2025

PROYECTO CUIDA+

Proyecto de fin de curso en Desarrollo de aplicaciones multiplataforma

Elena Navarro Guzmán

ÍNDICE

1. Introducción y Objetivos

Muchas personas, especialmente en el entorno familiar, asumen el cuidado diario de un ser querido que, por edad o enfermedad, ya no puede gestionar su salud por sí mismo. Esta responsabilidad suele recaer en cuidadores no profesionales: hijos, hijas, hermanos o incluso vecinos, que hacen lo posible por estar pendientes de citas médicas, tratamientos o medicamentos. Sin embargo, en medio de la rutina diaria, el trabajo y las obligaciones, es fácil olvidar una dosis o una cita médica, lo que puede tener graves consecuencias para la persona dependiente.

Cuida+ nace precisamente de esa realidad. Se trata de una aplicación móvil, desarrollada en el marco del **Ciclo Formativo de Grado Superior en Desarrollo de Aplicaciones Multiplataforma**, pensada específicamente para los cuidadores. No busca ser una solución compleja, sino una herramienta sencilla y útil que permita centralizar toda la información médica relevante de los pacientes a cargo: medicamentos, tratamientos y citas médicas, facilitando la colaboración entre distintas personas implicadas en el cuidado.

La idea surge al observar cómo, en muchas familias, la información médica queda dispersa en papeles, notas del móvil o en la memoria. Esto genera estrés adicional en los cuidadores, que deben dedicar tiempo a buscar información en diferentes lugares, aumentando la probabilidad de errores. Cuida+ centraliza esta información de forma organizada y accesible en cualquier momento y lugar, aliviando esta carga mental y permitiendo una gestión mucho más eficiente y segura del cuidado de los pacientes.

Además, permite crear grupos familiares, registrar pacientes, gestionar fácilmente tareas médicas y evitar depender exclusivamente de la memoria. La herramienta está diseñada con un enfoque especialmente práctico y realista, adaptada a la vida cotidiana y a las capacidades técnicas básicas de cualquier usuario.

Este proyecto representa no solo un importante reto técnico y formativo, sino también un ejercicio de aplicación práctica y útil de los conocimientos adquiridos durante el ciclo formativo. Supone la integración efectiva de diversas tecnologías modernas, incluyendo el desarrollo con Kotlin y Jetpack Compose, gestión de bases de datos y autenticación en Firebase, y buenas prácticas en arquitectura y diseño. Todo esto hace que Cuida+ sea un proyecto ambicioso, pero claramente definido, enfocado en resolver problemas reales y cotidianos con soluciones tecnológicas sencillas pero eficaces.

El objetivo principal es desarrollar una aplicación móvil que ayude a los cuidadores a organizar y centralizar la información médica de las personas a su cargo, ofreciendo una herramienta práctica y accesible para el control eficiente de citas médicas, tratamientos y medicamentos, minimizando errores u olvidos.

Para alcanzar este objetivo, se han planteado las siguientes metas concretas:

- Proporcionar una **interfaz sencilla e intuitiva**, diseñada específicamente para usuarios sin conocimientos técnicos avanzados, garantizando una experiencia amigable y efectiva en el uso diario.
- Facilitar la **creación y gestión de grupos familiares** para administrar uno o varios pacientes, manteniendo toda la información centralizada y claramente accesible para múltiples cuidadores simultáneamente.
- Ofrecer un **registro claro y organizado de citas médicas, tratamientos y medicamentos**, incluyendo detalles esenciales como fechas, horas, motivos, dosis, frecuencia, vía de administración y duración.
- Implementar funcionalidades adicionales como **marcar enfermedades como pasadas** para mantener un historial preciso y detallado de la evolución médica de cada paciente.
- **Asegurar la seguridad y privacidad de la información** mediante mecanismos robustos de autenticación y protección de datos personales y médicos, garantizando la confianza y tranquilidad de los usuarios.

La implementación técnica se llevará a cabo mediante **Kotlin con Jetpack Compose**, una tecnología moderna que permite desarrollar interfaces rápidas y adaptativas. **Firebase** será utilizado como base de datos en la nube, permitiendo una sincronización segura y constante entre dispositivos, facilitando el acceso inmediato y confiable a la información.

En definitiva, Cuida+ no solo busca ser una aplicación funcional desde el punto de vista tecnológico, sino también una herramienta empática y humana, que aporte tranquilidad y confianza a quienes día a día cuidan de otras personas.

2. Especificación de Requisitos

A. FUNCIONALES

Cuida+ proporciona a los cuidadores un conjunto de herramientas prácticas para organizar y gestionar la información médica de personas dependientes. El acceso a la aplicación se realiza mediante un sistema de autenticación con cuentas personales, desde las cuales cada usuario puede crear un nuevo grupo familiar o unirse a uno ya existente utilizando un nombre único que identifica al grupo. Esta funcionalidad facilita que varios cuidadores compartan la responsabilidad sobre los mismos pacientes, favoreciendo la colaboración entre familiares u otros responsables del cuidado.

Una vez dentro del grupo, los cuidadores pueden registrar pacientes, introduciendo sus datos personales como el nombre completo, la fecha de nacimiento y la dirección. Esta funcionalidad básica permite tener identificados claramente a todos los miembros atendidos dentro del grupo familiar.

Cada paciente puede tener asociadas diversas citas médicas, especificando para cada una de ellas la fecha, la hora, el lugar de atención y el motivo. Este registro permite mantener un historial ordenado y accesible de todas las consultas médicas, revisiones o intervenciones, ayudando al cuidador a tener siempre presente los próximos eventos y también a consultar lo que ya ha sucedido.

Además, Cuida+ contempla la gestión de enfermedades diagnosticadas. Las enfermedades no se introducen manualmente, sino que se seleccionan a partir de una lista precargada obtenida de una colección en la base de datos. Esta colección centralizada permite unificar criterios y evitar errores de escritura, garantizando que todas las enfermedades estén registradas de forma coherente y estructurada. Una vez seleccionada la enfermedad, esta puede ser registrada con el paciente correspondiente y, si es necesario, marcada como superada. Esta opción permite mantener un control claro sobre la evolución médica del paciente, distinguiendo entre patologías activas y pasadas.

A partir de cada enfermedad registrada, se pueden vincular tratamientos no farmacológicos (como fisioterapia, cambios en la dieta, terapias específicas, etc.) y/o medicamentos. Tanto los tratamientos como los medicamentos se seleccionan desde listas precargadas, de la misma manera que las enfermedades. Esta estrategia evita introducir información libremente, asegurando uniformidad en los datos y facilitando futuras estadísticas o filtrados. Esto permite adaptar la información a diferentes realidades clínicas: desde enfermedades crónicas que requieren seguimiento continuo hasta tratamientos temporales que deben ejecutarse por un periodo concreto.

Tanto tratamientos como medicamentos se vinculan siempre a una enfermedad específica para mantener una organización clara y coherente. De cada uno se registran datos como el nombre del fármaco, la dosis prescrita, la frecuencia (por ejemplo, cada 8 horas), la vía de administración (oral, tópica, inyectable, etc.) y la duración estimada del tratamiento. Esta estructura permite que el cuidador tenga toda la información relevante a mano para administrar correctamente la medicación.

Todos los datos almacenados en la aplicación pacientes, citas médicas, enfermedades, tratamientos y medicamentos pueden ser eliminados por los usuarios. Solo los datos personales del paciente y del perfil de usuario son directamente editables; el resto de la información se actualiza eliminando el elemento anterior y creando uno nuevo con los datos modificados.

La interfaz de usuario ha sido diseñada para ofrecer una visualización clara y ordenada de

toda esta información. Cada paciente cuenta con una pantalla de detalle en la que se muestran sus datos personales junto con secciones desplegables para visualizar sus enfermedades activas. Dentro de cada enfermedad, se organizan los tratamientos y la medicación mediante tarjetas colapsables, lo que permite al cuidador revisar rápidamente cualquier apartado sin sobrecargar la pantalla.

Además, los datos personales del paciente pueden actualizarse directamente desde esta misma pantalla, mediante un botón de edición que abre un diálogo rápido para cambiar el nombre, la dirección o la fecha de nacimiento. De este modo, el acceso y la modificación de información básica se realizan de forma sencilla y accesible.

B. NO FUNCIONALES

Además de las funcionalidades visibles para el usuario, Cuida+ cumple ciertos requisitos técnicos y de diseño esenciales para asegurar su correcto funcionamiento, facilidad de uso y protección de los datos sensibles gestionados.

En primer lugar, la **usabilidad** es clave en el desarrollo. La aplicación está pensada especialmente para usuarios adultos, posiblemente con cargas familiares y laborales importantes, que necesitan una **herramienta sencilla, rápida e intuitiva**. La interfaz evita elementos innecesarios o complicados y ofrece una experiencia fluida y clara, facilitando que cualquier acción se realice con el mínimo esfuerzo técnico posible.

