



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



TFG del Grado en Ingeniería
Informática

RedELA
Documentación Técnica



Presentado por Javier Martín Castro
en Universidad de Burgos — 2 de julio de 2024
Tutor: Pedro Renedo Fernández

Índice general

Índice general	i
Índice de figuras	iii
Índice de tablas	v
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal Redem	1
A.3. Planificación temporal RedELA	3
A.4. Estudio de viabilidad	6
Apéndice B Especificación de Requisitos	9
B.1. Introducción	9
B.2. Objetivos generales	10
B.3. Catálogo de requisitos	10
B.4. Especificación de requisitos	12
Apéndice C Especificación de diseño	27
C.1. Introducción	27
C.2. Diseño de datos	27
C.3. Diseño procedimental	29
C.4. Diseño arquitectónico	29
Apéndice D Documentación técnica de programación	31
D.1. Introducción	31
D.2. Estructura de directorios	31

D.3. Manual del programador	33
D.4. Compilación	43
D.5. Ejecución del proyecto	43
D.6. Pruebas del sistema	44
Apéndice E Documentación de usuario	47
E.1. Introducción	47
E.2. Requisitos de usuarios	47
E.3. Instalación	47
E.4. Manual del usuario	47
E.5. Manual del administrador	52
Apéndice F Anexo de sostenibilización curricular	55
F.1. Introducción	55

Índice de figuras

B.1. Diagrama flujo admin	12
B.2. Diagrama flujo gestor casos	13
B.3. Diagrama flujo paciente/cuidador	13
C.1. Diseño de datos actores	28
C.2. Entidades	28
C.3. Diagrama de secuencia Gestor de casos	29
C.4. MVVM	30
D.1. mvvm	32
D.2. Plataforma desarrollo	34
D.3. Comprobación correcta instalación Flutter	34
D.4. Añadir Flutter a Visual Studio Code	35
D.5. Paleta de comandos VS Code	36
D.6. Encrypt secure random	38
D.7. Agregar proyecto	39
D.8. Agregar proyecto	40
D.9. Secrets and variables	41
D.10. Respository secrets	41
D.11. Nueva versión	42
D.12. Subir bundle	43
D.13. Detalle versión	43
D.14. Seleccionar dispositivo	44
D.15. Test	45
D.16. Test OK	45
E.1. Acceder a RedELA	48
E.2. Acceder a RedELA	49

E.3. Registro en RedELA	50
E.4. Home	51
E.5. Citas	52
E.6. Home Admin	53

Índice de tablas

A.1. Costes realizar aplicación por un desarrollador.	6
A.2. Costes del material o subscripciones para realizar aplicación. . .	7
B.1. CU-1 Gestión Gestores de casos.	14
B.2. CU-2 Enviar invitación.	15
B.3. CU-3 Eliminar usuario.	15
B.4. CU-4 Listar usuarios.	16
B.5. CU-5 Ver usuario.	16
B.6. CU-6 Editar usuario.	17
B.7. CU-7 Gestión citas.	17
B.8. CU-8 Crear cita.	18
B.9. CU-9 Editar cita.	19
B.10.CU-10 Cancelar cita.	19
B.11.CU-11 Listar hospitales.	20
B.12.CU-12 Listar tratamientos.	20
B.13.CU-13 Crear hospital.	21
B.14.CU-14 Crear tratamiento.	22
B.15.CU-15 Editar hospital.	23
B.16.CU-16 Editar tratamiento.	24
B.17.CU-17 Eliminar hospital.	25
B.18.CU-18 Eliminar tratamiento.	25

Apéndice A

Plan de Proyecto Software

A.1. Introducción

La finalidad inicial de este proyecto era ofrecer la oportunidad para mejorar la calidad de vida de las personas que han sido diagnosticadas con algún tipo de Esclerosis Múltiple (EM). Para ello se empezó a desarrollar una aplicación frontend que de manera amigable que daría soporte a aquellas personas que hayan decidido utilizarla y padezcan la enfermedad.

En este apartado se valorarán y analizarán las la planificación temporal así como el estudio de viabilidad.

A.2. Planificación temporal Redem

Para la planificación temporal se tendrán en cuenta el tiempo que conlleva la curva de aprendizaje para poner en marcha el proyecto, así como las distintas funcionalidades que se van implementando según las necesidades aportadas por el cliente. A través del tiempo se realizan sprints para poder desarrollar y añadir las funcionalidades correspondientes al proyecto.

Este proyecto estará alojado en Github <https://github.com/jmc1005/redem>

- Se aplicó una estrategia de desarrollo a partir de (*sprints*) mediante iteraciones incrementales.
- Se realizan *sprints* que conllevan poco tiempo para la creación de funcionalidades.

- Al finalizar el *sprint* correspondiente se incorpora a la release que posteriormente formará parte del producto.
- Se realizaban reuniones con el cliente para revisar el producto. En estas revisiones se aportan mejoras/modificaciones.
- Tras la obtención de requisitos de cliente, se planifican las nuevas tareas y que deciden en que *sprint* estarán asignadas.
- Cada tarea incluida se estima en tiempo en un tablero *scrum*.
- Cada tarea estimada se priorizan dentro del tablero *scrum*.

Sprint 0 (19/10/2023)

Se realiza reunión con el tutor Pedro Renedo Fernández para asignación del proyecto para realizar la parte frontend de una aplicación para dar soporte a usuarios diagnosticados con Esclerosis Múltiples (EM).

En ella se comenta una idea inicial sobre que herramientas se van a emplear y la selección del framework para crear el proyecto.

Sprint 1 (27/10/2023)

Se realiza una primera toma de contacto con el cliente con los requisitos iniciales. En ella nos transmite que está muy ilusionado en que queramos aceptar el reto y poder dar a luz un aplicación que podrá facilitar la vida de mucha gente.

Para los objetivos iniciales nos informa que está interesado en crear una aplicación móvil para personas que hayan sido diagnosticadas con EM y así poder aportar una solución en su día a día.

Le indicamos que tenemos intención en realizar la aplicación en un lenguaje que podría utilizarse en distintos dispositivos así como podría ser usada en distintos Sistemas Operativos.

Inicialmente nos indica que una versión inicial estaría disponible para dispositivos con sistema operativo Android.

Sprint 2 (28/10/2023 al 04/02/2024)

Se realiza una primera toma de contacto con el framework Flutter después de analizar que ventajas e inconvenientes podríamos encontrarnos para realizar la aplicación móvil solicitada por el cliente.

Tras tomar la decisión se empieza una formación inicial en el lenguaje de programación Dart para entender su funcionamiento y poder realizar el desarrollo del proyecto.

Sprint 3 (05/10/2023 al 11/02/2024)

Se realiza la creación del proyecto con el framework Flutter y con el lenguaje de programación Dart. Además, se añaden las configuraciones iniciales y se crea la página inicial.

Sprint 4 (12/02/2024 al 18/02/2024)

Se añaden las páginas de de Login y Registro al proyecto.

Sprint 5 (19/02/2024 al 25/02/2024)

Crear página detalle del usuario y home.

Sprint 6 (26/02/2024 al 10/03/2024)

Crear aplicación responsive para varios dispositivos (web, tablet, móvil)
Corregir bug de idioma Enviar correo al cliente para resolver dudas a cerca de los requisitos del usuario.

Sprint 7 (11/03/2024 al 17/03/2024)

Actualizar datos usuario Actualizar documentación Crear página Administrador (tras las inquietudes transmitidas al tutor sobre la inconsistencia de datos, nos aconseja que nos creemos una página para que el administrador pueda realizar la gestión de síntomas, tipos de esclerosis múltiples,...)

A.3. Planificación temporal RedELA

Una vez tomada la decisión de enfocar el trabajo y el proyecto una vez nos pusimos en contacto con la Asociación de Esclerosis Lateral Amiotrófica de Castilla y León y nos informaron de los requisitos que necesitarían para crear una aplicación que facilitara la vida a pacientes con ELA nos planificamos de manera similar al proyecto inicial.

Este proyecto estará alojado en Github <https://github.com/jmc1005/red-ela>

- Se aplicó una estrategia de desarrollo a partir de (*sprints*) mediante iteraciones incrementales.
- Se realizan *sprints* que conllevan poco tiempo para la creación de funcionalidades.
- Al finalizar el *sprint* correspondiente se incorpora a la release que posteriormente formará parte del producto.
- Se realizaban reuniones con el cliente para revisar el producto. En estas revisiones se aportan mejoras/modificaciones.
- Tras la obtención de requisitos de cliente, se planifican las nuevas tareas y que deciden en que *sprint* estarán asignadas.
- Cada tarea incluida se estima en tiempo en un tablero *scrum*.
- Cada tarea estimada se priorizan dentro del tablero *scrum*.

Sprint 0 (08/04/2024 - 14/04/2024)

Analizar cambio rumbo del TFG. Reunión con Asociación ELACyL para presentar idea del proyecto. Analizar documento con requisitos iniciales enviado por la asociación

- Actores que interactúan (Gestor/a de Casos, Paciente, Cuidador principal)
- Secciones de la aplicación
- Fases entrega

Sprint 1 (15/04/2024 - 21/04/2024)

- Crear proyecto red-ela
- Añadir configuración inicial
- Añadir internacionalización, en esta primera versión sólo estará disponible en español.
- Crear página registro/acceso
- Crear página administrador

Sprint 2 (22/04/2024 - 28/04/2024)

- Crear servicio autenticación usuario
- Crear servicio gestión usuario
- Crear página listado de usuario para la gestión de usuarios
- Crear página detalle usuario

Sprint 3 (29/04/2024 - 05/05/2024)

- Analizar flujo de los actores Paciente, Cuidador principal y Enfermera Gestora de Casos
- Actualizar campos Detalle usuario

Sprint 4 (06/05/2024 - 12/05/2024)

- Daily cliente.
- Actualizar campos Detalle usuario

Sprint 5 (13/05/2024 - 26/05/2024)

- Analizar crear OTP (One Time Password) para el registro de la aplicación.
- Analizar enviar email con código de invitación para acceder página OTP (One Time Password)
- Crear página OTP (One Time Password)
- Crear funcionalidad OTP (One Time Password) con firebase a través del móvil
- Crear servicio cifrar datos usuarios

Sprint 6 (22/05/2024 - 02/06/2024)

- Separar en pantalla Admin los usuarios en Gestores de Casos, Pacientes y Cuidadores.
- Enviar email al invitar a usuarios a registrarse en la aplicación.
- Actualizar Gestor de Casos
- Actualizar Cuidador
- Añadir pantalla Home
- Añadir pantalla Citas

Sprint 7 (03/06/2024 - 16/06/2024)

- Crear servicio gestión de citas
- Añadir funcionalidad cancelar cita
- Actualizar pantalla home
- Ajustes envío email/invitación.
- Añadir notificaciones push

Sprint 8 (17/06/2024 - 30/06/2024)

- Corrección bugs
- Envío notificaciones
- Crear/Actualizar workflows para despliegue android/web
- Crear cuenta desarrollador para despliegue Android

A.4. Estudio de viabilidad

Viabilidad económica

En este apartado vamos a realizar un supuesto sobre el coste que tendría la realización del proyecto en una empresa.

