Universidad Politécnica de Chiapas

Ingeniería en Tecnologías de la Información e Innovación Digital

[Programacion Para moviles]

[C2 - A3 - EncryptedSharedPreferences]

[Nomenclatura del nombre de archivo C2 - A3 - EncryptedSharedPreferences-223216-Daniel Peregrino Perez.pdf]

[Alumno -Daniel Peregrino Perez] - [223216]

Docente: [José Alonso Macias Montoya]

Fecha de entrega: [17/06/2025]

1. DESCRIPCIÓN DE LA ACTIVIDAD

1.1. Enunciado del problema

Desarrollar una aplicación móvil nativa para Android desde cero utilizando Android Studio y el lenguaje Kotlin. El objetivo principal de la aplicación es ser capaz de almacenar de forma segura y persistente diversas configuraciones del usuario. La funcionalidad crítica es que estos datos (como nombre de usuario, preferencias de tema, etc.) deben ser guardados de manera encriptada para proteger la información de accesos no autorizados, incluso si el dispositivo es comprometido (por ejemplo, rooteado).

1.2. Objetivos de aprendizaje

- Comprender el funcionamiento y las limitaciones de las SharedPreferences estándar de Android.
- Integrar la librería de seguridad de AndroidX (androidx.security:security-crypto) en un proyecto de Android Studio.
- Implementar EncryptedSharedPreferences para crear un archivo de preferencias seguro.
- Dominar la escritura y lectura de diferentes tipos de datos (Strings, Booleans, Integers) en un archivo de preferencias encriptado.
- Verificar y demostrar que los datos almacenados en el dispositivo están efectivamente encriptados, utilizando el Device File Explorer de Android Studio.
- Diagnosticar y solucionar errores comunes, como la falta de dependencias o problemas de configuración del SDK mínimo requerido por la librería.

Definición clara

El problema a resolver es la implementación completa de un sistema de almacenamiento local seguro para datos de configuración en Android. Esto implica no solo la codificación de la interfaz de usuario para capturar los datos, sino también la correcta implementación de la librería de criptografía de AndroidX para garantizar que los datos persistan de forma encriptada, protegiendo la privacidad del usuario en cualquier estado de la aplicación.

Objetivos específicos

- Ser capaz de agregar la dependencia androidx.security:security-crypto al proyecto.
- Ser capaz de escribir el código Kotlin necesario para inicializar EncryptedSharedPreferences usando una clave maestra (MasterKey).
- Ser capaz de guardar y leer datos de la interfaz de usuario de forma encriptada.
- Ser capaz de verificar la encriptación del archivo .xml resultante en el sistema de archivos del dispositivo.
- Ser capaz de demostrar la diferencia entre un archivo de SharedPreferences normal (texto plano) y uno encriptado.

2. FUNDAMENTOS TEÓRICOS

SharedPreferences es una API del framework de Android que permite guardar y recuperar pares de datos clave-valor de tipos primitivos. Es ideal para almacenar preferencias de usuario o datos de configuración simples. Sin embargo, su principal desventaja es que, por defecto, almacena

los datos en un archivo XML en texto plano, haciéndolo vulnerable a ser leído por cualquier persona con acceso al sistema de archivos del dispositivo.

EncryptedSharedPreferences es una clase proporcionada por la librería de seguridad de AndroidX. Actúa como una envoltura (wrapper) sobre la clase SharedPreferences estándar, pero añade una capa de encriptación automática. Utiliza el **Android Keystore System** para generar y almacenar de forma segura las claves criptográficas, asegurando que ni siquiera la propia aplicación pueda extraerlas.

Componentes clave:

- **MasterKey:** Es la clave principal que se utiliza para encriptar las subclaves y los datos. Se genera y almacena de forma segura en el Keystore de Android, lo que la hace extremadamente difícil de comprometer.
- **Esquemas de encriptación:** EncryptedSharedPreferences permite configurar cómo se encriptan tanto las claves (PrefKeyEncryptionScheme) como los valores (PrefValueEncryptionScheme), utilizando algoritmos robustos como AES256-SIV y AES256-GCM.
- **Permisos:** A diferencia de otras funcionalidades, el uso de EncryptedSharedPreferences no requiere ningún permiso especial en el AndroidManifest.xml, ya que opera sobre el almacenamiento interno privado de la aplicación.