La **seguridad** es otro aspecto primordial. Al gestionar datos sensibles relacionados con información médica y personal, la protección adecuada de estos datos es crítica. Cuida+ implementa un sistema seguro de autenticación mediante cuentas personales, y utiliza Firebase como base de datos en la nube, aprovechando sus mecanismos integrados de seguridad y restricción de acceso. De esta manera, solo usuarios autorizados dentro del grupo familiar tienen acceso a la información médica almacenada.
En términos de fiabilidad y estabilidad, la aplicación está diseñada para funcionar continuamente sin errores significativos en dispositivos Android, incluso aquellos de gama

media o con recursos limitados. Las operaciones básicas como consultar datos, añadir información o editar registros, se realizan con rapidez y fluidez, minimizando posibles retardos perceptibles por el usuario.

La aplicación garantiza además la persistencia y sincronización de datos en tiempo real. Esto significa que cuando varios cuidadores acceden a la aplicación desde diferentes dispositivos, los cambios realizados por uno de ellos se actualizan inmediatamente en el resto. Esto es esencial para asegurar que todos los cuidadores trabajen siempre sobre la información más reciente y precisa disponible.

Finalmente, aunque el enfoque principal de Cuida+ es concreto y limitado en cuanto a funcionalidades para garantizar claridad y sencillez, la arquitectura implementada es modular y escalable. Esto permite que, en futuras versiones, se pueda ampliar fácilmente con nuevas funcionalidades como recordatorios automáticos, notificaciones o funcionalidades avanzadas como un calendario visual completo sin necesidad de realizar cambios profundos en la estructura ya creada.

Con estos requisitos no funcionales definidos claramente, Cuida+ ofrece no solo herramientas prácticas, sino también una aplicación segura, confiable, fácil de usar y adaptada específicamente a las necesidades diarias de las personas que asumen responsabilidades de cuidado.

3. Planificación Temporal y Evaluación de Costes

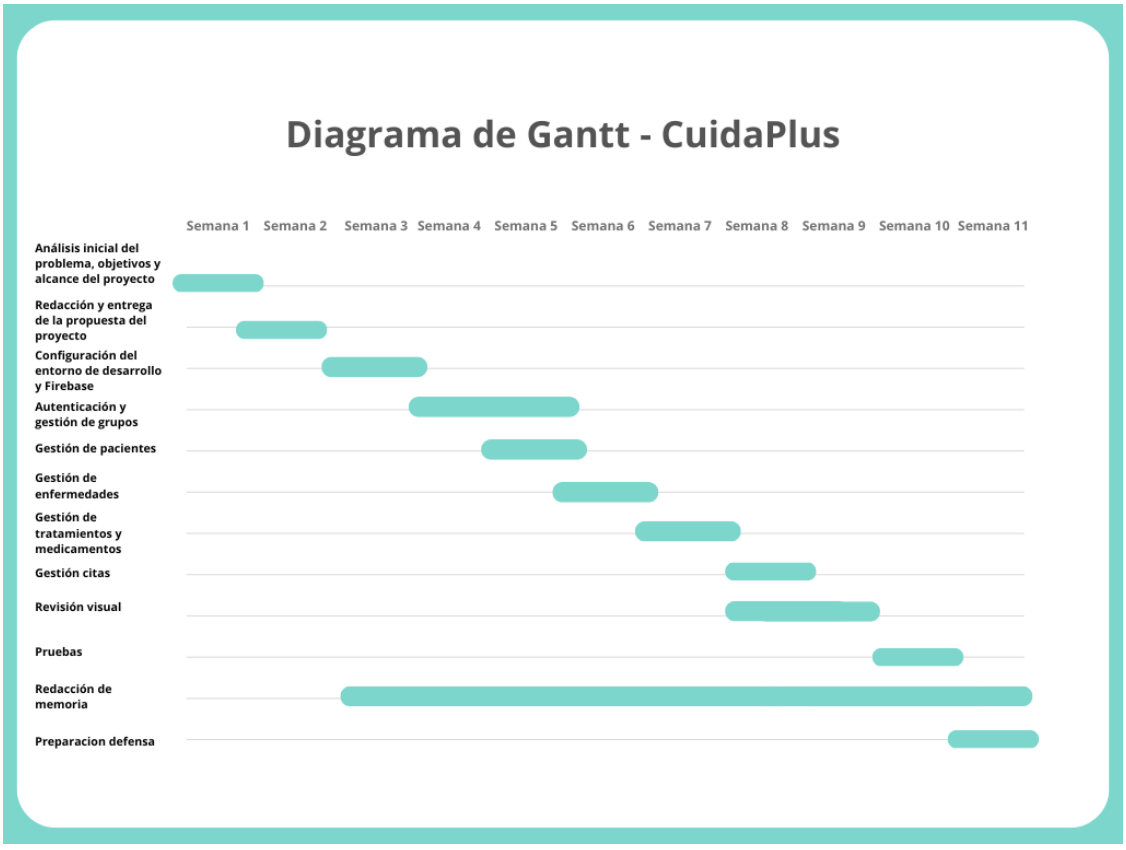
A. PLANIFICACIÓN TEMPORAL

Antes de comenzar el desarrollo técnico de Cuida+, fue necesario establecer una **planificación estructurada** que permitiera distribuir el trabajo de manera equilibrada en el tiempo disponible. Esta organización temporal ha sido clave para avanzar de forma ordenada, respetando los plazos del módulo de Proyecto y asegurando que todas las funcionalidades esenciales estuvieran implementadas y probadas a tiempo.

La planificación se elaboró en base a semanas de trabajo, asignando a cada una tareas específicas según su prioridad, complejidad y dependencia de otras fases. Se adoptó una metodología de desarrollo secuencial, pero flexible, permitiendo que algunas tareas se solaparan cuando fue necesario para optimizar recursos y avanzar en paralelo.

Además, se tuvo en cuenta que el desarrollo debía convivir con otras actividades del ciclo formativo, así como con la elaboración progresiva de esta memoria. Por ello, algunas tareas relacionadas con la documentación se realizaron en paralelo a las de programación.

A continuación, se presenta un **diagrama tipo Gantt** que refleja visualmente esta planificación.



El desarrollo del proyecto Cuida+ se ha llevado a cabo a lo largo de 11 semanas, comenzando con la fase de análisis previa a la aceptación oficial de la propuesta el 7 de abril y finalizando con la entrega del proyecto el 1 de junio. La planificación se estructuró por tareas

semanales, abordando de forma progresiva desde la configuración técnica inicial hasta la gestión de funcionalidades clave como pacientes, enfermedades, tratamientos y citas médicas.

A lo largo de estas 11 semanas se ha mantenido un ritmo constante de dedicación, estimando un promedio de entre 10 y 12 horas semanales. Esta carga de trabajo ha sido distribuida entre programación, diseño, pruebas y documentación. Por lo que en horas estimadas de trabajo serian entre 110 y 130 horas. La organización semanal permitió mantener un ritmo constante y alcanzar los objetivos marcados con margen suficiente para realizar pruebas y mejoras visuales.

B. EVALUACIÓN DE COSTES

El desarrollo del proyecto Cuida+ no ha supuesto costes económicos directos, ya que se ha realizado como parte del módulo de Proyecto del Ciclo Formativo de Grado Superior en Desarrollo de Aplicaciones Multiplataforma. No obstante, es importante valorar los recursos técnicos empleados, así como estimar el esfuerzo invertido, tanto en tiempo como en dedicación, para entender el alcance real del trabajo realizado.

A nivel de herramientas y tecnología, se han utilizado entornos y servicios gratuitos que ofrecen potentes funcionalidades sin coste para el desarrollador individual. El entorno Android Studio permitió programar en Kotlin con Jetpack Compose, una librería moderna que facilita el desarrollo de interfaces dinámicas y reactivas. Para la gestión de usuarios y almacenamiento de datos, se empleó Firebase, aprovechando sus servicios de autenticación, base de datos en la nube (Firestore), todo dentro del plan gratuito que cubre ampliamente las necesidades del proyecto. Además, se utilizó GitHub como plataforma de control de versiones, también sin coste.

Aunque no se ha realizado ninguna inversión económica directa, sí se ha dedicado una cantidad considerable de tiempo y esfuerzo personal. Se estima que el proyecto ha requerido entre 110 y 130 horas de trabajo, distribuidas en tareas de análisis, desarrollo,

diseño de interfaz, pruebas, depuración y redacción de esta memoria. Si se trasladara este tiempo a un entorno profesional, aplicando una tarifa orientativa de 15 euros por hora propia de un perfil junior en desarrollo de software, el coste estimado del desarrollo ascendería a una cifra situada entre 1.650 € y 1.950 €.

En resumen, Cuida+ ha sido posible gracias al aprovechamiento de recursos gratuitos, pero representa un trabajo de valor real, tanto por el tiempo invertido como por la complejidad de las funcionalidades implementadas. Esta valoración permite tomar conciencia del esfuerzo que implica un desarrollo completo y funcional, incluso cuando se realiza en un entorno educativo.

4. Tecnologías Utilizadas

El desarrollo del proyecto Cuida+ se ha apoyado en un conjunto de tecnologías modernas que han permitido implementar una aplicación funcional, modular y segura. A continuación, se describen una a una las tecnologías empleadas, justificando su elección y explicando su aplicación concreta dentro del proyecto.

