UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA EN TECNOLOGIAS DE LA INFORMACIÓN

Aplicación Móvil Offline multiplataforma para ingreso de datos de ficha familiar del Centro de Salud Chambo usando el framework Flutter

Trabajo de Titulación para optar al título de Ingeniero en Tecnologías de la Información

Autores:

Chafla Villacrés, Francis Jesús
Criollo Zambrano, Steven Eduardo

Tutor:

Ing. Gonzalo Allauca Peñafiel

Riobamba, Ecuador. 2025

DECLARATORIA DE AUTORÍA

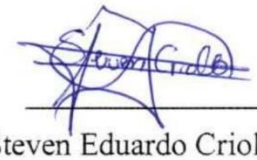
Yo, Francis Jesús Chafla Villacrés, con cédula de ciudadanía 0605966290 y Steven Eduardo Criollo Zambrano con cédula de ciudadanía 1720578960 autores del trabajo de investigación titulado: "Aplicación Móvil Offline multiplataforma para ingreso de datos de ficha familiar del Centro de Salud Chambo usando el framework Flutter", certifico que la producción, ideas, opiniones, criterios, contenidos y conclusiones expuestas son de mí exclusiva responsabilidad.

Asimismo, cedemos a la Universidad Nacional de Chimborazo, en forma no exclusiva, los derechos para su uso, comunicación pública, distribución, divulgación y/o reproducción total o parcial, por medio físico o digital; en esta cesión se entiende que el cesionario no podrá obtener beneficios económicos. La posible reclamación de terceros respecto de los derechos de autor de la obra referida será de mi entera responsabilidad; librando a la Universidad Nacional de Chimborazo de posibles obligaciones.

En Riobamba, 28 de febrero de 2025.

Francis Jesús Chafla Villacrés

C.I: 0605966290



Steven Eduardo Criollo Zambrano

C.I: 1720578960

ACTA FAVORABLE - INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN

En la Ciudad de Riobamba, a los **28** días del mes de **abril** del **2025**, luego de haber revisado el Informe Final del Trabajo de Investigación presentado por los estudiantes **Francis Jesús Chafla Villacrés** con CC: **0605966290**, y **Steven Eduardo Criollo Zambrano** con CC: **1720578960** de la carrera **Ingeniería en Tecnologías de la Información** y dando cumplimiento a los criterios metodológicos exigidos, se emite el **ACTA FAVORABLE DEL INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN** titulado **"Aplicación móvil offline multiplataforma para ingreso de datos de ficha familiar del centro de salud Chambo usando el framework Flutter"**, por lo tanto se autoriza la presentación del mismo para los trámites pertinentes.

Ing. Luis Gonzalo Allauca Peñafiel
TUTOR DE PROYECTO DE INVESTIGACIÓN

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

Quienes suscribimos, catedráticos designados Miembros del Tribunal de Grado para la evaluación del trabajo de investigación "Aplicación Móvil Offline multiplataforma para ingreso de datos de ficha familiar del Centro de Salud Chambo usando el framework Flutter", presentado por Francis Jesús Chafla Villacrés, con cédula de identidad número 0605966290 y por Steven Eduardo Criollo Zambrano, con cédula de identidad número 1720578960, bajo la tutoría del Ing. Luis Gonzalo Allauca Peñafiel, Mgs; certificamos que recomendamos la APROBACIÓN de este con fines de titulación. Previamente se ha evaluado el trabajo de investigación y escuchada la sustentación por parte de sus autores; no teniendo más nada que observar.

De conformidad a la normativa aplicable firmamos, en Riobamba el 30 de mayo de 2025.

Ing. María Isabel Uvidia, Mgs.
PRESIDENTE DEL TRIBUNAL DE GRADO

Ing. Lady Espinoza, Mgs.
MIEMBRO DEL TRIBUNAL DE GRADO

Ing. Fernando Molina, Mgs.
MIEMBRO DEL TRIBUNAL DE GRADO

C E R T I F I C A C I Ó N

Que, **Francis Jesús Chafla Villacrés** con CC: **0605966290** y **Steven Eduardo Criollo Zambrano** con CC: **1720578960**, estudiantes de la Carrera de Ingeniería en **Tecnologías de la Información**, Facultad de **Ingeniería**; han trabajado bajo mi tutoría el trabajo de investigación titulado "**Aplicación Móvil Offline multiplataforma para ingreso de datos de ficha familiar del Centro de Salud Chambo usando el framework Flutter**", cumple con el **8%**, de acuerdo al reporte del sistema Anti plagio **COMPILATIO**, porcentaje aceptado de acuerdo a la reglamentación institucional, por consiguiente autorizo continuar con el proceso.

Riobamba, 19 de mayo de 2025.

Firmado electrónicamente por:
LUIS GONZALO
ALLAUCA PENAFIEL
Validar únicamente con FirmaEC

Mgs. Gonzalo Allauca Peñafiel
TUTOR

DEDICATORIA

Dedico esta tesis con todo mi cariño y gratitud a mis queridos padres, Alex e Irma, quienes han sido un apoyo fundamental en cada paso de este camino. Gracias por su amor incondicional, su paciencia y por enseñarme a nunca rendirme ante los desafíos.

Gracias por estar siempre ahí, por alegrarse conmigo en cada logro y por animarme cuando sentía que ya no podía más.

Este logro también les pertenece, porque sin su amor y apoyo, no habría llegado hasta aquí. Los llevo en el corazón en cada paso de este camino, porque han sido mi mayor fuerza e inspiración.

Dedico este trabajo con mucho amor a mis queridos Toby y Silvestre. Gracias por acompañarme en los días difíciles, por estar ahí sin decir nada, solo con su presencia tranquila y su forma tan especial de dar cariño.

También a mis profesores, gracias por su dedicación, sabiduría y guía constante durante mi formación académica. Sus enseñanzas sentaron las bases de mis conocimientos y su compromiso fue una fuente de inspiración para alcanzar esta meta.

Francis Jesús Chafla Villacrés

Dedico este proyecto de investigación con todo mi cariño y gratitud a mis padres Eduardo y María también a mi hermana Sheyla, quienes me han demostrado su apoyo.

Gracias, mamá y papá, por su amor incondicional, por enseñarme el valor del esfuerzo, la perseverancia y la honestidad.

Su confianza en mí me ha impulsado a seguir adelante pase lo que pase. A mi hermana, por estar a mi lado, por su ánimo y compañía constante.

También lo dedico a mis profesores de la Universidad Nacional de Chimborazo, quienes, con dedicación, me han compartido su sabiduría, brindándome conocimientos, guía y motivación a lo largo de todo mi proceso de formación académica.

Steven Eduardo Criollo Zambrano

AGRADECIMIENTO

Quiero expresar mi más profundo agradecimiento a todas las personas que hicieron posible la realización de este proyecto.

A mis queridos padres, Alex e Irma, gracias por su amor incondicional, paciencia y apoyo constante.

A mi compañero Steven, por su colaboración, compromiso y amistad inquebrantable durante todo este proceso. Su apoyo fue clave para superar los todos retos y desafíos que enfrentamos juntos.

Y, finalmente, a mi tutor, Ing. Gonzalo Allauca Peñafiel, por su guía paciente, consejos acertados y constante acompañamiento. Su apoyo fue fundamental para mantener la calidad y dirección de este trabajo.

A todas las personas que, de alguna manera, aportaron a este proyecto, les agradezco su apoyo, ánimo y confianza en mí. Este logro no habría sido posible sin su valiosa colaboración.

Francis Jesús Chafla Villacrés

Quiero expresar el mayor agradecimiento a todas las personas que hicieron posible la realización de este proyecto de investigación.

En primer lugar, a mi compañero de tesis, Francis, por su valiosa colaboración, compromiso, apoyo constante y amistad incondicional. Gracias a su esfuerzo conjunto, logramos alcanzar este objetivo, superando las dificultades que surgieron en el camino.

Agradezco de manera especial al Ing. Gonzalo Allauca Peñafiel, por su guía, acompañamiento y disposición durante el desarrollo de este trabajo. Su asesoría fue fundamental para mantener el enfoque y la calidad de esta investigación.

Extiendo también mi sincero agradecimiento a todos los docentes de la Universidad Nacional de Chimborazo, quienes, a lo largo de mi formación académica, me brindaron no solo sus conocimientos, sino también su apoyo y motivación. Su dedicación fue un punto importante para alcanzar este logro.

A todos, mi gratitud infinita.

Steven Eduardo Criollo Zambrano

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

En el presente proyecto de investigación se desarrolló una aplicación móvil multiplataforma offline utilizando el framework Flutter con el objetivo de evaluar en base al estándar ISO/IEC 25010 la fiabilidad en la transferencia de datos de fichas familiares desde una aplicación offline a la base de datos del Centro de Salud tipo B de Chambo una vez que la aplicación cuente con acceso a internet.

La aplicación offline cuenta con módulos para la gestión de datos del establecimiento, de la familia, de los miembros, del registro de embarazadas, de la mortalidad familiar y del profesional responsable. El tipo de investigación utilizada fue cuantitativa y se establece una población de estudio considerada como infinita puesto que se definen escenarios de simulación con diferentes niveles de concurrencia para medir la fiabilidad en términos de ausencia de fallos, tolerancia a fallos y capacidad de recuperación.

Las herramientas utilizadas en las simulaciones fueron JMeter, Jest y scripts en JavaScript, donde se recopilaron registros de eventos y errores para analizar la fiabilidad. Para el desarrollo de la aplicación offline se usó la metodología Mobile-D con ciclos rápidos e iterativos propios del desarrollo de aplicaciones móviles. Se concluye que la implementación de transacciones, validaciones referenciales, manejo de errores con bloques try-catch, control de estado para sincronización asíncrona y registro de logs aseguró la fiabilidad de los datos, facilitando el ingreso de datos de la Ficha familiar. Los resultados mostraron una capacidad de recuperación de datos superior al 90 % con un MTTR de 17 segundos, un 100 % de ausencia de fallos en las funcionalidades principales y un 100 % de tolerancia ante fallos parciales.

Palabras claves: Aplicación móvil offline, Flutter, ingreso de datos, fiabilidad.

ABSTRACT

In the present research project, a multiplatform offline mobile application was developed using the Flutter framework, in order to evaluate, based on the ISO/IEC 25010 standard, the reliability of transferring family record data from the offline application to the database of the Chambo Type B Health Center, once the application gains internet access.

The offline application includes modules for managing facility, family members, pregnancy records, family mortality, and data of the responsible professional. The research type used was quantitative, and the study population was considered infinite, because simulation scenarios were defined with different concurrency levels, to measure reliability in faultlessness, fault tolerance, and recoverability.

The tools used in the simulations were JMeter, Jest, and JavaScript scripts, where event and error logs were collected to analyze reliability. For the development of the offline application, the Mobile-D methodology was used, featuring fast and iterative cycles specific to mobile application development. It is concluded the transactions implementation, referential validations, error handling with try-catch blocks, state control for asynchronous synchronization, and logs recording, ensured the data reliability, facilitating the entry Family Record data.

The results showed a data recovery capacity over 90% with an MTTR 17 seconds, 100% faultlessness performance in main functionalities, and 100% fault tolerance for partial failures.

Keywords: Offline mobile application, Flutter, data entry, reliability.

Reviewed by:

Dra. Myriam Trujillo Brito, Mgs.

English Professor

c.c. 0601823214

CAPÍTULO I. INTRODUCCIÓN

El Centro de Salud Tipo B, que se encuentra en el municipio de Chambo en la provincia de Chimborazo, ha llevado a cabo una serie de iniciativas con el fin de mejorar la gestión de la información médica utilizando la tecnología. Actualmente a través del proyecto de titulación titulado "Aplicación web y móvil para el expediente familiar en el Centro de Salud de Chambo" permitió facilitar el ingreso, modificación, búsqueda y eliminación de fichas familiares de manera fácil y efectiva. Ayudando a los profesionales de la salud acceder a la información de los grupos familiares en el sector rural, lo que les ayudó a planificar visitas a pacientes para registrar las fichas familiares y garantizar una cobertura eficiente de servicios de salud. Esto estableció un marco inicial para digitalizar datos médicos y mejorar la accesibilidad a la información de salud de los pacientes.

El entorno tecnológico ha evolucionado con un creciente enfoque en las aplicaciones móviles multiplataforma que permiten una gestión de datos más efectiva y accesible. Según Amazon “Flutter es un marco de código abierto creado por Google que permite a los desarrolladores de front-end y pila completa crear una interfaz de usuario (IU) de aplicación para varias plataformas con un único código base” [1]. Además, Netiris afirma que “Flutter no solo reduce el tiempo y los costos de desarrollo al permitir la creación de aplicaciones móviles que se ejecutan en Android e iOS utilizando una sola base de código, sino que también garantiza que la experiencia del usuario sea coherente a través de diferentes plataformas” [2].

En la actualidad, se han creado proyectos multiplataforma que utilizan frameworks como Flutter en áreas como la asesoría médica en línea que mejora el seguimiento de las personas expuestas al contagio de COVID-19 [3]. Además, se ha desarrollado aplicaciones para el agendamiento de citas médicas, facilitando la atención sanitaria en la ciudad de Cuenca [4]. Estos proyectos demuestran la versatilidad y eficiencia de Flutter, que permitió el desarrollo efectivo de aplicaciones multiplataforma. Los avances en este campo demuestran la posibilidad de crear soluciones para satisfacer las necesidades específicas de los centros de salud en contextos similares al del cantón Chambo.

La investigación actual propone el desarrollo de una aplicación móvil offline multiplataforma basada en el framework Flutter para facilitar la gestión de los datos de ficha familiar en lugares donde no exista acceso a internet y posteriormente cuando se disponga de conexión a internet actualizar las fichas familiares de los pacientes en el Centro de Salud de Chambo, así que la aplicación se dividió en cinco módulos principales:

A. Datos del establecimiento y familia:
Registra información básica sobre el centro de salud y la familia, como el nombre del centro, el código único, la ubicación geográfica, el estado de vulnerabilidad, los datos de contacto y las referencias del domicilio.

B. Miembros de la familia:

Almacena información personal de cada miembro de la familia, como nombre, parentesco, fecha de nacimiento, edad, ocupación, sexo, educación, estado de vacunación, controles de salud bucal, estado de riesgo, enfermedad, discapacidad y grupo de farmacia.

C. Embarazadas:
Registro de toda la información de las mujeres embarazadas en la familia, como su nombre, fecha de la última menstruación, fecha probable de parto, controles prenatales, semanas de gestación, esquema de vacunación, antecedentes gineco-obstétricos y patológicos.

D. Mortalidad familiar en los últimos 5 años:
Almacena información sobre los miembros de la familia que han fallecido en los últimos cinco años, como nombre, relación con el jefe de familia, edad al momento de la muerte y causa de la muerte.

E. Datos del profesional responsable:
Registre los nombres y apellidos, la fecha y hora de la visita, el número de identificación, la firma y el sello del profesional responsable de completar la ficha.

1.1 Planteamiento del problema

A pesar del avance obtenido a través del proyecto de titulación anterior “Desarrollo de una aplicación web y móvil para la ficha familiar en el Centro de Salud Chambo” [5], persisten desafíos significativos en la gestión de fichas familiares, particularmente en áreas sin conectividad a internet. La incapacidad de registrar datos médicos sin una conexión constante a internet ha generado problemas en el manejo de la información debido a que actualmente los médicos se desplazan a territorio y existen muchos lugares en los cuales no existe conectividad y a pesar de que tienen una aplicación no tienen forma de registrar o modificar esos datos de la ficha familiar.

Estudiar y abordar este problema es de vital importancia porque la fiabilidad de la información médica es esencial para el Centro de Salud Chambo. Una solución que permita el ingreso y acceso a datos médicos sin conexión mejoraría la eficiencia operativa y la calidad de la atención al paciente. Además, contribuiría a la modernización del sistema de salud local, alineándose con las tendencias globales hacia la digitalización y mejoramiento de la salud pública.

1.2 Justificación

La observación directa de las dificultades que enfrenta el Centro de Salud de Chambo en la gestión de datos médicos es la principal motivación para realizar esta propuesta. Existen tecnologías que facilitan la manipulación de datos sin necesidad de acceso a internet, lo que permite la gestión de datos en áreas rurales sin conectividad y facilita el almacenamiento de fichas familiares. La propuesta consiste en el desarrollo de una aplicación móvil offline que permita el ingreso y gestión de datos médicos de manera fiable sin necesidad de una conexión a internet para posteriormente cuando exista la conexión a internet estos datos

puedan ser almacenados en la base de datos de ficha familiar garantizando ausencia a fallos, tolerancia a fallos y capacidad de recuperación de la información.
Esta aplicación ofrecerá una interfaz intuitiva, con capacidades de almacenamiento local y posterior sincronización de datos cuando se restablezca la conectividad, facilitando una transición hacia la digitalización completa de los registros médicos y su integración con los sistemas existentes en el Centro de Salud de Chambo.

El objetivo de esta investigación es desarrollar una aplicación móvil multiplataforma offline, utilizando el framework Flutter, para ingresar datos de fichas familiares en el Centro de Salud Tipo B Chambo. La propuesta se centrará en garantizar la integridad y accesibilidad de los datos, desarrollando módulos como datos del establecimiento y familia, miembros de la familia, registro de embarazadas, mortalidad familiar, y datos del profesional responsable, con base en el proyecto de investigación anterior.

1.3 Formulación del problema

¿Cómo la aplicación móvil offline multiplataforma garantizará la fiabilidad en el proceso de registro de fichas familiares en el Centro de Salud Chambo?

1.4 Objetivos

Objetivo general

- Implementar una aplicación móvil offline multiplataforma para ingreso de datos de ficha familiar del Centro de Salud tipo B de Chambo usando el framework Flutter.

Objetivos específicos

- Investigar métodos y tecnologías que permitan garantizar la fiabilidad en aplicaciones offline.
- Desarrollar la aplicación móvil offline multiplataforma para el ingreso de datos de ficha familiar con Mobile-D.
- Evaluar la fiabilidad de la aplicación móvil offline multiplataforma utilizando la norma ISO / IEC 25010.

CAPÍTULO II. MARCO TEÓRICO

2.1 Aplicación móvil

Según información extraída de Anincubator "Una aplicación móvil, también conocida como aplicación móvil, es un tipo de aplicación que se puede ejecutar en un dispositivo móvil, como un teléfono inteligente o una tableta. Las aplicaciones pueden brindar a los usuarios servicios y experiencias de alta calidad, incluso si suelen ser unidades de software pequeñas con funciones limitadas" [6].

Las aplicaciones móviles no dependen de sistemas de software integrados, a diferencia de las aplicaciones para computadoras de escritorio. Por el contrario, cada aplicación móvil ofrece una funcionalidad única y limitada. Por ejemplo, puede ser un juego, una calculadora o un navegador web para teléfonos inteligentes.

Las aplicaciones móviles evitaban la multifuncionalidad debido a los recursos de hardware limitados de los primeros dispositivos móviles. Sin embargo, las aplicaciones móviles siguen siendo funcionales incluso si los dispositivos que se utilizan hoy en día son mucho más sofisticados. Como resultado, los propietarios de aplicaciones móviles permiten a los usuarios elegir las características que necesitan en sus dispositivos, a continuación, en la figura 1 se observan los tipos de aplicaciones móviles.

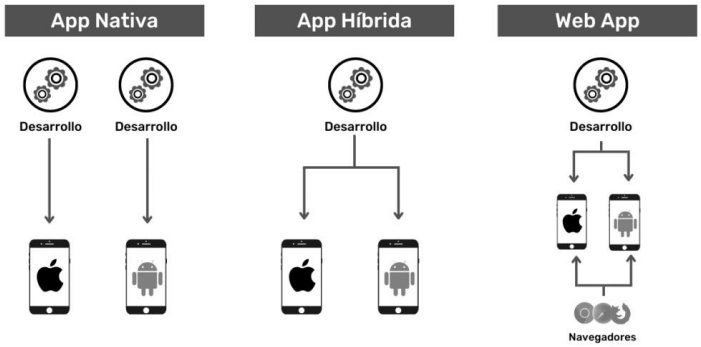


Figura 1: Tipos de aplicaciones móviles

Fuente: [7]

2.1.1 Características de una aplicación móvil

Una aplicación desarrollada para móviles suele tener ciertas características, a continuación, en la tabla 1 se describen.

Tabla 1: Características de una aplicación móvil

Aspecto	Descripción
Beneficios Únicos	La aplicación debe ofrecer beneficios únicos que respondan a necesidades específicas de los usuarios, centrándose en un nicho con pocos competidores. Además, debe ser fácil de usar, evitando complejidad y problemas que puedan frustrar a los usuarios.

Adaptación a Varios Sistemas Operativos	Es crucial conocer el porcentaje de usuarios por sistema operativo para decidir si la app será multiplataforma o no. También debe adaptarse a diferentes resoluciones para asegurar una experiencia de usuario óptima en todos los dispositivos.
Diseño Agradable y Capacidades de Interacción	Un diseño atractivo y funcional es clave. La interfaz debe ser simple, intuitiva, ordenada y coherente con la identidad de la empresa, brindando una experiencia de usuario positiva. Las funcionalidades útiles y la rapidez de navegación son esenciales.
Actualizaciones	Las actualizaciones son fundamentales para corregir errores y mejorar la app. Escuchar las críticas de los usuarios y responder a ellas ayuda a mantener una buena reputación, asegurando que las nuevas versiones resuelvan problemas sin crear otros.

Fuente: [8]

2.2 Ventajas y desventajas de las aplicaciones móviles

Según Parker las principales ventajas y desventajas de las aplicaciones móviles serian:

2.2.1 Ventajas de las aplicaciones móviles

Facilidad de comunicación: Las aplicaciones móviles permiten una comunicación más fácil y directa con los clientes. Esto puede incluir funcionalidades como chats en vivo, accesibilidad de información de contacto y botones de atención al cliente, lo que facilita la interacción con la empresa [9].