---

3. DESARROLLO DE LA ACTIVIDAD

3.1. Desarrollo

El proceso se dividió en cuatro fases principales:

1. **Configuración del Proyecto en Android Studio:** Se modificó el archivo build.gradle.kts (Módulo: app) para añadir la dependencia de la librería de seguridad de AndroidX.
    - implementation("androidx.security:security-crypto:1.0.0")
2. **Diseño de la Interfaz de Usuario:** Se creó un layout en activity_main.xml con varios componentes para que el usuario pudiera introducir y visualizar los datos: EditText para el nombre de usuario e idioma,

un SwitchMaterial para el tema oscuro, un SeekBar para el volumen y Button para las acciones de guardar y cargar.

3. **Implementación del Código Kotlin:**

    - Se creó una instancia de EncryptedSharedPreferences. Primero, se generó una MasterKey utilizando MasterKeys.getOrCreate() y luego se usó para inicializar EncryptedSharedPreferences con un nombre de archivo y los esquemas de encriptación.

    - Se implementó la función saveData(), que obtiene los valores de la UI, abre el editor de preferencias (.edit()), guarda cada dato con su respectiva clave (e.g., putString(), putBoolean()) y aplica los cambios con .apply().

    - Se implementó la función loadData(), que lee cada valor desde EncryptedSharedPreferences (e.g., getString(), getBoolean()) proporcionando un valor por defecto, y actualiza la UI con los datos recuperados.

4. **Pruebas y Depuración:**

    - Se ejecutó la aplicación, se introdujeron datos y se guardaron.

    - Se utilizó la herramienta **Device File Explorer** de Android Studio para navegar a data/data/<nombre_del_paquete>/shared_prefs/.

    - Se abrió el archivo XML generado (secure_user_prefs.xml) y se verificó que tanto las claves como los valores eran cadenas de texto ilegibles y encriptadas, demostrando el éxito de la implementación.

    - Se realizó una contraprueba generando un archivo con SharedPreferences normales para visualizar la diferencia y el riesgo de almacenar datos en texto plano.

build.gradle.kts (:app) | activity_main.xml | MainActivity.kt | secure_user_prefs.xml

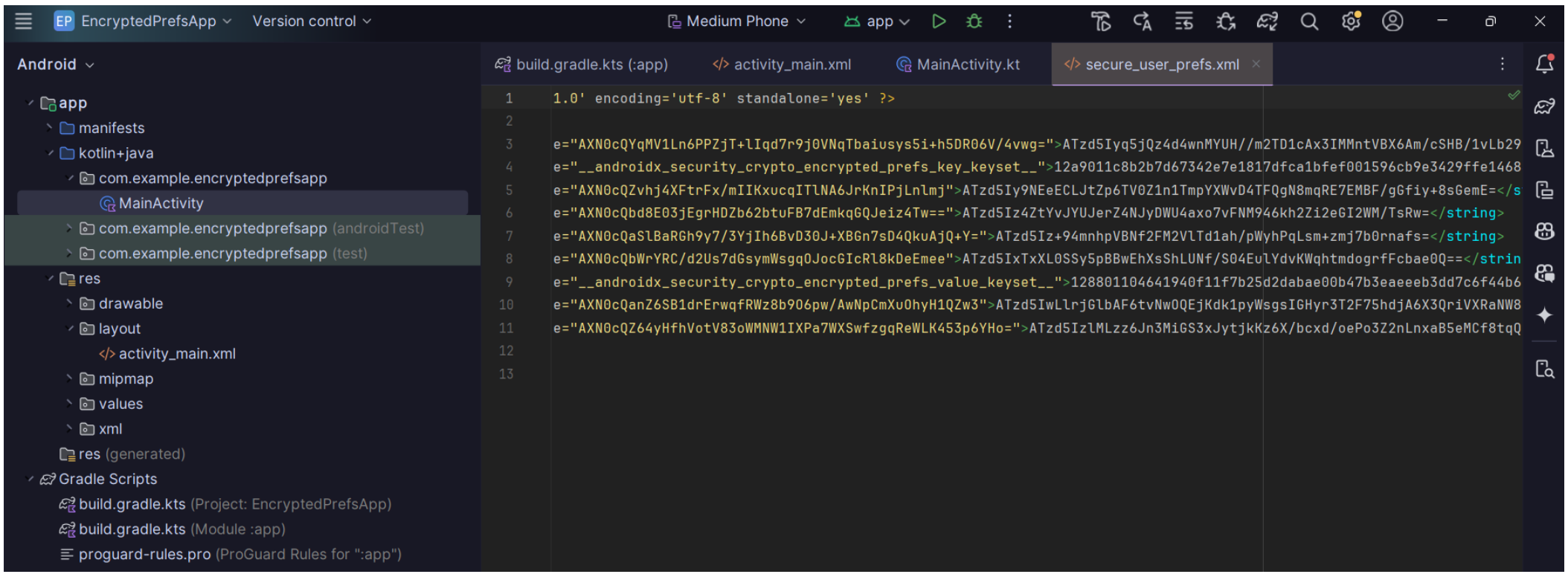
```xml
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:padding="16dp">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Configuraciones a Guardar"
            android:textSize="20sp"
            android:textStyle="bold" />

        <EditText
            android:id="@+id/etNombreUsuario"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="16dp"
            android:hint="Nombre del usuario" />

        <com.google.android.material.switchmaterial.SwitchMaterial
            android:id="@+id/switchTemaOscuro"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="8dp"
            android:text="Tema oscuro" />
```

