario FLAG_UPDATE_CURRENT

```
package dev.uta.example.dataencryptproject.dependencias

import ...

@Module
@InstallIn(ServiceComponent::class)
object ServicioModule {

    @ServiceScoped
    @Provides
    fun provideMainActivityPendingIntent(@ApplicationContext context: Context) =
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            PendingIntent.getActivity(
                context, requestCode: 420, Intent(context, HomeActivity::class.java).also {
                    it.action = ACTION_SHOW_TRACKING_FRAGMENT
                }, flags: PendingIntent.FLAG_UPDATE_CURRENT or PendingIntent.FLAG_IMMUTABLE
            )
        } else {
            PendingIntent.getActivity(
                context, requestCode: 420, Intent(context, HomeActivity::class.java).also {
                    it.action = ACTION_SHOW_TRACKING_FRAGMENT
                }, PendingIntent.FLAG_UPDATE_CURRENT
            )
        }

    @ServiceScoped
    @Provides
    fun provideNotificationBuilder(
        @ApplicationContext context: Context, pendingIntent: PendingIntent
    ) = NotificationCompat.Builder(context, NOTIFICATION_CHANNEL_ID).setAutoCancel(false)
        .setOngoing(true).setSmallIcon(R.drawable.ic_launcher_foreground)
        .setPriority(NotificationCompat.PRIORITY_HIGH)
        .setContentTitle("AppSecurity").setContentText("Servicio Activado")
        .setContentIntent(pendingIntent)

    @ServiceScoped
    @Provides
    fun provideNotificationManager(@ApplicationContext context: Context) =
        context.getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager

}
```

Figura 3.116: View Holder para adaptador de Datos Encriptados

Elaborado por: Alejandro Mejía

Se creó la clase ServicioSensor la cual extiende de LifecycleService() y de SensoreEventListener. Para acceder a los datos del dispositivo se declaran las referencias a las clases ViewModel. Notification Builder permite construir la notificación para el usuario, mientras que Sensor Manager monitorea los datos recopilados por el sensor.



```
package dev.uta.example.dataencryptproject.vista.activities

import ...

@AndroidEntryPoint
class ServicioSensor : LifecycleService(), SensorEventListener {
    companion object {
        val switchEvent = MutableLiveData<SwitchEvent>()
        val verSensor = MutableLiveData<Boolean>()
        val lapTime = MutableLiveData<String>()
    }

    private lateinit var sensorManager: SensorManager
    private var isServiceStopped = false
    private val datosprivados = ConfiguracionDataPhone()
    private val metodoscifrado = DataPrivateEncryptRepositorio()
    private val usuarioencrypt = UsuarioInfoEncryptRepositorio()
    private var whip = 0
    private var whip2 = 0
    private var whipCo = 0
    private var verificador = ""

    @Inject
    lateinit var notificationBuilder: NotificationCompat.Builder

    @Inject
    lateinit var notificationManager: NotificationManager

    override fun onCreate() {
        super.onCreate()
        initValues()
    }

    override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {
        intent?.let { it: Intent
            when (it.action) {
                ACTION_START_SERVICE -> {

                    startForegroundService()
                }
                ACTION_STOP_SERVICE -> {
```

Figura 3.117: Creación de servicio en primer plano

Elaborado por: Alejandro Mejía

El método initValues permite establecer el estado inicial de los componentes de la actividad principal. Mientas que startForegroundService inicia el servicio en primer plano, hasta que los datos recopilados por los sensores cumplan con ciertas condiciones de movimiento. Sensor Manager requiere que se le asigne el tipo de sensor del cual obtendrá sus datos. En vista de ello se agregó el sensor acelerómetro y vector de rotación del juego.

```
                    stopService()
                }
            }
        }
        return super.onStartCommand(intent, flags, startId)
    }

    private fun initValues() {
        switchEvent.postValue(SwitchEvent.OFF)
        lapTime.postValue( value: "")
    }

    private fun startForegroundService() {
        switchEvent.postValue(SwitchEvent.ON)
        lapTime.postValue( value: "")
        sensorManager = getSystemService(SENSOR_SERVICE) as SensorManager
        sensorManager.registerListener(
            listener: this,
            sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER),
            SensorManager.SENSOR_DELAY_NORMAL
        )
        sensorManager.registerListener(
            listener: this,
            sensorManager.getDefaultSensor(Sensor.TYPE_ROTATION_VECTOR),
            SensorManager.SENSOR_DELAY_NORMAL
        )

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            createNotificationChannel()
        }
        startForeground(NOTIFICATION_ID, notificationBuilder.build())
        lapTime.observe( owner: this, Observer { it: String!
        if (!isServiceStopped) {
            notificationBuilder.setContentText(
                "En ejecución"
            )
            notificationManager.notify(NOTIFICATION_ID, notificationBuilder.build())
        }
        })
    }
```

Figura 3.118: Creación de servicio en primer plano II

Elaborado por: Alejandro Mejía

Para que los datos sean encriptados el sensor acelerómetro debe cumplir con las siguientes condiciones

- Movimiento de izquierda a derecha sobre el eje X; es decir valor en X menor que 7 y valor en X mayor que 7, mientras el dispositivo se encuentra en posición vertical sobre el eje Y.
- El sensor vector de rotación del juego registra la orientación del dispositivo mayor a 45 grados en el eje Y menor que 45 grados sobre el mismo eje, es decir ubicar el dispositivo boca abajo y luego en su posición inicial.

Figura 3.119: Creación de servicio en primer plano III

Elaborado por: Alejandro Mejía

Cuando las condiciones mencionadas se cumplen los datos empiezan a encriptarse, este mecanismo está destinado a respaldar fotos y documentos.

Figura 3.120: Creación de servicio en primer plano IV

Elaborado por: Alejandro Mejía

3.2.5.10 Control de Vistas

Se diseñaron cuatro fragmentos para representar los datos obtenidos del dispositivo, a través de un controlador View Pager se trasladan las vistas para ver su contenido. El fragmento principal se encarga de controlador al resto de vistas, de tal forma que los datos cambios realizados sean visibles entre sí.

```
fun initViewPager2() {
    binding.viewpager2.adapter = AdaptadorViewPager(parentFragmentManager, lifecycle)
    val tabNames = arrayOf("Sel", "Foto", "Video", "Doc")
    TabLayoutMediator(binding.tablayout, binding.viewpager2) { tab, position ->
        tab.text = tabNames[position]
    }.attach()
}
```

Figura 3.121: Control de fragmentos con View Pager

Elaborado por: Alejandro Mejía

A. Proyección de Datos

Para mostrar los datos se diseñó el siguiente esquema encargado de recuperar los datos de la clase ViewModel y los agrega en el adaptador del RecyclerView. Se utilizó corrutinas para ejecutar las operaciones únicamente cuando la vista del fragmento inicie, Lifecycle.Starte.STARTED permite reservar el uso de la memoria para operaciones prioritarias. El contexto que se asigna a cada propiedad es importante

debido a que si se necesita una comunicación en tiempo real entre varios fragmentos es necesario referenciar el contexto de la Actividad principal.

```
viewLifecycleOwner.lifecycleScope.launch {  this: CoroutineScope
    viewLifecycleOwner.lifecycle.repeatOnLifecycle(Lifecycle.State.STARTED) {  this: CoroutineScope
        binding.contenedorimg.adapter?.notifyDataSetChanged()
        binding.contenedorimg.setHasFixedSize(true)
        binding.contenedorimg.layoutManager =
            LinearLayoutManager(activity?.applicationContext)
        datostelefonoviewmodel.fotos_a.observe(viewLifecycleOwner, { fotos ->
            binding.contenedorimg.adapter =
                AdaptadorFotos(fotos = fotos, onClickListener = {  it: Fotos
                    if (it.seleccionado == false) {
                        datostelefonoviewmodel.setSeleccionado(
                            TipoDato(
                                it.nombre_img,
                                it.direccion_img,
                                it.uri,
                                 identificador_seleccion: true,
                                 id_frg: "f",
                                 extFig: 0,
                                it.tam
                            )
                        )
                        it.seleccionado = true
                    } else if (it.seleccionado == true) {
                        datostelefonoviewmodel.removeSeleccionado(
                            TipoDato(
                                it.nombre_img,
                                it.direccion_img,
                                it.uri,
                                 identificador_seleccion: true,
                                 id_frg: "f",
                                 extFig: 0,
                                it.tam
                            )
                        )
                        it.seleccionado = false
                    }
                    if (fotos.all { it.seleccionado == false }) {
                        binding.checkAllItem.isChecked = false
```