Tiempo de desarrollo

Esta aplicación al ser personalizada y con funcionalidades complejas vamos a indicar que el tiempo de desarrollo completo de la primera versión serán de unos 6 meses.

Costes Desarrollador

Suponemos que la aplicación ha sido realizada por un desarrollador con contrato temporal durante 6 meses y se considera que los gastos ocasionados serían:

Concepto	Coste
Salario bruto	1.833,33 €
Complementos	0,00 €
Contingencias Comunes (23,60 %)	432,67 €
Accidentes de Trabajo y Enfermedades Profesionales (2,25 %)	41,25 €
Desempleo (6,70 %)	122,83 €
Formación Profesional (0,60 %)	11,00 €
Fondo de garantía salarial (FOGASA) (0,20 %)	3,67 €
Coste del empleado	2.444,75 €

Tabla A.1: Costes realizar aplicación por un desarrollador.

Costes Material

Los costes para que el desarrollador pueda realizar la aplicación y además, se tiene en cuenta que el material que necesita ronda una amortización durante 5 años, son los siguientes:

Concepto	Coste	Coste Amortizado
Portátil	1.400 €	140 €
Cuenta desarrollador	25 €	–
Dispositivo móvil android	280 €	28 €
Firebase	coste por uso	–
Total	1705 €	168 €

Tabla A.2: Costes del material o subscripciones para realizar aplicación.

Monetización

Esta aplicación va a ser una aplicación gratuita para ayudar a pacientes que hayan sido diagnosticadas con la enfermedad ELA, de momento será gratuita. Sería la Asociación de Esclerosis Lateral Amiotrófica de Castilla y León la que tendría que considerar la manera de sacarle rentabilidad a la aplicación. Algunas de las opciones podrían ser:

- Donaciones
- Publicidad
- Crowdfunding

Viabilidad legal

Para definir la viabilidad legal [?] de nuestro proyecto tenemos primeramente que comprender que tan importante es para nuestro desarrollo entender que hay que tener en cuenta lo siguiente:

- Debe ajustarse a las leyes y regulaciones.
- Afecta al coste, calidad, reputación e incluso a la duración.
- Durante el desarrollo del proyecto este tendrá la participación y colaboración de diversos colaboradores, en nuestro caso el equipo del proyecto, el cliente.

Nuestra aplicación, al estar relacionada con la sanidad pública debemos tener cuidado con la privacidad del paciente.

Software

Para Flutter [?] que es un framework para construir aplicaciones multi-plataforma, la viabilidad legal se centra en:

- La licencia Apache 2.0, para crear y redistribuir la aplicación de código abierto.
- La protección del framework mediante la propiedad intelectual de Google.
- Derechos de autor.

Para Firebase [?] que fue creado por Google contiene algunos puntos a tener en cuenta:

- Está creado bajo la licencia Apache 2.0, permitiendo a los desarrolladores poder utilizar, modificar y redistribuir sin tener alguna restricción y de manera gratuita.
- La propiedad intelectual pertenece a Google.
- Al crear la cuenta para utilizar la plataforma se aceptan los términos del servicio.
- Los desarrolladores cuentan con los derechos de autor sobre su aplicación.
- Los desarrolladores están obligados a cumplir las políticas de privacidad marcadas por Google.

Apéndice *B*

Especificación de Requisitos

B.1. Introducción

En este punto vamos a definir los requisitos que va a tener nuestra primera versión del proyecto. Para ello vamos a seguir la **especificación de requisitos de software (ERS)** [?], donde se recogen una serie de buenas prácticas y son las siguientes:

- **Completa.** Todos los requerimientos deben estar contemplados en ella y deben estar definidas todas las referencias .
- **Consistente.** Debe tener coherencia con los requerimientos y además, con otras especificaciones.
- **Inequívoca.** Claridad a la hora de redactar el documento para que no haya malas interpretaciones.
- **Correcta.** Se deben cumplir con los requisitos recogidos en la especificación.
- **Trazable.** Hace referencia a la posibilidad de comprobar la historia, aplicación o ubicación de un item mediante su identificación almacenada y documentada.
- **Priorizable.** Posibilidad de organizar los items según la relevancia y separándolos en esenciales, condicionales y opcionales.
- **Modificable.** Dar la posibilidad de poder ser modificable.
- **Verificable.** Tener la posibilidad de realizar las pruebas de manera finita y sin costo.
- **Clara.** Debe ser entendible.

B.2. Objetivos generales

Nuestro proyecto tendrá como objetivos principales:

- Crear una aplicación multiplataforma en Android y Web.
- Gestionar las citas a través de los gestores de casos y asignárselas a pacientes con ELA.
- Visualizar las citas que tengan en el día.
- Visualizar las citas que tengan en un calendario.
- Gestionar aquella información necesaria para los pacientes, como tratamientos, hospitales.

B.3. Catálogo de requisitos

Vamos a hacer en el catálogo de requisitos una mención de los requisitos específicos tras el análisis de los objetivos generales.

Requisitos Funcionales (RF)

RF-1 Gestión Gestor Casos

- **RF-1.1 Invitar gestor casos.** El gestor de casos puede enviar una invitación para que un gestor de casos se registre en la aplicación.
- **RF-1.2 Invitar paciente.** El gestor de casos puede enviar una invitación para que el paciente se registre en la aplicación.
- **RF-1.3 Eliminar paciente.** El paciente podrá ser eliminado por parte del gestor de casos.
- **RF-1.4 Listar pacientes.** El gestor de casos podrá ver un listado de pacientes asignados a él.
- **RF-1.5 Ver paciente.** El gestor de casos podrá ver los datos del paciente seleccionado.
- **RF-1.6 Editar gestor casos.** Podrá actualizar sus datos.
- **RF-1.7 Ver citas.** Ver citas.
- **RF-1.8 Crear cita.** Podrá crear cita.
- **RF-1.9 Editar cita.** Podrá crear cita.
- **RF-1.10 Cancelar cita.** Podrá cancelar cita.

RF-2 Gestión Paciente

- **RF-2.1 Editar paciente.** El paciente será el encargado de gestionar sus datos.
- **RF-2.2 Invitar cuidador principal.** El paciente puede enviar una invitación para que el cuidador se registre en la aplicación.
- **RF-2.3 Ver usuarios asociados.** El paciente podrá ver la información del cuidador y el gestor de casos asociados.
- **RF-2.4 Cancelar cita.** Podrá cancelar cita.

RF-3 Gestión Cuidador Principal

- **RF-3.1 Editar cuidador.** El paciente será el encargado de gestionar sus datos.
- **RF-3.2 Invitar cuidador principal.** El paciente puede enviar una invitación para que el cuidador se registre en la aplicación.
- **RF-3.3 Ver pacientes asociados.** El cuidador podrá ver la información del paciente.
- **RF-3.4 Cancelar cita.** Podrá cancelar cita.

RF-4 Gestión Citas

- **RF-4.1 Crear cita.** Crear una cita nueva.
- **RF-4.2 Cancelar cita.** Eliminar cita.
- **RF-4.3 Listar citas.** Ver las citas creadas.
- **RF-4.3 Editar cita.** Editar la cita.

RF-5 Gestión Admin

- **RF-5.1 Invitar admin.** Envía invitación para que un usuario pueda registrarse como admin.
- **RF-5.2 Invitar gestor casos.** Envía invitación para que un usuario pueda registrarse como gestor de casos.
- **RF-5.3 Listar admin.** Puede ver los administradores de la aplicación.
- **RF-5.4 Listar gestores de casos.** Puede ver los gestores de casos de la aplicación.
- **RF-5.5 Listar pacientes.** Puede ver los pacientes de la aplicación.
- **RF-5.6 Listar cuidadores.** Puede ver los cuidadores de la aplicación.
- **RF-5.7 Eliminar usuario.** Puede eliminar usuario.
- **RF-5.8 Ver usuario.** Ver usuario.

- RF-5.9 Listar hospitales. Ver las hospitales.
- RF-5.10 Crear hospital. Crear hospital.
- RF-5.11 Editar hospital. Editar hospital.
- RF-5.12 Eliminar hospital. Eliminar hospital.
- RF-5.13 Listar tratamientos. Ver las tratamientos.
- RF-5.14 Crear tratamiento. Crear tratamiento.
- RF-5.16 Editar tratamiento. Editar tratamiento.
- RF-5.17 Eliminar tratamiento. Eliminar tratamiento.
- RF-5.18 Editar admin. Editar admin.