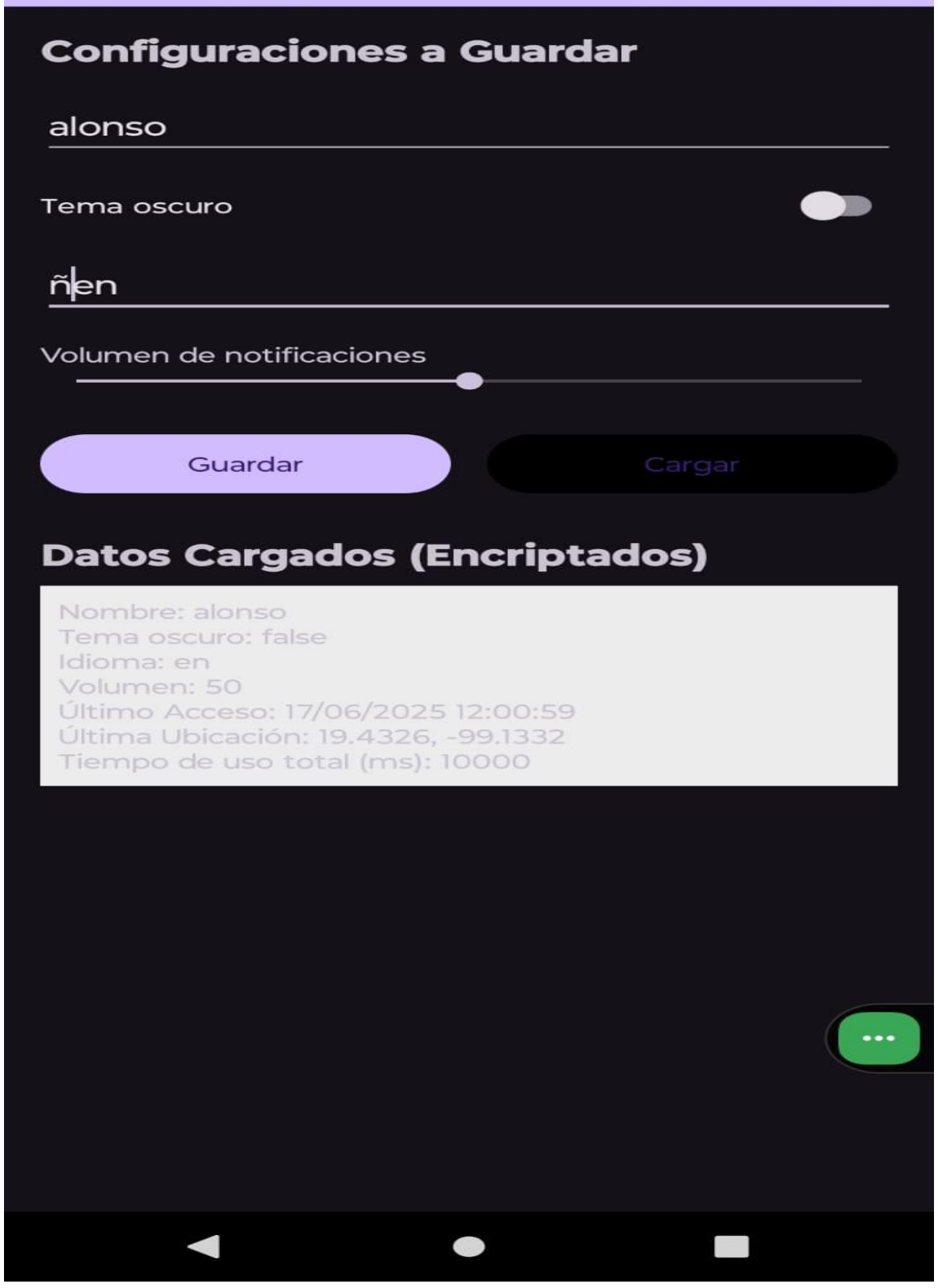
```kotlin
rivate fun initEncryptedSharedPreferences() {
    // 1. Crear o recuperar la clave maestra. Esta clave se almacena de forma segura en el Android Keystore.
    val masterKeyAlias = MasterKeys.getOrCreate(MasterKeys.AES256_GCM_SPEC)

    // 2. Inicializar EncryptedSharedPreferences
    encryptedPrefs = EncryptedSharedPreferences.create(
        PREFS_FILE_NAME,
        masterKeyAlias,
        context: this, // context
        EncryptedSharedPreferences.PrefKeyEncryptionScheme.AES256_SIV,
        EncryptedSharedPreferences.PrefValueEncryptionScheme.AES256_GCM
    )
```