Ampliación de la audiencia: Las aplicaciones móviles pueden ayudar a llegar a un público más amplio, especialmente aquellos que prefieren usar smartphones sobre ordenadores. Esto puede atraer a nuevos clientes que de otra manera no podrían acceder a los servicios de la empresa.

Comodidad de uso: Las aplicaciones móviles son muy convenientes ya que los usuarios siempre llevan consigo sus teléfonos. Esto facilita el acceso a los servicios o productos de la empresa en cualquier momento y lugar, lo que puede aumentar las ventas y el uso de la app.

Uso sin conexión: A diferencia de los sitios web, algunas aplicaciones móviles pueden funcionar sin conexión a Internet, lo que es una gran ventaja para los usuarios que no siempre tienen acceso a WiFi.

Funcionalidades Útiles: Las aplicaciones móviles pueden ofrecer funcionalidades que son difíciles de implementar en sitios web, como el rastreo de ubicación GPS, acceso a la cámara del dispositivo, y la capacidad de escanear documentos. Estas funciones pueden mejorar significativamente la experiencia del usuario.

2.2.2 Desventajas de las aplicaciones móviles

Dificultad de creación: Crear una aplicación móvil puede ser un proceso complicado y costoso. Puede requerir la contratación de desarrolladores especializados y un diseño diferente al del sitio web de la empresa.
Coste de Disponibilidad: Publicar una aplicación en varias plataformas (App Store de iOS, Google Play, etc.) implica costes adicionales. Además, mantener la app en estas plataformas puede resultar caro con el tiempo.

Seguridad: Garantizar la seguridad de una aplicación móvil es un desafío constante. Se requiere inversión continua en medidas de seguridad para proteger los datos de los usuarios contra hackers y programas malignos [9].

2.3 Sistema operativo Android

Según Urrutia "Android es un sistema operativo para teléfonos inteligentes desarrollado por la empresa estadounidense Google. Su objetivo inicial fue promover el uso de un sistema de tipo abierto, gratuito, multiplataforma, seguro y adaptado a dispositivos móviles como smartphones y tabletas, basado en Linux. El sistema ha hecho un gran esfuerzo para atraer a desarrolladores desde su lanzamiento, por lo que incluye una versión de Java llamada Dalvik, que facilita la creación de aplicaciones que utilicen las características de los dispositivos de manera fácil y sencilla donde el logotipo de la marca es un androide verde con formas redondeadas y un diseño simple" [10].

2.4 Arquitectura del sistema operativo Android

Además, Vanegas menciona que "La arquitectura de Android está formada por cuatro capas, o niveles, que dan al programador la capacidad de crear aplicaciones. Su arquitectura es tipo pila porque su distribución facilita el acceso a las diferentes capas a través de librerías y cada capa utiliza los elementos de la capa inferior para realizar sus funciones" [11], A continuación, como se observa en la figura 2 se muestra la arquitectura del sistema operativo Android.

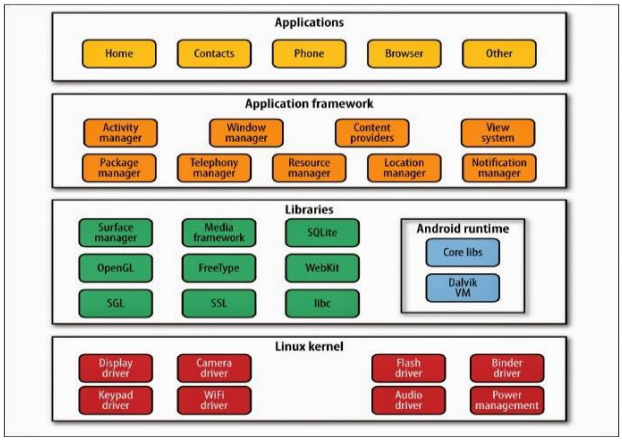


Figura 2: Arquitectura del sistema operativo Android

Fuente: [12]

2.5 Sistema operativo IOS

Según Rocío “iOS es un sistema operativo desarrollado por Apple, inicialmente conocido como iPhone OS debido a su lanzamiento para el iPhone. Más tarde, el sistema se extendió para ser utilizado en otros dispositivos de Apple como iPods y iPads (hasta la llegada de iPadOS). El nombre "iOS" fue adoptado en 2010 para reflejar su uso en una gama más amplia de productos de la compañía. iOS es un sistema operativo cerrado y exclusivo para los dispositivos de Apple, lo que significa que solo puede ejecutarse en hardware diseñado específicamente por la marca de Cupertino. A diferencia de Android, que es utilizado por múltiples fabricantes de dispositivos móviles, iOS está diseñado para ofrecer una integración profunda con el hardware y el ecosistema de servicios de Apple” [13].

2.6 Aplicaciones multiplataforma

Cetys menciona que “una aplicación multiplataforma es aquella que funciona en múltiples plataformas. Suele ser más fácil desarrollar y mantener este tipo de aplicaciones que las nativas. Esto se debe al hecho de que se crearon con un solo código base.
Una gran ventaja de estas aplicaciones es que les permite a los desarrolladores crear una aplicación una vez y luego desplegarla en una variedad de dispositivos. Esto ahorra tiempo y dinero en costos de desarrollo y mantenimiento.
En la actualidad, podemos encontrar una gran cantidad de herramientas que ayudan significativamente a los desarrolladores a crear estas aplicaciones. Sin ellas, deberían diseñarse como aplicaciones híbridas, páginas web o aplicaciones nativas independientes” [14].

Como se aprecia en la Figura 3, se compara el desarrollo de aplicaciones móviles usando Flutter versus el desarrollo nativo, así como las herramientas necesarias.

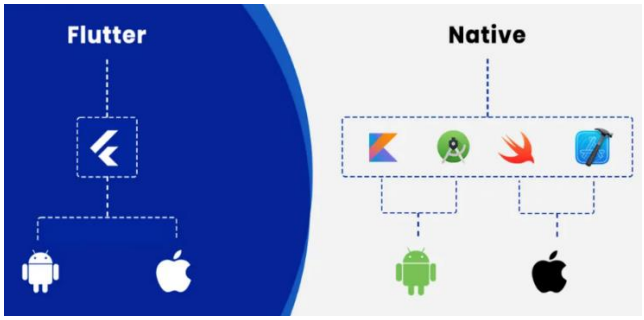


Figura 3: Flutter como framework multiplataforma en el desarrollo de software

Fuente: [15]

2.7 Framework

Meijomil, menciona que “un framework es un conjunto predefinido de herramientas, pautas y estructuras que se utilizan para desarrollar y organizar software de manera efectiva. Funciona como un "esqueleto" o plataforma base sobre la cual los desarrolladores y programadores pueden crear y personalizar sus aplicaciones. ofrece un conjunto de prácticas estandarizadas y partes reutilizables que aceleran el proceso de desarrollo. Más allá del desarrollo de software, los marcos se están volviendo cada vez más relevantes para sectores como marketing digital, desarrollo web y SEO porque permiten a los profesionales de estas

áreas trabajar de manera más organizada, eficiente y colaborativa, aprovechando las ventajas de la automatización y estandarización de procesos" [16] .

2.8 Flutter

Según Amazon Flutter es un marco de código abierto desarrollado por Google que permite la creación de aplicaciones móviles multiplataforma.
A continuación, en la tabla 2 se especifican las características clave de Flutter que impactan en el desarrollo y la experiencia de programación.

Tabla 2: Características claves del framework Flutter

Característica	Descripción
Compilación en código máquina	Se compila en código máquina que entiende directamente los dispositivos, lo que permite un rendimiento casi nativo.
Catálogo de widgets	Amplio catálogo de widgets desde el momento de la descarga, organizados en catorce categorías distintas, incluidos estilos, diseño de material y Cupertino (estilo iOS).
Recarga en caliente (Hot Reload)	La recarga en caliente, también conocida como (hot reload), es una herramienta para desarrolladores que permite previsualizar cambios de código casi al instante sin perder el estado.
Soporte y Comunidad	Soporte de Google y una comunidad de código abierto activa en sitios como Reddit, Discord, Stack Overflow, etc. Desde el lanzamiento de Flutter en 2018, Google lo ha actualizado continuamente

Fuente: [1]

2.9 Ventajas y desventajas de Flutter

2.9.1 Ventajas de Flutter

Según Ebim "Flutter facilita la creación de aplicaciones para los sistemas operativos iOS y Android.
Gracias a su compilación directa a código máquina, las aplicaciones creadas con Flutter tienen un rendimiento casi idéntico al de las aplicaciones nativas.
Al ofrecer una amplia gama de herramientas integradas y widgets personalizables, Flutter facilita el desarrollo de prototipos.
Utilizando widgets flexibles y animaciones fluidas, Flutter permite la creación de interfaces de usuario atractivas y personalizadas" [17].

2.9.2 Desventajas de Flutter

Debido a que Flutter es una tecnología relativamente nueva, lanzada en 2018, puede tener limitaciones o no soportar ciertas funcionalidades.
Aunque la curva de aprendizaje es corta, los desarrolladores sin experiencia previa en Dart o desarrollo multiplataforma pueden encontrarlo difícil.

Debido a que Flutter depende principalmente de su propio ecosistema, la integración con herramientas o librerías que no sean nativas de Flutter puede ser limitada [17].

2.10 Dart

Según Canals, "Dart es un lenguaje orientado a objetos desarrollado por Google, permite a los desarrolladores usar análisis de tipo estático. Han existido muchos cambios en el lenguaje de Dart y sus objetivos principales desde su primera versión estable en 2011. El sistema de tipo Dart pasó de ser opcional a ser estático desde su lanzamiento en la versión 2.0, y desde entonces, el lenguaje se ha enfocado en Flutter".
Ha demostrado ser un lenguaje versátil que puede resolver problemas en una variedad de plataformas, desde aplicaciones móviles hasta aplicaciones de escritorio, a pesar de su origen en el desarrollo web.
Una característica distintiva de Dart es su máquina virtual (VM), que permite una ejecución rápida y un desarrollo más fluido gracias a características como el "Hot Reload", que facilita la visualización instantánea de cambios en el código sin reiniciar toda la aplicación" [18].

2.11 Ventajas y desventajas de Dart

Según Massango Dart se ha convertido en un lenguaje integral que también funciona en el backend, a continuación, se analiza tanto los beneficios como los inconvenientes de usar Dart para el desarrollo.

2.11.1 Ventajas de Dart

Alto rendimiento: Los programas desarrollados en Dart generalmente funcionan más rápido que los desarrollados en JavaScript.

La compilación Ahead of Time (AOT) y Just in Time (JIT) están disponibles en Dart. AOT transforma directamente el código Dart en código de máquina nativo, mientras que JIT permite ciclos de desarrollo rápidos.

2.11.2 Desventajas de Dart

Comunidad limitada: En comparación con otros lenguajes, la comunidad de desarrolladores de Dart sigue siendo más pequeña a pesar de su crecimiento.
Dart sigue evolucionando: Por lo que puede causar cambios en la API y documentación incompleta [19].

2.12 Aplicaciones offline

Según Roces "una aplicación móvil offline es la que permite que los usuarios la usen sin necesidad de estar conectados a Internet. Esto es crucial debido a que internet está cada vez más presente en nuestras vidas, hay momentos en los que podemos quedarnos sin conexión a internet, como en un estacionamiento subterráneo, en la montaña o cuando viajamos a otro país sin pagar una tarifa de datos. Los usuarios de las aplicaciones están acostumbrados a utilizar las aplicaciones sin preocuparse por la conexión; por ejemplo, cuando se envía un mensaje a través WhatsApp o un correo electrónico, aunque no se posea conexión a internet la aplicación se encargará de enviar esta información cuando la conexión se reestablezca" [20].

2.13 Ficha familiar

Una ficha familiar es un documento en la salud pública y la asistencia social, según Montoya "sirve para recopilar información detallada sobre una familia específica. Para comprender y evaluar las necesidades y condiciones de vida de la familia en cuestión, esta ficha suele incluir datos sociodemográficos importantes, como la composición de la familia, las edades de los miembros, el nivel educativo, la ocupación laboral, el estado de salud general, las condiciones de vivienda y los ingresos económicos, entre otros aspectos relevantes" [21].

2.14 ProgressSQL

Las bases de datos relacionales, como PostgreSQL, según Microsoft hacen que sea más fácil organizar y entender cómo se relacionan los datos. La confiabilidad de PostgreSQL, una base de datos relacional de código abierto que ha sido desarrollada durante 30 años, destaca. Su gran flexibilidad e integridad lo hacen muy popular entre desarrolladores y administradores [22].
Por ejemplo, PostgreSQL admite consultas relacionales y no relacionales, y su naturaleza de código abierto permite a una comunidad activa de más de 600 colaboradores mejorar constantemente el sistema de base de datos.

Características eficaces:
Muchas características avanzadas de PostgreSQL incluyen registro de escritura previa, recuperación a un momento determinado, controles de acceso detallados, espacios de tabla, transacciones anidadas, copias de seguridad en línea y control de simultaneidad multiversión [22].

2.15 SQLite

SQLite es una biblioteca embebida que proporciona un motor de base de datos SQL independiente, en la cual no se requiere un servidor, configuración o instalación, y que soporta transacciones. Esta base de datos es de origen público por lo cual se pude usar para cualquier propósito, ya sea de carácter comercial o privado. Es la base de datos más empleada a nivel mundial, con usos en varios proyectos sobresalientes [23].

SQLite además es una pequeña biblioteca que pude tener un peso menor de 750 KiB con todas las funciones habilitadas. Aunque su rendimiento aumenta con más memoria, es demasiada eficiente en dispositivos con recursos limitados. Además, esta base de datos es incluso mucho más rápida que las operaciones de E/S directas en el sistema de archivos [23].

2.16 Fiabilidad offline con SQLite

La fiabilidad en bases de datos locales contiene la capacidad de almacenar y administrar datos de manera consistente y segura, incluso en condiciones de fallo. Es esencial para aplicaciones que operan offline y para la sincronización.

Tabla 3: Fiabilidad de la base de datos en SQLite

Método	Descripción
Uso de transacciones	Garantiza que las operaciones de base de datos sean atómicas, evitando inconsistencias al asegurar que todas las operaciones se completan correctamente o no se realizan.
Integridad referencial	Uso de restricciones como FOREIGN KEY y NOT NULL para mantener la coherencia y precisión de los datos, asegurando que las relaciones entre tablas no generen inconsistencias.
Manejo de errores	Utilización de bloques try-catch para capturar y manejar errores de manera controlada, evitando que una falla interrumpa todo el flujo de la aplicación.
Indicadores de Estado	Uso de campos de tipo bandera para marcar si los datos han sido sincronizados con el servidor, lo que permite controlar el flujo de datos y evitar duplicados u omisiones en la sincronización.
Uso de await en operaciones asíncronas	Await se utiliza para esperar de manera eficiente que las operaciones asíncronas, como las solicitudes de red o la actualización de la base de datos, se completen antes de proceder a la siguiente operación.

2.17 Jest

Jest es un framework de código abierto, diseñado principalmente para probar aplicaciones JavaScript, el cual puede gestionar la ejecución de las pruebas, incluyendo la paralelización y la recolección de resultados, Además no solo es útil para pruebas unitarias y de componentes, sino que también es una excelente para probar endpoints. Gracias a que Jest usa bibliotecas adicionales como Supertest, permite realizar solicitudes HTTP y verificar respuestas.

2.18 Jmeter

Es una herramienta de código abierto diseñada para realizar pruebas de rendimiento y carga en aplicaciones web, servicios REST/APIs, bases de datos, y otros servicios, siendo desarrollada en 1998, su principal objetivo es medir el rendimiento de sistemas bajo diferentes cargas de trabajo y analizar su escalabilidad.

2.19 JavaScript

El uso de scripts en JavaScript para simular fallos controlados y medir su respuesta es una técnica práctica y eficaz. Estos scripts permiten crear escenarios que interrumpen temporalmente el servicio y poder monitorean el proceso de recuperación, proporcionando datos clave sobre métricas como el tiempo medio de recuperación (MTTR) y la fiabilidad del sistema.

2.20 Node.js

Node.js es un entorno para realizar ejecución de JavaScript multiplataforma, además node.js es de código abierto y gratuito, diseñado para ejecutar código fuera de un navegador, se basa en la versión del motor V8 de Chrome, lo que permite crear servidores, aplicaciones web dinámicas, herramientas de línea de comando, scripts automatizados y sistemas backend eficientes. Debido a que su arquitectura se basada en modelos de entrada/salida no bloqueante, permite crear aplicaciones escalables y de alto rendimiento, como sistemas en tiempo real, APIs RESTful y microservicios.

2.21 Normas ISO 25010

Según ISO la norma ISO 25010 define ocho características principales de calidad de un producto de software, a continuación, en la tabla 4 se describen estas características:

Tabla 4: Criterios de la norma ISO 25010

Criterio	Descripción
Adecuación funcional	El grado en que el producto proporciona funciones que cumplen con los requisitos explícitos e implícitos cuando el software se usa en condiciones específicas.
Eficiencia de rendimiento	El grado en que el producto proporciona un rendimiento adecuado en condiciones establecidas, de acuerdo con la cantidad de recursos utilizados.
Compatibilidad	El grado en que un producto, sistema o componente puede comunicarse con otros productos, sistemas o componentes y/o realizar funciones requeridas mientras comparten el mismo entorno y recursos.
Usabilidad	El grado en que un producto puede ser utilizado por usuarios particulares para alcanzar objetivos específicos con eficacia, eficiencia y satisfacción en un contexto de uso particular.
Fiabilidad	Grado en que un sistema, producto o componente desempeña funciones especificadas bajo condiciones establecidas durante un período determinado.
Seguridad	La medida en que un producto o sistema protege la información y los datos, permitiendo el acceso a personas, productos o sistemas en función de sus tipos y niveles de autorización.
Mantenibilidad	El nivel de eficacia y eficiencia con el que el producto permite modificaciones e inspecciones.

Portabilidad	El grado en que un sistema, producto o componente puede ser transferido con efectividad y eficiencia entre diferentes hardware, software o entornos operativos o de uso.

Fuente: [24]

2.22 Metodología Mobile-D

Según Syntonize basándose en la programación no ágil, las metodologías Crystal y el Proceso Unificado Racional aplicados de manera estricta para otros tipos de desarrollos, esta metodología ágil tiene como objetivo permitir ciclos de desarrollo muy rápidos en equipos pequeños. a continuación, en la tabla 5 se especifican las etapas de exploración, iniciación, producto, estabilización y prueba.

Tabla 5: Etapas de la metodología ágil de desarrollo Mobile-D

Etapas	Descripción
Exploración	Se centra en los conceptos fundamentales del proyecto y la planificación. El alcance y las funcionalidades que se pretenden implementar están definidos.
Iniciación	El proyecto se configura identificando y preparando todos los recursos necesarios.
Producto	Las subfases de la fase de producto se repiten iterativamente. El desarrollo dirigido por pruebas (TDD) implica que, antes de desarrollar una funcionalidad, se realice una prueba para asegurar su correcto funcionamiento.
Estabilización	En esta fase se deben unir los diversos módulos en una sola aplicación.
Pruebas	Finalizado el desarrollo, se pasa a una fase de prueba hasta alcanzar una versión estable según lo acordado. De ser necesario, se corrigen errores, pero no se desarrollan nuevas funcionalidades.

Fuente: [25]

Adicionalmente en la Figura 4 se observan las cinco fases de la metodología Mobile-D.

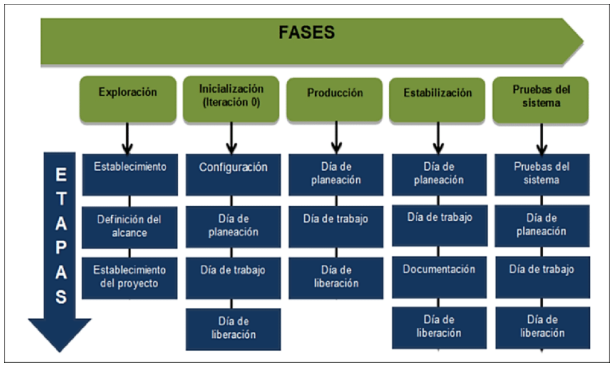


Figura 4: Fases de la metodología de desarrollo Mobile-D

Fuente: [26]

CAPÍTULO III. METODOLOGIA

3.1 Tipo de investigación

En el presente trabajo, el tipo de investigación desarrollado fue una investigación cuantitativa, ya que se creó una aplicación móvil offline multiplataforma para el Centro de Salud tipo B Chambo, utilizando tecnologías como Flutter y considerando los estándares ISO/IEC 25010. Esto permitió medir la fiabilidad en términos de: ausencia a fallos, tolerancia a fallos, capacidad de recuperación sobre la aplicación desarrollada para la transmisión de información hacia la base de datos de Ficha Familiar, facilitando la gestión eficiente de datos y beneficiando al personal médico.

3.2 Diseño de la investigación

La investigación aplicó métodos cuantitativos. El enfoque cuantitativo permitió medir y analizar la fiabilidad de los datos transmitidos y de sincronización, además se definió las tecnologías y métodos adecuados para la implementación de aplicaciones móviles offline mediante una revisión bibliográfica.

3.3 Población de estudio y tamaño de la muestra

La población para la investigación fue infinita, ya que se utilizaron herramientas de software y métodos con una carga de concurrencia simulada para medir cuantitativamente la fiabilidad de la aplicación móvil offline multiplataforma en términos de ausencia de fallos, tolerancia a fallos y capacidad de recuperación.

3.4 Técnicas de recolección de datos

Se utilizaron las siguientes técnicas de recolección de datos para evaluar la fiabilidad de la aplicación móvil offline multiplataforma para el ingreso de datos de ficha familiar, de acuerdo con la norma ISO/IEC 25010:

Simulación de escenarios de uso

Se probó cómo la aplicación manejó la entrada y sincronización de datos en diferentes condiciones de conectividad, para lo cual se crearon escenarios controlados, donde se evaluó la ausencia a fallos, tolerancia a fallos y capacidad de recuperación de datos de la ficha familiar.