Figura 3.122: Método para asignar datos en RecyclerView

Elaborado por: Alejandro Mejía

B. Guardar Estado de los Datos

El siguiente método permite iniciar las operaciones cuando la vista o fragmento se genere por primera vez, de esta manera el adaptador del Recycler View será notificado y obtendrá la lista de los datos solicitados

```
private fun loadOfflineData() {
    viewLifecycleOwner.lifecycleScope.launch { this: CoroutineScope
        viewLifecycleOwner.lifecycle.repeatOnLifecycle(Lifecycle.State.CREATED) { this: CoroutineScope
            datostelefonoviewmodel.setFoto(activity?.applicationContext!!)
        }
    }
}
```

Figura 3.123: Método para asignar datos en RecyclerView II

Elaborado por: Alejandro Mejía

3.2.5.11 Requisitos de interfaz de Usuario

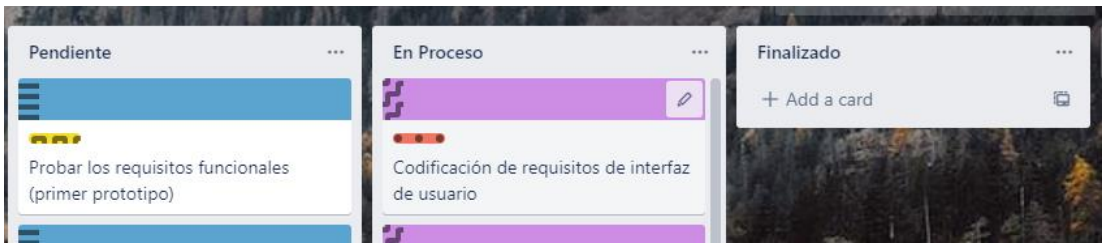


Figura 3.124: Codificación de requisitos de interfaz de usuario

Elaborado por: Alejandro Mejía

A. Página de Bienvenida

En esta sección el usuario debe leer las instrucciones proporcionadas por la aplicación y seleccionar el botón Iniciar para ir a la página de Ingreso.

Figura 3.125: Página de Bienvenida

Elaborado por: Alejandro Mejía

B. Página de Login

Una vez que el usuario seleccione el botón Iniciar de la página de Bienvenida podrá visualizar una interfaz de Login, debe ingresar el nombre de usuario y contraseña respectivamente para acceder al Menú Principal. Si es la primera vez que accede al sistema podrá ingresar a la Página de Registro seleccionando el botón Registrar.

Figura 3.126: Página de Login

Elaborado por: Alejandro Mejía

C. Página de Registro de Usuario

En esta página el usuario debe ingresar datos personales, datos de ingreso y parámetros de seguridad.

- Datos Personales: Nombre, apellido
- Datos de Ingreso: Nombre de usuario, contraseña
- Parámetros de Seguridad: PIN, clave de encriptación

Una vez finalizado el registro el usuario podrá acceder a la página de Login para confirmar sus credenciales e iniciar.

Figura 3.127: Página de Registro de Usuario

Elaborado por: Alejandro Mejía

D. Página de Recuperación de Contraseña

Esta etapa permite a los usuarios restablecer su cuenta en caso de olvidar su contraseña de inicio de sesión, para ello es importante que habilite como requisito el uso de Huellas Dactilares en el dispositivo, ya que la aplicación solicita la verificación de este recurso como medida de seguridad para restablecer los datos. Una vez que los datos se validen el usuario podrá visualizar una interfaz en la que tiene que validar su PIN y finalmente acceder a la página de registro y cambiar los datos.

Figura 3.128: Página recuperación de contraseña I

Elaborado por: Alejandro Mejía

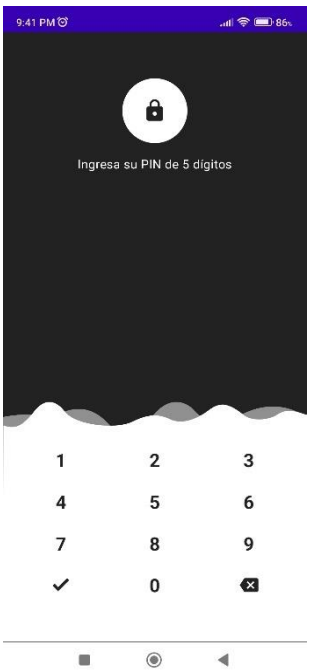


Figura 3.129: Página recuperación de contraseña II

Elaborado por: Alejandro Mejía

E. Página Principal

El usuario podrá acceder a las siguientes opciones para realizar las siguientes acciones:

- **Monitoreo de Sensores.** Activar un servicio en segundo plano, el cual monitorea mediante dos sensores (acelerómetro, vector de rotación del juego) un patrón de movimiento específico y luego encripta fotos y documentos del usuario
- **Buscar Archivos.** Ingresar a la página de búsqueda de archivos
- **Restaurar Archivos.** Restaurar el estado original de archivos encriptados
- **Cerrar Sesión.** Eliminar las preferencias de inicio de sesión para evitar que otras personas puedan acceder a la aplicación
- **Actualizar Datos.** Actualizar los datos del usuario, no se requiere el uso de verificación de identidad adicional como en la página de recuperación de contraseña

Figura 3.130: Página Principal

Elaborado por: Alejandro Mejía

F. Página de Búsqueda de Archivos

El usuario podrá acceder a las siguientes opciones para realizar las siguientes acciones:

- **Visualizar archivos.** Visualizar fotos, videos y documentos disponibles en el dispositivo
- **Seleccionar archivos.** Al hacer clic en cada item podrá añadir los elementos que posteriormente desea encriptar
- **Deseleccionar archivos.** Al hacer clic en el botón superior los datos seleccionados se eliminan para realizar una nueva selección
- **Encriptar archivos.** Cifra los datos seleccionados por el usuario y elimina los archivos originales
- **Encriptar archivos con respaldo.** Cifra los datos dentro de un archivo .zip, los respaldos se almacenan en el directorio /Respaldos del dispositivo
- **Encriptar archivos con respaldo y contraseña.** Para descomprimir el archivo es necesario ingresar la contraseña designada por el usuario.