4.1 KOTLIN

Kotlin ha sido el lenguaje de programación utilizado para el desarrollo de la aplicación Cuida+. Se trata de un lenguaje moderno y seguro, desarrollado por JetBrains y adoptado oficialmente por Google como lenguaje preferente para el desarrollo de aplicaciones Android. Kotlin combina lo mejor de la programación orientada a objetos y funcional, y permite a los desarrolladores escribir menos código, más claro y con menor posibilidad de errores, lo cual resulta especialmente valioso en entornos educativos y profesionales.
Uno de los principales motivos por los que se eligió Kotlin en este proyecto fue su perfecta integración con Android Studio, el entorno oficial para el desarrollo en Android. Kotlin no requiere configuraciones adicionales para empezar a trabajar y es compatible con todas las herramientas del ecosistema Android. Esta facilidad de uso ha permitido centrar los

esfuerzos en el diseño y la lógica de la aplicación sin tener que lidiar con problemas técnicos derivados del lenguaje o el entorno.

Además, Kotlin destaca por su concisión. Permite eliminar gran cantidad de código repetitivo que, en otros lenguajes, como Java, sería obligatorio. Por ejemplo, la creación de clases de datos (data class) en Kotlin se puede realizar en una sola línea, incluyendo automáticamente constructores, métodos equals, hashCode y toString. Este tipo de ventajas ha agilizado notablemente la implementación de los modelos de datos utilizados en Cuida+, como Paciente, CitaMedica o EnfermedadPaciente, mejorando también la legibilidad y el mantenimiento del código.

Otra característica destacada es su sistema de seguridad frente a valores nulos. Los errores por referencias nulas son una de las principales causas de fallos en aplicaciones Android, y Kotlin soluciona esto de raíz mediante su sistema de tipos nulos. Esto obliga al desarrollador a manejar de forma explícita los posibles valores nulos, lo que reduce significativamente las posibilidades de sufrir errores en tiempo de ejecución. Dado que Cuida+ trabaja con datos médicos sensibles y estructuras que se cargan desde una base de datos en la nube (Firebase), esta seguridad adicional ha sido muy útil para garantizar la estabilidad del proyecto.

En el contexto de este proyecto, Kotlin ha sido utilizado para **definir toda la lógica de negocio** de la aplicación. Se ha aplicado en:

- La creación de los **modelos de datos**, representando cada entidad del sistema (usuarios, pacientes, enfermedades, citas...).
- La implementación de los **ViewModel**, que gestionan el estado de cada pantalla y la lógica asociada a formularios, listas o carga de datos.
- La **conexión con Firebase Firestore y Firebase Authentication**, incluyendo operaciones de lectura, escritura, filtrado de datos y control de acceso.
- El control de la **navegación entre pantallas**, pasando parámetros entre rutas de forma segura y tipada.

En resumen, Kotlin ha ofrecido todas las ventajas necesarias para llevar a cabo un proyecto robusto, modular y escalable. Su sintaxis limpia, su integración con Compose y Firebase, y su orientación a buenas prácticas lo convierten en una tecnología no solo adecuada, sino ideal para el desarrollo de aplicaciones móviles modernas como Cuida+. El uso de Kotlin ha contribuido de forma directa a la calidad, mantenibilidad y estabilidad del código, demostrando su valor tanto en entornos académicos como en el mundo profesional del desarrollo Android.

4.2 JETPACK COMPOSE (JPC)

Jetpack Compose es el framework moderno de diseño de interfaces de usuario (UI) desarrollado por Google para aplicaciones Android. Supone un cambio importante respecto al enfoque tradicional basado en archivos XML, ya que utiliza un sistema declarativo, en el que las interfaces se describen directamente en código Kotlin. Esta forma de trabajar facilita la creación de componentes visuales dinámicos, reactivos y adaptables a diferentes contextos, mejorando la productividad del desarrollo y la legibilidad del código.

La elección de Jetpack Compose para Cuida+ ha sido clave para construir una aplicación moderna, visualmente clara y fácil de mantener. Compose permite definir cada elemento de la interfaz como una función **@Composable**, lo que ha facilitado dividir las pantallas en pequeñas unidades reutilizables, organizadas y coherentes. Este enfoque se alinea perfectamente con las buenas prácticas de desarrollo actuales, donde la modularidad y la reutilización del código son fundamentales.

En Cuida+, Jetpack Compose ha sido utilizado para crear todas las pantallas de la aplicación, siguiendo una estructura visual unificada basada en el uso de Scaffold, con TopBar, contenido desplazable (LazyColumn) y botón de acción fijo en la parte inferior. Gracias a Compose, ha sido posible diseñar componentes visuales como:

- **Formularios limpios**, con campos alineados, validaciones visuales y experiencia de usuario fluida.

- **Listas filtrables y reutilizables**, como las que se usan para seleccionar enfermedades desde Firebase.
- **Controles visuales personalizados**, como botones con estilo propio, íconos y elementos consistentes en toda la interfaz.

Además, Compose facilita la conexión entre la interfaz y los datos mediante observación de estados (State, mutableStateOf, collectAsState), lo que ha sido fundamental para actualizar automáticamente la UI al recibir información de Firebase sin necesidad de escribir código imperativo o gestionar manualmente los ciclos de vida.

A nivel de experiencia de usuario, Compose ha aportado fluidez, consistencia y claridad visual. Las pantallas de Cuida+ mantienen una **estética limpia**, con **jerarquía visual** clara, **botones accesibles**, y componentes que se adaptan al contenido sin forzar estructuras fijas. Gracias a esto, la aplicación resulta **sencilla de usar** incluso para personas sin conocimientos técnicos, uno de los objetivos fundamentales del proyecto.

En definitiva, Jetpack Compose no solo ha sido la herramienta utilizada para crear la interfaz gráfica de Cuida+, sino también una parte esencial para alcanzar un diseño profesional, moderno y funcional. Su integración con Kotlin, Firebase y el patrón de arquitectura adoptado ha permitido desarrollar una experiencia de usuario sólida, coherente y escalable, alineada con los estándares actuales del desarrollo Android.

4.3 VIEWMODEL Y ARCHITECTURE COMPONENTS

Una parte fundamental en el desarrollo de cualquier aplicación moderna es la correcta separación de responsabilidades. Para ello, en Cuida+ se ha implementado una arquitectura basada en **MVVM** (Model–View–ViewModel), haciendo uso del componente ViewModel proporcionado por los Architecture Components de Android.

El patrón MVVM permite dividir la aplicación en tres capas bien diferenciadas:

- **Model:** representa los datos y lógica del negocio (en este caso, modelos como Paciente, CitaMedica, EnfermedadPaciente, etc.).
- **ViewModel:** gestiona la lógica y el estado de cada pantalla. Actúa como intermediario entre los datos y la interfaz de usuario.
- **View (la UI):** muestra los datos y responde a las acciones del usuario, sin contener lógica propia.

A esta estructura se suma una capa adicional, el **Repository**, que es responsable de interactuar directamente con Firebase, encapsulando las operaciones de acceso a datos y proporcionando una interfaz clara y reutilizable al ViewModel. Esta decisión refuerza la separación de capas, mejora la mantenibilidad del código y facilita las pruebas unitarias.

En Cuida+, cada pantalla cuenta con su propio ViewModel, que trabaja en conjunto con un Repository específico. Algunos ejemplos implementados en el proyecto son:

- PacienteViewModel + PacienteRepository: para registrar pacientes, actualizar datos personales, obtener listas y escuchar cambios en tiempo real.
- CitaViewModel + CitaRepository: para añadir y eliminar citas médicas, agrupándolas por fecha o por paciente.
- EnfermedadPacienteViewModel + EnfermedadPacienteRepository: para asignar enfermedades, cargar tratamientos precargados desde Firebase y mantener sincronización entre pacientes y sus patologías activas.

Este enfoque garantiza que el ViewModel nunca acceda directamente a la base de datos, sino que delegue esta responsabilidad al repositorio, lo que simplifica el código y permite reutilizar la lógica de acceso a datos en diferentes partes del proyecto.
Además, gracias al uso combinado de StateFlow, mutableStateOf y funciones suspendidas (suspend), la UI responde automáticamente a los cambios en los datos gestionados por los ViewModel. Esto ha sido fundamental para mantener la sincronización en tiempo real con Firebase sin necesidad de operaciones manuales ni recargas.

Otra ventaja clave del uso de esta arquitectura es su **escalabilidad**. La aplicación está preparada para crecer sin necesidad de reescribir grandes bloques de código. Por ejemplo, si en una futura versión se deseara sustituir Firebase por una base de datos local o una API REST externa, bastaría con modificar los repositorios, sin tocar la lógica de los ViewModel ni la interfaz.

En resumen, el uso conjunto de ViewModel, Repository y los Architecture Components ha sido esencial para estructurar Cuida+ como una aplicación limpia, profesional y preparada para escalar. Esta arquitectura modular permite mantener cada parte del sistema bien definida, facilita la reutilización y asegura una experiencia de usuario estable, fluida y reactiva en todo momento.