B.4. Especificación de requisitos

Diagramas de flujo

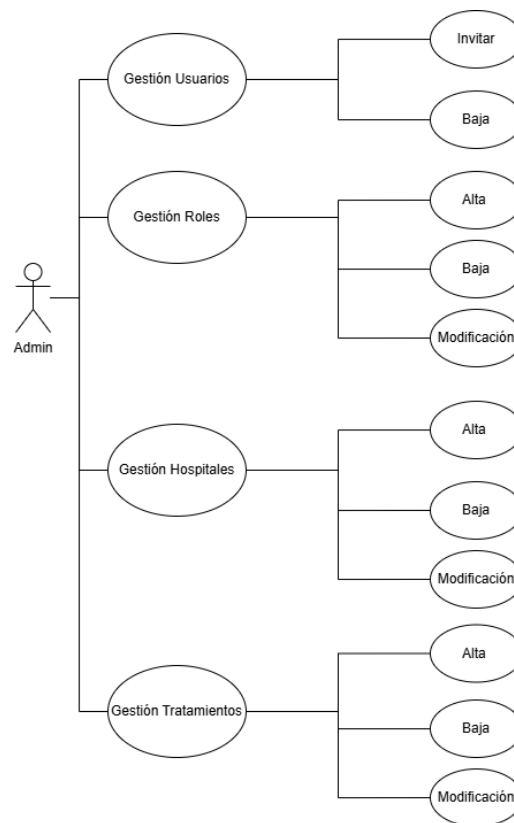


Figura B.1: Diagrama flujo admin

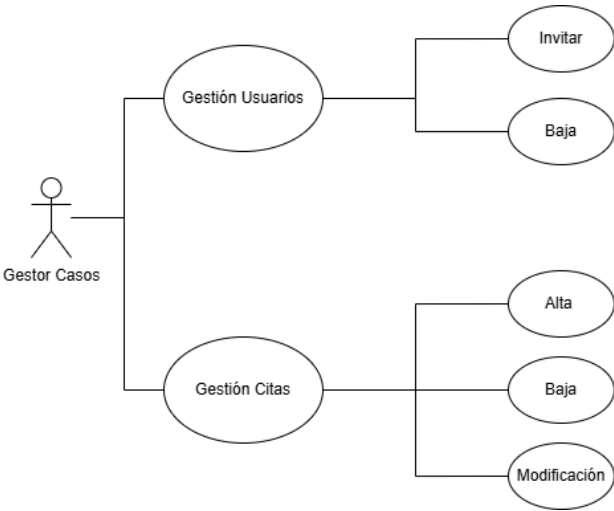


Figura B.2: Diagrama flujo gestor casos

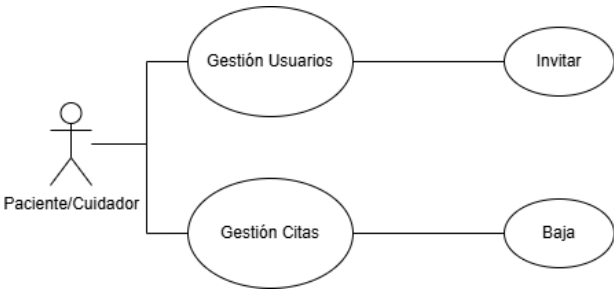


Figura B.3: Diagrama flujo paciente/cuidador

Casos de Uso

CU-1	Gestión usuarios
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-1, RF-2, RF-3, RF-5
Descripción	Permite gestionar los usuarios de la aplicación
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación 2. Ver información personal 3. Todos menos el admin ver las citas 4. Todos menos el admin ver usuarios asociados 5. Si es gestor o admin ver botón invitar a usuario 6. Solo admin ver usuarios (administradores, gestores casos, pacientes y cuidadores principales) 7. Solo admin ver tratamientos 8. Solo admin ver hospitales
Postcondición	Los pacientes y citas coinciden con las que hay en la base de datos
Excepciones	Error al obtener los pacientes o citas
Importancia	Alta

Tabla B.1: CU-1 Gestión Gestores de casos.

CU-2	Enviar invitación
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-1.1, RF-1.2, RF-2.2, RF-3.2, RF-5.1, RF-5.2
Descripción	Permite invitar a un usuario
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación 2. Ir a la pestaña de usuarios 3. Pulsar el botón (+) de añadir 4. Sale un pop up para rellenar los datos 5. Enviar invitación
Postcondición	La invitación es recibida por el usuario final
Excepciones	Error al enviar invitación
Importancia	Alta

Tabla B.2: CU-2 Enviar invitación.

CU-3	Eliminar usuario
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-1.3, RF-5.7
Descripción	Permite eliminar un usuario
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación 2. Ir a la pestaña de usuarios 3. Pulsar papelera para borrar
Postcondición	El usuario se borra
Excepciones	Error al borrar
Importancia	Alta

Tabla B.3: CU-3 Eliminar usuario.

CU-4	Listar usuarios
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-1.4, RF-5.3, RF-5.4, RF-5.5, RF-5.6
Descripción	Permite listar usuarios
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación 2. Ir a la pestaña de usuarios
Postcondición	Se muestran los usuarios
Excepciones	Error al obtener listado
Importancia	Alta

Tabla B.4: CU-4 Listar usuarios.

CU-5	Ver usuario
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-1.5, RF-2.3, RF-3.3, RF-5.8
Descripción	Permite ver los datos personales de un usuario como son nombre y apellidos, fecha de nacimiento, teléfono, correo electrónico. Según el rol que tiene el usuario, además tendrá información adicional relevante.
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación 2. Ir a la pestaña de usuarios 3. Pulsar en usuario para ver la información
Postcondición	El usuario se muestra con la información correcta
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.5: CU-5 Ver usuario.

CU-6	Editar usuario
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-1.6, RF-2.1, RF-3.1, RF-5.8, RF-5.18
Descripción	Permite editar un usuario
Precondición	Firestore está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación 2. Pulsar en el icono usuario 3. Editar la información
Postcondición	El usuario se borra
Excepciones	Error al borrar
Importancia	Alta

Tabla B.6: CU-6 Editar usuario.

CU-7	Gestión citas
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-1.8, RF-1.9, RF-1.10, RF-2.4, RF-3.4, RF-4
Descripción	Permite ver las citas
Precondición	Firestore está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación 2. Ver citas asociadas del día en la home 3. Ver todas las citas en el calendario
Postcondición	Se muestran las citas
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.7: CU-7 Gestión citas.

CU-8	Crear cita
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-1.8, RF-4.1
Descripción	Permite crear una cita
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación 2. Ir al calendario 3. Pulsar botón (+) para crear cita 4. Introducir los datos 5. Aceptar
Postcondición	Se crea la cita
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.8: CU-8 Crear cita.

CU-9	Editar cita
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-1.9 RF-4.4
Descripción	Permite editar una cita
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación 2. Ir al calendario 3. Pulsar sobre una cita 4. Cambiar los datos 5. Aceptar
Postcondición	Se modifica la cita
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.9: CU-9 Editar cita.

CU-10	Cancelar cita
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-1.10, RF-2.4, RF-3.4, RF-4.2
Descripción	Permite cancelar una cita
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación 2. Ir al calendario 3. Pulsar sobre una cita 4. Pulsar sobre cancelar cita
Postcondición	Se cancela la cita
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.10: CU-10 Cancelar cita.

CU-11	Listar hospitales
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-5.9
Descripción	Permite ver los hospitales
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación como admin 2. Pulsar sobre hospitales
Postcondición	Se obtienen los hospitales
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.11: CU-11 Listar hospitales.

CU-12	Listar tratamientos
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-5.13
Descripción	Permite ver los tratamientos
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación como admin 2. Pulsar sobre tratamientos
Postcondición	Se obtienen los tratamientos
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.12: CU-12 Listar tratamientos.

CU-13	Crear hospital
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-5.10
Descripción	Permite crear un hospital
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none">1. Accede a la aplicación como admin2. Pulsar sobre hospitales3. Pulsar sobre el botón (+)4. Introducir los datos5. Aceptar
Postcondición	Se guarda el hospital
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.13: CU-13 Crear hospital.

CU-14	Crear tratamiento
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-5.14
Descripción	Permite crear un tratamiento
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none">1. Accede a la aplicación como admin2. Pulsar sobre tratamientos3. Pulsar sobre el botón (+)4. Introducir los datos5. Aceptar
Postcondición	Se guarda el tratamiento
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.14: CU-14 Crear tratamiento.

CU-15	Editar hospital
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-5.11
Descripción	Permite crear un hospital
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none">1. Accede a la aplicación como admin2. Pulsar sobre hospitales3. Pulsar sobre un hospital4. Introducir los datos5. Aceptar
Postcondición	Se guarda el hospital
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.15: CU-15 Editar hospital.

CU-16	Editar tratamiento
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-5.15
Descripción	Permite editar un tratamiento
Precondición	Firebase está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación como admin 2. Pulsar sobre tratamientos 3. Pulsar sobre un tratamiento 4. Introducir los datos 5. Aceptar
Postcondición	Se guarda el tratamiento
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.16: CU-16 Editar tratamiento.

CU-17	Eliminar hospital
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-5.12
Descripción	Permite eliminar un hospital
Precondición	Firestore está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación como admin 2. Pulsar sobre hospitales 3. Pulsar sobre un hospital 4. pulsar sobre la papelera 5. Aceptar
Postcondición	Se elimina el hospital
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.17: CU-17 Eliminar hospital.

CU-18	Eliminar tratamiento
Versión	1.0
Autor	Javier Martín Castro
Requisitos asociados	RF-5.17
Descripción	Permite eliminar un tratamiento
Precondición	Firestore está disponible
Acciones	<ol style="list-style-type: none"> 1. Accede a la aplicación como admin 2. Pulsar sobre tratamientos 3. Pulsar sobre la papelera 4. Aceptar
Postcondición	Se elimina el tratamiento
Excepciones	Error al obtener los datos
Importancia	Alta

Tabla B.18: CU-18 Eliminar tratamiento.

Apéndice C

Especificación de diseño

C.1. Introducción

En este anexo vamos a definir en base a los objetivos y especificaciones que datos va a manejar la aplicación, que estructura, arquitectura y diseño tendrá.

C.2. Diseño de datos

Nuestra aplicación cuenta con un diseño de base de datos **NoSQL**, lo que nos permite mediante la flexibilidad de la estructura de datos adaptarnos rápidamente a los requisitos y necesidades del negocio. Mediante los documentos que darán vida a nuestras entidades estructurados con formato JSON, nos permitirán crear una aplicación donde el usuario no esté relacionado directamente como sucede en las bases de datos relacionales y nos permita mayor rendimiento cuando la aplicación vaya creciendo.

A continuación, vamos a definir el diseño de datos con el que cuenta nuestra aplicación. Para poder entenderlo más rápidamente, vamos a representar cada elemento de nuestra aplicación como si fuera una entidad relacional.