3.5 Procedimiento

A continuación, se definieron las actividades que se realizaron en el trabajo de investigación de acuerdo con los objetivos.

- Realizar una investigación bibliográfica sobre los métodos y tecnologías aplicables para garantizar fiabilidad en aplicaciones móviles offline.

- Elaborar tablas con las características generales tanto de métodos como tecnologías aplicables al desarrollo de aplicaciones móviles offline.

- Escoger los métodos y tecnologías a utilizar para desarrollar la aplicación móvil offline.

- Realizar una investigación de la metodología de desarrollo a usar para el desarrollo ágil usando la metodología Mobile-D.

- Identificar los requisitos funcionales y no funcionales de acuerdo con el requerimiento del centro de salud Chambo para la elaboración de la aplicación móvil offline.

- Definir la estructura funcional de sincronización que garantice el correcto funcionamiento de la aplicación móvil, tanto en modo offline como durante la sincronización de la ficha familiar.

- Desarrollar los módulos e interfaces necesarias para asegurar el funcionamiento eficiente de la aplicación móvil offline.

- Adecuar los métodos y tecnologías aplicables para garantizar fiabilidad en aplicaciones móviles offline.

- Evaluar la métrica de ausencia a fallos del sistema bajo condiciones de funcionamiento normales aplicando Jest.

- Valorar la métrica de tolerancia a fallos verificando su capacidad para mantener la operatividad ante fallos parciales usando logs.

- Evaluar la métrica de capacidad de recuperación con el número de recuperaciones exitosas como el tiempo medio de recuperación, usando JavaScript y Jmeter.

- Finalmente se realizaron las conclusiones y recomendaciones de los objetivos del trabajo de investigación.

3.6 Métodos de análisis y procesamiento de datos

Para evaluar cómo manejó la aplicación móvil offline multiplataforma las diversas situaciones, se llevaron a cabo simulaciones de fallos o interrupciones durante las pruebas de campo y simulaciones de escenarios de uso con distintos métodos o programas de software. Los datos recopilados durante estas simulaciones permitieron analizar cómo la aplicación respondió a estas distintas situaciones.

La información recopilada durante las simulaciones de fallos e interrupciones determinó cómo se procesaron los datos. Para evaluar cómo manejó las interrupciones y las fallas, los datos fueron analizados para examinar el comportamiento de la aplicación en diversas situaciones. Asimismo, para comprender el funcionamiento de la aplicación y su capacidad de recuperación, se examinaron los registros de eventos y errores. Al asegurar que cumplió con los objetivos de sincronización y gestión de datos en diversas condiciones operativas, la información recopilada permitió realizar una evaluación general de la fiabilidad de la aplicación.

3.7 Identificación de variables

3.7.1 Variable dependiente

Fiabilidad en el proceso de registro de ficha familiar.

3.7.2 Variable independiente

Aplicación móvil offline multiplataforma.

3.8 Operacionalización de variables

A continuación, la tabla 6, presenta la operacionalización de variables

Tabla 6: Operacionalización de variables

PROBLEMA	TEMA	OBJETIVOS	VARIABLES	CONCEPTUALIZACION	DIMENSION	INDICADORES
¿Cómo la aplicación móvil offline multiplataforma garantizara la fiabilidad en el proceso de registro de fichas familiares en el Centro de Salud Chambo?	Aplicación Móvil Offline multiplataforma para ingreso de datos de ficha familiar del Centro de Salud Chambo usando el framework Flutter.	GENERAL Implementar una aplicación móvil offline multiplataforma para ingreso de datos de ficha familiar del centro de salud tipo b de chambo usando el framework Flutter. ESPECIFICOS • Investigar métodos y tecnologías que permitan garantizar la fiabilidad en aplicaciones offline. • Desarrollar la aplicación móvil offline multiplataforma para el ingreso de datos de ficha familiar con Mobile-D. • Evaluar la fiabilidad de la aplicación móvil offline multiplataforma utilizando la norma ISO / IEC 25010.	INDEPENDIENTE Aplicación móvil offline multiplataforma DEPENDIENTE Fiabilidad en el proceso de registro de ficha familiar.	Una aplicación móvil offline es un programa multiplataforma que funciona con múltiples sistemas operativos móviles como Android e iOS sin conexión a internet. La fiabilidad de la Aplicación móvil offline multiplataforma según el estándar ISO 25010 implica verificar la capacidad de la aplicación para funcionar sin fallos significativos en condiciones normales de operación, basándose en los principios de calidad del software enfocado en la fiabilidad.	Desarrollar una aplicación móvil offline multiplataforma. Fiabilidad de Software ISO 25010	Independiente. • Módulos Implementados. • Tamaño de la aplicación. • Tiempo de desarrollo. Dependiente. • Ausencia a fallos. • Tolerancia a fallos. • Capacidad de Recuperación.

3.9 Metodología de desarrollo de la aplicación móvil offline

La metodología de desarrollo que se usó fue Mobile-D que tuvo un enfoque ágil diseñado específicamente para el desarrollo rápido y eficiente de aplicaciones móviles, sus fases fueron: Exploración, Iniciación, Producción, Estabilización y pruebas.

3.9.1 Exploración

En esta fase se llevó a cabo una evaluación del requisito inicial para el desarrollo de una aplicación móvil multiplataforma en Flutter, con el objetivo de abordar las limitaciones identificadas en el proyecto de titulación previo. Si bien el proyecto anterior representó un avance en la gestión de datos de ficha familiar, se evidenció que la aplicación desarrollada no cubría la necesidad de operar en zonas con conectividad limitada, un escenario recurrente en la parte rural del cantón Chambo.

Durante una reunión con el Dr. Pablo Álvarez, responsable del apoyo técnico de la gestión de las fichas familiares en el Centro de Salud Chambo, se identificó el principal requerimiento para esta nueva solución. También, se expuso la necesidad de una aplicación offline que permitiera ingresar datos de ficha familiar y modificar información médica de manera local en los dispositivos móviles, con la capacidad de sincronizar la información automáticamente cuando se restableciera la conectividad. Además, este nuevo desarrollo debía aprovechar la estructura de bases de datos diseñada previamente, garantizando compatibilidad e integración con el sistema existente.

a) Definición de roles y responsabilidades

Desarrolladores: Responsables del diseño y programación de la aplicación multiplataforma que emplea Flutter. Su obligación consistió en establecer una arquitectura que facilite el registro y sincronización de datos de forma offline, garantizando que estos se sincronicen de manera automática cuando se restablezca la conexión.

Especialistas del Centro de Salud Chambo: Principales usuarios de la aplicación. Su feedback durante las etapas de diseño y evaluación fue esencial para asegurar que la herramienta responda a los requerimientos en situaciones reales.

Supervisor del sistema: Encargado de incorporación y conservación de la aplicación en los sistemas ya existentes del centro de salud chambo, así como otorgar los accesos necesarios a los sistemas funcionales.

Paciente: Fueron las personas beneficiarias del sistema a pesar de que no tiene una interacción directa con la aplicación.

b) Requisitos funcionales

Se desarrolló una aplicación móvil multiplataforma utilizando el framework Flutter con el IDE de Visual Studio Code, con el propósito de gestionar de manera eficiente las fichas familiares en el Centro de Salud Chambo, abordando las limitaciones detectadas en base a la información recopilada durante la reunión con el Dr. Pablo Álvarez responsable del apoyo técnico en este proyecto, A continuación, en la tabla 7 se visualizan los principales requerimientos.

Tabla 7: Requerimientos funcionales de la aplicación móvil offline

Identificador	Nombre del Requerimiento	Descripción	Prioridad
RF01	Login	Valida el acceso al personal del Centro de Salud Chambo	Alta
RF02	Nueva ficha familiar offline	Permite registrar datos del domicilio, integrantes de la familia, embarazadas, personas fallecidas y datos del doctor responsable.	Alta
RF03	Modificación de una ficha familiar offline	Permite modificar el personal responsable, datos de mortalidad, embarazadas y los miembros de la familia.	Alta
RF04	Registro Offline	Permite que toda la aplicación funcione sin conexión a internet, garantizando la operatividad en áreas sin cobertura.	Alta
RF05	Migración de Datos	Los datos recopilados offline deben migrarse al servidor con possgress cuando haya conexión. Las fichas incompletas se mantendrán en una bandeja de pendientes.	Alta
RF06	Estado de Avance	Visualiza el avance de las fichas familiares a nivel de menú.	Media
RF07	Almacenamiento Incremental	Guarda automáticamente la información para evitar pérdida de datos ante apagones, desconexiones o problemas del dispositivo.	Alta
RF08	Navegación entre Secciones	Permite navegar entre diferentes partes de la ficha sin perder datos ya ingresados, con subformularios interrelacionados para una recopilación más fluida.	Alta
RF09	Controles de Coherencia	Implementa validaciones que bloqueen selecciones inconsistentes, como seleccionar categorías incompatibles según edad, sexo o estado (ej., un hombre no puede ser embarazada).	Media

RF10	Captura de Fotos	Activa la funcionalidad de captura de imágenes directamente desde el dispositivo, optimizando el tamaño a menos de 1 MB para evitar sobrecargas de almacenamiento.	Bajo
RF11	Registro y Control de Campos Obligatorios	Define campos obligatorios específicos como "Observaciones" y asegura que estos puedan ser completados con información relevante.	Media

c) Requisitos no funcionales

Los requerimientos no funcionales otorgaron el funcionamiento de la aplicación móvil, a continuación, en la tabla 8 se observan estas características principales.

Tabla 8: Requerimientos no funcionales de la aplicación móvil offline

Identificador	Nombre del Requerimiento	Descripción	Prioridad
RNF01	Lenguaje de Desarrollo	La aplicación móvil usara el framework Flutter con Dart.	Alta
RNF02	Compatibilidad de Plataformas	La aplicación móvil será compatible con varios dispositivos Android y IOS.	Alta
RNF03	Interfaz de Usuario	La aplicación móvil tendrá una interfaz intuitiva para conceder a los médicos una experiencia óptima.	Alta
RNF04	Base de Datos	La información de la aplicación offline será almacenada en SQLite y la sincronización de esta en possgress SQL.	Alta
RNF05	Sincronización	La sincronización de datos entre la base de datos offline y online se realizará a través de endpoints.	Alta

d) Planeación y estimación

A continuación, la tabla 9 presenta el cronograma de actividades planificadas para el desarrollo de la aplicación móvil offline de ficha familiar.

Tabla 9: Cronograma de actividades

N.º	ACTIVIDADES	SEMANAS															
		MES 1				MES 2				MES 3				MES4			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1.	**Diseño de la Investigación**																
1.1	Planificación del trabajo del proyecto investigación.	X															
1.2	Desarrollo y Aprobación del Tema.		X														
1.3	Elaboración de perfil de tesis.			X	X												
2.	**Tutorías del proyecto de Investigación**																
2.1	Análisis de la elaboración de la ficha familiar en el Centro de salud Chambo					X	X	X	X								
2.2	Revisión de la documentación del trabajo de titulación.					X	X	X	X	X	X	X	X	X	X	X	X
3.	**Trabajo Autónomo**																
3.1	Investigación sobre diseño de bases de datos offline.					X	X	X	X								
3.2	Levantamiento de información de la ficha familiar					X	X	X									
3.3	Desarrollo del diseño de la aplicación móvil.					X	X	X	X	X	X	X	X				
3.4	Implementación de la funcionalidad offline en Flutter.									X	X	X	X				
3.5	Pruebas de funcionalidad offline y sincronización de datos.									X	X	X	X				
4.	**Desarrollo de proyecto de Investigación**																
4.1	Análisis de los requerimientos para el desarrollo de la Aplicación						X	X									
4.2	Desarrollo de la aplicación móvil offline multiplataforma						X	X	X	X	X	X	X				
4.3	Lanzamiento de la Aplicación Móvil Offline.												X	X			
4.4	Evaluar la ausencia a fallos de la Aplicación Móvil Offline. utilizando la ISO 25010														X	X	X
5.	**Documentación**																
5.1	Desarrollo del Trabajo escrito de Titulación.					X	X	X	X	X	X	X	X	X	X	X	X

3.9.2 Inicialización

En esta etapa se llevaron a cabo actividades relacionadas con el desarrollo y diseño de la aplicación móvil offline.

a) Diseño de la arquitectura de la aplicación móvil

En esta fase se definió la estructura de la aplicación funcional, en la figura 5 se observaron los aspectos clave de la arquitectura.

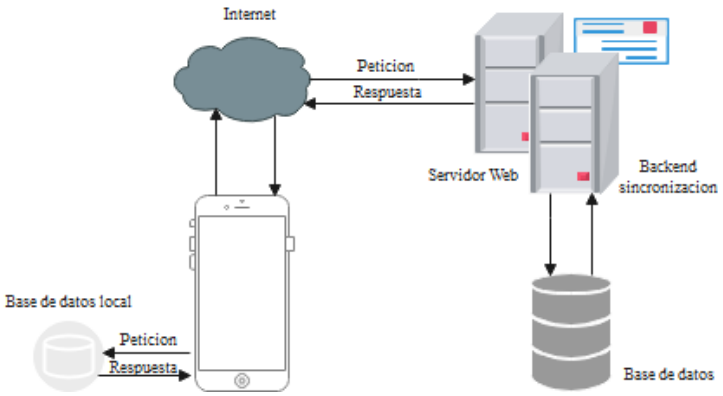


Figura 5: Arquitectura de la aplicación móvil offline

b) Diagrama de casos de uso

Se identificaron en la figura 6 las interacciones entre los usuarios y el sistema, describiendo diferentes escenarios de uso.

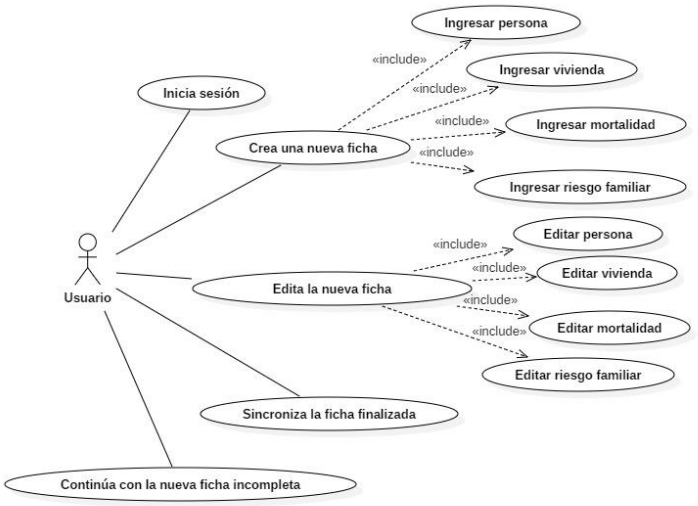


Figura 6: Casos de uso de la aplicación móvil offline

c) Diagrama de actividades

Se modelaron los flujos de trabajo y procesos internos de la aplicación, en la figura 7 se detallaron las acciones y decisiones.

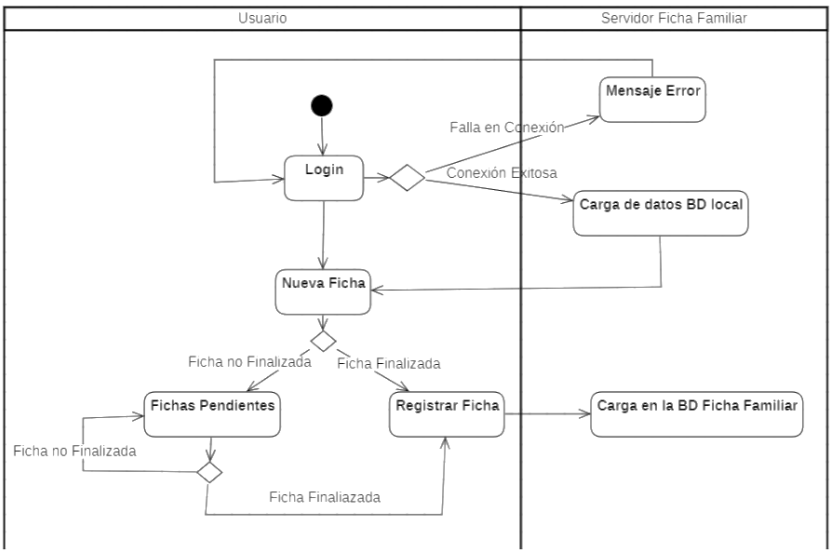


Figura 7: Diagrama de actividades de la aplicación móvil offline

d) Diagrama de componentes

Se especificaron en la figura 8 los elementos técnicos y su interacción, representando la organización de módulos.

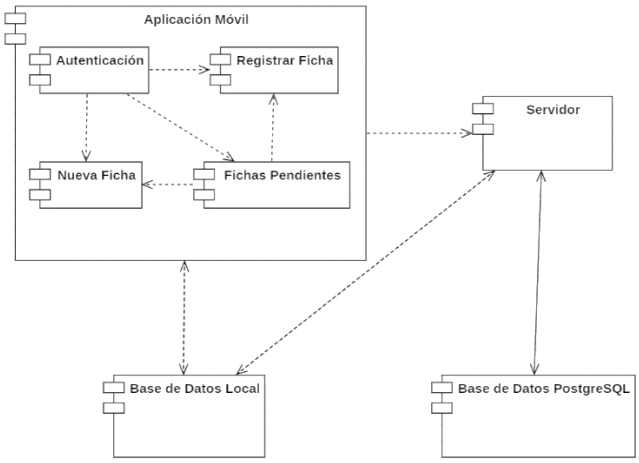


Figura 8: Diagrama de componentes de la aplicación móvil offline

e) Diagrama de base de datos

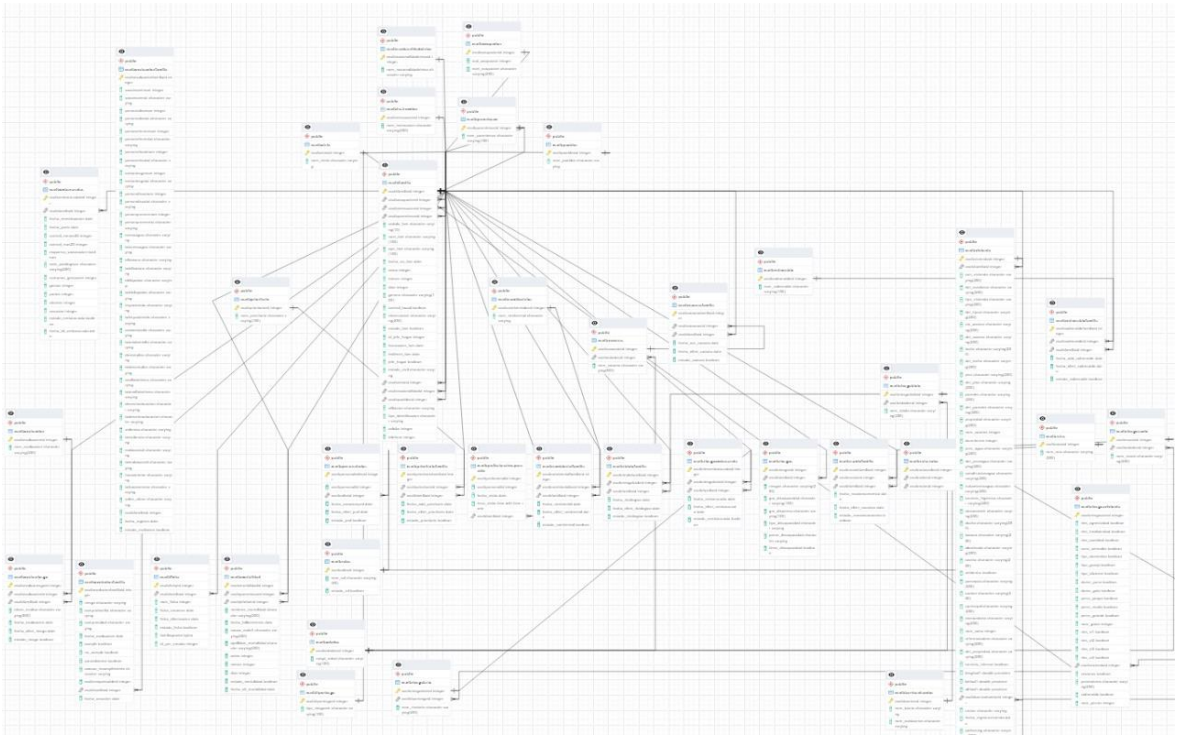


Figura 9: Diagrama físico de la base de datos

f) Diccionario de datos

Tabla 10: Diccionario de datos

Nombre de la Tabla	Atributo	Tipo	Descripción
MSP_DATOS_HOGAR	id_datos_hogar	serial	Llave primaria de la tabla
	nom_institucion	character varying	Nombre de la institución de salud
	unicodigo	character varying	Código de la institución de salud
	nom_establecimiento	character varying	Nombre del establecimiento de salud
	latitud_vivienda	double precision	Dato geográfico
	longitud_vivienda	double precision	Dato geográfico
	altitud_vivienda	double precision	Dato geográfico
	cond_vivienda	character varying	Condición de ocupación de la vivienda
	obser_cond	character varying	Observación de la ocupación

	tipo_vivienda	character varying	Tipo de vivienda que ocupa
	obser_tipo	character varying	Observación tipo de vivienda
	via_acceso	character varying	Principal vía de acceso a la vivienda
	obser_via	character varying	Observación principal vía de acceso
	material_techo	character varying	Material predominante del techo

En el **Anexo 1** se encuentra la extensión detallada del diccionario de datos, que contiene la estructura de datos y especificaciones de los demás módulos de la ficha familiar offline.

g) Esquema de navegabilidad

Se especificaron en la figura 10 los elementos técnicos y las rutas de navegación de la aplicación móvil.