4.4 FIREBASE AUTHENTICATION

Para garantizar la seguridad y privacidad de los datos gestionados en Cuida+, se ha implementado un sistema de autenticación de usuarios utilizando **Firebase Authentication**, una solución proporcionada por la plataforma Firebase de Google. Esta tecnología permite gestionar de forma segura el registro, acceso y control de sesiones de los usuarios, sin necesidad de desarrollar desde cero un sistema propio ni mantener servidores dedicados.

En Cuida+ se ha optado por la autenticación mediante correo electrónico y contraseña, un método simple y accesible para cualquier tipo de usuario. Firebase se encarga de gestionar el cifrado, validación y control de sesiones, permitiendo que los usuarios permanezcan identificados entre usos de la aplicación y que puedan cerrar sesión en cualquier momento desde el menú de perfil. Toda la información relativa a la cuenta queda asociada al identificador único del usuario (UID), el cual se utiliza internamente para vincular sus datos en la base de datos y restringir el acceso.

Una de las principales ventajas de Firebase Authentication es su integración directa con Firestore y las reglas de seguridad, lo que permite que únicamente los usuarios autenticados puedan acceder a los datos correspondientes a su grupo familiar. Esta autenticación también

permite estructurar la lógica de navegación según el estado de la sesión, redirigiendo automáticamente al usuario a la pantalla principal si ya ha iniciado sesión previamente. En caso contrario, se muestra el formulario de acceso.

La combinación de Firebase Authentication con Firestore ha permitido crear un sistema seguro, escalable y fácilmente ampliable, en el que los datos están protegidos y cada usuario tiene acceso exclusivamente a la información autorizada. Este servicio, gratuito dentro del plan básico de Firebase, ha sido esencial para dotar al proyecto Cuida+ de una gestión profesional de accesos, sin añadir complejidad innecesaria al desarrollo.

4.5 FIREBASE FIRESTORE

Firebase Firestore ha sido utilizado como base de datos principal en Cuida+. Se trata de una **base de datos NoSQL en la nube**, que permite almacenar, consultar y sincronizar datos en tiempo real de forma estructurada y escalable. Su integración nativa con Android Studio y su compatibilidad directa con Kotlin lo convierten en una opción ideal para proyectos móviles que requieren rapidez, sencillez y actualización constante de la información entre múltiples usuarios.

En el caso de Cuida+, Firestore ha permitido implementar toda la lógica de almacenamiento y acceso a la información médica de los pacientes, de forma centralizada, segura y organizada. Dentro de la base de datos se han creado diversas colecciones para almacenar las entidades clave del sistema: usuarios, grupos familiares, pacientes, enfermedades, tratamientos, medicamentos y citas médicas. Cada documento representa una unidad de datos concreta y está vinculado mediante identificadores únicos que permiten establecer relaciones entre ellos, como por ejemplo, pacientes dentro de un grupo, o tratamientos asociados a una enfermedad concreta.

Firestore ha sido esencial para que múltiples cuidadores, dentro de un mismo grupo familiar, puedan acceder de forma simultánea y sincronizada a la información médica de las personas a su cargo. Gracias a su capacidad de sincronización en tiempo real, cualquier cambio

realizado por un usuario (por ejemplo, añadir una cita o eliminar un medicamento) se refleja de inmediato en el dispositivo de los demás cuidadores del grupo, sin necesidad de actualizar manualmente ni recargar la aplicación.

Además, se ha aprovechado la flexibilidad de Firestore para precargar colecciones estáticas como enfermedades, tratamientos y medicamentos, que luego son filtradas y seleccionadas desde la interfaz por parte del cuidador. Esta estrategia mejora la coherencia de los datos y evita errores de escritura o duplicidades.

Toda la comunicación con la base de datos se realiza desde repositorios específicos, que encapsulan las operaciones de lectura, escritura y escucha de cambios, respetando la arquitectura MVVM del proyecto. Los ViewModel se encargan únicamente de gestionar el estado de la pantalla y solicitar datos a los repositorios, manteniendo una separación clara de responsabilidades.

Gracias a las reglas de seguridad configuradas en Firebase, cada usuario solo tiene acceso a los documentos que pertenecen a su grupo familiar. Esto garantiza la privacidad de los datos médicos y asegura que ningún usuario pueda acceder o modificar información que no le corresponde. Además, la estructura escalable y modular de Firestore permitirá, en versiones futuras, añadir nuevas funcionalidades sin necesidad de reconstruir toda la base de datos.

```
rules_version = '2';

service cloud.firestore {
  match /databases/{database}/documents {

  //solo usuarios autenticados pueden leer o escribir
  match /usuarios/{usuarioId} {
    allow read, write: if request.auth !=null && request.auth.uid == usuarioId;
  }
  match /gruposFamiliares/{grupoId} {
    allow read, write: if request.auth != null;
  }
  match /pacientes/{pacienteId} {
    allow read, write: if request.auth != null;
  }
  match /enfermedades/{enfermedadId} {
    allow read, write: if request.auth != null;
  }
  match /enfermedadesPaciente/{docId} {
    allow read, write: if request.auth != null;
  }
  match /medicamentos_maestro/{medicamentoId} {
    allow read, write: if request.auth != null;
  }
  match /tratamientos_maestro/{tratamientoId} {
    allow read, write: if request.auth != null;
  }
  match /tratamientos/{tratamientoId} {
    allow read, write: if request.auth != null;
  }
  match /medicamentos/{medicamentoId} {
    allow read, write: if request.auth != null;
  }
  match /citas_medicas/{citaId} {
    allow read, write: if request.auth != null;
  }
  match /{document=**} {
    allow read, write: if request.auth != null;
  }

  }
}
```

En definitiva, Firebase Firestore ha sido una pieza clave en el desarrollo de Cuida+, permitiendo construir una base de datos flexible, actualizable en tiempo real, segura y perfectamente integrada con el resto de tecnologías utilizadas en la aplicación.

4.6 GIT Y GITHUB

Durante todo el desarrollo del proyecto Cuida+ se ha utilizado Git como sistema de control de versiones y GitHub como plataforma de alojamiento remoto del repositorio. Esta combinación ha permitido mantener un flujo de trabajo ordenado, seguro y profesional, facilitando tanto la organización del proyecto como el seguimiento del progreso a lo largo del tiempo.

Git ha sido fundamental para gestionar de forma eficiente los cambios realizados en el código. Desde el inicio del proyecto se creó un repositorio local donde se han ido registrando

todos los avances mediante commits frecuentes. Gracias a esta práctica, se ha podido mantener un historial completo del proyecto, identificar fácilmente cuándo y por qué se introdujo una funcionalidad concreta, y volver a versiones anteriores si era necesario. Esta capacidad de retroceso ha sido especialmente útil en las fases de prueba y refactorización, donde ha sido necesario modificar o descartar temporalmente ciertas implementaciones.

Paralelamente, se ha utilizado GitHub para alojar el repositorio de forma remota. Esto ha permitido acceder al proyecto desde diferentes dispositivos, mantener copias de seguridad automáticas y disponer de una plataforma organizada desde la que controlar el estado actual del desarrollo. GitHub también ha ofrecido una visión clara del historial del proyecto, la estructura de archivos, las ramas creadas y los cambios aplicados en cada commit. A pesar de que el desarrollo ha sido individual, GitHub ha permitido mantener una metodología profesional, como si se tratara de un proyecto en equipo.

En definitiva, Git y GitHub han sido herramientas fundamentales para el desarrollo de Cuida+, no solo por la utilidad técnica que ofrecen, sino también por su valor formativo. Su uso ha permitido aplicar buenas prácticas de desarrollo profesional, organizar el trabajo de forma clara y asegurar la trazabilidad de todo el proceso, desde la primera línea de código hasta la versión final entregada.

4.7 ANDROID STUDIO

Android Studio ha sido el **entorno de desarrollo integrado** (IDE) utilizado para programar toda la aplicación Cuida+. Se trata del **IDE oficial** para el desarrollo Android, con **soporte completo** para Kotlin, Jetpack Compose y herramientas de Firebase, lo que ha facilitado enormemente la creación, prueba y depuración del proyecto.

Desde el inicio, Android Studio permitió configurar el proyecto de forma rápida, integrando las dependencias necesarias a través de Gradle y conectando fácilmente con Firebase. Su compatibilidad con Kotlin ha sido total, ofreciendo sugerencias de código, correcciones automáticas y refactorización inteligente, lo que ha mejorado la productividad y la calidad.

Además, Android Studio cuenta con herramientas de análisis, consola de depuración y terminal integrada para gestionar versiones de Git, lo que ha facilitado el control del proyecto desde un único entorno.

En definitiva, Android Studio ha sido una herramienta completa, estable y fundamental para el desarrollo profesional de Cuida+, permitiendo trabajar con eficiencia y garantizando la calidad técnica del resultado final.

4.8 INKSCAPE

Para el diseño del logotipo oficial del proyecto Cuida+ se ha utilizado **Inkscape**, una herramienta gratuita y de código abierto especializada en **gráficos vectoriales**. Su uso ha permitido crear un logotipo escalable, limpio y adaptado a diferentes tamaños sin pérdida de calidad, ideal para integrarlo tanto en la aplicación como en la memoria del proyecto.