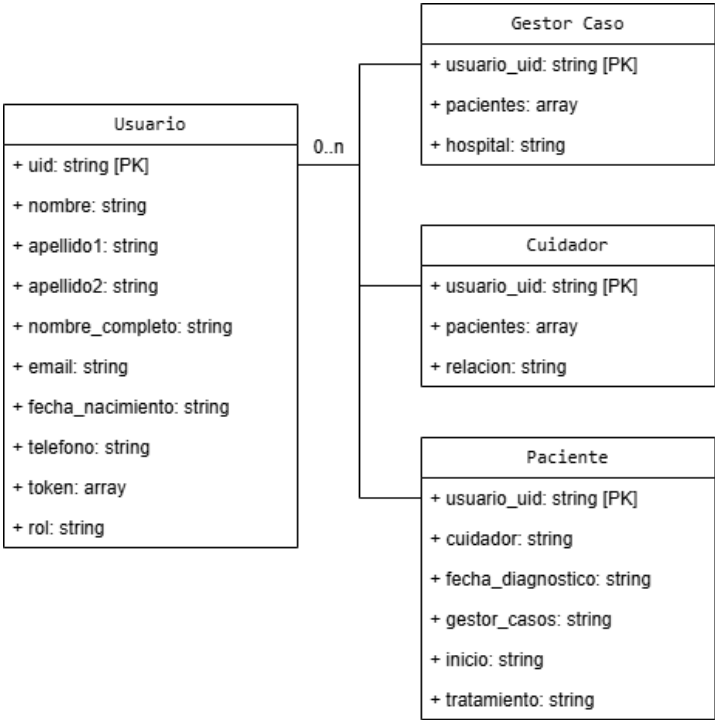


Figura C.1: Diseño de datos actores

Otras entidades que necesitaremos para nuestra aplicación son:

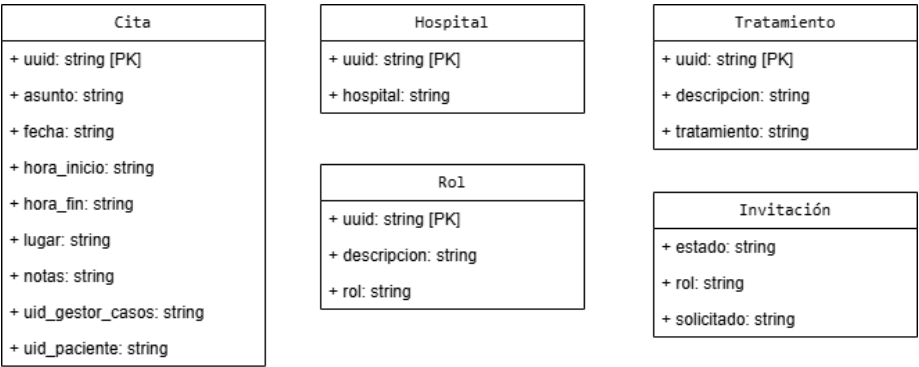


Figura C.2: Entidades

C.3. Diseño procedimental

En este punto vamos a definir el aspecto más relevante de la aplicación y es la gestión de las citas por parte del gestor de casos. En ella mostramos el diagrama de secuencia con la creación y borrado de una cita donde se notificará a los pacientes.

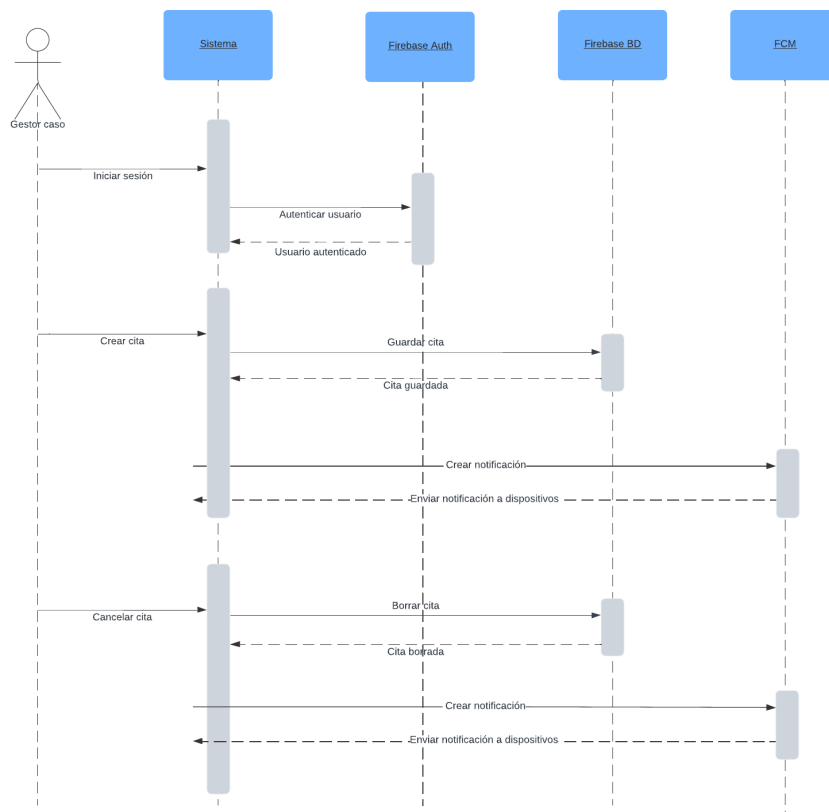


Figura C.3: Diagrama de secuencia Gestor de casos

C.4. Diseño arquitectónico

Al elegir hacer la aplicación con Flutter nos vimos ante la necesidad de realizar una arquitectura limpia y por esa misma razón escogimos usar MVVM (Model-View-ViewModel) estructurando nuestras carpetas de la siguiente forma:

- nuestro model contendrá las entidades que controla la aplicación

- nuestro view contendrá las vistas de la aplicación
- nuestro viewmodel hará de intermediario entre la parte de los datos y la interfaz.

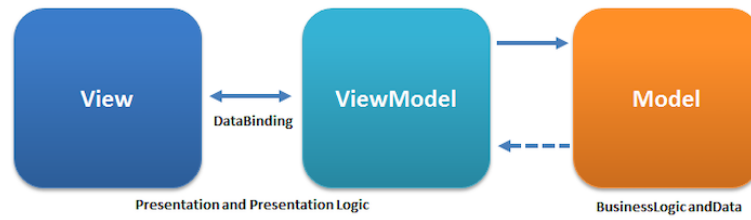


Figura C.4: MVVM

Apéndice D

Documentación técnica de programación

D.1. Introducción

Este anexo va orientado a describir y documentar la información necesaria sobre la poder realizar la instalación de las herramientas necesarias para el desarrollo, que estructura tiene la aplicación, como compilarlo, que servicios se necesitan.

- Cuenta firebase - Generar dotenv - Descargar el proyecto - Comandos para arrancar el proyecto

D.2. Estructura de directorios

La estructura de nuestro proyecto está orientado en la arquitectura MVVM y será la siguiente;



Figura D.1: mvvm

- [/.github](#): carpeta contiene los actions para los despliegues CI/CD en web y android al realizar push sobre la rama **main**.
- [/android](#): carpeta que contiene el código para la aplicación android.
- [/assets](#): carpeta que contendrá las imágenes y fichero dotenv para las variables de entorno.

- `/doc`: carpeta que contendrá la memoria y anexos.
- `/lib`: directorio donde se encuentra el código fuente del proyecto.
 - `main.dart`: fichero principal del proyecto.
 - `/app`: contendrá la estructura de carpetas de nuestro Model-View-ViewModel
 - `/data`: contiene las implementaciones a los repositorios
 - `/domain`: contiene los modelos y las interfaces de los repositorios
 - `/presentation`: contiene los modulos que representan las vistas de nuestra aplicación
 - `/l10n`: contendrá los archivos para la internacionalización de la aplicación
- `/web`: carpeta que contiene el código para la aplicación web
- `analysis_options.yaml`
- `build.yaml`
- `pubspec.yaml`: archivo que contendrá los paquetes necesarios para el correcto funcionamiento de la aplicación
- `l10n.yaml`: archivo para la internacionalización de la aplicación

D.3. Manual del programador

Este manual está pensado para que futuros programadores puedan poner en marcha el proyecto y saber que pasos deben seguir para el correcto funcionamiento.

Cómo primer requisito es instalar Flutter [?], para ello tendremos que ir a la página oficial de flutter <https://docs.flutter.dev/get-started/install> y elegir el sistema donde vamos a realizar la instalación y para que tipo de aplicación queremos desarrollar (en nuestro caso android).

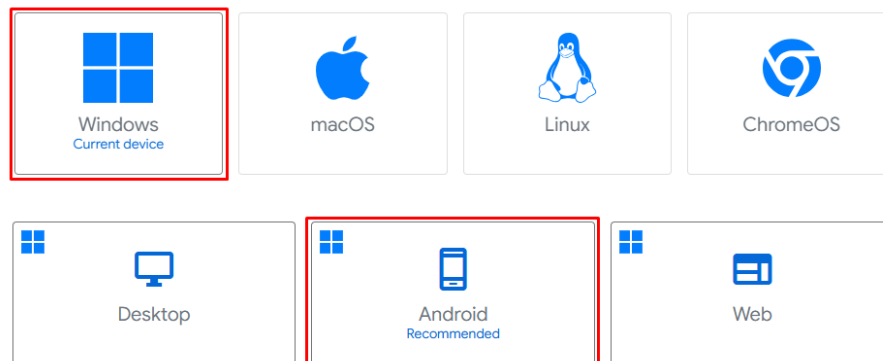


Figura D.2: Plataforma desarrollo

Una vez realizada las instalaciones y configuraciones correspondientes, debemos comprobar que todo ha ido correctamente, para ello utilizaremos la siguiente instrucción en el terminal de **VS Code flutter doctor**

```
PS D:\UBU\2023-24\TFG\ELACyL\red-ela> flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.22.2, on Microsoft Windows [Versión 10.0.19045.4598], locale es-ES)
[✓] Windows Version (Installed version of Windows is version 10 or higher)
[✓] Android toolchain - develop for Android devices (Android SDK version 34.0.0)
[✓] Chrome - develop for the web
[✓] Visual Studio - develop Windows apps (Visual Studio Community 2022 17.6.4)
[✓] Android Studio (version 2023.3)
[✓] VS Code (version 1.90.2)
[✓] Connected device (3 available)
[✓] Network resources

• No issues found!
```

Figura D.3: Comprobación correcta instalación Flutter

Visual Studio Code

Para ello debemos de tener instalado VS Code, en caso contrario descargar de la página oficial <https://code.visualstudio.com/>.