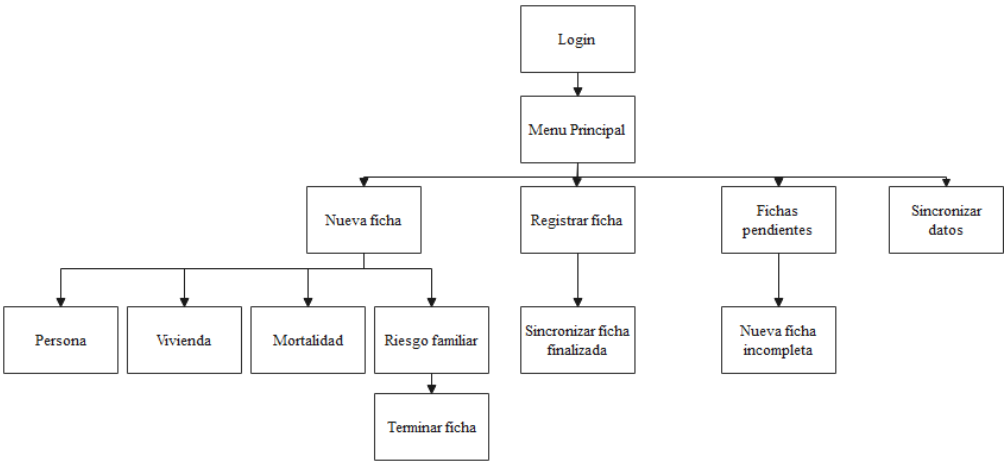


Figura 10: Esquema de navegabilidad de la aplicación móvil offline

h) Modelado de la interfaz

En la figura 11 se encuentra la interfaz del login de la ficha familiar.

Figura 11: Modelo del inicio de sesión

En la figura 12 se observa la pantalla del menú y del ingreso de los módulos de la ficha familiar.

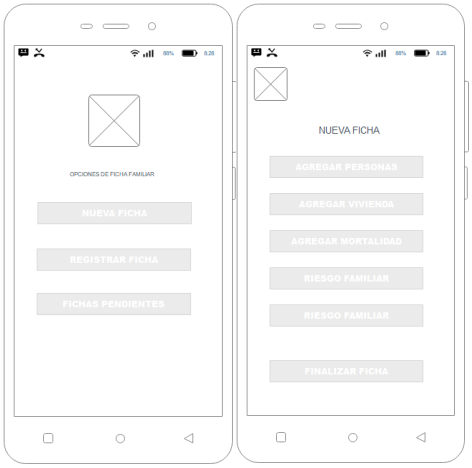


Figura 12: Modelo del menú y módulos.

Se visualiza en la figura 13 la pantalla de agregar datos de cada módulo.

Figura 13: Modelo del ingreso de datos de ficha familiar

En la figura 14 se ilustra las interfaces de las fichas incompletas, así como del registro de las fichas finalizadas.

Figura 14: Modelo para fichas pendientes y finalizadas.

3.9.3 Producción y estabilización

En la fase de producción, las subfases se repiten de manera iterativa. El desarrollo de la aplicación móvil se rige por pruebas (TDD) para desarrollar una funcionalidad de forma independiente y garantizar su funcionamiento. Además, en la estabilización se integran los distintos módulos en una sola aplicación.

a) **Arquitectura limpia basado en componentes**

En la figura 15 se observa la arquitectura de desarrollo organizada en capas con la lógica distribuida de adentro hacia afuera.

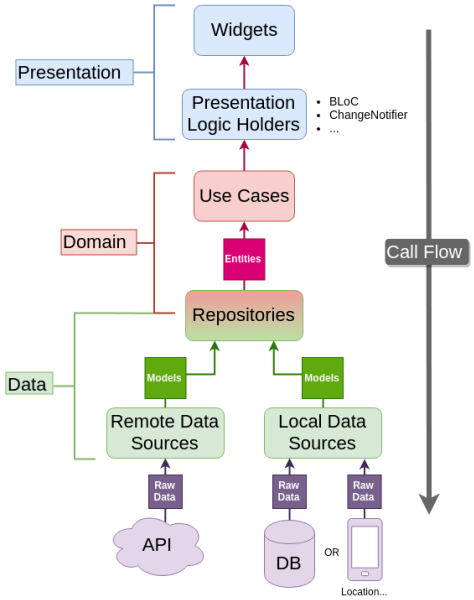


Figura 15: Arquitectura limpia por componentes
Fuente: [27]

b) Capas de codificación

La arquitectura por capas organiza una aplicación en Presentación, Dominio y Datos. La capa de **Presentación** contiene los widgets y maneja la lógica de la interfaz, delegando las tareas complejas a los casos de uso y validando los datos de entrada.

La capa de **Dominio** encapsula la lógica de negocio principal mediante casos de uso y entidades, siendo independiente de otras capas.

La capa de **Datos** implementa los repositorios definidos en el dominio, manejando fuentes de datos remotas (APIs) y locales.

c) Codificación

En la figura 16 se puede observar la conexión y sincronización de la base de datos de ficha familiar a la base de datos offline de la aplicación móvil.

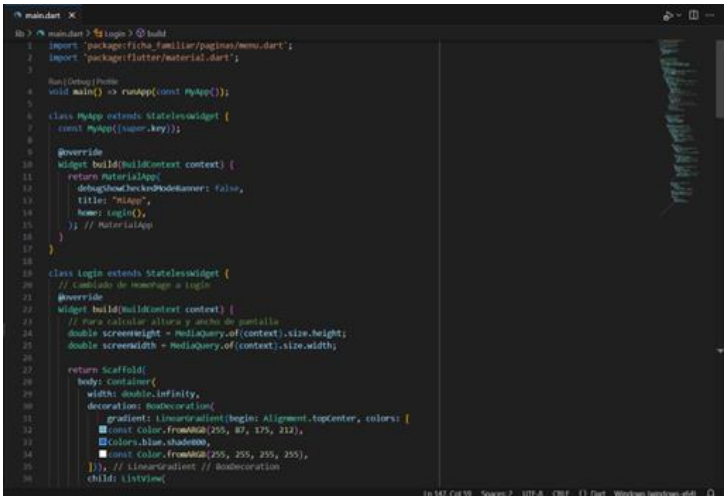


Figura 16: Conexión y sincronización a la base de datos de ficha familiar

La figura 17 el usuario agrega sus credenciales para poder iniciar sesión en la aplicación móvil offline de ficha familiar.

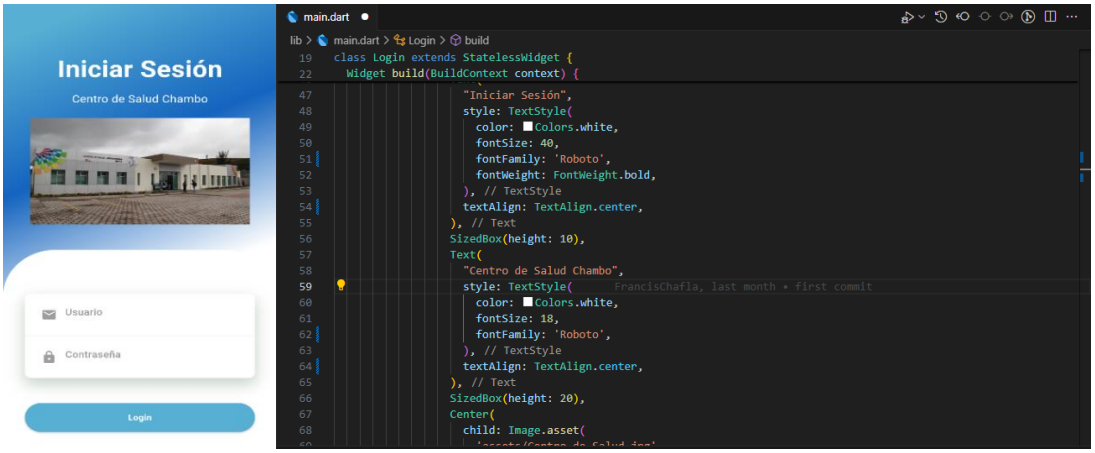


Figura 17: Inicio de sesión

La figura 18 muestra el menú de ficha familiar, donde se presentan las alternativas para crear una nueva ficha, registrar ficha para cargar la ficha familiar completa a la base de datos y fichas pendientes para que el usuario complete las fichas aún sin completar.

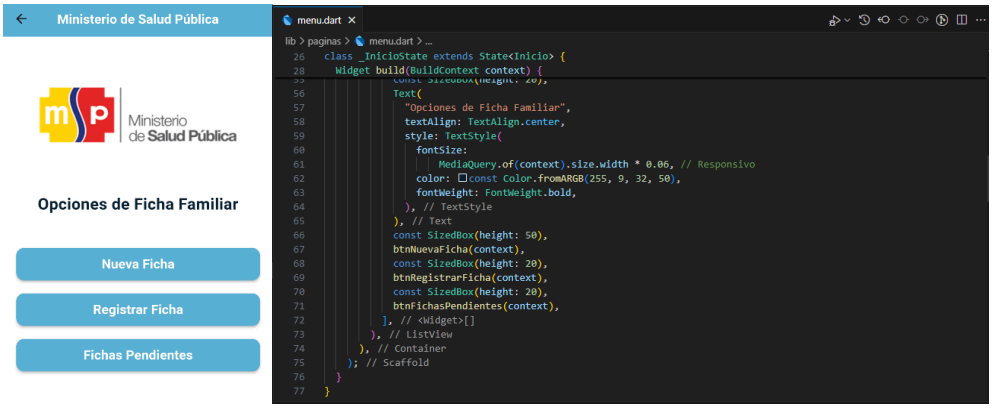


Figura 18: Menú de la ficha familiar

La figura 19 ilustra el módulo para agregar a los miembros de la familia, que incluye datos personales, riesgos, enfermedades, discapacidades, grupos de atención, riesgos familiares y vacunas.

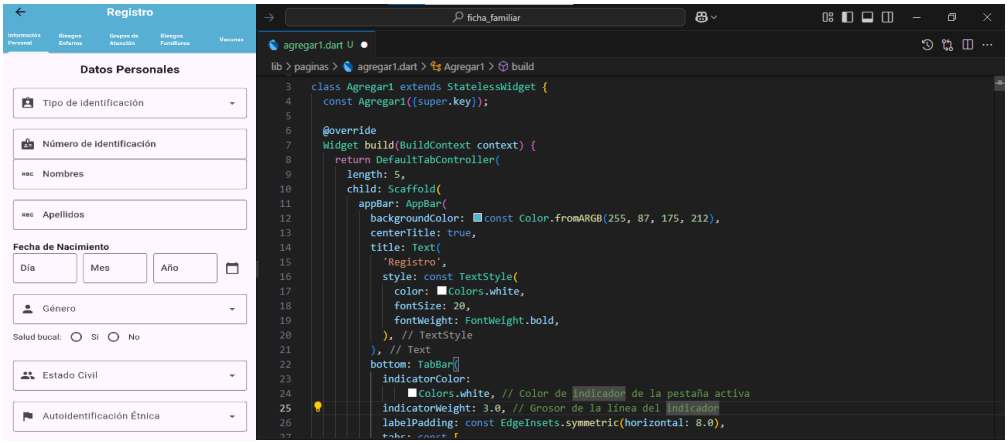


Figura 19: Apartado para agregar los integrantes de familia

La figura 20 ilustra el módulo para agregar información de vivienda, que incluye datos de vivienda, servicios, factores de riesgo y ubicación.

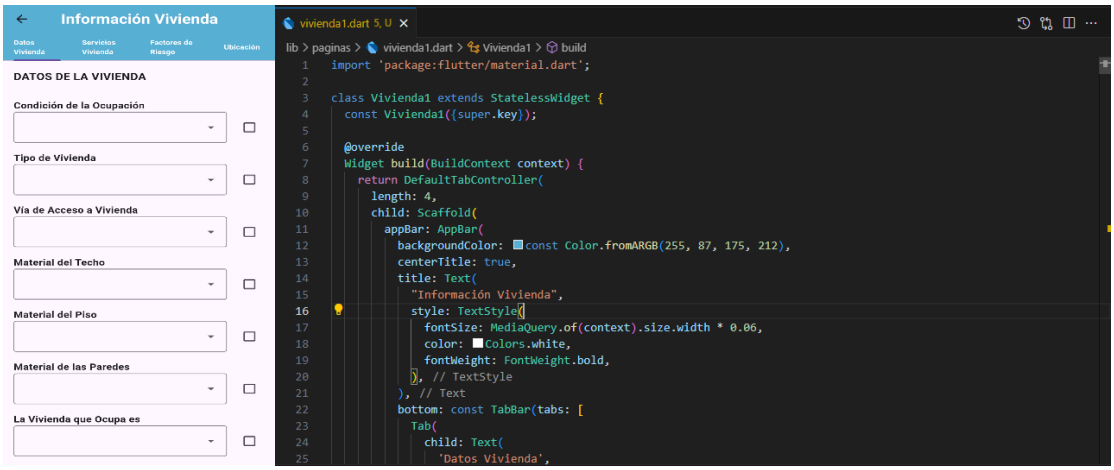


Figura 20: Apartado para agregar información de vivienda

La figura 21 ilustra el módulo para agregar de embarazadas.

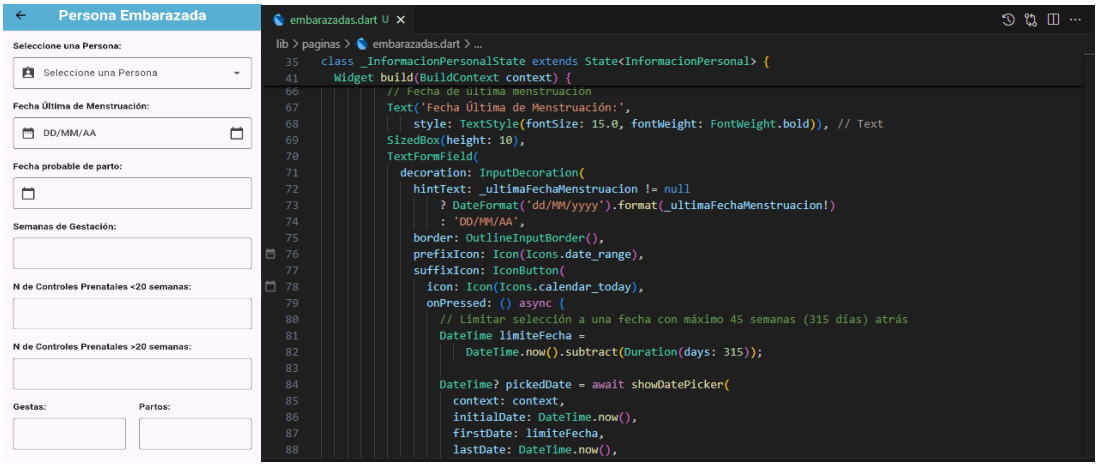


Figura 21: Apartado para agregar persona embarazada

3.9.4 Pruebas de simulación

a) Planificación de las pruebas

Se planificó las pruebas de fiabilidad de software de la aplicación móvil offline de acuerdo con la norma ISO 25010.

1. **Ausencia a fallos.**

Se verificó que las funcionalidades principales del sistema funcionan correctamente en condiciones normales.

Tabla 11: Planificación de ausencia a fallos

Categoría	Descripción
Pruebas realizadas	Creación y sincronización de datos de fichas familiares.
Herramientas utilizadas	Jest para pruebas funcionales.

Indicadores evaluados	**Tasa de errores:** Número de pruebas fallidas / Número total de pruebas. **Meta**: < 10% errores.

2. Tolerancia a fallos.

Se evaluó si el sistema puede continuar funcionando ante fallos parciales.

Tabla 12: Planificación de tolerancia a fallos

Categoría	Descripción
Pruebas realizadas	**Pruebas de desconexión:** Simuló la pérdida de conexión y verifica que la app pueda trabajar offline (guardando datos en SQLite). **Pruebas de datos:** Se insertó datos en la base de datos local y se verificó que no provoquen el colapso de la aplicación.
Herramientas utilizadas	• SQLite Inspector para Flutter. • Simulaciones de fallos para el backend.
Indicadores evaluados	**Porcentaje de continuidad operativa** Número de operaciones exitosas / Total de operaciones probadas. Meta: > 90%.

3. Capacidad de recuperación.

Se comprobó la eficacia para recuperar la funcionalidad después de un fallo en el servicio.

Tabla 13: Planificación de capacidad de recuperación

Categoría	Descripción
Pruebas realizadas	**Recuperación tras desconexión:** Simula desconexión y reconexión, verificando que los datos guardados offline se sincronicen con el backend. **Recuperación tras cierre forzado:** Fuerza el cierre de la app durante una operación y verifica si los datos se restauran correctamente.
Herramientas utilizadas	Script para la simulación de interrupción del backend, así como carga concurrente con Jmeter.
Indicadores evaluados	**Porcentaje de recuperación exitosa:** Número de recuperaciones exitosas / Total de fallos simulados. **Meta:** > 90%.

MTTR ideal : < 5 minutos

---

b) Ejecución de las pruebas.

Indicador de fiabilidad 1 - ausencia a fallos.

Jest para el backend

Se verificó las respuestas de sincronización inicial de la ficha familiar offline haciendo el uso de Jest en el backend, como se observa en la figura 22, bajo distintos niveles de concurrencia, desde 10 hasta 1000 solicitudes simultáneas.

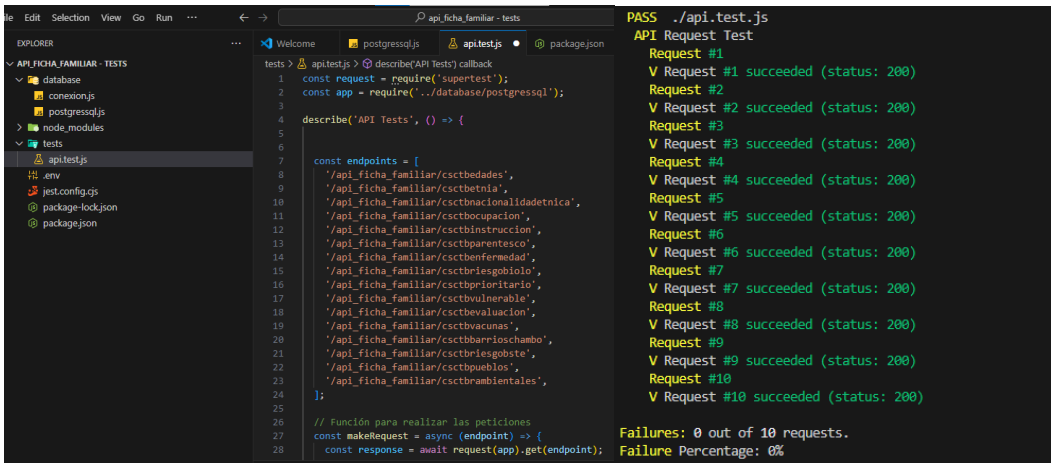
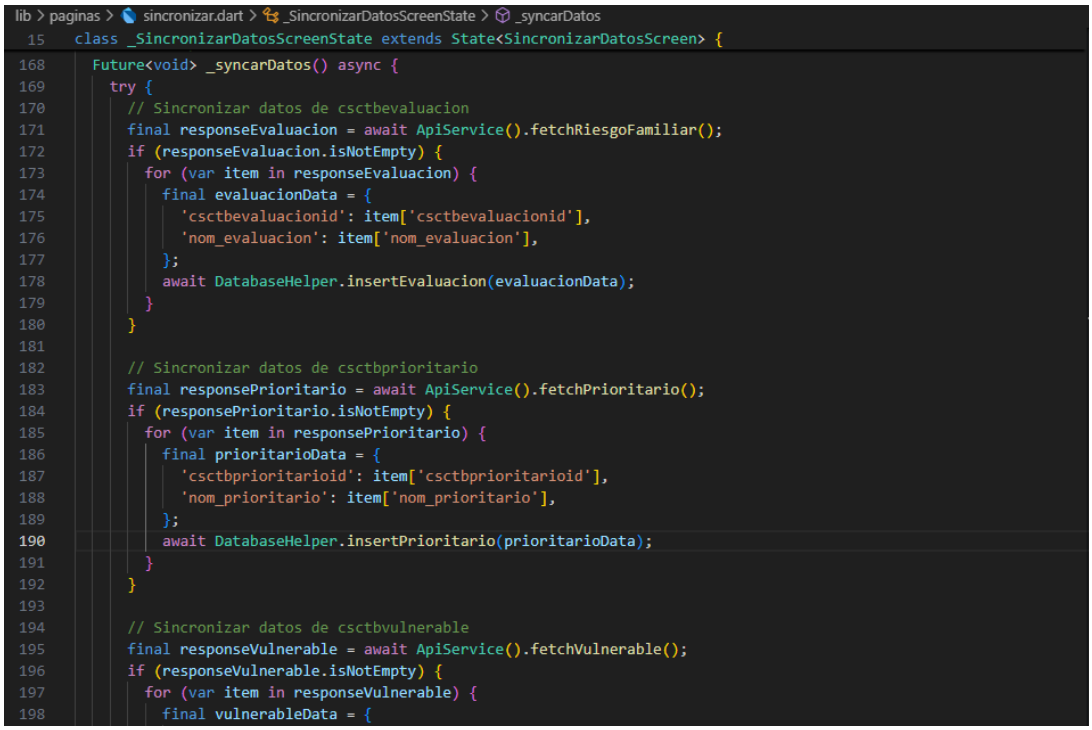


Figura 22: Respuesta de ausencia a fallos de la ficha familiar

Número de pruebas fallidas / Número total de pruebas.
Meta: < 10% errores.

Indicador de fiabilidad 2 - tolerancia a fallos.