El logo fue diseñado con una estética sencilla y moderna, representando visualmente los valores del proyecto: **cuidado, salud y unión**. Gracias a Inkscape se ha podido exportar en formatos como PNG y SVG, facilitando su incorporación en Android Studio y en todos los materiales gráficos del trabajo.

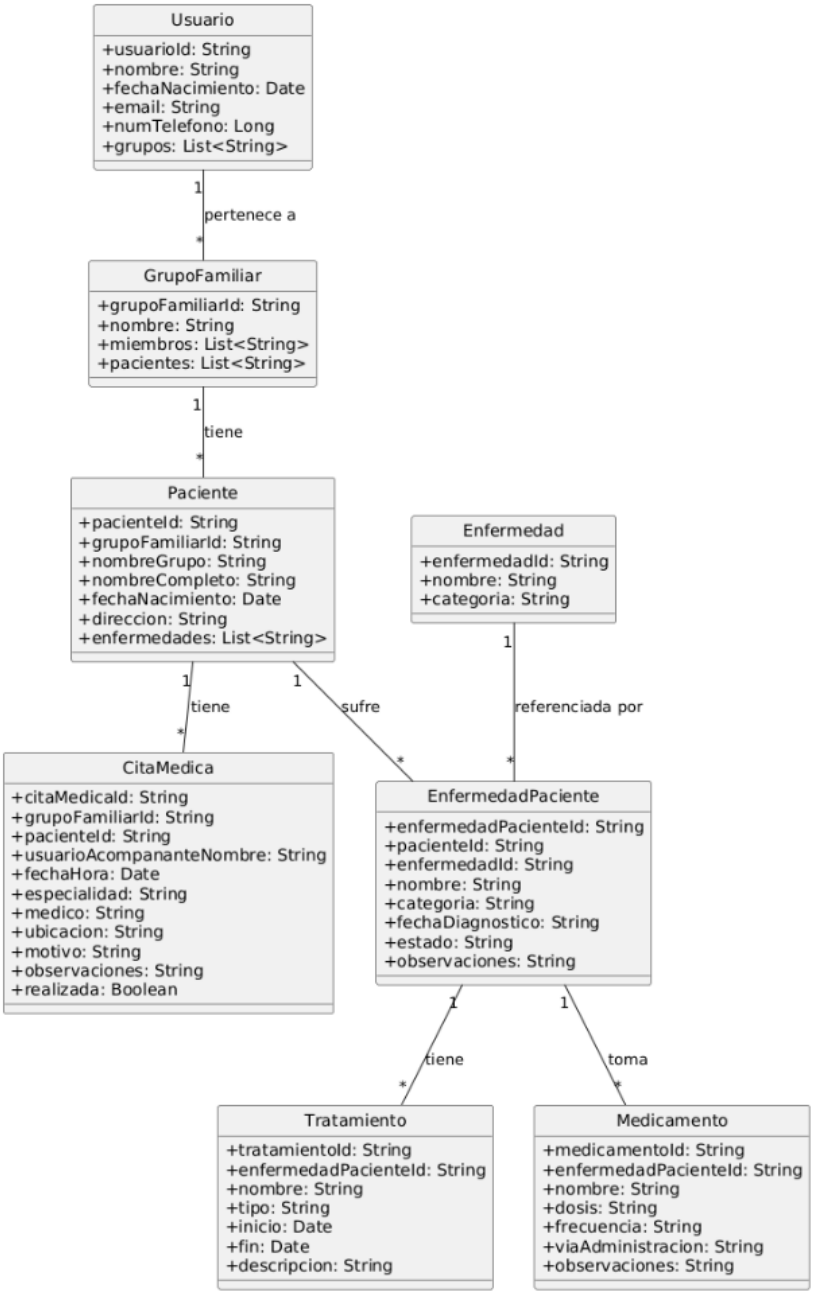
Aunque no forma parte directa del desarrollo funcional, el logotipo aporta una identidad visual clara y coherente, haciendo que la presentación del proyecto sea más profesional y cuidada.

5. Desarrollo e implementación

5.1 MODELO DE DATOS

La aplicación Cuida+ trabaja sobre un modelo de datos pensado para organizar y representar de forma clara la información médica y familiar de los pacientes. Para definir este modelo se ha seguido un enfoque basado en entidades principales relacionadas entre sí, respetando la estructura de una base de datos NoSQL como Firebase Firestore, donde no se crean relaciones estrictas como en una base relacional, pero sí se gestionan referencias lógicas entre documentos.

Como referencia para la definición del modelo se ha utilizado el siguiente diagrama de clases UML, que representa de forma visual las principales entidades del sistema, sus atributos y las relaciones lógicas entre ellas:



Usuario

Representa al cuidador que accede a la aplicación. Está autenticado mediante Firebase y, aunque en el modelo se ha contemplado la posibilidad de que pueda pertenecer a varios grupos familiares, en la versión actual de la app solo puede formar parte de un único grupo.

Al iniciar sesión, si su correo ya está vinculado a un grupo, se le redirige automáticamente a dicho grupo sin necesidad de volver a unirse.

```
data class Usuario(
    var usuarioId: String = "",
    var nombre: String = "",
    var fechaNacimiento: Date = Date(),
    var email: String = "",
    var numTelefono: Long = 0L,
    var grupos: List<String> = emptyList()
)
```

GrupoFamiliar

Es el núcleo compartido donde varios cuidadores gestionan pacientes. Cada grupo puede tener múltiples pacientes y miembros.

Recibe referencias desde Usuario (miembros) y desde Paciente.

```
data class GrupoFamiliar(
    var grupoFamiliarId: String = "",
    var nombre: String = "",
    var miembros: List<String> = emptyList(),
    var pacientes: List<String> = emptyList()
)
```

Paciente

Representa a la persona a la que se cuida. Contiene información personal y está asociada a un grupo familiar.

Pertenece a un GrupoFamiliar. Se relaciona con EnfermedadPaciente y con CitaMedica.

```
data class Paciente(
    var pacienteId: String = "",
    var grupoFamiliarId: String = "",
    var nombreGrupo: String = "",
    var nombreCompleto: String = "",
    var fechaNacimiento: Date = Date(),
    var direccion: String = "",
    var enfermedades: List<String> = emptyList(),
    var archivosAdjuntos: List<String> = emptyList()
)
```

Enfermedad

Catálogo maestro de enfermedades disponibles para asignar a un paciente.

Es referenciada desde EnfermedadPaciente.

```
data class Enfermedad(
    var enfermedadId: String = "",
    var nombre: String = "",
    var categoria: String = ""
)
```

EnfermedadPaciente

Representa una enfermedad concreta diagnosticada a un paciente, con información adicional sobre su estado.

Conecta Paciente y Enfermedad. Se relaciona con Tratamiento y Medicamento.

```
data class EnfermedadPaciente(
    var enfermedadPacienteId: String = "",
    var pacienteId: String = "",
    var enfermedadId: String = "",
    var nombre: String = "",
    var categoria: String = "",
    var fechaDiagnostico: String = "",
    var estado: String = "",
    var observaciones: String = ""
)
```

Tratamiento

Tratamiento no farmacológico asociado a una enfermedad concreta de un paciente.

Relacionado con una única EnfermedadPaciente.

```
data class Tratamiento(
    var tratamientoId: String = "",
    var enfermedadPacienteId: String = "",
    var nombre: String = "",
    var tipo: String = "",
    var inicio: Date = Date(),
    var fin: Date = Date(),
    var descripcion: String = ""
)
```

Medicamento

Medicamento administrado a un paciente como parte de una enfermedad diagnosticada.

Relacionado con una única EnfermedadPaciente.

```
data class Medicamento(
    var medicamentoId: String = "",
    var enfermedadPacienteId: String = "",
    var nombre: String = "",
    var dosis: String = "",
    var frecuencia: String = "",
    var viaAdministracion: String = "",
    var horario: List<String> = emptyList(),
    var observaciones: String = ""
)
```

CitaMedica

Cita médica programada para un paciente, con datos de fecha, médico, especialidad, etc.

Asociada a un Paciente y a su GrupoFamiliar.

```
data class CitaMedica(
    var citaMedicaId: String = "",
    var grupoFamiliarId: String = "",
    var pacienteId: String = "",
    var usuarioAcompananteNombre: String = "",
    var fechaHora: Date = Date(),
    var especialidad: String = "",
    var medico: String = "",
    var ubicacion: String = "",
    var motivo: String = "",
    var observaciones: String = "",
    var realizada: Boolean = false
)
```

TratamientoMaestro y MedicamentoMaestro

Catálogo general de tratamientos y medicamentos disponibles en la aplicación.

```
data class MedicamentoMaestro(
    var medicamentoId: String = "",
    var nombre: String = "",
    var tipo: String = ""
)
```

```
data class TratamientoMaestro(
    var tratamientoId: String = "",
    var nombre: String = "",
    var tipo: String = ""
)
```

5.2 ARQUITECTURA DE LA APLICACIÓN

Para garantizar un desarrollo limpio, escalable y mantenible, Cuida+ se ha construido siguiendo el patrón arquitectónico **MVVM (Model–View–ViewModel)**, ampliamente adoptado en el desarrollo de aplicaciones Android modernas. Este patrón permite dividir la aplicación en capas separadas con responsabilidades claramente definidas, favoreciendo la modularidad, reutilización y claridad del código.