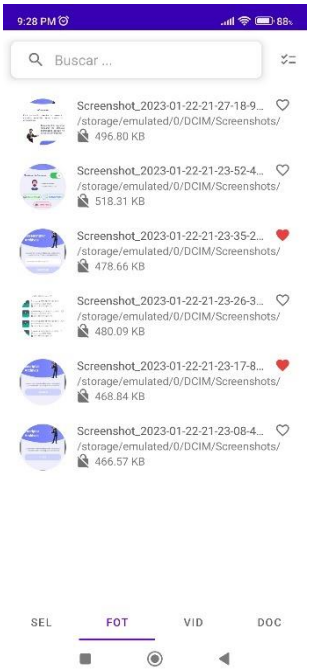


Figura 3.131: Página de búsqueda de archivos

Elaborado por: Alejandro Mejía

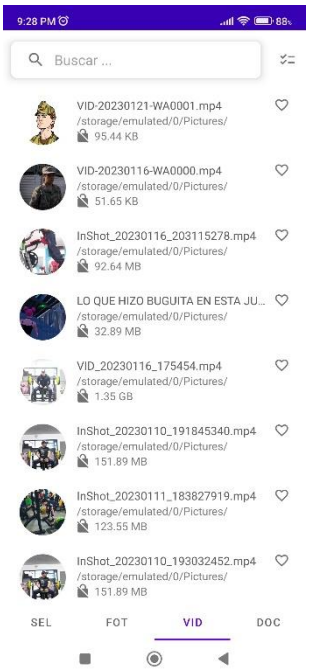


Figura 3.132: Página de búsqueda de archivos II

Elaborado por: Alejandro Mejía

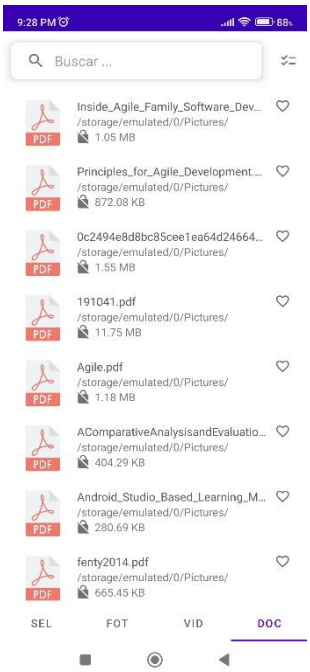


Figura 3.133: Página de búsqueda de archivos III

Elaborado por: Alejandro Mejía

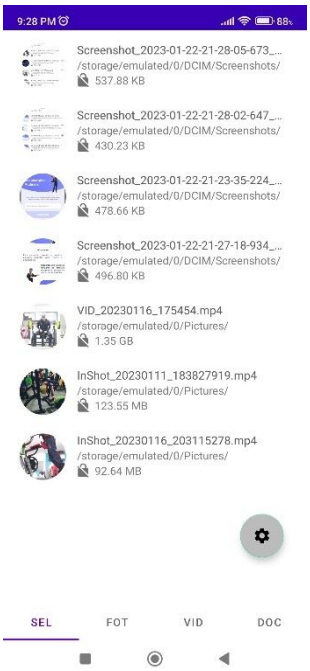


Figura 3.134: Página de búsqueda de archivos IV

Elaborado por: Alejandro Mejía

Figura 3.135: Página de encriptación

Elaborado por: Alejandro Mejía

Figura 3.136: Encriptación con respaldo y clave

Elaborado por: Alejandro Mejía

Figura 3.137: Encriptación con respaldo sin uso de clave

Elaborado por: Alejandro Mejía

G. Página de Restauración de Archivos

El usuario podrá descifrar los datos encriptados ingresando la clave de encriptación ingresada durante la fase de registro de usuario.

Figura 3.138: Página de restauración de archivos

Elaborado por: Alejandro Mejía

Figura 3.139: Página de restauración de archivos II

Elaborado por: Alejandro Mejía

Además, si el usuario utilizó una clave distinta para encriptar los datos, puede cambiar el método de descifrado presionando el Switch de la interfaz gráfica para que los datos se recuperen mediante esta clave personalizada.

Figura 3.140: Página de restauración de archivos III

Elaborado por: Alejandro Mejía

3.2.6 Fase de Prototipado

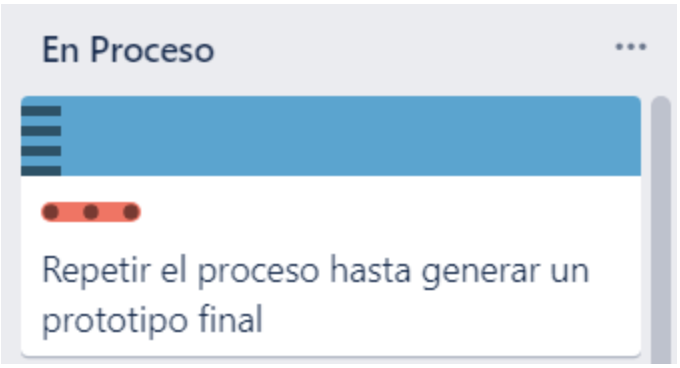


Figura 3.141: Tarea Fase de Prototipado

Elaborado por: Alejandro Mejía

Se determinó el prototipo final de la aplicación con base en los criterios de los usuarios y de los involucrados del proyecto referentes con: usabilidad, apariencia y seguridad.

3.2.7 Fase de Pruebas

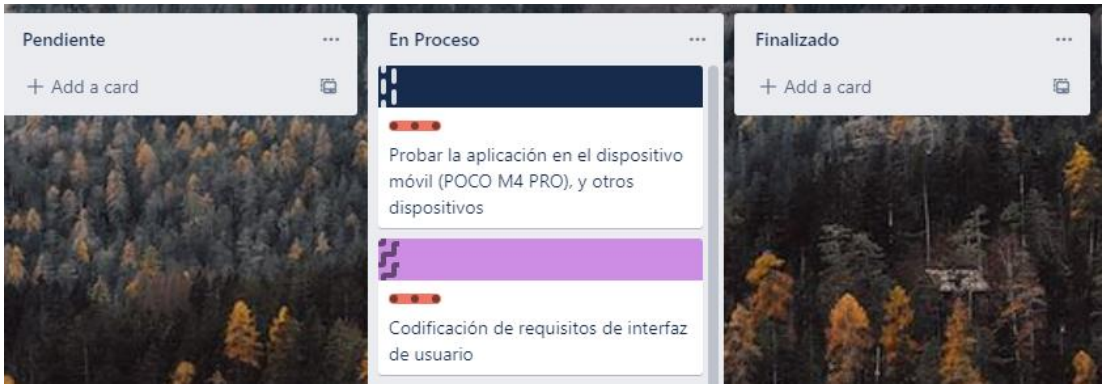


Figura 3.142: Prueba de aplicación en el Dispositivo

Elaborado por: Alejandro Mejía

Las pruebas se efectuaron empleando los mecanismos de encriptación de la aplicación móvil: cifrado de archivos por selección múltiple, cifrado de datos por sensores, y cifrado con respaldo.

3.2.7.1 Cifrado de archivos aleatorios

Inicialmente seleccionar los archivos de la sección Fotos

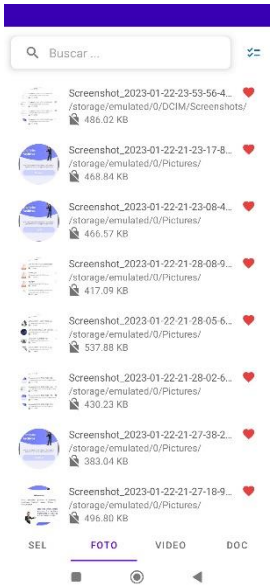


Figura 3.143: Selección de Archivos para cifrado

Elaborado por: Alejandro Mejía

A continuación, verificar los archivos en la sección de selección:

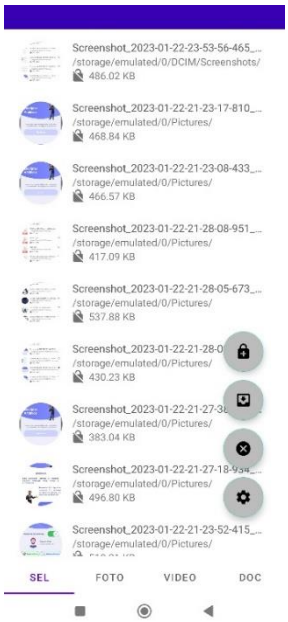


Figura 3.144: Verificación de archivos seleccionados

Elaborado por: Alejandro Mejía

Luego presionar el botón Empezar. La Figura 3.145 resume la información provista por la aplicación al usuario durante la encriptación

Figura 3.145: Encriptación en Proceso

Elaborado por: Alejandro Mejía

En la sección de restauración seleccionar todos los archivos e iniciar ingresando la clave de encriptación. Ver la Figura 3.147.