Una vez instalado añadimos la extensión de Flutter

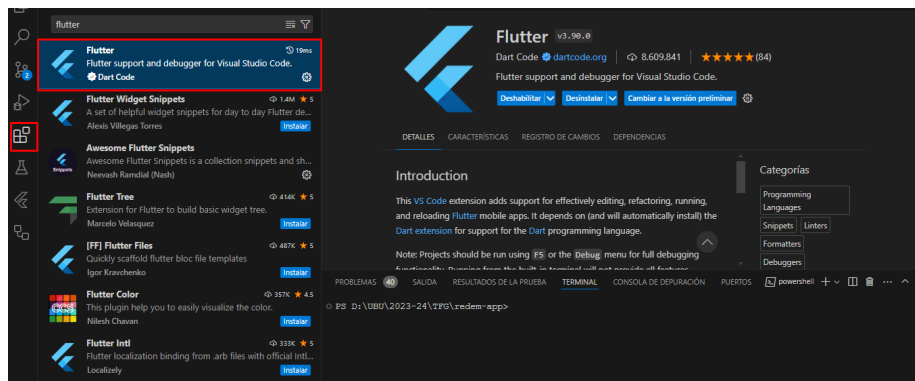


Figura D.4: Añadir Flutter a Visual Studio Code

Flutter SDK

Siguiendo los pasos que indican en la guía de Flutter, nosotros escogimos instalar Flutter SDK desde VS Code, para ello debemos hacer:

1. Abrir **VS Code**
2. Pulsar **Ctrl+Shift+P** para abrir la paleta de comandos
3. Seleccionar **New Project**
 - Si no lo tienes descargado, pulsa en **Download SDK**.
 - Si lo tienes descargado, pulsa **Locate SDK**
4. Cuando te pregunte que plantilla usar, hay que ignorarlo.

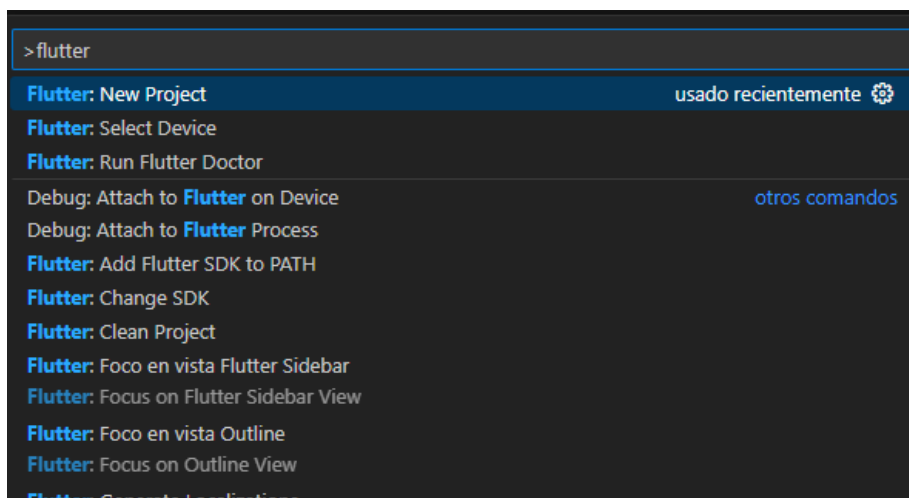


Figura D.5: Paleta de comandos VS Code

Android Studio

Para ello si no lo tenemos instalado, tendremos que realizar la instalación de **Android Studio** desde <https://developer.android.com/studio/install#windows> en mi caso lo instalo para Windows. Una vez instalado deberemos instalar el plugin de Flutter para IntelliJ <https://plugins.jetbrains.com/plugin/9212-flutter>

Para poder crear aplicación con Flutter, necesitamos comprobar que tenemos instalados los siguientes componentes de Android:

- Android SDK Platform, API 34.0.5
- Android SDK Command-line Tools
- Android SDK Build-Tools
- Android SDK Platform-Tools
- Android Emulator

Configuración Android Studio

Realizar la siguiente configuración de android studio.

1. Habilite la aceleración de VM en su computadora de desarrollo.

2. Inicie **Android Studio**.
3. Vaya al diálogo de Configuración para ver el **Administrador de SDK**.
 - Si tiene un proyecto abierto, vaya a **Herramientas > Administrador de dispositivos**.
 - Si se muestra el diálogo de bienvenida, haga clic en el icono pulsar **más opciones** que sigue al botón **abrir** y seleccione **administrador de dispositivos** desde el menú desplegable.
4. Haga clic en **virtual**.
5. Haga clic en **crear dispositivo**. El diálogo de configuración de dispositivo virtual se muestra.
6. Seleccione el tipo de dispositivo bajo la categoría.
7. Seleccione una definición de dispositivo. Puede explorar o buscar por el dispositivo.
8. Haga clic en **siguiente**.
9. Haga clic en **imagenes x86**.
10. Haga clic en una imagen del sistema para la versión de Android que desee emular.
 - Si la imagen deseada tiene un icono de descarga al lado del nombre de lanzamiento, haga clic en él.
 - Cuando complete la descarga, haga clic en **finalizar**.
11. Haga clic en **siguiente**. La configuración del dispositivo virtual se muestra en su paso de verificación.
12. Para renombrar el **Dispositivo Virtual de Android (AVD)**, cambie el valor en el cuadro de texto **nombre AVD**.
13. Haga clic en **mostrar configuraciones avanzadas** y desplace hasta **rendimiento emulado**.
14. Desde el menú desplegable **gráficos**, seleccione **Hardware - GLES 2.0**. (Esto habilita la aceleración de hardware y mejora el rendimiento de renderizado).

15. Verifique la configuración de su AVD. Si todo está correcto, haga clic en **finalizar**.
16. En el diálogo del Administrador de dispositivos, haga clic en el icono **ejecutar** al lado derecho del AVD deseado. El emulador se inicia y muestra la pantalla predeterminada para la versión de Android OS y dispositivo seleccionados.

Fichero variables de entorno

Antes de generar el fichero, debemos seguir los pasos descritos en el paquete de **Flutter encrypt** <https://pub.dev/packages/encrypt>:

- Activar el paquete `pub global activate encrypt`

Para generar las claves seguras de manera aleatoria utilizaremos la siguiente instrucción `secure-random`. Con ella podremos generar claves que nos servirán para encriptar de manera segura los datos tratados por nuestra aplicación.

```
PS D:\UBU\2023-24\TFG\ELACyL\red-ela> secure-random
Can't load Kernel binary: Invalid kernel binary format version.
Building package executable... (3.4s)
Built encrypt:secure-random.
S/95q1MQQVLKBCj8KIj0INd915NIe4ladQJ/iquNLZk=
PS D:\UBU\2023-24\TFG\ELACyL\red-ela>
```

Figura D.6: Encrypt secure random

Este fichero lo tenemos que generar dentro de la carpeta **assets** con el nombre **dotenv**, ya que está ignorado en las subidas a **Github**. El fichero contendrá unas claves necesarias para el correcto funcionamiento de la aplicación:

- **ENCRYPT_KEY** esta clave la generaremos con la instrucción anterior y cuya longitud debe de ser de 32
- **ENCRYPT_IV** esta clave la generaremos con la instrucción anterior y cuya longitud debe de ser de 12
- **USER_EMAIL**=redelaubutfg@gmail.com
- **USER_PASS**=qdmucyqwausbluzw

- **VAPID_KEY** esta clave la necesitamos para poder obtener notificaciones push en nuestra web. Explicado en **D.3 Firebase**

Firestore

Crearemos una cuenta Firestore y agregaremos un proyecto con el nombre **RedELA**