El registro de datos se valida mediante el análisis de los logs de ingreso, utilizando la teoría de Chaos Monkey, en esta técnica se ingresaron datos al sistema para evaluar si existen errores inesperados verificando si los datos se registran correctamente, los logs proporcionan evidencia de cómo maneja el sistema los fallos, garantizando la fiabilidad del proceso de registro de datos, como se observa en la figura 24, la cual se encuentra bajo distintos niveles de concurrencia, desde 10 hasta 1000 solicitudes simultáneas.



```
lib > paginas > sincronizar.dart > _SincronizarDatosScreenState > _syncarDatos
15  class _SincronizarDatosScreenState extends State<SincronizarDatosScreen> {
168   Future<void> _syncarDatos() async {
169     try {
170       // Sincronizar datos de csctbevaluacion
171       final responseEvaluacion = await ApiService().fetchRiesgoFamiliar();
172       if (responseEvaluacion.isNotEmpty) {
173         for (var item in responseEvaluacion) {
174           final evaluacionData = {
175             'csctbevaluacionid': item['csctbevaluacionid'],
176             'nom_evaluacion': item['nom_evaluacion'],
177           };
178           await DatabaseHelper.insertEvaluacion(evaluacionData);
179         }
180       }
181
182       // Sincronizar datos de csctbprioritario
183       final responsePrioritario = await ApiService().fetchPrioritario();
184       if (responsePrioritario.isNotEmpty) {
185         for (var item in responsePrioritario) {
186           final prioritarioData = {
187             'csctbprioritarioid': item['csctbprioritarioid'],
188             'nom_prioritario': item['nom_prioritario'],
189           };
190           await DatabaseHelper.insertPrioritario(prioritarioData);
191         }
192       }
193
194       // Sincronizar datos de csctbvulnerable
195       final responseVulnerable = await ApiService().fetchVulnerable();
196       if (responseVulnerable.isNotEmpty) {
197         for (var item in responseVulnerable) {
198           final vulnerableData = {
```

Figura 23: Código de ingreso de ficha familiar en tolerancia a fallos

```
I/flutter ( 7711): *** WARNING ***
I/flutter ( 7711):
I/flutter ( 7711): Invalid argument false with type bool.
I/flutter ( 7711): Only num, String and Uint8List are supported. See https://github.com/tekartik/sqflite/blob/master/sqflite/doc/supported_types.md for details
I/flutter ( 7711):
I/flutter ( 7711): This will throw an exception in the future. For now it is displayed once per type.
I/flutter ( 7711):
I/flutter ( 7711):
I/flutter ( 7711): #0      _checkArg (package:sqflite_common/src/value_utils.dart:30:7)
I/flutter ( 7711): #1      checkNonNullValue (package:sqflite_common/src/value_utils.dart:51:5)
I/flutter ( 7711): #2      new SqlBuilder.insert.<anonymous closure> (package:sqflite_common/src/sql_builder.dart:180:11)
I/flutter ( 7711): #3      _LinkedHashMapMixin.forEach (dart:collection-patch/compact_hash.dart:633:13)
I/flutter ( 7711): #4      new SqlBuilder.insert (package:sqflite_common/src/sql_builder.dart:169:14)
I/flutter ( 7711): #5      SqfliteDatabaseExecutorMixin.insert (package:sqflite_common/src/database_mixin.dart:61:32)
I/flutter ( 7711): #6      DatabaseHelper.insertaridFamilia (package:ficha_familiar/paginas/agregar_personas/DatabaseHelper.dart:1233:21)
I/flutter ( 7711): <asynchronous suspension>
I/flutter ( 7711): #7      _InformacionPersonalState.guardarFamilia (package:ficha_familiar/paginas/agregar_personas/informacion_personal.dart:944:25)
I/flutter ( 7711): <asynchronous suspension>
Lost connection to device.
```

Figura 24: Logs registrados en la tolerancia a fallos

Número de operaciones exitosas / Total de operaciones probadas.

Meta: > 90%.

Indicador de fiabilidad 3 - capacidad de recuperación.

Se desarrolló un script en JavaScript que simuló la caída del sistema, con el objetivo de evaluar su capacidad para manejar interrupciones temporales en el servicio, en este caso el servicio dejó de estar disponible y posteriormente se restableció, permitiendo analizar su recuperación, como se observa en la figura 25, manteniendo distintos niveles de concurrencia, desde 10 hasta 1000 solicitudes simultáneas.

```
PS E:\api_ficha_familiar - tests\database> node .\test.js
Iniciando prueba de desconexión, Iteración 1...
Desconectando la base de datos...
Consulta inicial exitosa sin necesidad de reconectar.
Iniciando prueba de desconexión, Iteración 2...
Desconectando la base de datos...
Consulta inicial exitosa sin necesidad de reconectar.
Iniciando prueba de desconexión, Iteración 3...
Desconectando la base de datos...
Consulta inicial exitosa sin necesidad de reconectar.
Iniciando prueba de desconexión, Iteración 4...
Desconectando la base de datos...
Consulta inicial exitosa sin necesidad de reconectar.
Iniciando prueba de desconexión, Iteración 5...
Desconectando la base de datos...
Consulta inicial exitosa sin necesidad de reconectar.
Iniciando prueba de desconexión, Iteración 6...
Desconectando la base de datos...
Consulta inicial exitosa sin necesidad de reconectar.
Iniciando prueba de desconexión, Iteración 7...
Desconectando la base de datos...
Consulta inicial exitosa sin necesidad de reconectar.
Iniciando prueba de desconexión, Iteración 8...
Desconectando la base de datos...
Consulta inicial exitosa sin necesidad de reconectar.
Iniciando prueba de desconexión, Iteración 9...
```

Figura 25: Capacidad de recuperación de la base de datos de la ficha familiar

Porcentaje de recuperaciones exitosa: Número de recuperaciones exitosas / Total de fallos simulados. Meta: > 90%.

Posteriormente, se realizaron consultas a la API de sincronización utilizada en Flutter como se ve en la figura 27, llevando a cabo una simulación concurrente con distintos niveles de concurrencia, desde 10 hasta 1000 solicitudes simultáneas.

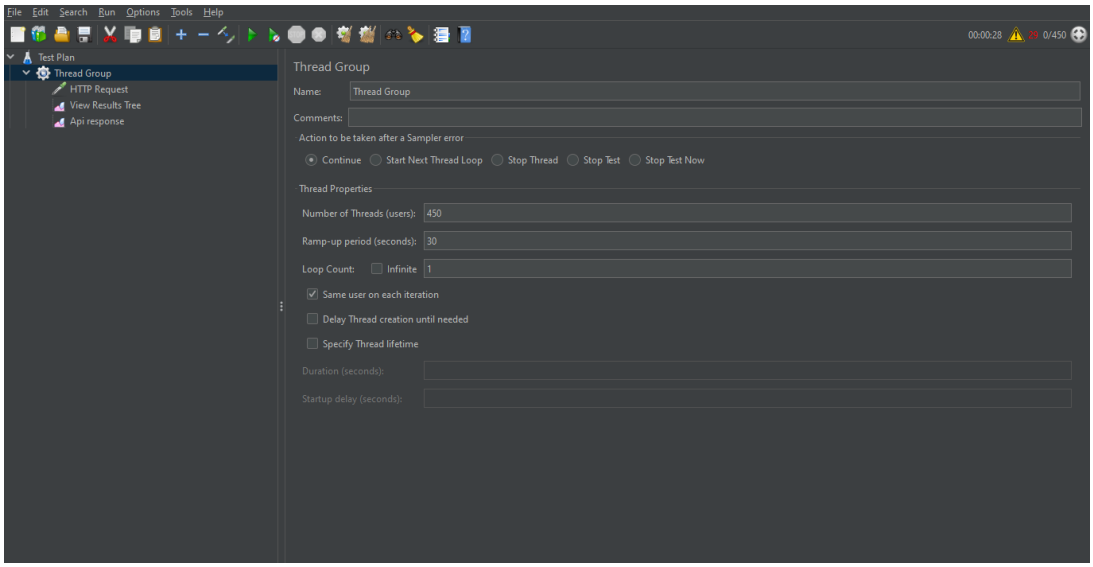


Figura 26: Consultas realizadas a la sincronización de la ficha familiar.

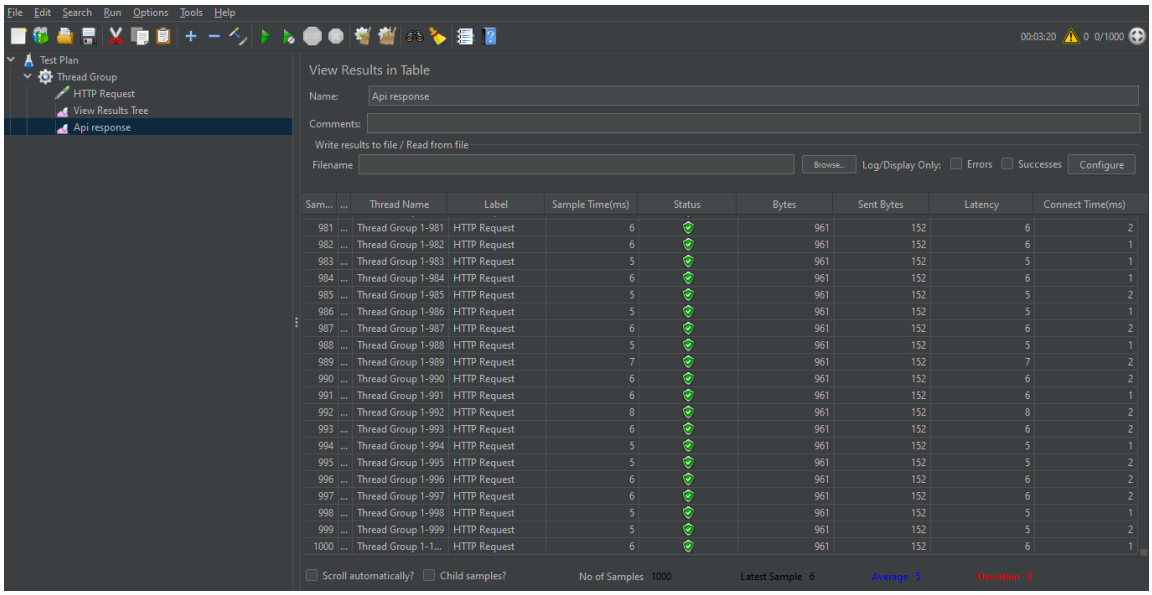


Figura 27: Simulación de solicitudes de ficha familiar con carga concurrente

El gestor de procesos de PM2 reinició automáticamente la aplicación 17 segundos después de cualquier fallo como se observa en la figura 29, por lo que está configurado para reiniciar automáticamente la aplicación cuando detecta que se ha detenido.

```
Reconectando la base de datos...
Respuesta después de reconectar la base de datos: [
  { csctbedadesid: 1, rango_edad: '0 - 28 DIAS' },
  { csctbedadesid: 2, rango_edad: '29 DIAS - 11 MESES' },
  { csctbedadesid: 3, rango_edad: '1 - 4 AÑOS' },
  { csctbedadesid: 4, rango_edad: '5 - 9 AÑOS' },
  { csctbedadesid: 5, rango_edad: '10 - 19 AÑOS' },
  { csctbedadesid: 6, rango_edad: '20 - 64 AÑOS' },
  { csctbedadesid: 7, rango_edad: 'MAYOR 65 AÑOS' },
  { csctbedadesid: 8, rango_edad: 'MENORES DE UN AÑO' },
  { csctbedadesid: 9, rango_edad: '12 - 23 MESES' },
  { csctbedadesid: 10, rango_edad: '24 - 35 MESES' },
  { csctbedadesid: 11, rango_edad: '36 - 59 MESES' },
  { csctbedadesid: 12, rango_edad: '5 AÑOS' },
  { csctbedadesid: 13, rango_edad: '9 AÑOS' },
  { csctbedadesid: 14, rango_edad: '15 AÑOS' },
  { csctbedadesid: 15, rango_edad: 'ADULTOS' }
]
```

Figura 28: Validación de la respuesta de la ficha familiar tras un fallo concurrente

```
PM2        | 2025-01-27T22:34:24: PM2 log: Concurrent actions   : 2
PM2        | 2025-01-27T22:34:24: PM2 log: SIGTERM timeout      : 1600
PM2        | 2025-01-27T22:34:24: PM2 log: ============================================================
PM2        | 2025-01-27T22:35:00: PM2 log: App [api:0] starting in -fork mode-
PM2        | 2025-01-27T22:35:00: PM2 log: App [api:0] online
PM2        | 2025-01-27T22:49:24: PM2 log: Stopping app:api id:0
PM2        | 2025-01-27T22:49:24: PM2 log: App [api:0] exited with code [1] via signal [SIGINT]
PM2        | 2025-01-27T22:49:24: PM2 log: pid=6916 msg=process killed
PM2        | 2025-01-27T22:49:37: PM2 log: App [api:0] starting in -fork mode-
PM2        | 2025-01-27T22:49:37: PM2 log: App [api:0] online
PM2        | 2025-01-27T23:01:55: PM2 log: Stopping app:api id:0
PM2        | 2025-01-27T23:01:56: PM2 log: App [api:0] exited with code [1] via signal [SIGINT]
PM2        | 2025-01-27T23:01:56: PM2 log: pid=1860 msg=process killed
PM2        | 2025-01-27T23:02:13: PM2 log: App [api:0] starting in -fork mode-
PM2        | 2025-01-27T23:02:13: PM2 log: App [api:0] online
```

Figura 29: Registros generados PM2 de monitorización para aplicaciones Node.js

Tiempo medio de recuperación: < 5 minutos

CAPÍTULO IV. RESULTADOS Y DISCUSIÓN

4.1 Resultados

- **Resultado 1 .- La investigación de métodos y tecnologías que permiten garantizar la fiabilidad en aplicaciones offline**, permitió observar diversas estrategias y herramientas con el objetivo de garantizar la mejora en la fiabilidad.

En la tabla 14 en base a la revisión bibliográfica respectiva se establecen distintos métodos aplicables al desarrollo de apps offline y se realiza una descripción referente a su uso y aplicabilidad; así como en la tabla 15 se establecen distintas tecnologías.

Tabla 14: Métodos para garantizar fiabilidad en aplicaciones offline

Métodos	Descripción
Replicated Data Types (CRDTs)	Estructuras de datos que garantizan consistencia eventual sin conflictos en entornos distribuidos. Útiles para sincronización automática de datos. [28]
Búsqueda por eventos	Permite registrar los cambios en los datos como una serie de eventos en lugar de modificar los datos directamente, es decir en lugar de actualizar un valor en una base de datos el sistema guarda cada cambio como un evento en una secuencia.
Uso de transacciones	Garantiza que las operaciones de base de datos sean atómicas, evitando inconsistencias al asegurar que todas las operaciones se completan correctamente o no se realizan.
Integridad referencial	Uso de restricciones como FOREIGN KEY y NOT NULL para mantener la coherencia y precisión de los datos, asegurando que las relaciones entre tablas no generen inconsistencias.
Manejo de errores	Utilización de bloques try-catch para capturar y manejar errores de manera controlada, evitando que una falla interrumpa todo el flujo de la aplicación.
Indicadores de Estado	Uso de campos de tipo bandera para marcar si los datos han sido sincronizados con el servidor, lo que permite controlar el flujo de datos y evitar duplicados u omisiones en la sincronización.
Uso de await en operaciones asíncronas	Await se utiliza para esperar de manera eficiente que las operaciones asíncronas, como las solicitudes de red o la actualización de la base de datos, se completen antes de proceder a la siguiente operación. [29]

Mecanismos de redundancia	Duplicación de datos críticos en múltiples almacenamientos, por ejemplo, guardar registros en SQLite y en archivos JSON como respaldo.

- **Comparativa de métodos para garantizar fiabilidad**

Los métodos para asegurar la fiabilidad en aplicaciones offline varían según la complejidad y los requisitos de cada sistema. Los CRDTs (Conflict-Free Replicated Data Types), por ejemplo, son ideales para escenarios con sincronización bidireccional y alta probabilidad de conflictos, pero implica un mayor consumo de memoria debido al almacenamiento de metadatos.

Por lo tanto, el manejo de transacciones y la integridad referencial son métodos más sencillos que garantizan atomicidad y coherencia en operaciones locales, adecuados para aplicaciones sin alta concurrencia offline, por lo que una aplicación médica con claves foráneas y transacciones asegura que las actualizaciones de stock no generen registros huérfanos o inconsistentes.

Finalmente los métodos investigados de son suficientes para apps offline con requisitos de sincronización y sin alta concurrencia a nivel local, por lo cual el manejo de transacciones, la integridad referencial, el manejo de errores, los indicadores de estado y el uso de Await en operaciones asíncronas son la opción ideal para su respectivo uso en la aplicación móvil offline de la Ficha familiar, otros métodos adicionales como (CRDTs, Event Sourcing, o redundancia) introducen complejidad innecesaria si no hay riesgo de conflictos frecuentes, además de que consumen más memoria y no son necesarias si no se requiere sincronización automática bidireccional.

Tabla 15 Tecnologías para el desarrollo de aplicaciones offline

Tabla 15: Tecnologías empleadas en el proceso de pruebas para garantizar fiabilidad

Tecnologías	Descripción
Jest	Eficaz para pruebas unitarias y de endpoints (con Supertest). Ideal para validar lógica de sincronización o APIs que interactúan con SQLite.
Jmeter	Permite simular carga masiva en el servidor durante la sincronización.
JavaScript	Flexible para simular fallos controlados durante por ejemplo cortes de red durante la sincronización y poder medir índices de fiabilidad.

Mocha	Permite escribir, ejecutar y gestionar pruebas automatizadas para garantizar la fiabilidad en el código.

- **Limitaciones de las herramientas empleadas en el proceso de pruebas para garantizar fiabilidad.**

Mocha es flexible para la elaboración de pruebas básicas, su falta de soporte nativo avanzado lo hace menos adecuado para aplicaciones offline complejas, donde escenarios como la sincronización parcial requieren simulaciones precisas.
JMeter es potente para pruebas de carga, pero tiene limitaciones ya que no simula adecuadamente la reconexión intermitente o redes inestables.
JavaScript permite simular fallos controlados pruebas de resiliencia son manuales y propensas a errores, ya que dependen de implementaciones personalizadas.

Por lo que finalmente se encontró que tecnologías como Mocha no ofrecen una buena integración con pruebas simuladas, por lo que se seleccionó Jest ya que es el framework de pruebas más popular y perfecto para validar lógica de sincronización, Jmeter la cual es ideal para simular pruebas de carga y estrés, y finalmente JavaScript ya que es ampliamente soportado para simular fallos controlados y poder medir la resiliencia.

- **Resultado 2 .- Como resultado de desarrollar la aplicación móvil offline multiplataforma para el ingreso de datos de ficha familiar con Mobile-D**; se identificó los requisitos funcionales y no funcionales como se describe en la tabla 16.

Tabla 16: Requisitos funcionales de la aplicación móvil offline

Identificador	Nombre del Requerimiento	Descripción	Prioridad
RF01	Login	Valida el acceso al personal del Centro de Salud Chambo	Alta
RF02	Nueva ficha familiar offline	Permite registrar datos del domicilio, integrantes de la familia, embarazadas, personas fallecidas y datos del doctor responsable.	Alta
RF03	Modificación de una ficha familiar offline	Permite modificar el personal responsable, datos de mortalidad, embarazadas y los miembros de la familia.	Alta
RF04	Registro Offline	Permite que toda la aplicación funcione sin conexión a internet, garantizando la operatividad en áreas sin cobertura.	Alta

RF05	Migración de Datos	Los datos recopilados offline deben migrarse al servidor con possgress cuando haya conexión. Las fichas incompletas se mantendrán en una bandeja de pendientes.	Alta
RF06	Estado de Avance	Visualiza el avance de las fichas familiares a nivel de menú.	Media
RF07	Almacenamiento Incremental	Guarda automáticamente la información para evitar pérdida de datos ante apagones, desconexiones o problemas del dispositivo.	Alta
RF08	Navegación entre Secciones	Permite navegar entre diferentes partes de la ficha sin perder datos ya ingresados, con subformularios interrelacionados para una recopilación más fluida.	Alta
RF09	Controles de Coherencia	Implementa validaciones que bloqueen selecciones inconsistentes, como seleccionar categorías incompatibles según edad, sexo o estado (ej., un hombre no puede ser embarazada).	Media
RF10	Geolocalización	Activa la funcionalidad de geolocalización directamente la aplicación móvil offline en la ficha familiar.	Medio
RF11	Registro y Control de Campos Obligatorios	Define campos obligatorios específicos como "Observaciones" y asegura que estos puedan ser completados con información relevante.	Media

Tabla 17: Requisitos no funcionales de la aplicación móvil offline

Identificador	Nombre del Requerimiento	Descripción	Prioridad
RNF01	Lenguaje de Desarrollo	La aplicación móvil usara el framework Flutter con Dart.	Alta
RNF02	Compatibilidad de Plataformas	La aplicación móvil será compatible con varios dispositivos Android y IOS.	Alta
RNF03	Interfaz de Usuario	La aplicación móvil tendrá una interfaz intuitiva para conceder a los médicos una experiencia óptima.	Alta

RNF04	Base de Datos	La información de la aplicación offline será almacenada en SQLite y la sincronización de esta en possgress SQL.	Alta
RNF05	Sincronización	La sincronización de datos entre la base de datos offline y online se realizará a través de endpoints.	Alta

Así mismo se estableció la estructura de la aplicación funcional, del manejo de bases de datos con SQLite, backend de sincronización para PostgreSQL.

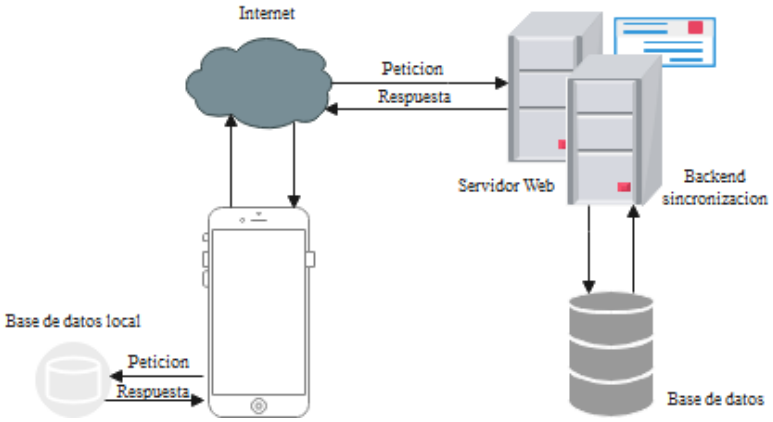


Figura 30: Estructura funcional de la aplicación móvil offline

Se diseño la navegabilidad de la aplicación funcional.