Figura 3.146: Búsqueda de archivos cifrados

Elaborado por: Alejandro Mejía

Figura 3.147: Desencriptar Archivos

Elaborado por: Alejandro Mejía

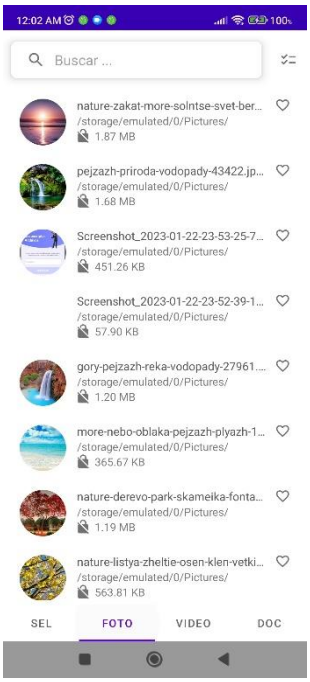


Figura 3.148: Verificación de Archivos restaurados

Elaborado por: Alejandro Mejía

3.2.7.2 Cifrado con Sensores Android

Verificar el número total de archivos antes de empezar con el cifrado

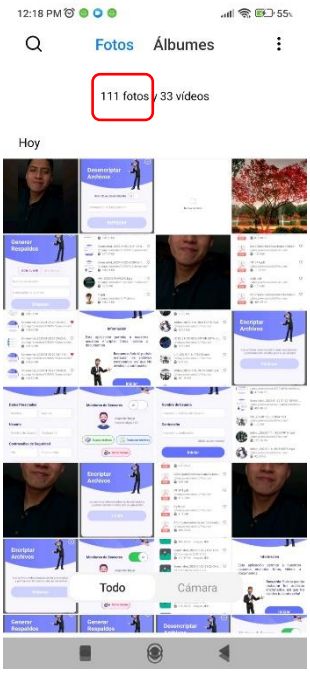


Figura 3.149: Número total de fotos en galería

Elaborado por: Alejandro Mejía

Luego de presionar el switch el servicio en segundo plano el servicio entre en ejecución, durante esta etapa los sensores recopilan los datos de movimiento efectuados por el usuario esperando cumplir con las condiciones de movimiento para que el cifrado de datos inicie.

Figura 3.150: Activación de servicio por sensores

Elaborado por: Alejandro Mejía

En la barra de notificaciones del dispositivo se puede visualizar el aviso de la aplicación para indicar al usuario que el servicio se encuentra en ejecución.

Figura 3.151: Notificación de activación de servicio en segundo plano

Elaborado por: Alejandro Mejía

Luego, efectuar el patrón de movimiento para iniciar con el cifrado de datos. Durante el proceso es posible observar el tipo y la cantidad de datos.

Figura 3.152: Cifrado de fotos con Servicio sensores

Elaborado por: Alejandro Mejía

Figura 3.153: Cifrado de documentos con Servicio sensores

Elaborado por: Alejandro Mejía

El proceso se llevó a cabo en un minuto aproximadamente. En la galería y el sistema de archivos del dispositivo comprobar si los archivos fueron cifrados. La Figura 3.154 resume la cantidad de fotos del dispositivo luego de cifrar los datos.

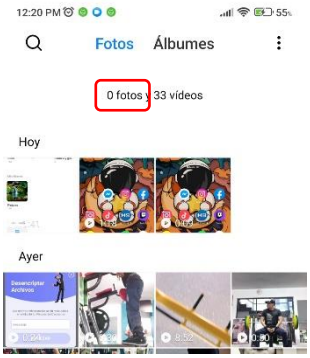


Figura 3.154: Revisión de archivos cifrados en el dispositivo

Elaborado por: Alejandro Mejía

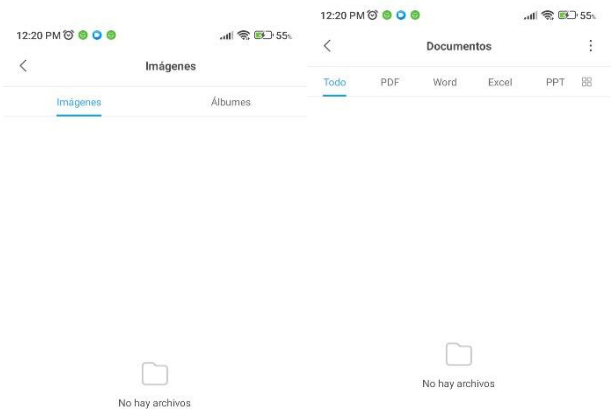


Figura 3.155: Revisión de archivos cifrados en el dispositivo

Elaborado por: Alejandro Mejía

Ingresar en la aplicación y verificar los archivos encriptados. En la Figura 3.156 se pueden observar los archivos cifrados, los cuales únicamente son visibles para la aplicación.

Figura 3.156: Revisión de archivos cifrados en la aplicación móvil

Elaborado por: Alejandro Mejía

Posteriormente, seleccionar todos los archivos encriptados, ingresar la clave e iniciar con el proceso para restaurar el estado original de los datos.

Figura 3.157: Ingreso de clave para descifrar datos

Elaborado por: Alejandro Mejía

Finalmente, en la aplicación móvil, galería y sistema de archivos del dispositivo verificar que el proceso fue realizado correctamente y que los datos sean visibles.

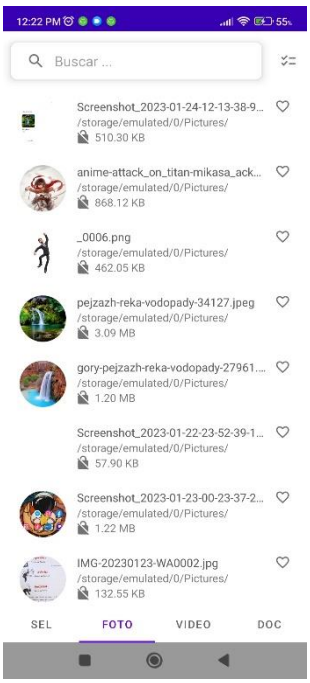


Figura 3.158: Verificación de restauración de archivos en la aplicación móvil

Elaborado por: Alejandro Mejía

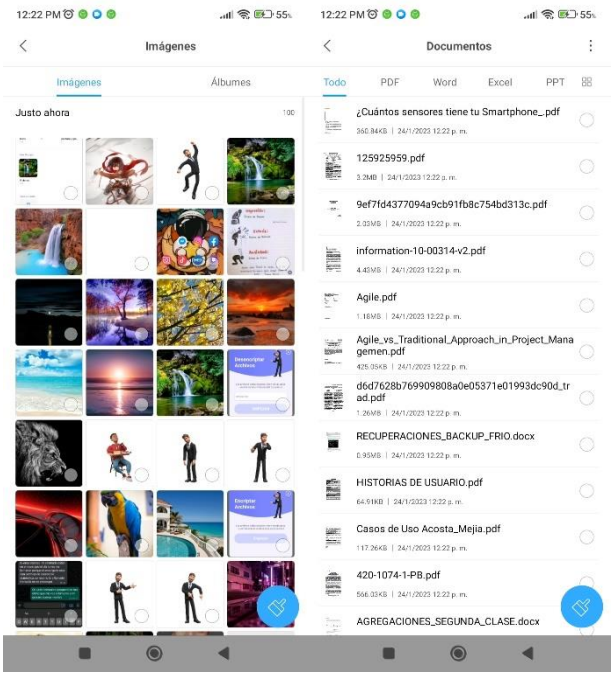


Figura 3.159: Verificación de restauración de archivos en el sistema de archivos

Elaborado por: Alejandro Mejía

3.2.7.3 Cifrado con respaldo

Para generar un respaldo con clave, seleccionar datos aleatoriamente e ingresar un nombre y contraseña.