- **Model:** esta capa incluye las entidades (data class) como Paciente, CitaMedica, Enfermedad, EnfermedadPaciente, Tratamiento, etc., así como todos los repositorios que encapsulan la lógica de acceso a datos. Los repositorios se comunican con Firebase Firestore y Authentication, y se encargan de leer, escribir, actualizar o escuchar datos en tiempo real.

- **ViewModel:** representa la capa intermedia entre la vista y los datos. Cada pantalla principal de la aplicación (gestión de pacientes, detalle de enfermedades, citas, tratamientos...) cuenta con su propio ViewModel. En ellos se define el estado de la UI y se centraliza toda la lógica de presentación. Los ViewModel reciben y exponen datos desde los repositorios, utilizando herramientas como mutableStateOf, StateFlow o LiveData para notificar a la interfaz cuando hay cambios.

- **View (UI):** construida por completo con Jetpack Compose, esta capa se encarga exclusivamente de mostrar la información y recoger las acciones del usuario. Las pantallas están estructuradas de forma modular, usando Scaffold, TopBar, listas (LazyColumn), formularios y tarjetas visuales. Cada pantalla observa el estado expuesto por su ViewModel y se actualiza automáticamente ante cualquier cambio.

Se ha seguido una estructura clara en paquetes separados por funcionalidad: view, viewmodel, repository, model. Esta organización ha permitido mantener el código ordenado, con una separación total entre la lógica de negocio y la presentación.

Cada entidad tiene su propio repositorio (PacienteRepository, CitaRepository, EnfermedadPacienteRepository, etc.). Esto permite que la lógica de acceso a datos esté centralizada y reutilizable, y que pueda modificarse sin alterar el resto de la aplicación. Por ejemplo, si en el futuro se decide sustituir Firebase por una API externa, bastará con modificar los repositorios.

La combinación de **ViewModel + estado observable + Firebase** en tiempo real ha permitido desarrollar una **aplicación completamente reactiva**. Cuando un cuidador registra una enfermedad, elimina una cita o añade un tratamiento, los cambios se reflejan al instante en todos los dispositivos del grupo familiar. Esta reactividad se ha logrado sin estructuras complejas, gracias a la integración natural entre Compose, Firebase y los observables del ViewModel.

Las ventajas que podemos observar con esta arquitectura son las siguientes:

- Claridad y mantenibilidad del código.
- Escalabilidad: nuevas funciones pueden añadirse sin afectar a las ya existentes.
- Separación total entre UI, lógica y datos.
- Reutilización de componentes visuales y funciones.
- Sincronización en tiempo real sin necesidad de recargar la app manualmente.
- Posibilidad de migrar Firebase por otra fuente de datos en el futuro sin romper la arquitectura.

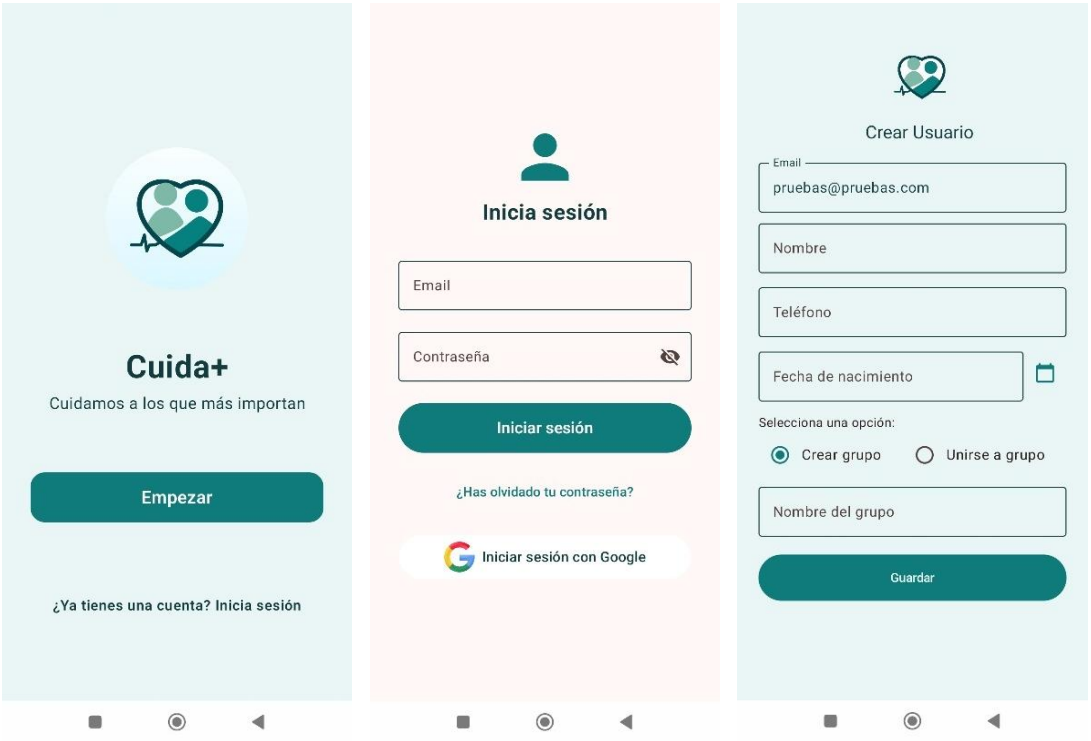
5.3 FLUJO DE AUTENTICACIÓN Y GESTIÓN DE USUARIOS

Cuida+ utiliza **Firebase Authentication** como sistema de registro y acceso de usuarios. El flujo está diseñado para ser sencillo y seguro: al iniciar la aplicación, el usuario puede registrarse con su correo y contraseña, o iniciar sesión si ya tiene cuenta. Una vez autenticado, se le solicita crear o unirse a un grupo familiar, que define el entorno en el que trabajará durante el uso de la aplicación.

El identificador único (uid) que genera Firebase para cada usuario se utiliza para asociarlo a un grupo y controlar el acceso a todos los datos Esto garantiza que solo los miembros de un mismo grupo puedan visualizar, registrar o eliminar información como pacientes, citas o enfermedades. Por diseño, únicamente los datos personales del paciente y del usuario pueden editarse directamente.

La autenticación activa se mantiene durante la sesión y se gestiona automáticamente. También se ofrece la posibilidad de cerrar sesión desde la pantalla de perfil, eliminando así el acceso a los datos hasta que se vuelva a iniciar sesión.

Este sistema de autenticación ha sido fundamental para asegurar la privacidad de los datos, limitar el acceso únicamente a cuidadores autorizados y estructurar la experiencia de usuario en torno a su grupo familiar.



5.4 GESTIÓN DE GRUPO FAMILIAR

Una vez registrado en Cuida+, el usuario debe crear un grupo familiar o unirse a uno existente. Este grupo actúa como unidad central de acceso: todos los datos que verá o podrá gestionar un cuidador están vinculados al grupo al que pertenece.

El grupo familiar contiene un identificador único (grupoFamiliarId), un nombre y una lista de miembros (usuarios) y pacientes. Este ID se guarda en los documentos asociados (pacientes, citas, enfermedades) y se utiliza en las consultas para filtrar los datos disponibles para cada usuario.

Desde el punto de vista funcional, esto permite que varios cuidadores colaboren en el seguimiento de los mismos pacientes, compartiendo la información médica de forma sincronizada y segura. Todas las funcionalidades (añadir pacientes, registrar enfermedades, gestionar citas) están aisladas por grupo, garantizando que los datos no se mezclen entre diferentes entornos familiares.

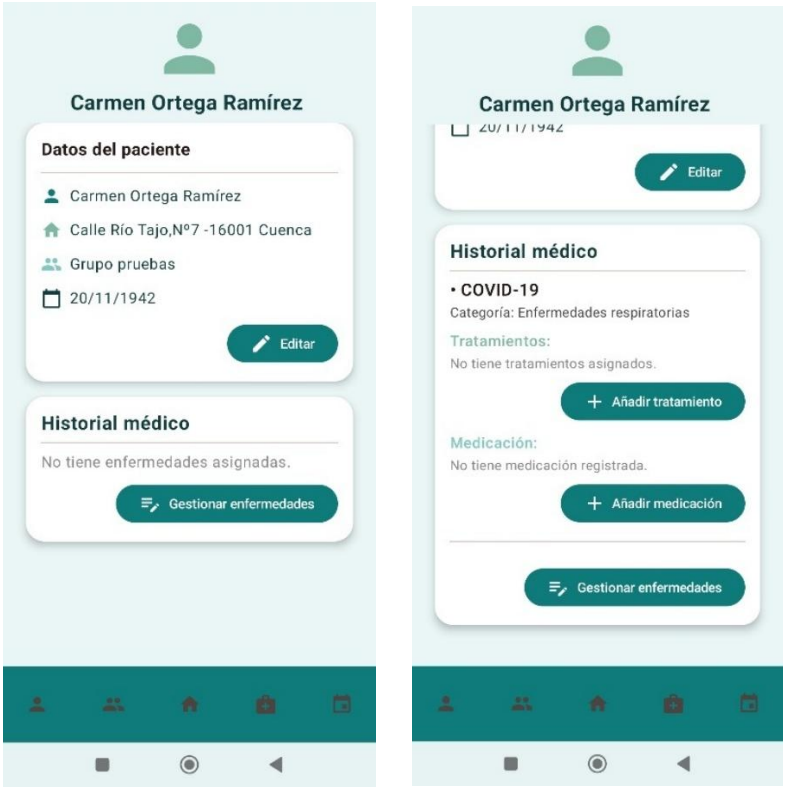
La lógica del grupo familiar está integrada en los repositorios y ViewModels, y su gestión se realiza automáticamente tras la autenticación. Esta estructura ha sido clave para permitir una experiencia compartida entre cuidadores, manteniendo al mismo tiempo privacidad y organización.