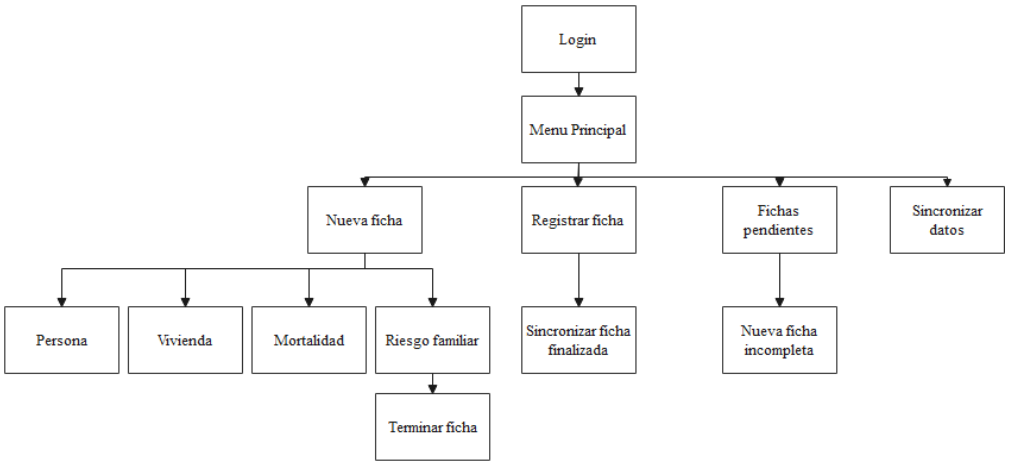


Figura 31: Navegabilidad funcional de la aplicación móvil offline

En la figura 29 se presenta el resultado final de la implementación a través de las pantallas principales del login y el menú principal de la aplicación offline. El detalle de pantallas y menús de los módulos de datos del establecimiento y familia, miembros de la familia, embarazadas, mortalidad familiar en los últimos 5 años, así como la sincronización de las fichas familiares se encuentra en el anexo 29.

Figura 32: Resultado de la pantalla de login y menú

Las otras pantallas generadas se encuentran disponibles en el anexo 3.

- **Resultado 3.- Como resultado final de la investigación se evaluó la fiabilidad de la aplicación móvil offline multiplataforma utilizando la norma ISO / IEC 25010.**

Para la evaluación del criterio de fiabilidad en la aplicación móvil offline multiplataforma se utilizaron las métricas de los subcriterios: ausencia a fallos, tolerancia a fallos y la capacidad de recuperación, realizando para tal efecto pruebas de simulación.

Evaluación de la métrica de ausencia a fallos. - Se verificó que las funcionalidades principales del sistema funcionan correctamente en condiciones normales, en la cual obtuvimos un resultado de 0% de error interpretado como ausencia de fallos. Las mediciones de esta métrica se detallan en la tabla 18.

Tabla 18: Resultados del indicador de ausencia a fallos

Subcriterio	Número de pruebas fallidas	Número total de pruebas	Resultado %	Meta de acuerdo con el criterio
Ausencia a fallos	0	10	0% errores	< 10% errores
	0	100	0% errores	< 10% errores
	0	500	0% errores	< 10% errores
	0	1000	0% errores	< 10% errores

Evaluación de la métrica de tolerancia a fallos. - Se evaluó si el sistema puede continuar funcionando ante fallos parciales, en la cual dio una meta: > 90%, dando como resultado de 100% de tolerancia a fallos en el ingreso de los datos, las mediciones se observan en la tabla 19.

Tabla 19: Resultados del indicador de tolerancia a fallos

Subcriterio	Total de	Resultado %

	operaciones probadas.	Número de operaciones exitosas		Meta de acuerdo con el criterio
Tolerancia a fallos	10	10	100% de tolerancia	Meta: > 90%.
	100	100	100% de tolerancia	Meta: > 90%.
	500	500	100% de tolerancia	Meta: > 90%.
	1000	1000	100% de tolerancia	Meta: > 90%.

Evaluación de la métrica de capacidad de recuperación. - Se evaluó la capacidad de recuperación, tanto el número de recuperaciones exitosas como el tiempo medio de recuperación (MTTR), el primer subcriterio, se alcanzó un 100% de recuperaciones exitosas, cumpliendo con la meta mayor al 90% de recuperaciones exitosas, y de tiempo de recuperación, el sistema logró un MTTR fue de 17 segundos, menos de los 5 minutos; los resultados de estas mediciones se visualizan en la tabla 20.

Tabla 20: Resultados del indicador de capacidad de recuperación

Subcriterio	Número de recuperaciones exitosas	Total de fallos simulados	Resultado %	Meta de acuerdo con el criterio
Capacidad de recuperación	10	10	100% recuperación	>90% recuperación exitosa
	100	100	100% recuperación	>90% recuperación exitosa
	500	500	100% recuperación	>90% recuperación exitosa
	1000	1000	100% recuperación	>90% recuperación exitosa

Tabla 21: Resultados del indicador de capacidad de recuperación (MTTR)

Subcriterio	Resultado %	Meta de acuerdo con el criterio (MTTR)
Capacidad de recuperación	17 segundos	< 5 minutos

4.2 Discusión

En la presente investigación las aplicaciones diseñadas para funcionar sin conexión a internet requieren ciertos mecanismos para garantizar la fiabilidad de los datos, donde SQLite enfatiza esa necesidad en bases de datos locales [23], así como proveer servicios de datos rápidos y fiables aun existiendo una conexión considerada como intermitente [23], estos factores coinciden con los resultados del proyecto, al usar SQLite con transacciones ACID

se garantiza que después de un cierre inesperado, los datos se mantengan sin errores. De hecho, se encontró que apps se recuperan correctamente, mientras que si manejaban otros archivos binarios presentaban corrupción de datos [30], descrita en la publicación de Labs America.

Adicionalmente las transacciones, integridad referencial, manejo de errores, indicadores de estado y Await en operaciones asíncronas es suficiente para garantizar estabilidad sin introducir complejidad innecesaria [31], estas prácticas permitieron evitar que las operaciones de la aplicación se congelen, especialmente durante sincronización e ingreso de datos, como lo describe publicación de Valeriu en aplicaciones offline. Otros métodos como CRDTs o Event Sourcing no fueron priorizados debido a que suelen consumir más memoria y su aplicabilidad en escenarios de sincronización bidireccional, que no son críticos como lo menciona Salman, quienes proponen una estrategia optimista de control de concurrencia sin recurrir a mecanismos complejos como los CRDTs [32], en cuanto a pruebas, se seleccionaron Jest, JMeter y JavaScript por su integración eficiente en la validación de la lógica de sincronización y la simulación de fallos controlados.

El desarrollo de la aplicación móvil para el Centro de Salud Chambo usando los principios de la metodología ágil Mobile-D específico para el desarrollo de aplicaciones móviles prioriza iteraciones rápidas, adaptabilidad y entrega, permitiendo la implementación de funcionalidades offline y la sincronización de datos, mediante ciclos de desarrollo iterativos centrados en módulos independientes [33] esto debido a que la metodología se usa en entornos con requisitos móviles, lo cual coincidió con las necesidades dinámicas del sistema de salud rural, donde la retroalimentación de médicos fue integrada progresivamente.

En cuanto a la evaluación de la fiabilidad el resultado del 0% de fallos en las pruebas es excelente, ya que superó la meta establecida <10% de errores, esto puede deberse a que el sistema mantiene funcionalidades robustas, código y validaciones, en la publicación de Labs America introdujeron sistemáticamente desconexiones de red y cierres abruptos de la app para evaluar la fiabilidad, observando que muchos servicios caían o perdían datos cuando fallaba la sincronización, así como la tolerancia a fallos [30], permitiendo su evaluación.

El 100% de éxito en tolerancia a fallos se debe a que se usó en el ingreso mecanismos denominadas transacciones atómicas, las cuales garantizan operaciones completas o reversibles, así como la integridad referencial.

El 100% de recuperaciones exitosas y un MTTR de 17 segundos reflejan un sistema adaptable, superando las metas establecidas del >90% de éxito y <5 minutos de recuperación. Estos resultados se debieron a los métodos usados en la recuperación utilizando gestores de procesos responsable del reinicio de servicios para corregir fallos, que minimizan la intervención manual, por lo que el MTTR reflejó la efectividad de estos enfoques, por ejemplo, Sergio ha documentado que implementar estrategias de auto-recuperación puede reducir el tiempo de inactividad hasta en un 50% respecto a sistemas sin tales mecanismos [34].

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- La investigación sobre métodos y tecnologías para garantizar la fiabilidad en aplicaciones offline permitió identificar diversas estrategias clave, tales como el uso de transacciones, integridad referencial, manejo de errores, indicadores de estado y operaciones asíncronas controladas mediante el uso de Await. Estas prácticas fueron fundamentales para mantener la coherencia, integridad y disponibilidad de los datos, incluso en ausencia de conectividad. La implementación de transacciones atómicas evitó estados inconsistentes, asegurando que las acciones sean completadas. Asimismo, la integridad referencial mediante claves foráneas y restricciones protege la precisión de los datos y la relación de las variables. El manejo de errores a través de bloques try-catch facilita las recuperaciones controladas, y el uso de indicadores de estado mejora la comunicación y la sincronización al restablecer la conexión. Además, la utilización de Await en operaciones asíncronas evita conflictos de ejecuciones paralelas. Las herramientas de pruebas como JMeter y Jest, junto con el uso de logs en Flutter, garantizaron la fiabilidad al haber simulado condiciones de carga y posibles fallos en la red.

- El desarrollo de la aplicación móvil offline multiplataforma fue facilitado por la metodología Mobile-D, enfocado en el desarrollo de aplicaciones móviles la cual fue importante en la implementación del ciclo ágil de la metodología, con fases de exploración, inicialización, producto, estabilización y pruebas. Esta metodología, adaptativa y flexible, resultó apropiada para el trabajo de un equipo pequeño y permitió una planificación eficaz, además de la entrega iterativa y rápida de los diferentes módulos de la aplicación. La integración de las funcionalidades de la aplicación con la base de datos local y el servidor, así como la implementación de un sistema funcional de sincronización offline/online, fueron aspectos muy importantes en la fiabilidad del sistema. Este enfoque permitió gestionar correctamente el ingreso de datos de fichas familiares del Centro de Salud Chambo, incluso en situaciones de desconexión de red.

- La evaluación de la fiabilidad de la aplicación se realizó siguiendo los principios de la norma ISO/IEC 25010, con un enfoque en la ausencia de fallos, tolerancia a fallos y capacidad de recuperación. En cuanto a la ausencia de fallos, el sistema logró un 0% de fallos durante las pruebas realizadas, lo que cumple con la meta establecida de mantener el error por debajo del 10%. Esto se debe a la robustez de las funcionalidades principales del sistema, las cuales operaron correctamente en condiciones normales. Respecto a la tolerancia a fallos, el sistema demostró su capacidad para continuar funcionando ante fallos parciales, alcanzando un 100% en el ingreso de datos, lo que cumple con la meta de más del 90% en la tolerancia a fallos. Finalmente, en términos de recuperación de fallos, el sistema logró un tiempo medio de recuperación (MTTR) de 17 segundos, superando la meta de <5 minutos de recuperación, estos resultados se deben a los métodos usados en la recuperación

utilizando gestores de procesos responsable del reinicio de servicios para corregir fallos, que minimizan la intervención manual, finalmente el MTTR sugiere una detección rápida de errores y procesos de recuperación.

5.2 Recomendaciones

- Implementar pruebas basadas en escenarios de interrupción de red, fallos de almacenamiento y caídas abruptas permitirá identificar vulnerabilidades que puedan pasar desapercibidas en condiciones controladas.
- Registrar errores en condiciones de desconexión y reconexión, lo que facilitará la identificación de problemas recurrentes.
- Utilizar herramientas para la gestión de proyectos para mejorar la planificación y seguimiento de tareas dentro del enfoque iterativo de la metodología Mobile-D.
- Documentar los procesos de sincronización y manejo de datos offline, manteniendo el enfoque de la metodología Mobile-D permitiendo una actualización continua del sistema sin comprometer la coherencia de los datos.
- Ampliar los escenarios de prueba para incluir situaciones más variables, como fluctuaciones en la calidad de la red. Esto permitirá garantizar un rendimiento óptimo en otras situaciones.

BIBLIOGRAFÍA

[1] Amazon, «¿Qué es Flutter?,» 2024. [En línea]. Available: https://aws.amazon.com/es/what-is/flutter/. [Último acceso: 2024].

[2] Netiris, «Flutter: Desarrollo de aplicaciones móviles,» 5 Abril 2024. [En línea]. Available: https://blog.neteris.com/stepforward/flutter-desarrollo-de-aplicaciones-moviles.

[3] Chiriboga y Collaguazo, «Desarrollo de una aplicación móvil multiplataforma basado en la tele asesoría médica que permita mejorar el seguimiento a personas expuestas al contagio de covid-19 utilizando analítica de datos en la Universidad de las Fuerzas Armadas ESPE Sede Latacunga,» 2021. [En línea]. Available: https://repositorio.espe.edu.ec/handle/21000/26024.

[4] Reinoso y Zhirzhan, «DESARROLLO DE UNA APLICACIÓN MÓVIL PARA EL AGENDAMIENTO DE CITAS DE CONSULTAS MÉDICAS UTILIZANDO TÉCNICAS DE PROCESAMIENTO DE LENGUAJE NATURAL APLICADAS A UN ASISTENTE VIRTUAL.,» 2022. [En línea]. Available: https://dspace.ups.edu.ec/bitstream/123456789/22064/1/UPS-CT009620.pdf.

[5] Coro y Alverca, «DESARROLLO DE UNA APLICACIÓN WEB Y MÓVIL PARA LA FICHA FAMILIAR EN EL CENTRO DE SALUD CHAMBO,» 2024. [En línea]. Available: http://dspace.unach.edu.ec/handle/51000/13056.

[6] L. Herazo, «¿Qué es una aplicación móvil?,» 2022. [En línea]. Available: https://anincubator.com/que-es-una-aplicacion-movil/.

[7] L. Nunes, «Arcana,» 2023. [En línea]. Available: https://emma.io/blog/tipos-aplicaciones-caracteristicas-ejemplos/.

[8] Molero, «Características que una aplicación debe tener,» 2022. [En línea]. Available: https://creatuaplicacion.com/caracteristicas-que-una-aplicacion-debe-tener/.

[9] Parker, «Ventajas y desventajas de las aplicaciones móviles,» 2023. [En línea]. Available: https://www.oppizi.com/es/post/advantages-and-disadvantages-of-mobile-apps/.

[10] Urrutia, «Android,» 2020. [En línea]. Available: https://www.arimetrics.com/glosario-digital/android.

[11] Vanegas, «ANDROID ?....DE QUÉ ME HABLAN,» 2014. [En línea]. Available: https://revistas.udistrital.edu.co/index.php/vinculos/article/download/8022/9872?inline=1.

[12] Davelou, «Davelou,» 2024. [En línea]. Available: https://www.develou.com/aprendiendo-la-arquitectura-de-android/.

[13] R. G, «¿Qué es iOS? Todo sobre el sistema operativo de Apple.,» 2020. [En línea]. Available: https://www.adslzone.net/reportajes/software/que-es-ios/.

[14] Cetys, «¿Qué es una aplicación multiplataforma?,» 2022. [En línea]. Available: https://www.ufv.es/cetys/blog/que-es-una-aplicacion-multiplataforma/.

[15] GuruTechLabs.inc, «Guru,» 22 Marzo 2022. [En línea]. Available: https://www.gurutechnolabs.com/flutter-vs-native-technology/.

[16] Meijomil, «Definición de framework, ventajas y aplicaciones en diferentes ámbitos.,» 2024. [En línea]. Available: https://www.inboundcycle.com/diccionario-marketing-online/framework.

[17] Ebim, «Conoce las ventajas y desventajas de Flutter,» 2023. [En línea]. Available: https://www.grupoebim.com/blog/ventajas-desventajas-flutter/.

[18] Canals, «¿Qué es el lenguaje de programación Dart?,» 2020. [En línea]. Available: https://inlab.fib.upc.edu/es/uncategorized-ca-es/que-es-el-lenguaje-de-programacion-dart/2020/.

[19] Massango, «Dart for backend: pros and cons. DEV Community.,» 2023. [En línea]. Available: https://dev.to/pedromassango/dart-for-backend-pros-and-cons-2jh.

[20] Roces, «¿Qué tal funciona tu app en modo offline?,» 2022. [En línea]. Available: https://www.hiberus.com/crecemos-contigo/que-tal-funciona-app-modo-offline/.

[21] Montoya, «INSTRUCTIVO DE LA FICHA FAMILIAR PARA LA ESTRATEGIA: COMUNIDADES CON AUTOCUIDADO PROMOTORAS DE SALUD (CAPS).,» 2018. [En línea]. Available: https://manizalessalud.net/wp-content/uploads/2018/08/INSTRUCTIVO-FICHA-FAMILIAR-2018.pdf.

[22] Microsoft, «¿Qué es PostgreSQL?,» [En línea]. Available: https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-postgresql#:~:text=relaciones%20entre%20ellos.-,PostgreSQL%20es%20una%20base%20de%20datos%20relacional%20de%20c%C3%B3digo%20abierto,su%20gran%20flexibilidad%20e%20integridad.. [Último acceso: 2024].

[23] R. Hip, «SQLite,» 10 Octubre 2023. [En línea]. Available: https://www.sqlite.org/about.html.

[24] ISO, «Norma ISO 25010,» 2023. [En línea]. Available: https://normasiso.org/norma-iso-25010/.

[25] Syntonize, «Metodologías para el desarrollo de aplicaciones móviles,» 2022. [En línea]. Available: https://www.syntonize.com/metodologias-desarrollo-de-aplicaciones-moviles/.

[26] M. Saira, «ResearchGate,» Diciembre 2020. [En línea]. Available: https://www.researchgate.net/profile/Saira-Edith-Marquez-De-La-Cruz.

[27] R. J. Romo, «Flutter Arquitectura,» 2019. [En línea]. Available: https://rubenjromo.com/flutter-arquitectura-limpia-1-explicacion-y-estructura/.

[28] M. Shapiro, «Replicated Data Types,» de *Replicated Data Types*, HAL Open Science, p. 8.

[29] D. Engel, «Microsoft,» 02 01 2025. [En línea]. Available: https://learn.microsoft.com/es-es/sql/connect/ado-net/asynchronous-programming?view=sql-server-ver16.

[30] N. A. C. Younghwan, «Reliable, Consistent, and Efficient Data Sync for Mobile Apps,» 2015. [En línea]. Available: https://pages.cs.wisc.edu/~nitina/Publications/simbaclient-fast15.pdf.

[31] V. Crudu, «Building Offline-First Xamarin Apps with SQLite Database - A Comprehensive Guide,» 2025. [En línea]. Available: https://moldstud.com/articles/p-building-offline-first-xamarin-apps-with-sqlite-database-a-comprehensive-guide.

[32] A. Salman y R. Lakshmi, «Cornell University,» 2010. [En línea]. Available: https://arxiv.org/abs/1005.1747.

[33] A. Pekka, «Research Gate,» 2004. [En línea]. Available: https://www.researchgate.net/publication/221322054_Mobile-D_An_Agile_Approach_for_Mobile_Application_Development.

[34] S. Vergara, «ItDo,» 2025. [En línea]. Available: https://www.itdo.com/blog/diseno-de-aplicaciones-auto-recuperables-garantizando-resiliencia-ante-fallos.

[35] G. Contreras, M. Fernández y S. Martínez, «Congreso Nacional de Control Automatico,» 2015. [En línea]. Available: https://amca.mx/memorias/amca2015/files/0048_JuAT1-03.pdf.

[36] R. Hipp, «Appropriate Uses For SQLite,» 2025. [En línea]. Available: https://www.sqlite.org/whentouse.html.