Figura 3.160: Cifrado de datos con Clave

Elaborado por: Alejandro Mejía

Luego, ingresar al directorio Respaldos del sistema de archivos e ingresar la clave asignada para visualizar su contenido.

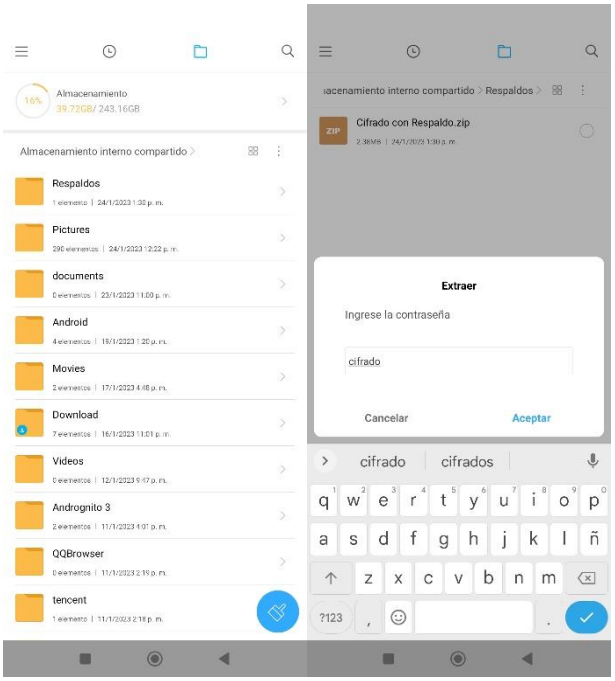


Figura 3.161: Verificación de creación de respaldo con clave

Elaborado por: Alejandro Mejía

Finalmente, comprobar que el archivo .zip contiene los archivos encriptados.

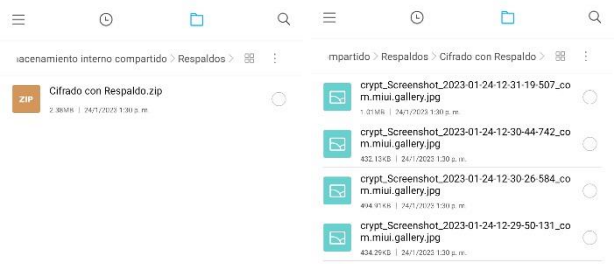


Figura 3.162: Comprobación de contenido de archivo respaldado

Elaborado por: Alejandro Mejía

CAPITULO IV

CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

- Se aplicó apropiadamente los mecanismos de MVVM vinculados en el acceso a los datos entre los ciclos de vida de cada actividad o fragmento, previniendo posibles fugas de memoria, adquisición de datos incoherentes e inconsistencia en la información visualizada por los usuarios.
- La búsqueda de una metodología para encriptación de datos permitió establecer que por medio de Cipher, se pueden cifrar los datos basándose en algoritmos específicos de encriptación que impacten efectivamente en el rendimiento, velocidad, autenticidad de los datos y optimización de los recursos de memoria de los dispositivos móviles.
- Cipher facilitó el establecimiento de un esquema de autenticación para restablecer el estado original de los datos, mediante claves secretas que se proporcionen por los propietarios, ofreciendo así un aporte sustancial en la privacidad personal.
- De acuerdo con el análisis sobre las funciones de los distintos sensores integrados en dispositivos Android, se obtuvo que el sensor acelerómetro en conjunto con el sensor de orientación, facilitan la detección precisa de los cambios de posición y patrones de movimiento efectuados. Esto favorece al desarrollo de aplicaciones que ejecuten procesos o servicios en segundo plano de acuerdo con los datos recopilados por los sensores.
- La versatilidad de los sensores es un antecedente que motivó su utilización dentro del proyecto, debido a que gran parte de los dispositivos actuales e inclusive en versiones antiguas los integran, siendo un factor determinante para utilizar sus capacidades en áreas de seguridad y protección de información.

- Se desarrolló una aplicación móvil que cumple con los requerimientos iniciales establecidos, obteniendo una herramienta útil que garantice a los usuarios la protección de sus datos frente a diversas fuentes de robo de información y eventos que puedan afectar su reputación.

4.2 RECOMENDACIONES

A continuación, se detallan una serie de recomendaciones cuya aplicación es fundamental para la implementación de proyectos similares.

- Utilizar dispositivos con versiones superiores de Android API 28 (P), para ejecutar procesos que requieren altos niveles de recursos de memoria como es el caso de la encriptación de videos.
- Es recomendable durante el proceso de encriptación, actualizar los metadatos adquiridos por MediaStore, en vista de que, para determinadas versiones de Android los cambios aplicados (actualizar, eliminar), en archivos multimedia no son visibles a menos de que el proveedor de metadatos escanee el sistema de archivos en busca de nuevos medios o que el dispositivo se restablezca.
- Definir un tamaño de buffer específico para almacenar los bytes encriptados considerando la cantidad de memoria RAM, a causa de que forzar la asignación de un gran conjunto de bytes para un espacio limitado, ocasionaría la interrupción de las operaciones de la aplicación en dispositivos con capacidades limitadas de lectura y escritura.
- Se sugiere verificar el estado de los sensores y calibrarlos en caso de que presenten anomalías, con el fin de evitar que la recopilación de los datos genere valores inconsistentes o con variaciones sin compensación.
- Se debe fortalecer la integración y mantenimiento de los mecanismos de seguridad en dispositivos móviles y la capacitación de los usuarios para su utilización.

BIBLIOGRAFÍA

[1] "Protección de datos personales en la aplicación de telefonía móvil whatsapp messenger." https://repositorio.uchile.cl/handle/2250/131714 (accessed May 27, 2022).

[2] "El robo de identidad y sus cifras en América Latina | WeLiveSecurity." https://www.welivesecurity.com/la-es/2011/04/08/robo-identidad-cifras-america-latina/ (accessed May 27, 2022).

[3] "¿Cómo evitar que nos roben datos personales del móvil? - ¿Cómo evitar que nos roben datos personales del móvil?" https://reportajes.lavanguardia.com/bq-evitar-robo-datos-personales-movil/ (accessed May 27, 2022).

[4] "El teléfono celular, en camino de ser la principal vía de entrada de las ciberamenazas | Doctor Tecno | La Revista | El Universo." https://www.eluniverso.com/larevista/tecnologia/el-telefono-celular-en-camino-de-ser-la-principal-via-de-entrada-de-las-ciberamenazas-nota/ (accessed May 27, 2022).

[5] "Potencialidades de los celulares inteligentes para investigaciones biológicas. Parte 1: Sensores integrados." https://redib.org/Record/oai_articulo3188207-potencialidades-de-los-celulares-inteligentes-para-investigaciones-biol%C3%B3gicas-parte-1-sensores-integrados (accessed May 26, 2022).

[6] "(PDF) Con la Física a todas partes: experiencias utilizando el teléfono inteligente." https://www.researchgate.net/publication/273441580_Con_la_Fisica_a_todas_partes_experiencias_utilizando_el_telefono_inteligente (accessed May 26, 2022).

[7] T. Fin *et al.*, "Universidad de Valladolid".

[8] A. O. M. Benitez and B. Ospina, "Benchmark para determinar el sistema de cifrado con mejor rendimiento sobre dispositivos inteligentes," *Informador Técnico*, vol. 84, no. 2, pp. 175–191, Sep. 2020, doi: 10.23850/22565035.2782.