1. Agregaremos el proyecto

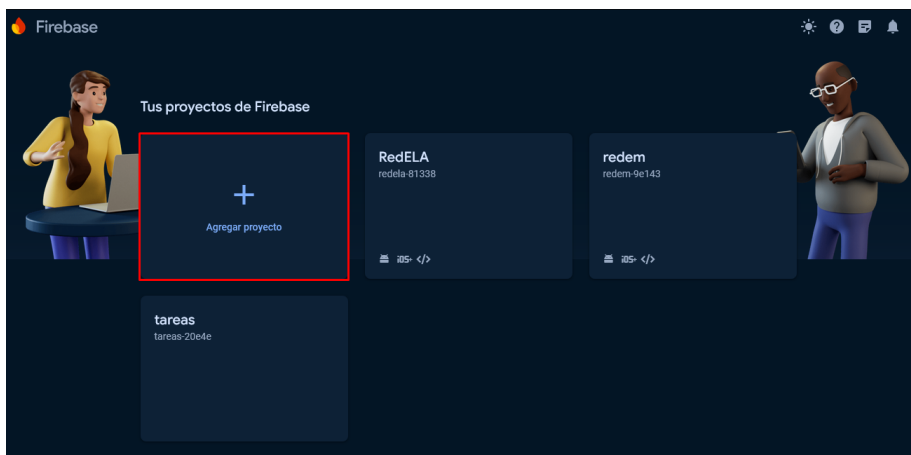


Figura D.7: Agregar proyecto

2. Introduciremos el nombre del proyecto. Pulsaremos **continuar**

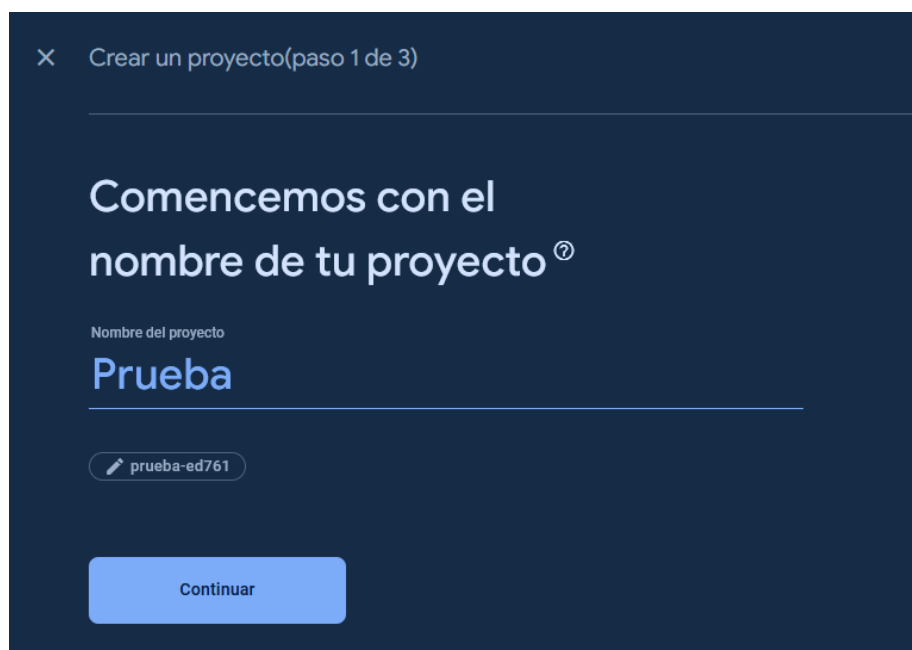


Figura D.8: Agregar proyecto

3. Deshabilitaremos Google Analytics y pulsaremos **crear proyecto**

Tras unos segundos tendremos creado nuestro proyecto, ahora necesitamos enlazar ese proyecto con nuestro código descargado desde **D.3 Github** <https://github.com/jmc1005/red-ela>

Una vez creado nuestro proyecto en Firebase, debemos obtener el **VAPID_KEY** en **descripción general configuración del proyecto** vamos a la sección de **cloud messaging** y al final en **configuración web** en **certificados push web** encontraremos en **par de claves** nuestro **VAPID_KEY**.

Github

Para empezar, necesitamos crear nuestros secrets en Github. Para acceder a **secrets** debemos acceder a **Settings** y luego sobre el menú lateral izquierdo ir a **Secrets and variables** y pulsaremos sobre **actions**

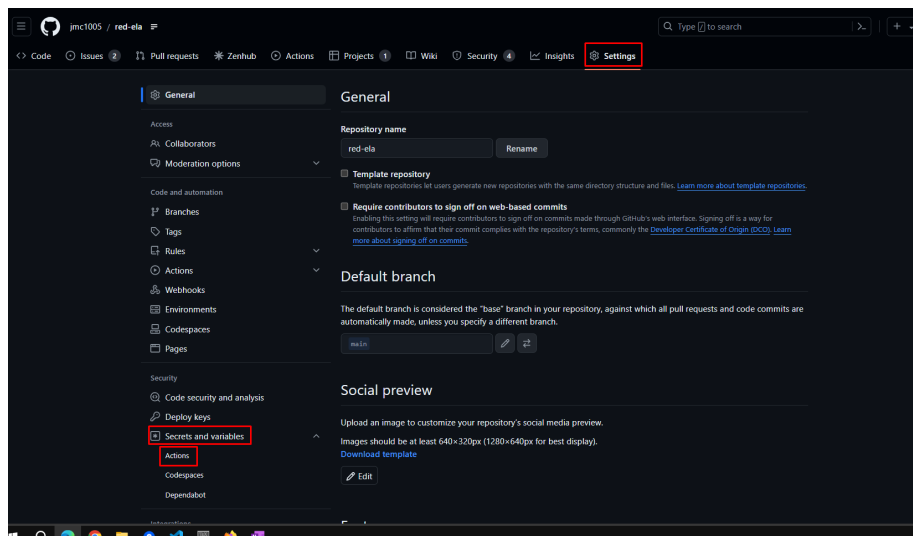


Figura D.9: Secrets and variables

Posteriormente, sobre **Repository secrets** añadiremos nuestras variables necesarias para los despliegues mediante los workflow.

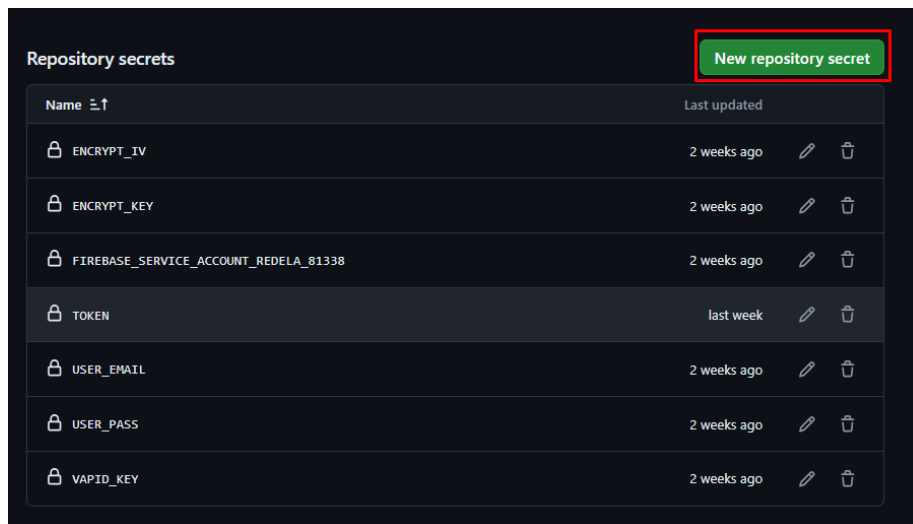


Figura D.10: Repository secrets

En este punto vamos a explicar un poco a cerca de los workflows definidos en nuestro proyecto y además crear los secrets que serán necesarios para nuestro proyecto.

Los workflow son acciones que nos van a permitir desplegar en el hosting de firebase nuestra web y generar nuestras apks

Google Play

De momento sólo estamos haciendo despliegues en plan Prueba Interna, para poder desplegar la aplicación para prueba interna necesitamos generar:

1. Modificar nuestro fichero **pubspec.yaml** cambiando la versión por uno más es decir, si tenemos **version: 1.0.3+3** poner **version: 1.0.4+4**
2. un **bundle** con el comando `flutter build appbundle`, una vez generado el **bundle** accedemos a nuestra **consola de google play** <https://play.google.com/console/u/0/developers>
3. pulsar en **crear nueva versión**

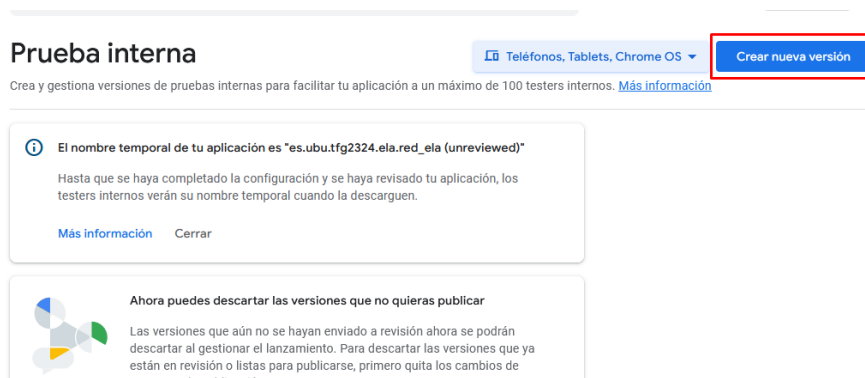


Figura D.11: Nueva versión

4. Subir el bundle generado y actualizar la versión, por ejemplo a 4 (1.0.4)

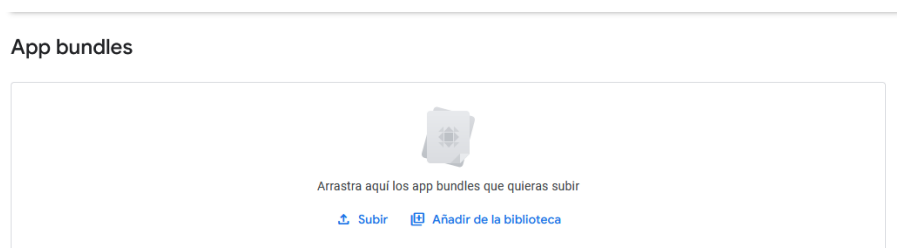


Figura D.12: Subir bundle

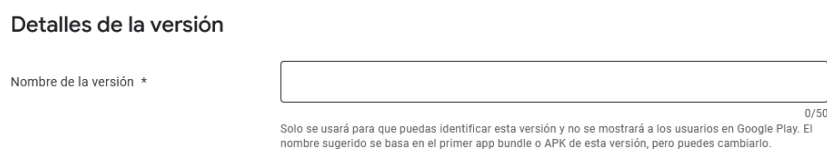


Figura D.13: Detalle versión

D.4. Compilación

Para poder compilar el proyecto debemos ejecutar estos comandos `dart run build_runner build` y `flutter pub get`. El primer comando nos generará el código automático de las clases que están con la anotación `@freezed` en nuestro proyecto. Estas anotaciones nos permiten generar código automático con pocas líneas de código.

D.5. Ejecución del proyecto

La aplicación se puede ejecutar tanto en la web como en un dispositivo móvil. Para ejecutar las pruebas podemos realizarlas tanto en un dispositivo móvil con Android que tenga API 34 o bien accediendo a la web.

También se pueden hacer pruebas en local desde VSCode `flutter run -d chrome` si queremos que se ejecute en chrome o bien seleccionando un dispositivo como se muestra en la imagen.