ANEXOS

Anexo 1. Diccionario de datos

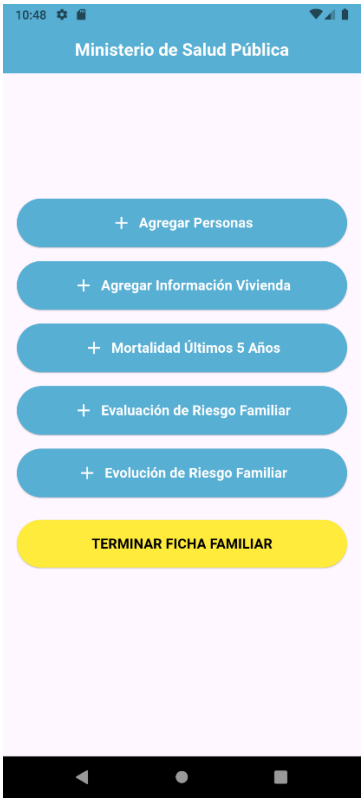
Nombre de la Tabla	Atributo	Tipo	Descripción
Embarazadas	csctbembarazadasid	integer	Identificador único de embarazadas (clave primaria).
	csctbfamiliaid	integer	Identificador de la familia asociada.
	fecha_menstruacion	date	Fecha de la última menstruación.
	fecha_parto	date	Fecha probable de parto.
	control_menos20	integer	Controles prenatales realizados antes de las 20 semanas.
	control_mas20	integer	Controles prenatales realizados después de las 20 semanas.
	esquema_vacunacion	boolean	Indica si tiene el esquema de vacunación completo.
	ante_patologicos	character varying	Antecedentes patológicos de la embarazada.
	semanas_gestacion	integer	Semanas de gestación actuales.
	gestas	integer	Número de gestaciones previas.
	partos	integer	Número de partos previos.
	abortos	integer	Número de abortos previos.
	cesarias	integer	Número de cesáreas previas.
	estado_embarazada	boolean	Estado actual de la embarazada (activa/inactiva).

	fecha_eli_embarazada	date	Fecha de eliminación del registro de la embarazada.
csctbfamilia	csctbfamiliaid	integer	Identificador único de la familia (clave primaria).
	csctbocupacionid	integer	Identificador de la ocupación.
	csctbinstruccionid	integer	Identificador del nivel de instrucción.
	csctbparentescoid	integer	Identificador del parentesco familiar.
	cedula_fam	character varying	Número de cédula del miembro familiar.
	nom_fam	character varying	Nombres del miembro familiar.
	ape_fam	character varying	Apellidos del miembro familiar.
	fecha_na_fam	date	Fecha de nacimiento del miembro familiar.
	anios	integer	Años cumplidos del miembro familiar.
	meses	integer	Meses cumplidos del miembro familiar.
	dias	integer	Días cumplidos del miembro familiar.
	genero	character varying	Género del miembro familiar.
	control_bucal	boolean	Indica si el miembro familiar tiene control bucal.
	observacion	character varying	Observaciones generales.
	estado_fam	boolean	Estado actual del miembro familiar (activo/inactivo).
	id_jefe_hogar	integer	Identificador del jefe de hogar.

	fecreacion_fam	date	Fecha de creación del registro familiar.
	feelimini_fam	date	Fecha de eliminación del registro familiar.
	jefe_hogar	boolean	Indica si el miembro es jefe de hogar.
	estado_civil	character varying	Estado civil del miembro familiar.
	csctbetniaid	integer	Identificador de la etnia.
	csctbnacionalidadid	integer	Identificador de la nacionalidad.
	csctbpueblosid	integer	Identificador del pueblo indígena o comunidad.
	afiliacion	character varying	Tipo de afiliación médica.
	tipo_identificacion	character varying	Tipo de identificación del miembro familiar.
	celular	integer	Número de teléfono celular.
	telefono	integer	Número de teléfono convencional.
csctbmortalidad	csctbmortalidadid	integer	Identificador único de mortalidad (clave primaria).
	csctbparentescoid	integer	Identificador del parentesco del fallecido.
	csctbjefefamid	integer	Identificador del jefe familiar asociado.
	nombres_mortalidad	character varying	Nombres de la persona fallecida.
	fecha_fallecimiento	date	Fecha de fallecimiento.
	causa_nodefi	character varying	Causa del fallecimiento (no definida).
	apellidos_mortalidad	character varying	Apellidos de la persona fallecida.

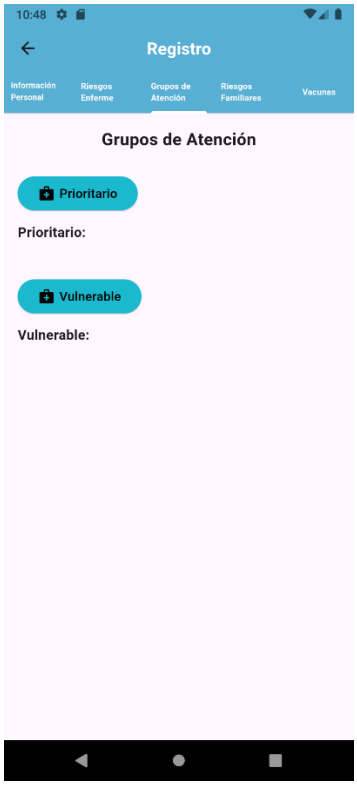
	anios	integer	Edad en años al fallecer.
	meses	integer	Edad en meses al fallecer.
	dias	integer	Edad en días al fallecer.
	estado_mortalidad	boolean	Estado actual del registro (activo/inactivo).
	fecha_eli_mortalidad	date	Fecha de eliminación del registro de mortalidad.
csctbevaluacionfamilia	csctbevaluacionfamiliaid	integer	Identificador único de evaluación familiar (clave primaria).
	vacuincomnum	integer	Número de vacunas incompletas.
	vacuincomtxt	character varying	Observaciones sobre vacunas incompletas.
	persomalnunum	integer	Número de personas con malnutrición.
	persomalnutxt	character varying	Observaciones sobre malnutrición.
	estado_evolucion	boolean	Estado de evolución de la familia (activo/inactivo).

Anexo 2. Pantallas de la ficha familiar offline



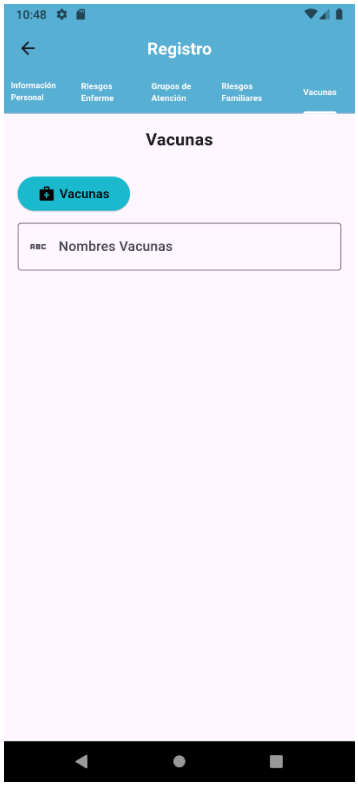
(a) Menú Ficha

(b) Integrante Familia

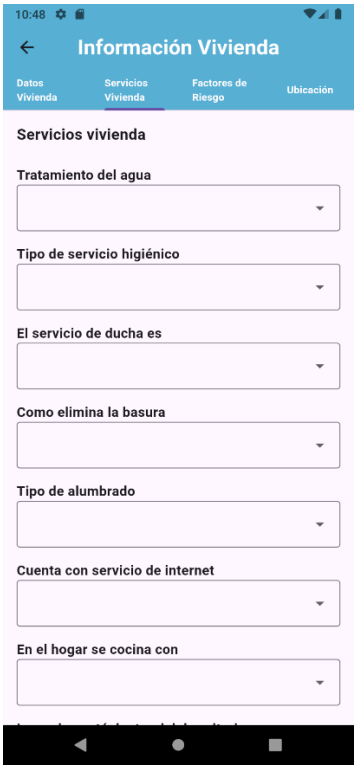


(c) Grupos de atención

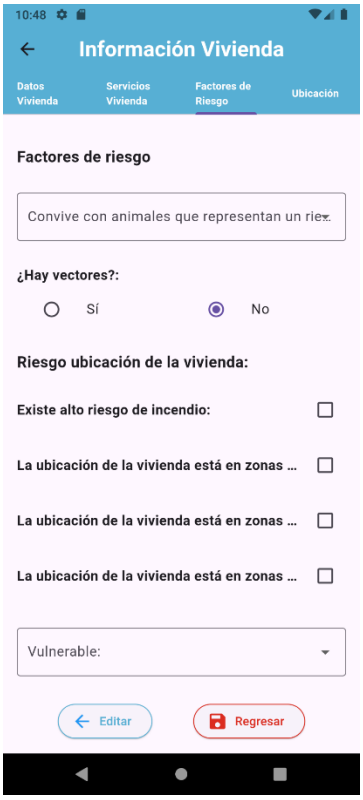
(d) Riesgo Familiar



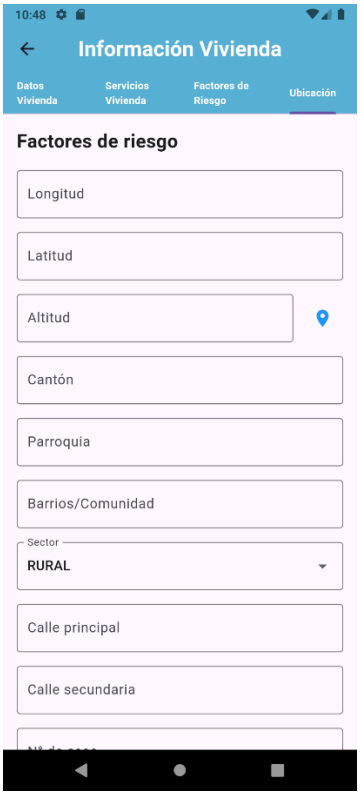
(e) Vacunas



(f) Servicios de Vivienda



(g) Factores de Riesgo



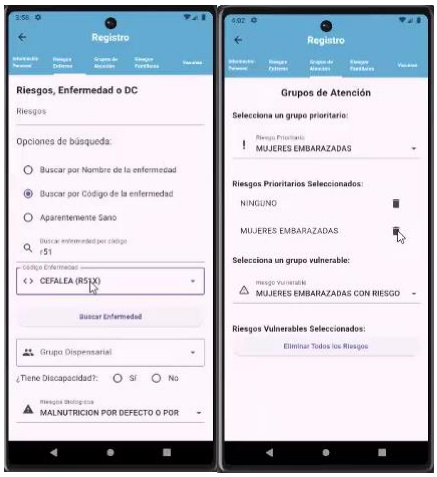
(h) Ubicación

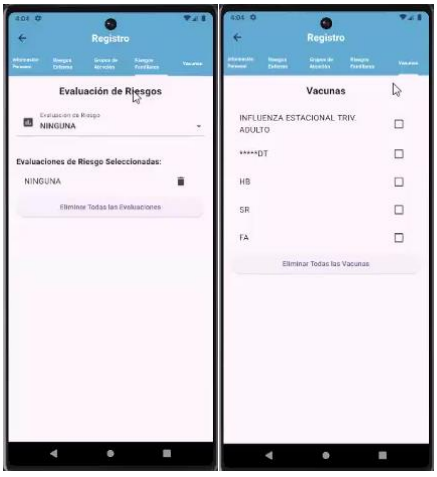
(i) Persona Fallecida

(j) Terminar Ficha

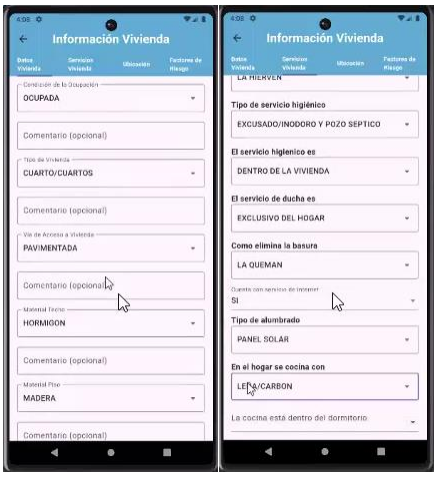
Anexo 3. Pantallas de la ficha familiar offline resultados

Se creó el módulo de registro de personas para la ficha familiar, teniendo en cuenta la parte lógica de la ficha medica de acuerdo con el Centro de Salud Chambo.

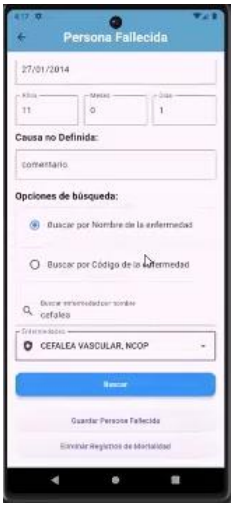


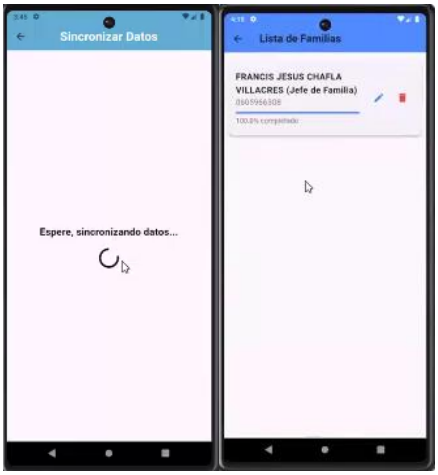


Se generó el módulo de registro de vivienda para la ficha familiar, teniendo en cuenta la parte lógica de la ficha medica de acuerdo con el Centro de Salud Chambo.

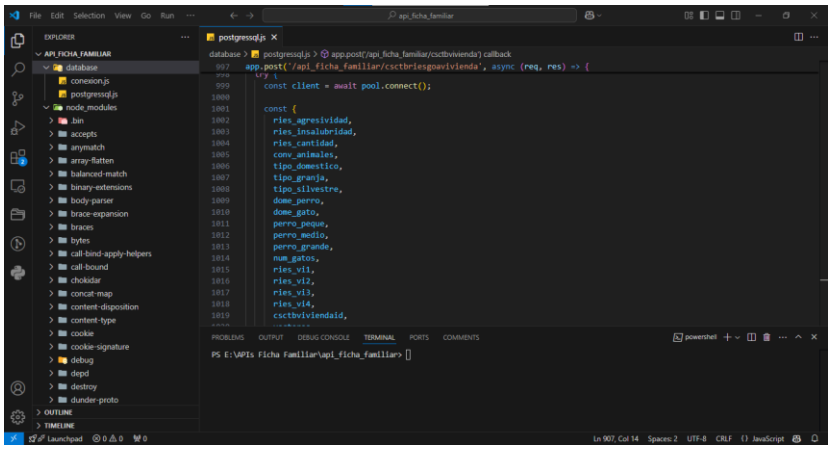


Se formó el módulo de registro de persona fallecida para la ficha familiar, de sincronización y fichas pendientes de acuerdo con el Centro de Salud Chambo.



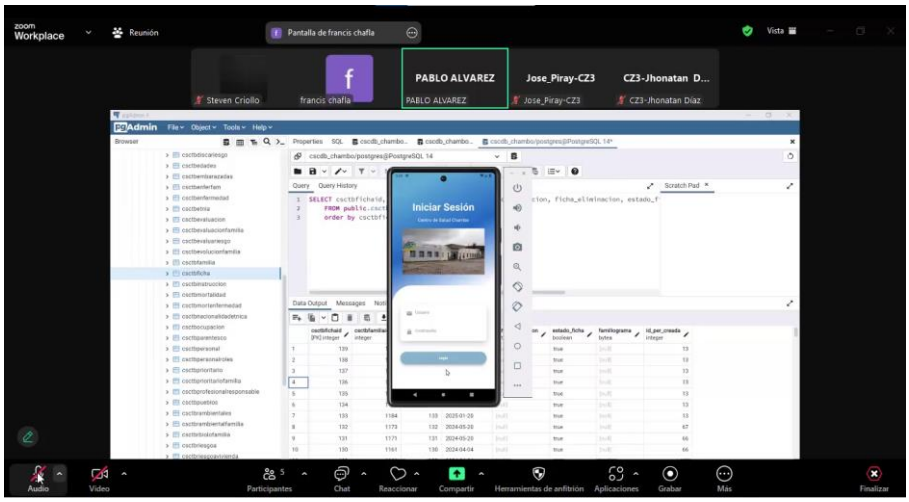


Se desarrolló el backend API REST usando node.js.



Anexo 4. Reunión para visualización de requerimientos de la ficha familiar offline.

Anexo 5. Reunión virtual de la aplicación móvil offline de ficha familiar.



Anexo 6. Manual de usuario de la aplicación móvil offline de ficha familiar.

MANUAL DE USUARIO

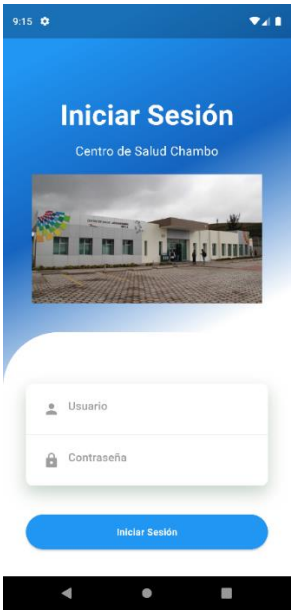
CENTRO DE SALUD CHAMBO

INDICE

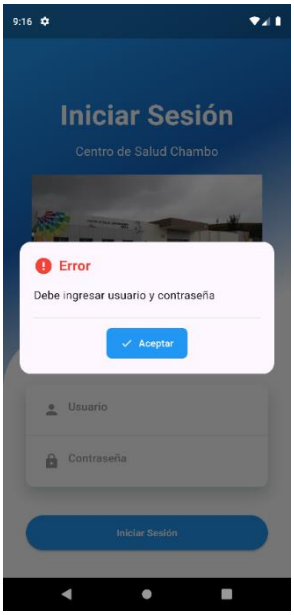
1. INICIO DE SESIÓN

I. INICIO DE SESION Y ACCESO OFFLINE

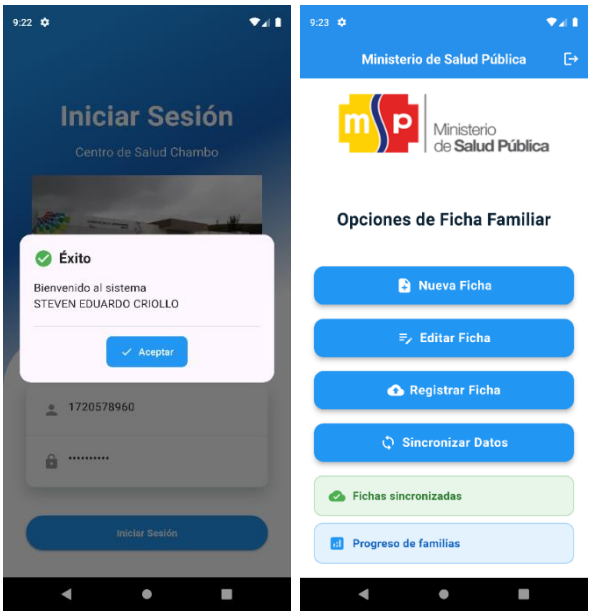
En la ventana principal, se deberán ingresar el nombre de usuario y la contraseña, tal como se muestra a continuación. Es importante tener en cuenta que, al acceder por primera vez, se requiere una conexión a internet para realizar la autenticación correspondiente.



En caso de que las credenciales sean incorrectas, se mostrará el siguiente mensaje de advertencia.



Una vez iniciada la sesión, se mostrará la interfaz principal con el menú de la ficha familiar. Para iniciar sesión sin conexión a internet, simplemente se debe abrir la aplicación.

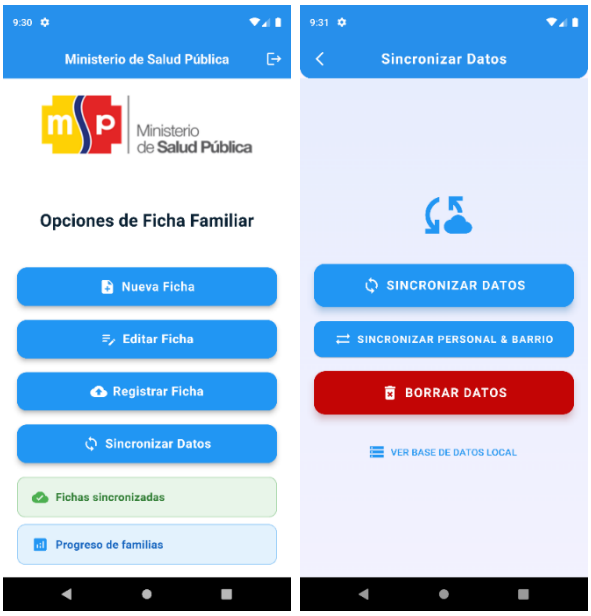


2. SINCRONIZAR DATOS

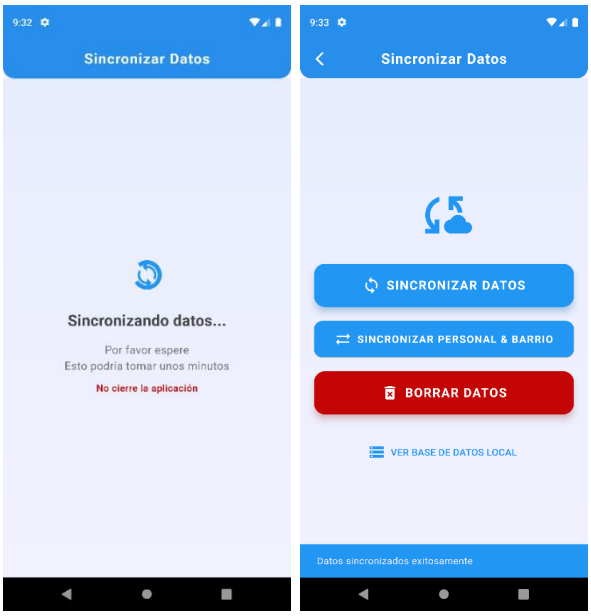
I. SINCRONIZACION DE DATOS DE FICHA FAMILIAR

Antes de crear una ficha familiar por primera vez, es necesario realizar una sincronización. De lo contrario, se mostrará el siguiente aviso.

El proceso inicial de sincronización de los datos necesarios para utilizar la ficha familiar requiere conexión a internet. Para ello, se debe seleccionar el botón "**Sincronizar datos**" desde el menú y su respectivo submenú.



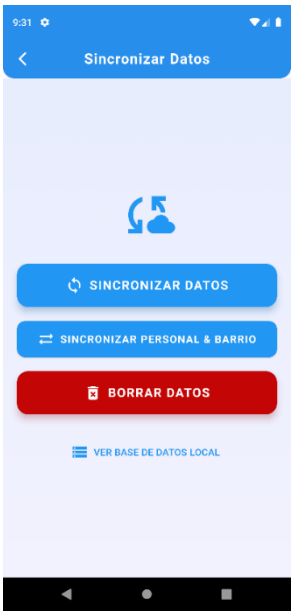
Durante el proceso de sincronización se mostrará el mensaje "*Sincronizando datos*". Una vez finalizado, será posible acceder sin conexión a internet.



En caso de requerir una actualización de los datos utilizados en la ficha familiar, se debe repetir el proceso de sincronización.

II. SINCRONIZACION DE DATOS DE PROFESIONALES

De igual forma, el proceso inicial de sincronización de los datos de profesionales requiere conexión a internet. Para ello, se debe seleccionar el botón "**Sincronizar personal y barrio**".



Finalizado el proceso de sincronización aparecerá el mensaje "*Profesional y barrio profesional sincronizado*", en ese momento se podrán tener acceso sin conexión a internet.

En caso de requerir una actualización de los datos del profesional y barrios, se debe repetir el proceso de sincronización.