[9] “Derechos Digitales: cómo nos vigilan | Asociación para el Progreso de las Comunicaciones.” https://www.apc.org/es/news/derechos-digitales-c%C3%B3mo-nos-vigilan (accessed May 26, 2022).

[10] “Gestion de procesos Android.” https://es.slideshare.net/frankojur/gestion-de-procesos-26985517 (accessed May 29, 2022).

[11] “Descripción general de sensores | Desarrolladores de Android | Android Developers.” https://developer.android.com/guide/topics/sensors/sensors_overview?hl=es-419 (accessed Nov. 29, 2022).

[12] D. Giacomelli and E. R. Faria, “Study and characterization of the main tools for human activity recognition using accelerometer sensors,” *ACM International Conference Proceeding Series*, pp. 285–292, Jun. 2018, doi: 10.1145/3229345.3229385.

[13] G. Priyadharshini and J. S. F. Josephin, “A comprehensive review of various data collection approaches, features, and algorithms used for the classification of driving style,” *IOP Conf Ser Mater Sci Eng*, vol. 993, no. 1, Nov. 2020, doi: 10.1088/1757-899X/993/1/012098.

[14] A. Mimouna and A. ben Khalifa, “A Survey of Human Action Recognition using Accelerometer Data,” *Smart Sensors, Measurement and Instrumentation*, vol. 38, pp. 1–32, 2021, doi: 10.1007/978-3-030-71225-9_1.

[15] S. Sáez Bombín, “Reconocimiento de actividades físicas con sensores inerciales y Redes Neuronales de Aprendizaje Profundo,” 2018, Accessed: Nov. 18, 2022. [Online]. Available: https://uvadoc.uva.es/handle/10324/33059

[16] M. Nain, “Implementing MATLAB with Android sensors,” 2018, Accessed: Nov. 18, 2022. [Online]. Available: http://www.theseus.fi/handle/10024/144352

[17] S. Naval, A. Pandey, S. Gupta, G. Singal, V. Vinoba, and N. Kumar, “PIN Inference Attack: A Threat to Mobile Security and Smartphone-controlled Robots,” *IEEE Sens J*, Sep. 2021, doi: 10.1109/JSEN.2021.3080587.

[18] S. Odenwald, “A Guide to Smartphone Sensors Experimeter’s Guide To Smartphone Sensors”.

[19] M. Matuz-Cruz, C. J. Garcia-Aquino, E. Reyes-Sanchez, D. Mujica-Vargas, and S. M. Villalobos, “Methodology of measurement of the opening and coverage of the canopy implementing artificial vision techniques,” *IEEE Latin America Transactions*, vol. 18, no. 12, pp. 2138–2146, Dec. 2020, doi: 10.1109/TLA.2020.9400442.

[20] D. K. Lee, F. Nedelkov, and D. M. Akos, “Assessment of Android Network Positioning as an Alternative Source of Navigation for Drone Operations,” *Drones*, vol. 6, no. 2, Feb. 2022, doi: 10.3390/DRONES6020035.

[21] Q. Fu *et al.*, “Ambient light sensor based colorimetric dipstick reader for rapid monitoring organophosphate pesticides on a smart phone,” *Anal Chim Acta*, vol. 1092, pp. 126–131, Dec. 2019, doi: 10.1016/J.ACA.2019.09.059.

[22] H. S. Utama, Kevin, and H. Tanudjaja, “Smart street lighting system with data monitoring,” *IOP Conf Ser Mater Sci Eng*, vol. 1007, no. 1, Dec. 2020, doi: 10.1088/1757-899X/1007/1/012148.

[23] V. Hugo Rodríguez Ontiveros, P. García, and I. Medrano Sanchez, “Aplicaciones de sensores vestibles y teléfonos inteligentes en el bienestar personal: Cuantificación de la actividad física y control de la práctica de mindfulness / Víctor Hugo Rodríguez Ontiveros”, Accessed: May 26, 2022. [Online]. Available: http://zaguan.unizar.es

[24] A. Jalal, M. A. K. Quaid, S. B. Ud Din Tahir, and K. Kim, “A study of accelerometer and gyroscope measurements in physical life-log activities detection systems,” *Sensors (Switzerland)*, vol. 20, no. 22, pp. 1–23, Nov. 2020, doi: 10.3390/S20226670.

[25] “UCI Machine Learning Repository: OPPORTUNITY Activity Recognition Data Set.” https://archive.ics.uci.edu/ml/datasets/opportunity+activity+recognition (accessed Nov. 17, 2022).

[26] "UCI Machine Learning Repository: Heterogeneity Activity Recognition Data Set." https://archive.ics.uci.edu/ml/datasets/heterogeneity+activity+recognition (accessed Nov. 17, 2022).

[27] "UCI Machine Learning Repository: REALDISP Activity Recognition Dataset Data Set." https://archive.ics.uci.edu/ml/datasets/REALDISP+Activity+Recognition+Dataset (accessed Nov. 17, 2022).

[28] N. Halim, "Stochastic recognition of human daily activities via hybrid descriptors and random forest using wearable sensors," *Array*, vol. 15, Sep. 2022, doi: 10.1016/J.ARRAY.2022.100190.

[29] J. Suto, "The effect of hyperparameter search on artificial neural network in human activity recognition," *Open Computer Science*, vol. 11, no. 1, pp. 411–422, Jan. 2021, doi: 10.1515/COMP-2020-0227.

[30] D. Roy, S. Girdzijauskas, and S. Socolovschi, "Confidence-calibrated human activity recognition," *Sensors*, vol. 21, no. 19, Oct. 2021, doi: 10.3390/S21196566.

[31] "Materiales formativos - Acércate a las TIC | Cooperación." https://www.fundaciondedalo.org/COOPERACION/materiales-formativos-acercate-a-las-tic.html (accessed May 29, 2022).

[32] "HERRAMIENTAS PARA EL CUMPLIMIENTO DE LA RESPONSABILIDAD PROACTIVA".

[33] "Seguridad informática y seguridad de la información." http://repository.unipiloto.edu.co/handle/20.500.12277/2821 (accessed May 29, 2022).

[34] M. I. Romero *et al.*, "Mecanismo Correctivos en seguridad informática," *Introducción a la seguridad informática y el análisis de vulnerabilidades*, 2018.

[35] “Repositorio Institucional de la Universidad Politécnica Salesiana: Estrategias algorítmicas orientadas a la ciberseguridad: Un mapeo sistemático.” https://dspace.ups.edu.ec/handle/123456789/20933 (accessed May 29, 2022).

[36] C. A. C. Chávez, “La encriptación de datos empresariales: ventajas y desventajas,” *RECIMUNDO*, vol. 3, no. 2, pp. 980–997, Apr. 2019, doi: 10.26820/RECIMUNDO/3.(2).ABRIL.2019.980-997.

[37] G. Lemke, “The software development life cycle and its application,” *Senior Honors Theses and Projects*, Jan. 2018, Accessed: Nov. 23, 2022. [Online]. Available: https://commons.emich.edu/honors/589

[38] G. Gurung, R. Shah, and D. P. Jaiswal, “Software Development Life Cycle Models-A Comparative Study,” *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, pp. 30–37, Jul. 2020, doi: 10.32628/CSEIT206410.

[39] S. Al-Ratrout, O. Husain Tarawneh, M. HusniAltarawneh, and M. Yosef Altarawneh, “Mobile Application Development Methodologies Adopted in Omani Market: A Comparative Study,” *International Journal of Software Engineering & Applications*, vol. 10, no. 2, pp. 13–22, Mar. 2019, doi: 10.5121/IJSEA.2019.10202.