4. RESULTADOS

4.1. Resultados obtenidos

- Se ha construido exitosamente una aplicación Android funcional que se integra con la librería de seguridad de AndroidX para el almacenamiento de datos.
- La aplicación es capaz de guardar y recuperar configuraciones de usuario de forma persistente y segura.
- Se ha verificado visualmente que los datos almacenados en el dispositivo están correctamente encriptados, cumpliendo con el requisito principal de la actividad.
- Se ha implementado correctamente el flujo de escritura y lectura, gestionando diferentes tipos de datos primitivos.
- El proyecto no presentó errores de ejecución y demostró ser una solución robusta para el almacenamiento local seguro.


EncryptedPrefsApp
Version control
Medium Phone
app
Android
build.gradle.kts (:app)
activity_main.xml
MainActivity.kt
secure_user_prefs.xml
app
manifests
kotlin+java
com.example.encryptedprefsapp
MainActivity
com.example.encryptedprefsapp (androidTest)
com.example.encryptedprefsapp (test)
res
drawable
layout
activity_main.xml
mipmap
values
xml
res (generated)
Gradle Scripts
build.gradle.kts (Project: EncryptedPrefsApp)
build.gradle.kts (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
1.0' encoding='utf-8' standalone='yes' ?>
e="AXN0cQYqMV1Ln6PPZjT+lIqd7r9j0VNqTbaiusys5i+h5DR06V/4vwg=">ATzd5Iyq5jQz4d4wnMYUH//m2TD1cAx3IMMntVBX6Am/cSHB/1vLb29
e="__androidx_security_crypto_encrypted_prefs_key_keyset__">12a9011c8b2b7d67342e7e1817dfca1bfef001596cb9e3429ffe1468
e="AXN0cQZvhj4XFtrFx/mIIKxucqITlNA6JrKnIPjLnlmj">ATzd5Iy9NEeECLJtZp6TV0Z1n1TmpYXWvD4TFQgN8mqRE7EMBF/gGfiy+8sGemE=</s
e="AXN0cQbd8E03jEgrHDZb62btuFB7dEmkqGQJeiz4Tw==">ATzd5Iz4ZtYvJYUJerZ4NJyDWU4axo7vFNM946kh2Zi2eGI2WM/TsRw=</string>
e="AXN0cQaSlBaRGh9y7/3YjIh6BvD30J+XBGn7sD4QkuAjQ+Y=">ATzd5Iz+94mnhpVBNf2FM2VlTd1ah/pWyhPqLsm+zmj7b0rnafs=</string>
e="AXN0cQbWrYRC/d2Us7dGsymWsgqOJocGIcRl8kDeEmee">ATzd5IxTxXL0SSy5pBBwEhXsShLUNf/S04EulYdvKWqhtmdogrfFcbae0Q==</strin
e="__androidx_security_crypto_encrypted_prefs_value_keyset__">128801104641940f11f7b25d2dabae00b47b3eaeeeb3dd7c6f44b6
e="AXN0cQanZ6SB1drErwqfRWz8b9O6pw/AwNpCmXuOhyH1QZw3">ATzd5IwLlrjGlbAF6tvNwOQEjKdk1pyWsgsIGHyr3T2F75hdjA6X3QriVXRaNW8
e="AXN0cQZ64yHfhVotV83oWMNW1IXPa7WXSwfzgqReWLK453p6YHo=">ATzd5IzlMLzz6Jn3MiGS3xJytjkKz6X/bcxd/oePo3Z2nLnxaB5eMCf8tqQ