5.5 GESTIÓN DE PACIENTES

La gestión de pacientes es una de las funcionalidades centrales de Cuida+. Cada paciente está asociado a un grupo familiar y es accesible solo por los cuidadores miembros de dicho grupo. Desde la pantalla de pacientes, el usuario puede ver la lista de pacientes registrados, añadir uno nuevo y eliminarlo o acceder a sus detalles.

El formulario de alta de paciente recoge información básica como nombre completo, fecha de nacimiento y dirección. Esta información se almacena en Firebase Firestore y queda vinculada al grupoFamiliarId, permitiendo que todos los cuidadores del grupo accedan a ella.

Cada paciente cuenta con una pantalla de detalle desde donde se pueden consultar y gestionar sus enfermedades, tratamientos, medicamentos. Esta pantalla está diseñada con tarjetas visuales para que el acceso a la información sea cómodo y ordenado.



Toda la lógica relacionada con pacientes se gestiona desde PacienteViewModel y PacienteRepository, asegurando una correcta separación entre la interfaz y la lógica de acceso a datos. Gracias a esta estructura, los datos se sincronizan en tiempo real entre todos los cuidadores del grupo, manteniéndose siempre actualizados.

5.6 ENFERMEDADES DE PACIENTES

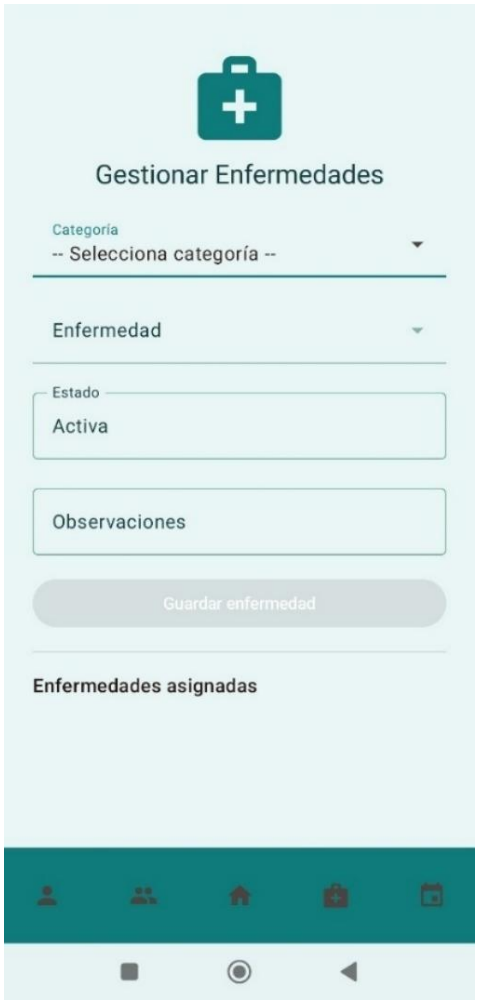
Cuida+ permite a los cuidadores asignar enfermedades a cada paciente a partir de una colección precargada de enfermedades almacenada en Firebase (enfermedades_maestro). Esta colección asegura que los nombres y categorías sean coherentes, evitando errores manuales y facilitando una futura explotación de datos.

Cuando se asigna una enfermedad, se crea un documento en la colección

enfermedadPaciente, que enlaza al paciente correspondiente e incluye información adicional como la fecha de diagnóstico, estado actual (activo, controlado, superado) y observaciones del cuidador. Las enfermedades se pueden marcar como pasadas, lo que permite mantener un historial médico limpio y ordenado.

Visualmente, cada enfermedad se muestra como una tarjeta dentro del perfil del paciente, donde también se accede a sus tratamientos y medicación asociada. La carga y eliminación de enfermedades se gestiona desde el EnfermedadPacienteViewModel, garantizando que los datos estén sincronizados y correctamente filtrados por paciente y grupo familiar.

Esta estructura permite mantener un control preciso de la evolución médica de cada persona atendida, facilitando la coordinación entre varios cuidadores.



5.7 TRATAMIENTOS Y MEDICAMENTOS

Cada enfermedad activa registrada en un paciente puede tener asociados uno o varios tratamientos no farmacológicos y medicamentos, que se gestionan desde Firebase mediante colecciones específicas. Para evitar errores de entrada, ambos se seleccionan desde catálogos precargados (tratamientos_maestro y medicamentos_maestro).

Los tratamientos incluyen información como nombre, tipo (ej. fisioterapia, dieta), fechas de inicio y fin, y una descripción. Los medicamentos registran nombre, dosis, frecuencia, vía de administración y observaciones. Ambos tipos de datos se relacionan directamente con la enfermedad concreta (enfermedadPacienteId) y no con el paciente en general, asegurando una organización médica precisa.

Desde la interfaz, los tratamientos y medicamentos se presentan como tarjetas dentro de cada enfermedad, facilitando la visualización y edición sin necesidad de cambiar de pantalla.

Toda esta gestión se realiza mediante los TratamientoViewModel, MedicamentoViewModel y sus respectivos repositorios, lo que garantiza una estructura limpia, mantenible y sincronizada entre todos los cuidadores del grupo.

5.8 CITAS MÉDICAS

Cuida+ permite registrar y gestionar citas médicas para cada paciente del grupo familiar. Estas citas se almacenan en la colección citas, vinculadas tanto al pacienteId como al grupoFamiliarId, lo que asegura que solo los cuidadores autorizados puedan acceder a ellas.

Cada cita médica incluye información detallada como fecha y hora, médico, especialidad, ubicación, motivo y observaciones, además de un campo que indica si la cita fue realizada o no. El formulario de alta permite registrar todos estos datos de forma sencilla desde la interfaz.

Las citas se agrupan por paciente y por fecha, y se muestran tanto en la pantalla de inicio (como resumen general) como en la sección de citas, permitiendo al cuidador consultar la planificación médica de forma clara y ordenada.

La lógica de gestión de citas se encuentra en CitaViewModel y CitaRepository, donde se realiza la carga, creación y eliminación de documentos. Gracias a la sincronización en tiempo real de Firebase, cualquier cambio realizado se refleja al instante en los dispositivos de los demás cuidadores del grupo.

Esta funcionalidad es clave para coordinar adecuadamente la atención médica de los pacientes y evitar olvidos o solapamientos entre cuidadores.

5.9 INTERFAZ DE USUARIO Y NAVEGACIÓN

La interfaz de Cuida+ ha sido diseñada completamente con Jetpack Compose, utilizando un enfoque modular y limpio que facilita la experiencia de uso. Cada pantalla está estructurada con un Scaffold que incluye una barra superior (TopBar), contenido desplazable (LazyColumn) y un botón de acción flotante (FAB) para las operaciones principales.

Además, los formularios siguen una estructura ordenada con validaciones básicas, y los botones están posicionados de forma consistente en todas las pantallas para mantener una navegación intuitiva.

La app utiliza un sistema de navegación compuesto por rutas claramente definidas. Al navegar entre pantallas, se pasan parámetros como el ID del paciente o de la enfermedad, asegurando que la información mostrada siempre esté correctamente filtrada y actualizada.

Toda la navegación está controlada por el estado de la sesión: si el usuario no está autenticado, se redirige automáticamente a la pantalla de login. Este comportamiento garantiza una experiencia segura y coherente en todo momento.

5.10 SINCRONIZACIÓN Y ESTADO

Una de las ventajas principales de Cuida+ es su capacidad para trabajar en tiempo real gracias a Firebase Firestore, lo que permite que varios cuidadores puedan ver y gestionar la misma información simultáneamente, sin necesidad de actualizar manualmente la aplicación.

Para lograr esto, todos los datos se sincronizan automáticamente desde Firebase, y la interfaz se mantiene actualizada mediante herramientas reactivas de Jetpack Compose como mutableStateOf y StateFlow. Los ViewModel observan los cambios en los datos y actualizan la UI en consecuencia usando funciones como collectAsState.

Por ejemplo, si un cuidador añade una enfermedad o edita una cita, el cambio se refleja inmediatamente en los dispositivos de los demás cuidadores del grupo. Esta sincronización ha sido clave para garantizar que todos trabajen siempre sobre la información más reciente.

Además, se ha implementado la gestión de estados de carga, error y éxito para mejorar la experiencia de usuario. Esto permite mostrar indicadores visuales cuando se está esperando una respuesta, o mensajes informativos si ocurre algún problema al conectarse con la base de datos.