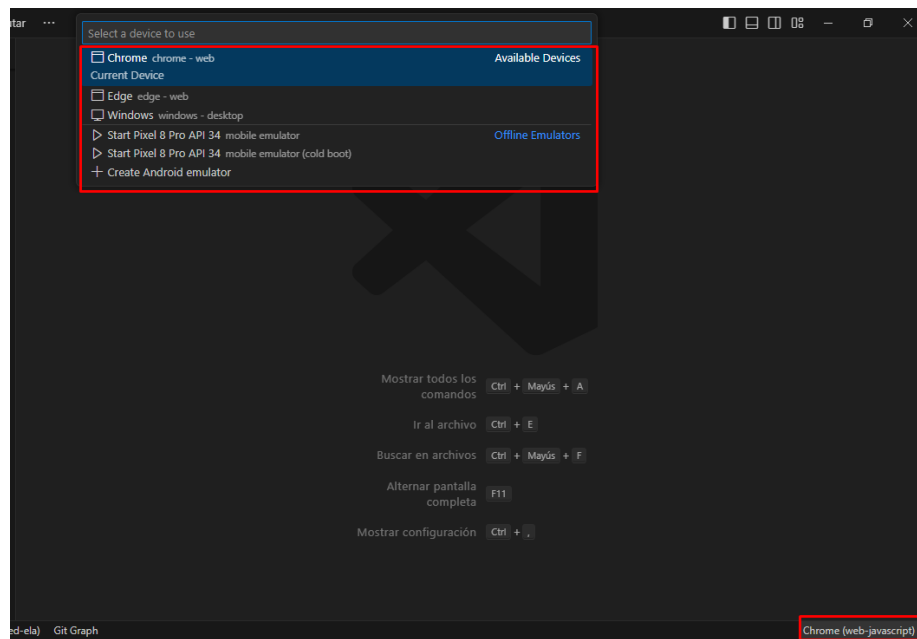


Figura D.14: Seleccionar dispositivo

D.6. Pruebas del sistema

Se ha generado una clase que testea mediante **flutter_test** Firebase que será el gestione los datos que utiliza la aplicación.

Para poder ejecutar los test deberemos pulsar los iconos que serán un play

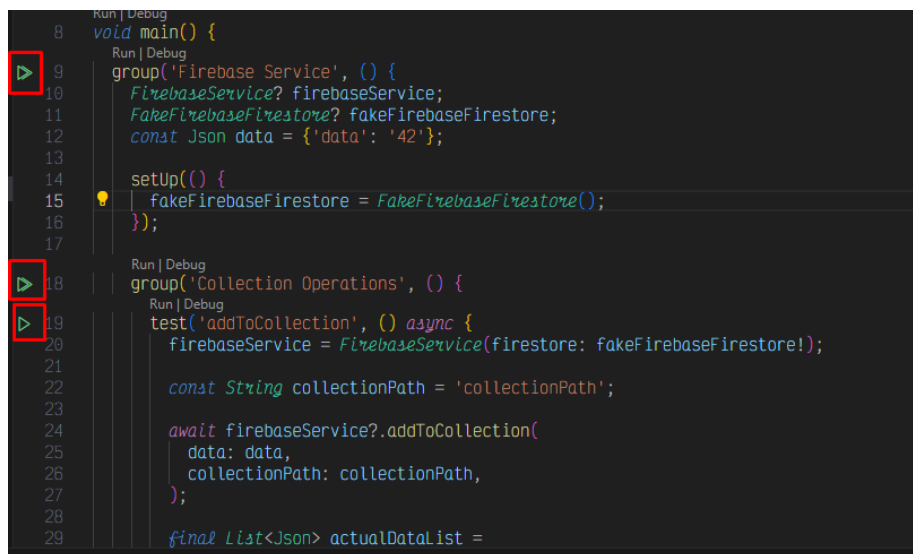


Figura D.15: Test

Cuando los tests pasan se verá de la siguiente manera.

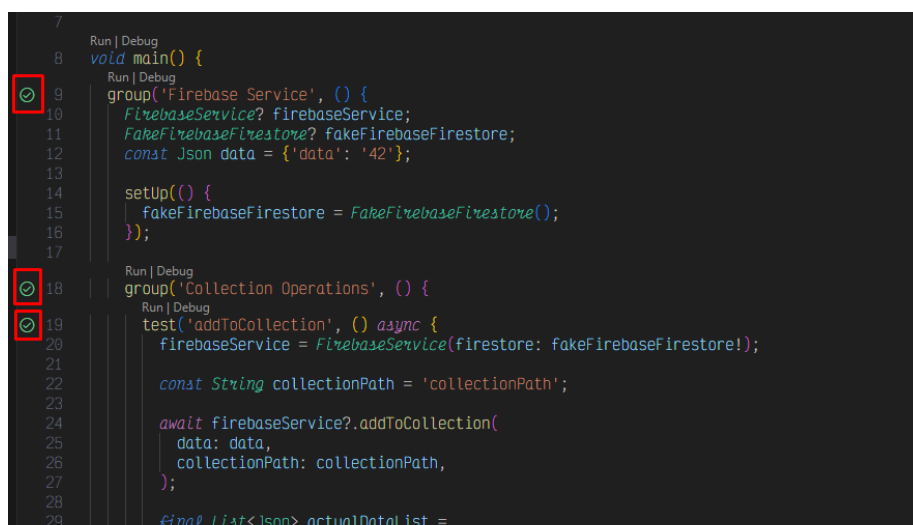


Figura D.16: Test OK

Apéndice E

Documentación de usuario

E.1. Introducción

En este manual se indican que requisitos mínimos debe tener para poder utilizar la aplicación en un dispositivo móvil con sistema operativo Android 13 o superior.

E.2. Requisitos de usuarios

Los requisitos mínimos para hacer uso de la aplicación Android son:

- Contar con un dispositivo Android 13 (Tiramisú – API 33) o superior.

Para acceder a la web se podrá hacer con cualquier dispositivo que contenga un navegador.

E.3. Instalación

Se podrá instalar la aplicación en un dispositivo móvil con Android desde **Google play** o a partir desde la apk activando el modo desarrollador en el dispositivo android.

E.4. Manual del usuario

Para empezar esta aplicación consta por un registro por invitación, es decir, que para que un usuario pueda realizar el registro debe ser invitado por

otra persona. En un primer momento, las invitaciones serán las siguientes según los actores de la aplicación:

- El **administrador** puede invitar a nuevos **administradores** o a **gestores de casos**
- El **gestor de casos** puede invitar a nuevos **gestores** o a **pacientes**
- El **paciente** puede invitar a un único **cuidador**

En la parte superior de la aplicación podremos actualizar y modificar nuestros datos o salir de la aplicación.



Figura E.1: Acceder a RedELA

Las pantallas de la aplicación serán:

- Acceder mediante **correo electrónico** y **contraseña**



The screenshot shows a mobile application interface for RedELA. At the top, there is a status bar with the time 13:30. Below it is the RedELA logo, which consists of a blue square with a white stylized figure and the letters 'REDELA' in white. Under the logo are two input fields: the first is labeled 'Correo electrónico' and the second is labeled 'Contraseña' with an eye icon to its right. Below these fields is a dark blue button labeled 'Acceder'. At the bottom, there is a link that says 'Acceder con código [Registrarse](#)'.

Figura E.2: Acceder a RedELA

- Registro. Para poder registrarse se necesita haber recibido un correo electrónico que permite registrarse inicialmente a partir de una **contraseña de un sólo uso**.

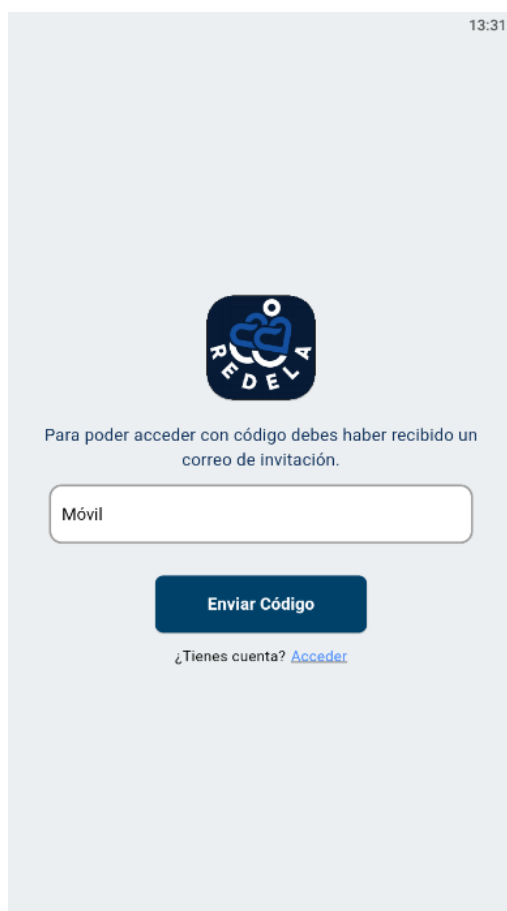


Figura E.3: Registro en RedELA

- Home. En esta pantalla se verán las citas que se tendrán en el día.

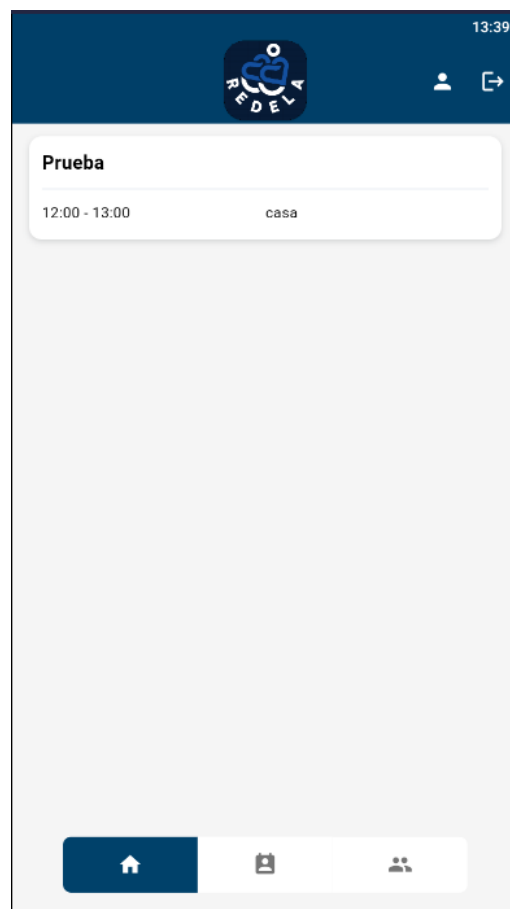


Figura E.4: Home

- Citas. En esta parte de la aplicación podremos ver el calendario con las citas, podemos ver las citas en el calendario mensual o en el calendario de listas programadas.