12:10
Configuraciones a Guardar
alonso
Tema oscuro
ñen
Volumen de notificaciones
Guardar
Cargar
Datos Cargados (Encriptados)
Nombre: alonso
Tema oscuro: false
Idioma: en
Volumen: 50
Último Acceso: 17/06/2025 12:00:59
Última Ubicación: 19.4326, -99.1332
Tiempo de uso total (ms): 10000

5. CONCLUSIONES

Aprendizaje Adquirido: El desarrollo de esta aplicación ha permitido consolidar conocimientos fundamentales sobre la persistencia de datos en Android y, más importante aún, sobre la seguridad de dicha información. Se ha comprendido la arquitectura de EncryptedSharedPreferences, el rol crucial de la MasterKey y el Android Keystore, y la importancia de no almacenar nunca datos sensibles en texto plano. La habilidad de verificar la encriptación directamente en el sistema de archivos es una lección práctica invaluable para el desarrollo de aplicaciones robustas y seguras.

Relación con los Objetivos: Todos los objetivos de aprendizaje planteados fueron cumplidos. Se logró configurar la dependencia, implementar la lógica de encriptación/desencriptación y probar exitosamente que el sistema funciona como se esperaba. La experiencia práctica de ver un archivo XML ilegible en contraste con uno legible fue fundamental para consolidar el concepto de seguridad de datos en el cliente.

---

6. DIFICULTADES Y SOLUCIONES

- **Identificación del problema (Dificultad encontrada):**
  Al intentar ejecutar la aplicación en un emulador o dispositivo con un nivel de API bajo (ej. API 21), la aplicación fallaba al iniciar con un error relacionado con la librería de seguridad.

- **Análisis de causas (Investigación de los factores que originaron el problema):**
  El problema se originó porque la librería androidx.security:security-crypto tiene un requisito de versión mínima del SDK. Específicamente, requiere que la aplicación tenga un minSdk de 23 (Android 6.0 Marshmallow) o superior para poder funcionar, ya que depende de características del Keystore introducidas en esa versión.

- **Implementación de solución (Aplicación de estrategias para resolver la dificultad):**
  Se modificó el archivo build.gradle.kts del módulo de la aplicación. Se ajustó el valor de minSdk dentro del bloque defaultConfig a 23.

---

7. REFERENCIAS

Google Developers. (s.f.). Almacenar datos clave-valor de forma segura. Android Developers. Obtenido de https://developer.android.com/training/data-storage/shared-preferences?hl=es#kts

Google Developers. (s.f.). EncryptedSharedPreferences class reference. Android Developers. Obtenido de https://developer.android.com/reference/androidx/security/crypto/EncryptedSharedPreferences

Google Developers. (s.f.). Introducción a la biblioteca de seguridad. Android Developers. Obtenido de https://developer.android.com/topic/security/data?hl=es