Gracias a esta estructura reactiva y sincronizada, Cuida+ mantiene un comportamiento estable, fluido y actualizado en todo momento, sin que el usuario tenga que preocuparse por guardar, recargar o sincronizar manualmente.

5.11 PRUEBAS Y AJUSTES FINALES

Antes de la entrega final, se realizaron diversas pruebas funcionales para asegurar que todas las funcionalidades de Cuida+ funcionaran correctamente. Estas pruebas se llevaron a cabo tanto en el emulador de Android Studio como en dispositivos reales, comprobando la

estabilidad, fluidez y comportamiento general de la aplicación.

Se validó que el flujo de navegación fuera coherente, que los datos se cargaran y actualizaran en tiempo real, y que las operaciones de lectura y escritura en Firebase funcionaran sin errores. También se probaron los formularios de alta, edición y eliminación en cada pantalla, así como la persistencia de la sesión de usuario.

Además, se realizaron ajustes visuales en la interfaz: márgenes, tamaños de texto, posicionamiento de botones, colores y organización de secciones. Se mejoraron las tarjetas informativas, se optimizaron los formularios y se revisó la consistencia visual de toda la app para ofrecer una experiencia clara y agradable.

También se revisaron y corrigieron posibles errores en los ViewModel y Repositorios, asegurando que las operaciones fueran seguras, controladas y que los datos no se duplicaran o perdieran. Gracias a estas pruebas finales, Cuida+ ha alcanzado un estado estable y funcional, listo para su presentación y posible evolución futura.

6. Conclusiones y líneas futuras

A. CONCLUSIONES

El desarrollo de Cuida+ ha supuesto un proceso completo de diseño, análisis e implementación de una solución tecnológica centrada en mejorar la organización del cuidado sanitario dentro del entorno familiar. A lo largo del proyecto se han tomado decisiones técnicas fundamentadas en buenas prácticas, buscando no solo cumplir los requisitos funcionales, sino también construir una aplicación sólida, mantenible y con potencial de crecimiento.

Uno de los principales logros ha sido la correcta aplicación del patrón arquitectónico MVVM, que ha permitido mantener la lógica de negocio separada de la interfaz gráfica, lo cual ha favorecido la claridad del código, su reutilización y la facilidad de mantenimiento. Esta

arquitectura se ha complementado con el uso de repositorios específicos por entidad, ViewModels independientes por pantalla y un modelo de datos coherente, lo que ha hecho posible una estructura robusta y escalable.

En cuanto al almacenamiento y sincronización de datos, el uso de Firebase Firestore ha sido clave para garantizar la integridad y actualización en tiempo real de toda la información. Gracias a esta tecnología, múltiples cuidadores pueden colaborar simultáneamente en el seguimiento de un mismo paciente, desde diferentes dispositivos y sin necesidad de realizar sincronizaciones manuales. La integración con Firebase Authentication ha garantizado la seguridad del acceso, permitiendo que cada usuario tenga control únicamente sobre los datos de su grupo familiar.

Desde el punto de vista visual y de experiencia de usuario, el uso de Jetpack Compose ha permitido diseñar una interfaz moderna, clara y funcional. La aplicación resulta accesible para usuarios sin conocimientos técnicos, gracias a su estructura por tarjetas, menús consistentes y navegación sencilla.

Además de alcanzar los objetivos iniciales, Cuida+ ha demostrado que es posible desarrollar una herramienta tecnológica realista, útil y coherente dentro de un entorno académico, aplicando principios de desarrollo profesional como modularidad, separación de responsabilidades, reutilización de código y pruebas continuas.

En definitiva, Cuida+ no solo ha sido un proyecto técnico, sino también una propuesta con utilidad práctica. Ha sido concebida con un enfoque humano, centrado en las personas que cuidan de otras, y se ha construido con la intención de ser una herramienta real de apoyo en su día a día. Su implementación ha sido una experiencia formativa completa, que ha permitido aplicar conocimientos, tomar decisiones fundamentadas y construir una solución funcional y preparada para su evolución.

B. LÍNEAS FUTURAS DE MEJORA

Aunque la versión actual de Cuida+ cumple satisfactoriamente con los objetivos funcionales definidos al inicio del proyecto, existe un amplio margen de evolución. Gracias a su estructura modular, la aplicación está preparada para incorporar nuevas funcionalidades que amplíen su alcance, mejoren la experiencia de uso y se adapten a nuevos perfiles y necesidades.

A continuación, se detallan algunas de las principales líneas de mejora que podrían considerarse para futuras versiones del proyecto:

- **Sistema de notificaciones:** Implementar notificaciones locales y push que alerten al cuidador sobre próximas citas médicas, tomas de medicación, tratamientos o revisiones. Esto mejoraría la adherencia y reduciría olvidos.

- **Calendario visual interactivo:** Añadir un calendario con vistas por día, semana o mes, donde se visualicen todos los eventos médicos de forma organizada, filtrables por paciente.

- **Gestión de roles y permisos:** Definir distintos niveles de acceso (administrador, cuidador, colaborador), con funcionalidades adaptadas según el perfil de cada usuario.

- **Historial clínico cronológico:** Incluir un historial detallado con todos los eventos del paciente: citas pasadas, enfermedades superadas, tratamientos finalizados, etc.

- **Adjuntar documentos e imágenes:** Ampliar el uso de Firebase Storage para asociar archivos como informes médicos, analíticas o imágenes diagnósticas.

- **Accesibilidad mejorada:** Integrar lectura por voz, tamaño de fuente ajustable, y botones más grandes, especialmente útiles para personas mayores.

- **Internacionalización:** Traducir la aplicación a varios idiomas para adaptarse a familias multiculturales o usuarios de otros países.

- **Gráficas de evolución médica:** Representar visualmente indicadores de salud (glucosa, tensión, temperatura...) para facilitar el seguimiento del estado del paciente.

- **Exportación en PDF:** Permitir generar informes médicos en formato PDF con los datos principales de cada paciente para uso médico o personal.

- **Modo offline:** Implementar funcionalidad limitada sin conexión, con sincronización automática al recuperar acceso a internet.

- **Autenticación biométrica:** Añadir acceso mediante huella o reconocimiento facial, facilitando un login más rápido y seguro.

- **Compatibilidad con wearables o sensores:** Integrar dispositivos médicos o relojes inteligentes que envíen datos automáticamente a la app.

- **Compatibilidad con otros sistemas operativos:** Explorar la posibilidad de desarrollar una versión multiplataforma (iOS o web) para ampliar el alcance de la app.

- **Edición completa de datos clínicos:** Permitir editar no solo los datos personales, sino también enfermedades, tratamientos o citas ya creadas, para corregir errores sin tener que eliminar y volver a crear elementos.

- **Múltiples grupos por usuario:** Habilitar que un mismo usuario pueda pertenecer a más de un grupo familiar y cambiar entre ellos según el contexto de cuidado.

Estas líneas futuras no solo ampliarían las funcionalidades de Cuida+, sino que también la convertirían en una solución tecnológica más flexible, colaborativa y adaptada a la realidad cambiante de muchas familias cuidadoras. La evolución hacia estos objetivos permitiría aumentar su impacto como herramienta digital de apoyo en el ámbito del cuidado médico no profesional.

7. Bibliografía

A continuación, se listan las fuentes bibliográficas y documentales que han sido consultadas y utilizadas durante el desarrollo del proyecto Cuida+. Estas fuentes han sido fundamentales para comprender y aplicar correctamente tecnologías como Firebase, Kotlin, Jetpack Compose, Git, así como para estructurar el modelo de datos y la arquitectura de la aplicación. También se incluye la referencia al modelo de lenguaje ChatGPT de OpenAI, empleado como herramienta de apoyo en tareas de redacción técnica, aclaración conceptual y estructuración del trabajo.

- Google. (s.f.). Firebase documentation. Firebase. Recuperado de https://firebase.google.com/docs

- Google. (s.f.). Jetpack Compose overview. Android Developers. Recuperado de https://developer.android.com/jetpack/compose

- Google. (s.f.). *Firebase Security Rules*. Firebase. Recuperado de https://firebase.google.com/docs/rules

- JetBrains. (s.f.). Kotlin language documentation. Recuperado de https://kotlinlang.org/docs/home.html

- GitHub. (s.f.). GitHub Docs. Recuperado de https://docs.github.com/

- Oracle. (s.f.). OpenJDK documentation. Recuperado de https://openjdk.org/

- PlantUML. (s.f.). UML diagrams with PlantUML. Recuperado de https://plantuml.com/

- Inkscape. (s.f.). Inkscape user manual. Recuperado de https://inkscape.org/

- Google. (s.f.). Architecture components: Guide to app architecture. Android Developers. Recuperado de https://developer.android.com/jetpack/guide

- Google. (s.f.). Android Studio User Guide. Android Developers. Recuperado de https://developer.android.com/studio

- Google. (s.f.). *Material Design guidelines*. Recuperado de https://m3.material.io/

- OpenAI. (2024). ChatGPT (versión GPT-4) [Modelo de lenguaje grande]. OpenAI. https://chat.openai.com/