[40] N. Magaia, P. Gomes, L. Silva, B. Sousa, C. X. Mavromoustakis, and G. Mastorakis, “Development of Mobile IoT Solutions: Approaches, Architectures, and Methodologies,” *IEEE Internet Things J*, vol. 8, no. 22, pp. 16452–16472, Nov. 2021, doi: 10.1109/JIOT.2020.3046441.

[41] S. Saeed, N. Z. Jhanjhi, M. Naqvi, and M. Humayun, “Analysis of software development methodologies,” *International Journal of Computing and Digital Systems*, vol. 8, no. 5, pp. 445–460, 2019, doi: 10.12785/IJCDS/080502.

[42] K. Risener, “A Study of Software Development Methodologies,” *Computer Science and Computer Engineering Undergraduate Honors Theses*, May 2022, Accessed: Nov. 23, 2022. [Online]. Available: https://scholarworks.uark.edu/csceuht/103

[43] E. Puik and D. Ceglarek, “Application of Axiomatic Design for Agile Product Development,” *MATEC Web of Conferences*, vol. 223, p. 01004, Oct. 2018, doi: 10.1051/MATECCONF/201822301004.

[44] D. J. Hutahaean, “Pengembangan Sistem Informasi Penyewaan Gedung Berbasis Web Dengan Metode Rational Unified Process (Rup) (Studi Kasus : Wisma Rata Medan),” Jul. 2019.

[45] J. Gaete *et al.*, “Enfoque de aplicación ágil con Serum, Lean y Kanban,” *Ingeniare. Revista chilena de ingeniería*, vol. 29, no. 1, pp. 141–157, Mar. 2021, doi: 10.4067/S0718-33052021000100141.

[46] R. Delima, H. B. Santoso, N. Andriyanto, and A. Wibowo, “Development of Purchasing Module for Agriculture E-Commerce using Dynamic System Development Model,” *undefined*, vol. 9, no. 10, pp. 86–96, 2018, doi: 10.14569/IJACSA.2018.091012.

[47] R. Kumar, P. Maheshwary, and T. Malche, “Inside Agile Family Software Development Methodologies,” *International Journal of Computer Sciences and Engineering*, vol. 7, no. 6, pp. 650–660, Jun. 2019, doi: 10.26438/IJCSE/V7I6.650660.

[48] A. Granulo and A. Tanovic, “Comparison of SCRUM and KANBAN in the Learning Management System implementation process,” *27th Telecommunications Forum, TELFOR 2019*, Nov. 2019, doi: 10.1109/TELFOR48224.2019.8971201.

[49] A. A. Shari *et al.*, “Mobile Application of Food Recommendation for Allergy Baby Using Rule-Based Technique,” *2019 IEEE International Conference on Automatic Control and Intelligent Systems, I2CACIS 2019 - Proceedings*, pp. 279–282, Jun. 2019, doi: 10.1109/I2CACIS.2019.8825026.

[50] M. Y. Darus, M. Hazani, and N. Awang, “Mobile Self-Management System for University Students using Mobile Application Development Lifecycle (MADLC),” *undefined*, 2017.

[51] C. M N, K. Ambareen, and S. M. Sundaram, “Developers Guide for Android Applications,” pp. 210–214, Jun. 2018, doi: 10.21467/PROCEEDINGS.1.36.

[52] S. Ersoy and F. Özdöşemeci, “Reading and playing musical notes with image processing techniques with mobile application,” *Vibroengineering Procedia*, vol. 44, pp. 111–116, Aug. 2022, doi: 10.21595/VP.2022.22589.

[53] B. G. Mateus and M. Martinez, “On the adoption, usage and evolution of Kotlin features in Android development,” *International Symposium on Empirical Software Engineering and Measurement*, Oct. 2020, doi: 10.1145/3382494.3410676.

[54] B. G. Mateus, M. Martinez, and C. Kolski, “Learning migration models for supporting incremental language migrations of software applications,” *Inf Softw Technol*, vol. 153, p. 107082, Jan. 2023, doi: 10.1016/J.INFSOF.2022.107082.

[55] A. Mishra, “The MVVM Architectural Pattern,” *iOS Code Testing*, pp. 43–60, 2017, doi: 10.1007/978-1-4842-2689-6_3.

[56] R. Lotfi, “Patterns for development of windows form applications and web applications,” *International Journal of Grid and Distributed Computing*, vol. 9, no. 6, pp. 181–196, 2016, doi: 10.14257/IJGDC.2016.9.6.18.

[57] S. Komolov, G. Dlamini, S. Megha, and M. Mazzara, “Towards Predicting Architectural Design Patterns: A Machine Learning Approach,” *Computers*, vol. 11, no. 10, Oct. 2022, doi: 10.3390/COMPUTERS11100151.

[58] S. A. Pitchay, W. A. A. Alhiagem, F. Ridzuan, and S. Perumal, “Mobile application design for protecting the data in cloud using enhanced technique of encryption,” *International Journal of Engineering and Technology(UAE)*, vol. 7, no. 4, pp. 98–102, 2018, doi: 10.14419/IJET.V7I4.15.21427.

[59] P. P., P. Dhanokar, M. K. Chaithanya, and M. U. Patil, “Secure Storage of Data on Android Based Devices,” *International Journal of Engineering and Technology*, vol. 8, no. 3, pp. 177–182, Mar. 2016, doi: 10.7763/IJET.2016.V6.880.

ANEXOS

Anexo A

ENCUESTA SOBRE SEGURIDAD EN DISPOSITIVOS MÓVILES

1. **¿Qué tipo de información almacenada en su dispositivo móvil es más importante para usted?**

○	Fotos
○	Videos
○	Documentos PDF/ Word/ Legales
○	Música

2. **¿Con qué constancia considera son atacados los dispositivos móviles iOS**

○	Nunca
○	Diariamente
○	Semanalmente
○	Mensualmente
○	Anualmente

3. **¿Con qué frecuencia considera ocurren ataques a dispositivos móviles Android?**

○	Nunca
○	Diariamente
○	Semanalmente
○	Mensualmente
○	Anualmente

4. **¿Qué dispositivo electrónico considera tiene un nivel de seguridad bajo, que lo hace vulnerable a hackeos?**

○	Celular
○	Computadora
○	Laptop
○	iPad
○	Ninguno

5. **¿Conoce personas que hayan sido víctima de robo de información privada de sus dispositivos móvil?**

○	Si
○	No

6. **¿Qué grupo de personas consideraría es más probable que roben su información privada?**

○	**Jóvenes**
○	Adultos Mayores
○	Niños
○	Adolescentes
○	Ninguno

7. **¿Quién se ve más afectado en caso de que su información personal sea expuesta en Internet?**

○	Jóvenes
○	Adultos Mayores
○	Niños
○	Adolescentes
○	Ninguno

8. **Califique el tipo de información que comprometería su integridad personal en caso de ser hackeada o vulnerada de su dispositivo móvil.**

Información	Nada Comprometedora	Poco Comprometedora	Comprometedora	Muy Comprometedora
Contactos Telefónicos				
Videos				
Documentos				
Fotos				

9. ¿Qué método de seguridad utiliza para seguridad de su dispositivo móvil?

○	Contraseña
○	Huella Digital
○	Reconocimiento Facial
○	PIN
○	Ninguno

10. ¿Con que frecuencia descarga e instala aplicaciones en su dispositivo móvil?

○	Frecuentemente
○	Casi Nunca
○	Siempre
○	Nunca

11. ¿Considera que las apps de tiendas de aplicaciones oficiales como PlayStore? Aptoide, Uptodown son seguras y confiables?

○	La mayor parte son seguras
○	Ninguna es segura
○	Pocas
○	Todas

12. ¿Cuál considera es el factor principal para que ocurra el robo o hackeo de información privada de los dispositivos móviles?