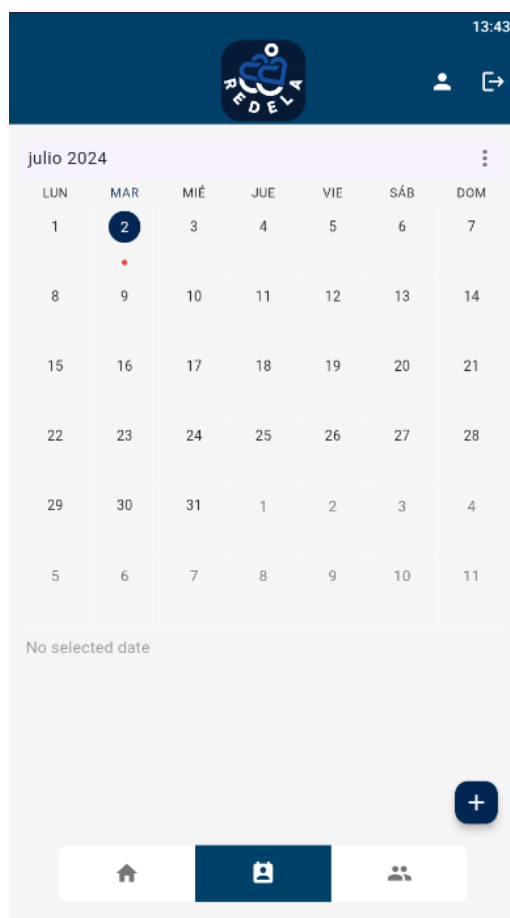


Figura E.5: Citas

- Usuarios asociados. En esta parte veremos los usuarios relacionados. Es decir, si accede un gestor de casos, verá todos los pacientes asignados a él, el en caso de ser un paciente, se verá la información del cuidador principal y del gestor de casos, y en caso de ser un cuidador principal, aparecerá la información del paciente relacionado con él.

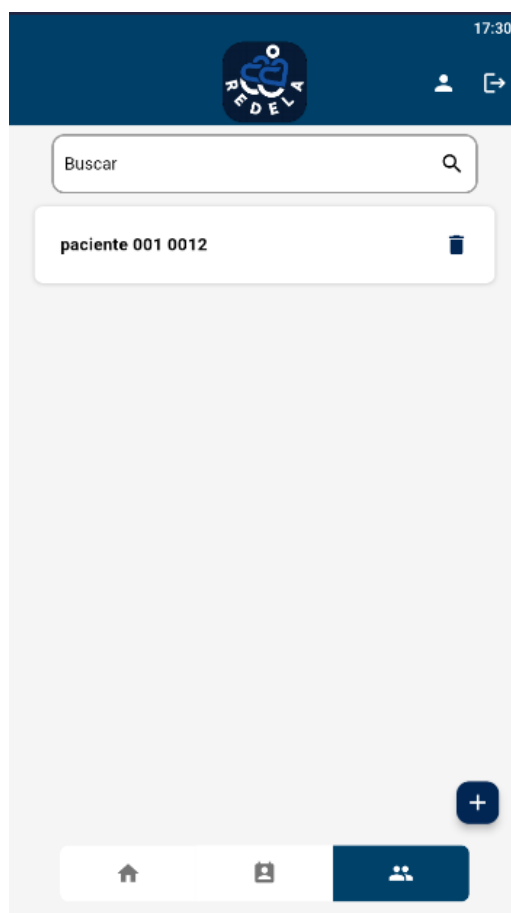


Figura E.6: Pacientes asociados gestor de casos

17:32

Cuidador/a

Nombre y apellidos
cuidador 001 001

Móvil
+34 638123001

Relación
Hermano

Enfermero/a gestor/a casos

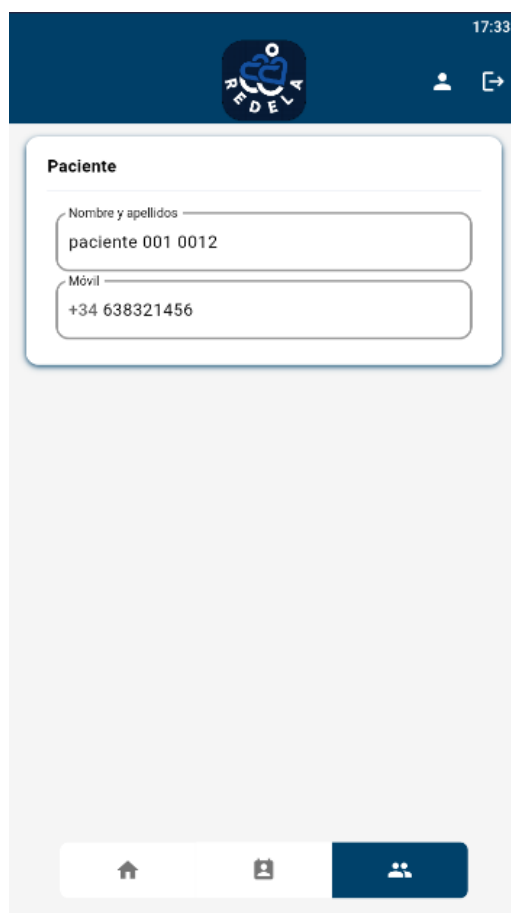
Nombre

Hospital

Móvil

Home, Profile, and Group icons in the bottom navigation bar.

Figura E.7: Cuidador y gestor de casos asociado a paciente



The screenshot displays the REDELY mobile application interface. At the top, a dark blue header bar contains the REDELY logo on the left, a user profile icon and a share icon on the right, and the time 17:33 in the top right corner. Below the header, a white card titled "Paciente" contains two input fields. The first field, labeled "Nombre y apellidos", contains the text "paciente 001 0012". The second field, labeled "Móvil", contains the text "+34 638321456". At the bottom of the screen, a white navigation bar features three icons: a home icon, a calendar icon, and a group of people icon, which is currently highlighted with a dark blue background.

Figura E.8: Paciente asociado a cuidador

E.5. Manual del administrador

Una vez accedido como admin veremos la pantalla principal donde podrá gestionar a los usuarios, roles, hospitales y tratamientos. Además de poder salir de la aplicación.

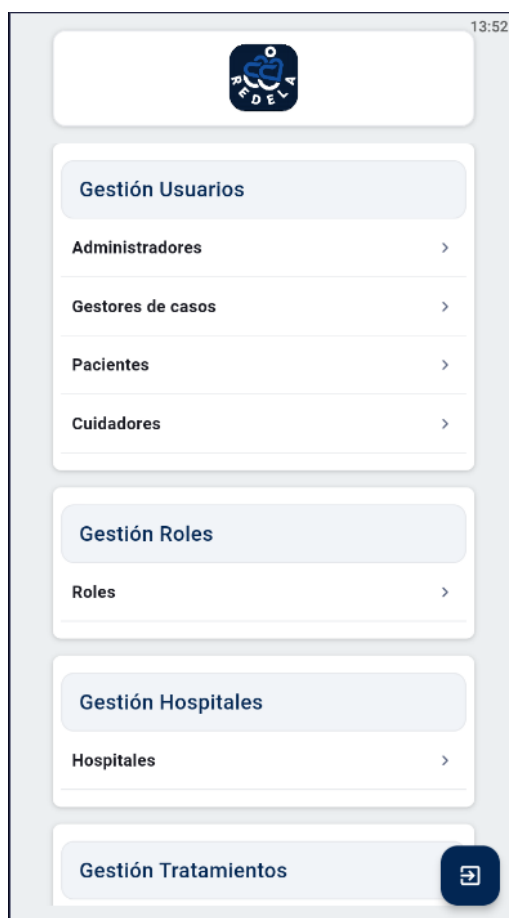


Figura E.9: Home Admin

Gestión de Usuarios

Inicialmente se concibe la aplicación para que un usuario en la **gestión del usuario** sólo pueda agregar a **administradores** o **gestores de casos**, ya que consideramos que la información de los pacientes por protección de datos es un tema que hay que proteger.

Gestión de Roles

Gestión de Hospitales

Gestión de Tratamientos

Gestión de Usuarios

Anexo de sostenibilización curricular

F.1. Introducción

El objetivo del proyecto es crear una aplicación que facilite el día a día de los pacientes diagnosticados con ELA (Esclerosis Lateral Amiotrófica) [?]. Esta aplicación está creada con el framework Flutter [?] y con la plataforma Firebase [?] que nos permite gestionar inicialmente las citas y tratamientos que tiene el paciente.

Además, con el desarrollo de la aplicación, se pretende concienciar y dar sostenibilidad al cuidado de personas con ELA haciendo que se comprometan aquellos profesionales de la salud y cuidadores principales ante los cuidados necesarios para hacer la vida más fácil a personas que hayan sido diagnosticadas con la enfermedad.

Para realizar el anexo de sostenibilidad se utiliza como referencia el documento de la CRUE [?]

Objetivos

Algunos de los objetivos que podemos destacar son:

- Concienciar la importancia sobre la sostenibilidad sobre los cuidados que necesitan los paciente con ELA.
- Concienciar a crear soluciones sostenibles para el cuidado de pacientes con ELA.

- Realizar desarrollos que promuevan la responsabilidad para dar solución a pacientes con ELA.

Estrategias

Algunas estrategias que se deben de usar son:

- Realizar formaciones que fomenten la colaboración entre los profesionales de la salud, pacientes y cuidadores principales.
- Sensibilizar y dar visibilidad a los pacientes que padezcan esta enfermedad.
- Realizar desarrollos que promuevan la responsabilidad para dar solución a pacientes con ELA.

Conclusión

Sacamos como conclusión que la realización del proyecto aporta a la soluciones a la sociedad facilitando la vida a personas enfermas de ELA, a los cuidadores principales e incluso a los profesionales sanitarios. Además, a nivel medio ambiental reduce el uso de emisiones contaminantes ahorrando en el gasto de papel. Con ello se tendrá en cuenta la aceptación, demanda y aceptación por parte del usuario de la aplicación para realizar ajustes y nuevas funcionalidades a lo largo del tiempo