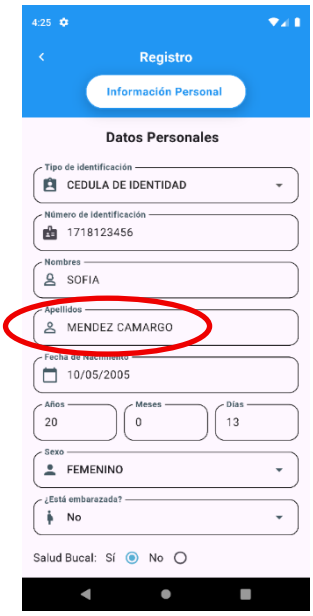
3. GENERAR FICHA FAMILIAR

I. CREAR FICHA FAMILIAR

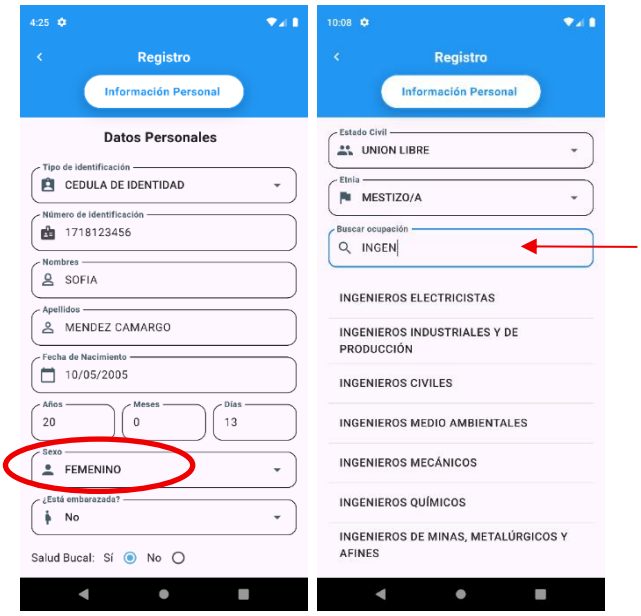
Las fichas familiares se pueden ingresar a través de la opción "**Nueva ficha**", lo que permitirá acceder a todos los módulos. Recordando que se completan en el orden establecido.

II. PROCEDIMIENTOS PARA EL LLENADO DE DATOS

El proceso de llenado de la ficha varía según el tipo de dato. Los campos de ingreso permiten teclear libremente la información, como por ejemplo nombres, direcciones o comentarios.

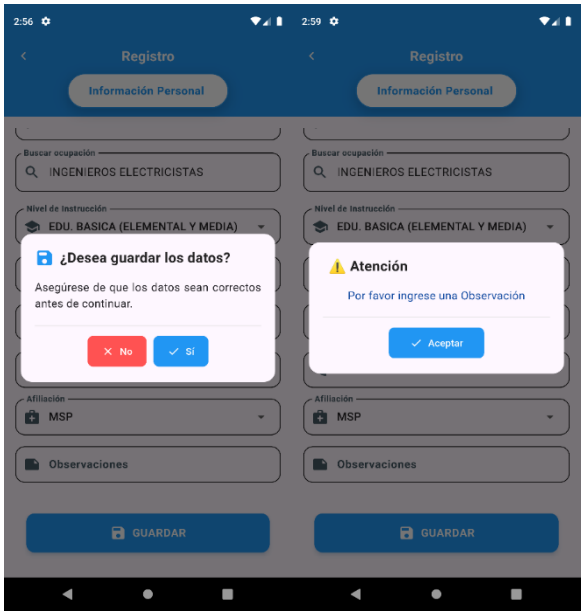


En los campos de selección se ofrece una lista desplegable de opciones, en algunos casos se pueden buscar directamente.

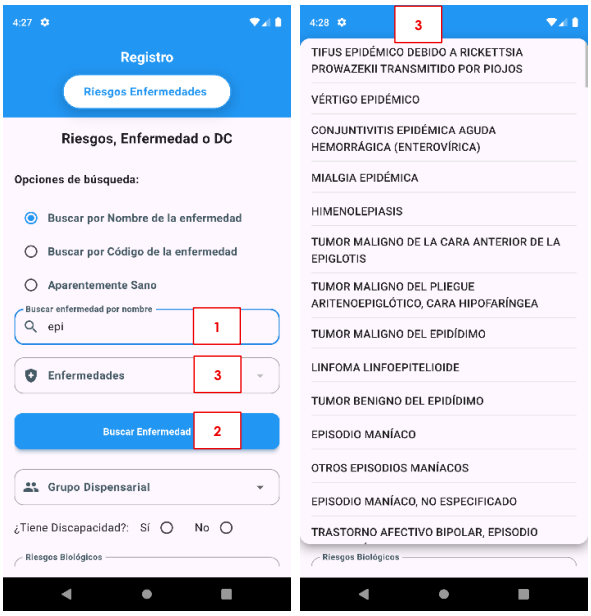


Una vez finalizado el registro en el apartado de *información personal*, la información debe guardarse haciendo clic en el botón "Guardar".

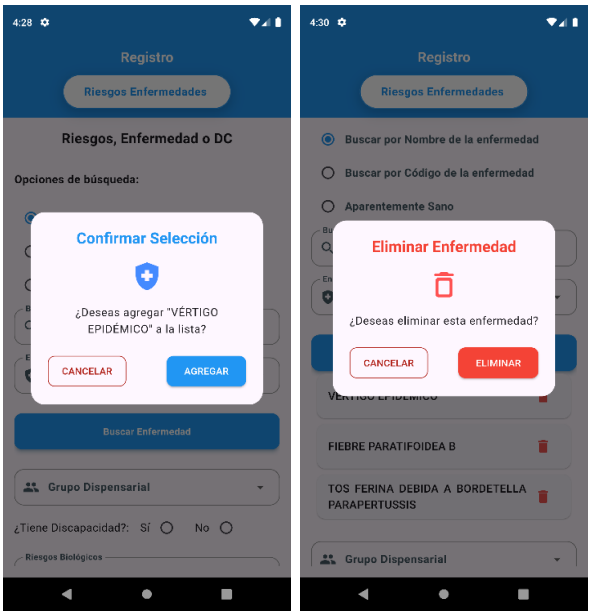
Nota: Es obligatorio completar todos los campos requeridos; de lo contrario, se mostrarán alertas indicando la información faltante.



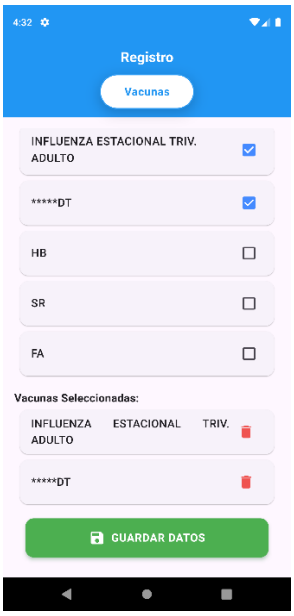
En los campos de búsqueda y selección se requiere ingresar un término de búsqueda, a continuación, seleccionar "**Buscar**", y escoger el ítem correspondiente de los resultados.



Al realizar una selección, se abrirá un cuadro de diálogo para confirmar la opción elegida. Además, se ofrecerá la posibilidad de eliminar la selección en caso de ser necesario.

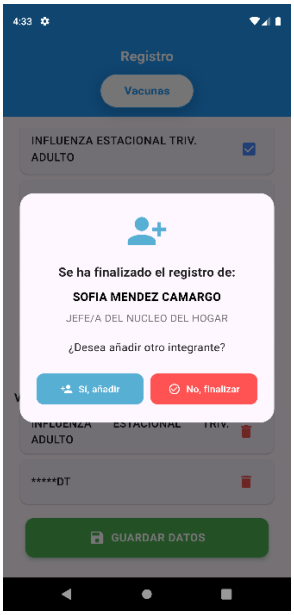


Finalmente, existirán secciones en las que será posible realizar selecciones directamente desde cuadros de opción.



III. AGREGAR MIEMBROS DE FAMILIA

Se ofrecerá la posibilidad de añadir más miembros a la familia mediante la opción "**Sí, añadir**". En caso de no agregar más integrantes y continuar con el llenado de los demás módulos, se deberá seleccionar "**No, finalizar**".

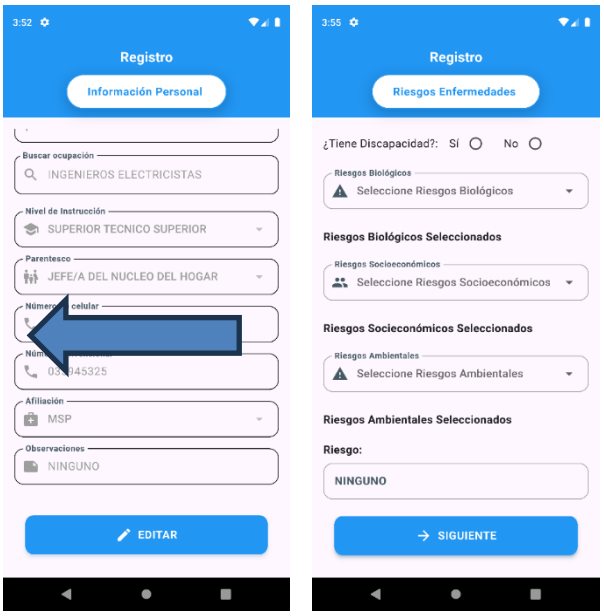


IV. NAVEGABILIDAD EN EL INGRESO DE DATOS

La aplicación facilita la navegación durante el ingreso de datos, permitiendo moverse entre secciones.

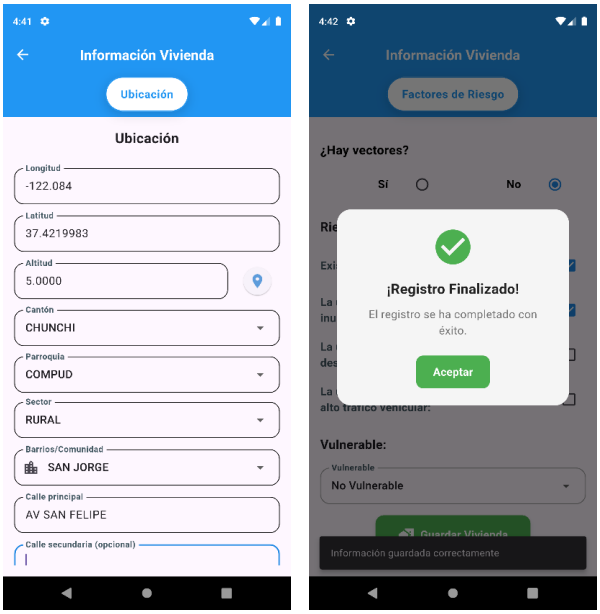
Gestos táctiles: Se puede deslizar hacia la izquierda o derecha para avanzar o retroceder entre pantallas.

Nota: Es obligatorio completar los campos requeridos.



V. GEOLOCALIZACIÓN

En el apartado de ubicación dentro del módulo de información de vivienda, se requiere capturar la geolocalización. Para ello, se deberá seleccionar el ícono .



VI. EVALUACION Y EVOLUCION FAMILIAR

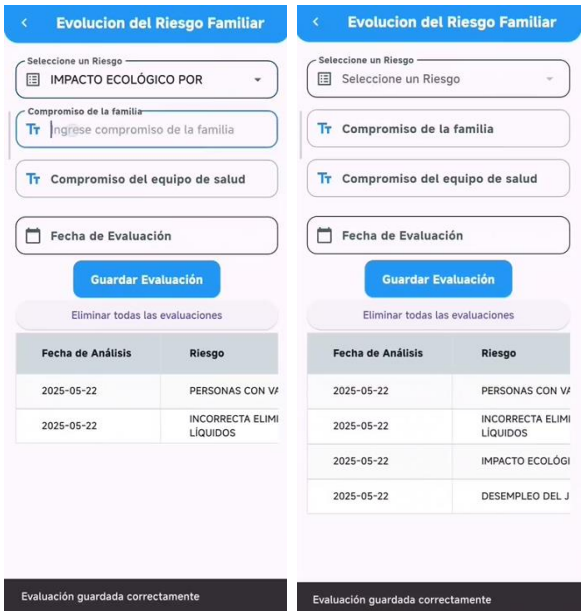
EVALUACION

La evaluación depende de los riesgos de la familia y de las condiciones de la vivienda. Para iniciar este proceso, se debe hacer clic en "**Agregar**".

Después aparecerán los indicadores correspondientes a la evaluación. Para guardar la información, seleccione la opción "**Guardar**".

EVOLUCION

Este apartado se basa en la evaluación de la familia. Del mismo modo, los cambios realizados podrán guardarse seleccionando la opción "Guardar".

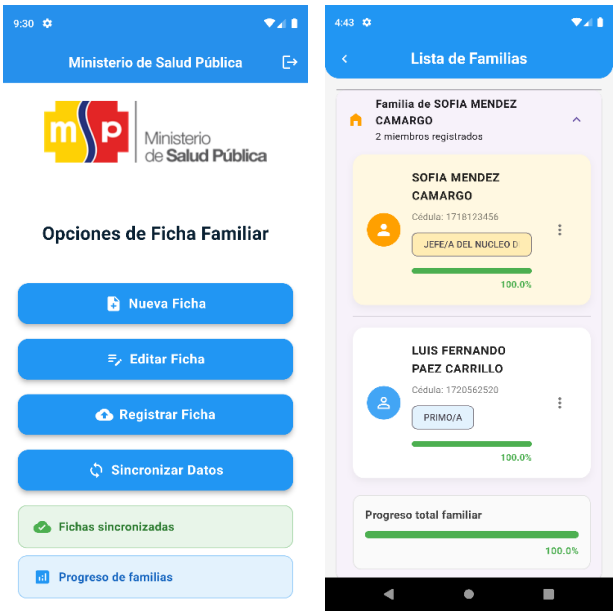


VII. TERMINAR FICHA FAMILIAR

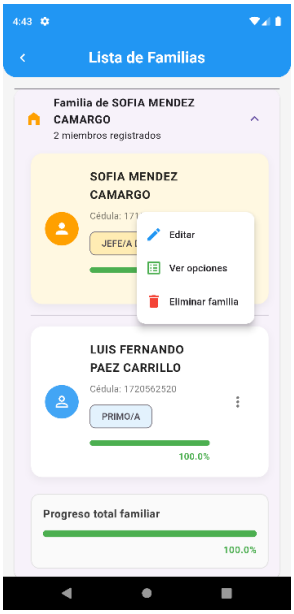
Para finalizar el registro de una ficha, se deberá hacer clic en el botón "**Terminar ficha familiar**".

4. EDITAR FICHA FAMILIAR

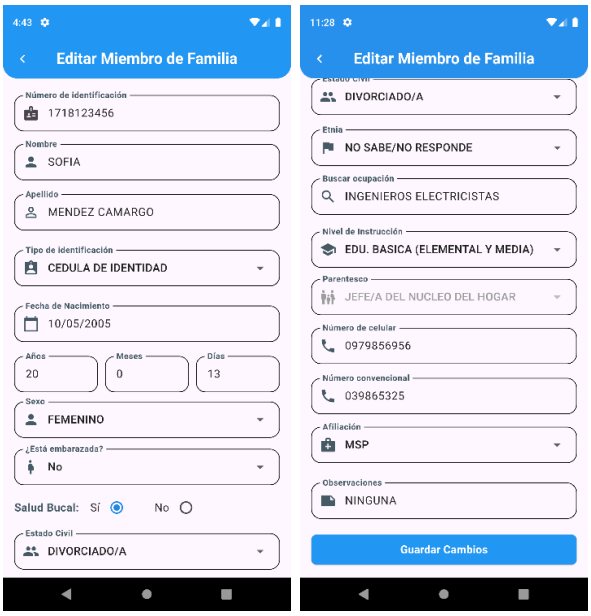
Una vez finalizado el registro de la ficha familiar, en el apartado "**Editar ficha**" se podrán visualizar todos los miembros e integrantes del grupo familiar.



Cada miembro contará con las siguientes opciones: **Editar**, para modificar los datos de información personal; **Ver opciones**, para acceder y editar los demás módulos; y **Eliminar**, para eliminar a la familia del sistema.

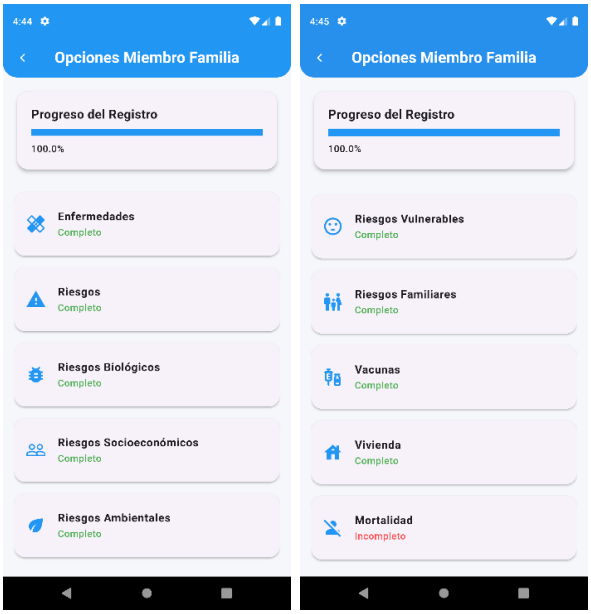


En la opción **editar** se encuentran los datos principales de la información personal.

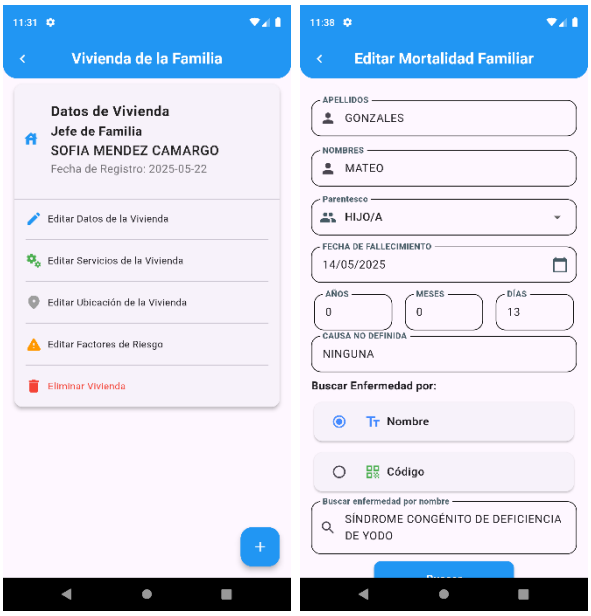


En **Ver opciones** se podrá acceder a los datos registrados en los respectivos módulos. Además, se podrán visualizar los campos pendientes por completar, así como el avance del proceso del integrante expresado en porcentaje.

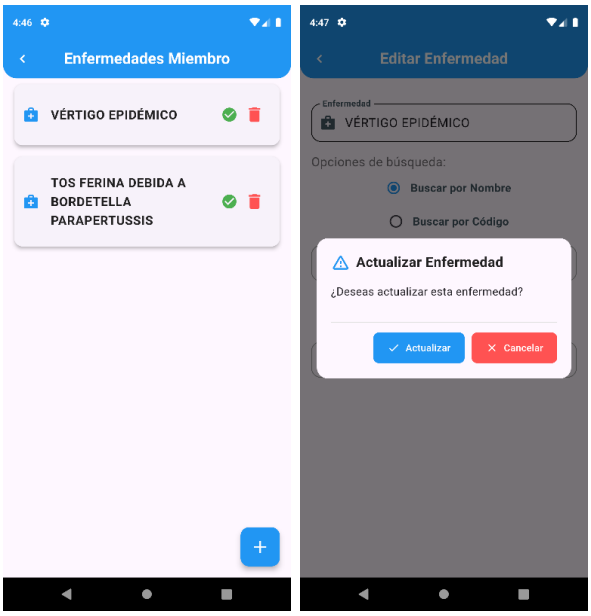
Cabe recordar que únicamente el jefe de familia tendrá habilitadas las opciones de vivienda y mortalidad.



Como se muestra a continuación, los módulos de vivienda y mortalidad son accesibles únicamente para el jefe de familia.



En cada módulo y segmento será posible **agregar**, **editar**, **modificar** y **eliminar** opciones, tal como se muestra en el siguiente ejemplo.



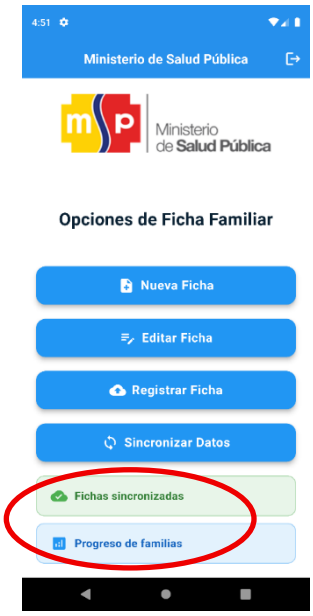
5. REGISTRAR FICHA FAMILIAR

Cuando el registro de las familias esté completo al 100%, en el apartado "**Registrar ficha**" se podrán visualizar las familias listas para ser subidas al sistema.

Cuando la ficha de la familia se haya subido, se mostrará el mensaje de "*Sincronización exitosa*".

6. INDICADORES DE ESTADOS DE FICHA

En el menú principal de la ficha familiar se mostrarán *indicadores de estado* de las fichas, como se observa a continuación.



En la sección "**Fichas sincronizadas**" se podrá visualizar la cantidad de fichas que han sido sincronizadas y aquellas que aún no lo han sido.

En el apartado "**Proceso de familias**" se puede visualizar cuántas fichas están completas al 100% (incluyendo tanto al jefe de familia como a los integrantes) y cuántas permanecen incompletas.

7. CERRAR SESION

Para cerrar sesión, haga clic en la opción "**Cerrar sesión** ⎋". Para volver a ingresar, será necesario autenticarse nuevamente.