○	Desinformación de los usuarios
○	Aplicaciones Inseguras
○	Beneficio Económico
○	No se utilizan contraseñas
○	Ninguno

13. ¿Consideraría útil contar con una aplicación móvil para proteger los datos privados almacenados en los dispositivos móviles?

○	Poco útil
○	Muy útil
○	Útil
○	Nada útil

Anexo B

FICHAS BIBLIOGRÁFICAS

ESQUEMA DE FICHA BIBLIOGRÁFICA	
TEMA	
TESIS	
PROPÓSITO	
IDEAS CENTRALES	
CONCEPTOS CLAVES	
CONCLUSIONES	
APORTE A TEMA ELEGIDO	
CONCLUSIÓN GENERAL	

Anexo C

MANUAL DE USUARIO

INICIO DE APLICACIÓN

La aplicación móvil muestra una pantalla de inicio con información adicional sobre la aplicación, una vez seleccionado la opción empezar accederá a la pantalla de inicio de sesión.

INICIO DE SESIÓN

Para utilizar las funciones de la aplicación es necesario ingresar sus credenciales de usuario, caso contrario no podrá acceder. El sistema valida si los datos son correctos y en caso de ser correctos lo traslada directamente al Menú Principal.

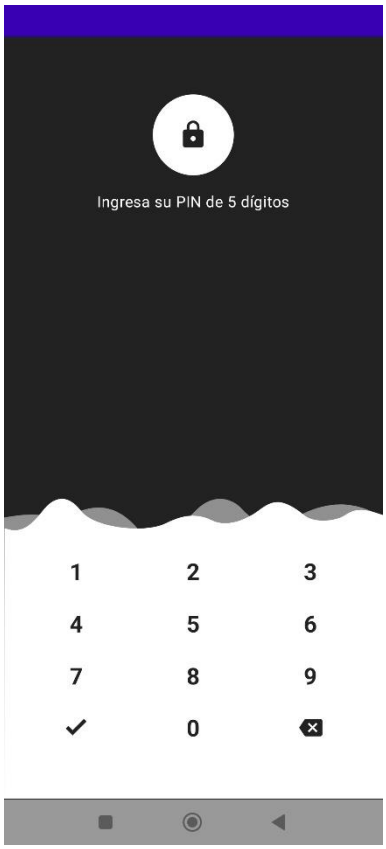
REGISTRO DE USUARIO

La aplicación móvil requiere de una cuenta local que le permitirá administrar eficientemente los datos, para esto es necesario ingresar nombre, apellido, nombre de usuario, contraseña, pin de seguridad y clave de encriptación. La clave de encriptación le permitirá descifrar los datos, por lo tanto, una persona sin la clave de encriptación no podrá restablecer sus datos encriptados. Si el sistema detecta que ya existe un usuario local creado, no podrá registrarse a menos de que usted sea el propietario, en ese caso debe acceder a la opción Olvido su contraseña y verificar su identidad.

RECUPERACIÓN DE INICIO DE SESIÓN

En caso de olvidar sus credenciales de inicio la aplicación le permite recuperar su cuenta a través de un protocolo de seguridad. ¿Luego de que el usuario seleccione la opción “Olvido su contraseña?”, se le solicitará verificar su identidad a través del reconocimiento dactilar.

Una vez realizada la verificación dactilar, el usuario debe ingresar el PIN asignado durante el registro de usuario. Si la verificación fue satisfactoria podrá ingresar en el menú de registro y actualizar la información.



INFORMACIÓN DE USUARIO

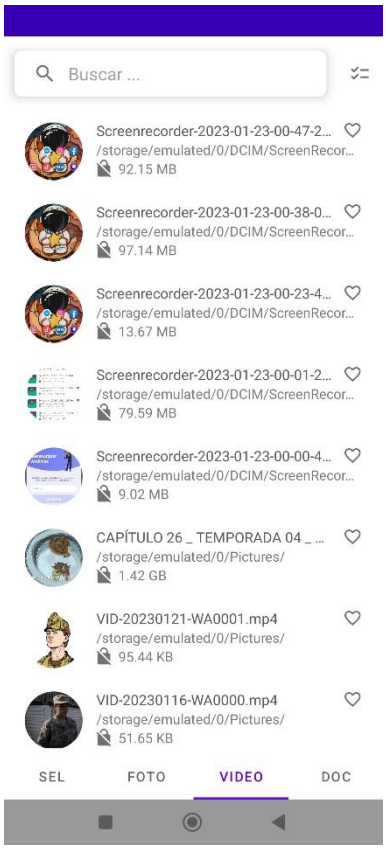
Al ingresar en el Menú Principal el usuario podrá observar un panel con información general del usuario, no se incluyen datos privados como contraseñas.

MENÚ PRINCIPAL

La aplicación mostrará un menú de opciones en el cual se podrá elegir si el usuario necesita encriptar archivos, restablecer archivos o si desea cerrar la sesión de la cuenta local.

CONTENIDO MULTIMEDIA

Al ingresar en la opción “Buscar archivos” el usuario podrá visualizar tres secciones que incluyen un tipo de contenido multimedia; fotos, videos y documentos. Adicionalmente la aplicación incluye el panel principal en el que el usuario puede observar de manera más eficiente los datos seleccionados sin necesidad de buscar en toda la lista de archivos. Al hacer clic en uno de los ítems el sistema agrega automáticamente el dato a una lista de espera, si desea deseleccionar el dato solo haga clic en el mismo ítem o desde el panel principal.



CIFRAR DATOS

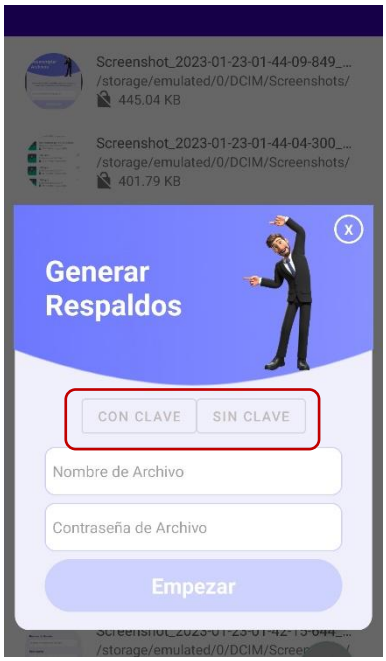
Una vez que el usuario seleccionó en el panel de Opciones la primera opción, podrá visualizar un cuadro de confirmación antes de empezar con el procedimiento, de manera que si el usuario desea cancelar la operación lo puede hacer cerrando el cuadro de diálogo. Los datos encriptados ya no serán visibles, y solo se podrán visualizar desde la aplicación.

DESCIFRAR DATOS

La aplicación requiere que para descifrar un archivo encriptado se ingrese en primer lugar la clave de encriptación del usuario, de manera que sin esa clave no se puede descifrar la información.

RESPALDO DE ARCHIVOS CIFRADOS

Esta sección permitirá a los usuarios cifrar los archivos e incluirlos en un archivo con extensión .zip. Así mismo puede crear el archivo de datos cifrados, pero con una contraseña de seguridad, por lo tanto, para descomprimir el archivo necesitará primero ingresar la clave de seguridad que el usuario asignó.



SERVICIO SENSOR

El usuario puede activar el mecanismo desde el Menú Principal cuando sea conveniente, los sensores integrados en el dispositivo estarán recopilando datos hasta que el usuario efectué un patrón de movimiento específico, esta es la orden para iniciar con los subprocesos que incluye el servicio, es decir encriptar sus archivos personales (fotos, documentos), los videos no se incluyen.


10:19 AM
54%
Menú de Opciones
Monitoreo de Sensores
off
Alejandro Mejia
Usuario: samj25121
Buscar Archivos
Restaurar Archivos
Cerrar Sesión
Actualizar Datos