Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Aplicación móvil para la gestión de pacientes con Mieloma Múltiple.

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Fenollar Onrubia, Daniel

Tutor: Manjón Herrera, Jose Vicente

Cotutor: Bayo Montón, Jose Luis

2018-2019

Resumen.

El sistema NuMielo ha sido ideado en colaboración con la Unidad de Investigación Clínica Servicio de Hematología del Hospital Universitario y Politécnico La Fe, cuyo objetivo es llevar un control de los pacientes crónicos con Mieloma Múltiple y así mejorar su calidad de vida. El sistema permite introducir el peso, los síntomas, rellenar cuestionarios para llevar un control del estado anímico del paciente y la calidad de vida del mismo. El sistema se integra con otra aplicación web, donde los profesionales sanitarios pueden consultar los datos enviados desde la aplicación móvil y posteriormente realizar diagnósticos o llevar el control del paciente.

Palabras clave: Aplicación, cáncer, móvil, Mieloma Múltiple, telemedicina, teleasistencia, crónico, Android, dispositivo.

Abstract.

The NuMielo system has been designed in collaboration with the Clinical Research Unit of the Haematology Service of the La Fe University and Polytechnic Hospital, whose objective is to control chronic patients with Multiple Myeloma and thus improve their quality of life. The system allows you to introduce the weight, the symptoms, to fill in questionnaires to keep track of the patient's mood and quality of life. The system is integrated with another web application, where health professionals can consult the data sent from the mobile application and then make diagnoses or keep track of the patient.

Keywords: Application, cancer, smartphone, Multiple Myeloma, telemedicine, telecare, chronic, Android, device.

Índice de contenidos.

Índice de ilustraciones.

Índice de tablas.

1. Introducción.

Primeramente, y antes de profundizar en este TFG, debemos hablar del proyecto NuMielo y la enfermedad que engloba, el Mieloma Múltiple (MM). Dicha enfermedad es un tipo de cáncer de la sangre que afecta principalmente a personas mayores; esta provoca debilidad, formación de tumores óseos y fracturas óseas e insuficiencia renal entre otros de sus síntomas, acompañado de dolor que se acaba cronificando.

El MM afecta a 2.466 personas en España y su tasa de mortalidad es de 4,6 por 100.000 habitantes, las infecciones son una de las principales causas de muerte (1). Esta cantidad incluye el 2% del total de cánceres, el 10% de las neoplasias hematológicas y es la causa de mortalidad de 20% de las neoplasias hematológicas (2).

El Mieloma Múltiple es, a día de hoy, una enfermedad sin cura que además de afectar físicamente por la toxicidad asociada al tratamiento farmacológico, merma el estado psicológico y nutricional del paciente. Actualmente no se dispone de atención que cubra estas necesidades para pacientes oncohematológicos como parte de la práctica clínica habitual. Esta carencia genera una gran brecha entre clases sociales ya que aquellos con menos recursos no tienen acceso a tratamientos más completos que cuidan estos aspectos (3).

Por todo ello, se propone evaluar el estado nutricional, la calidad de vida y estatus psicosocial de los pacientes con MM del Servicio de Hematología del Hospital La Fe, ofreciendo al mismo tiempo una asistencia personalizada que ofrezca soluciones que se ajusten a las necesidades de cada paciente. NuMielo nace con la intención de empoderar al paciente con MM, dotándole de los recursos suficientes para gestionar su salud, integrando el aspecto nutricional y psicológico.

En resumen, NuMielo plantea la implementación de una solución software, compuesta por una aplicación móvil Android, destinada a los pacientes que sufren MM, y una aplicación web, para la gestión de los pacientes por parte de los profesionales de la salud. Profundizaremos en el diseño, desarrollo y pruebas de la aplicación móvil a lo largo de este TFG.

1.1. Motivación.

Como se ha comentado en la introducción, este TFG forma parte de un proyecto de mayor envergadura. El motivo principal del autor es lograr que el subsistema de la aplicación sea funcional y que contribuya al objetivo del proyecto NuMielo.

Respecto al objetivo del autor, la selección de este es la posibilidad de estrechar más los lazos entre el mundo de la informática y el mundo de la medicina, creando una aplicación que pueda contribuir con el tratamiento de personas que sufren este tipo de cáncer. También añade tanto la ocasión de ayudar a la mayor cantidad de personas posible como profundizar en el desarrollo de aplicaciones móviles en el entorno Android.

1.2. Objetivos.

El objetivo del proyecto NuMielo es crear un sistema para el seguimiento de pacientes que permita mejorar el tratamiento y, en consecuencia, ganar calidad de vida. Durante este seguimiento, se recogen datos del proceso de la enfermedad que facilita estudios posteriores para mejorar dicho proceso de cuidado para futuros pacientes y, además, permite conocer mejor la enfermedad.

Como se explica detalladamente en el apartado de solución propuesta, este sistema se divide en dos subsistemas: una aplicación móvil destinada para los pacientes y una aplicación web para los profesionales de la salud junto con la administración y recogida de datos.

Dentro de los objetivos del proyecto se encuentra el objetivo del autor y, en consecuencia, del TFG. Este es desarrollar el subsistema de aplicación móvil, una herramienta para el seguimiento y recolección de datos que posteriormente visualizan los profesionales de la salud.

Para poder alcanzar este objetivo, en este TFG se realizan las fases de análisis de requisitos, planificación, desarrollo e implementación y, finalmente, pruebas y validación.

Como objetivos secundarios, uno de los aspectos más importantes es el diseño de la interfaz, ya que esta está destinada a personas de una avanzada edad. Se siguen los estilos y las líneas de diseño de *Material Design*[1] de Google. Así, se obtiene una

[1] https://material.io/design/

aplicación más atractiva, fácil de utilizar, con un aspecto más profesional, mayor usabilidad y accesibilidad.

La traducción de la aplicación a cualquier lenguaje, mediante el uso de los recursos[2] de esta, es otro aspecto necesario, consiguiendo así, ampliar el número de usuarios que podrán utilizar la aplicación.

Con la finalidad de poder mantener o modificar la aplicación, se deben seguir un conjunto de buenas prácticas, como una buena documentación del código, la utilización de patrones arquitectónicos, proporcionando una estructura robusta y mantenible.

[2] https://developer.android.com/training/basics/supporting-devices/languages

1.3. Metodología.

En este apartado se va a hacer una breve introducción de la metodología seguida por el autor.

Se ha llevado a cabo una metodología ágil siguiendo un proceso incremental (4) haciendo uso del tablero Kanban (5) que podemos observar en la ilustración 1. Este proceso consiste en la creación de versiones del producto a medida que se van terminando ítems o unidades de trabajo, estas pueden ser nuevos requisitos, mejoras sobre un requisito ya introducido o correcciones de fallos.

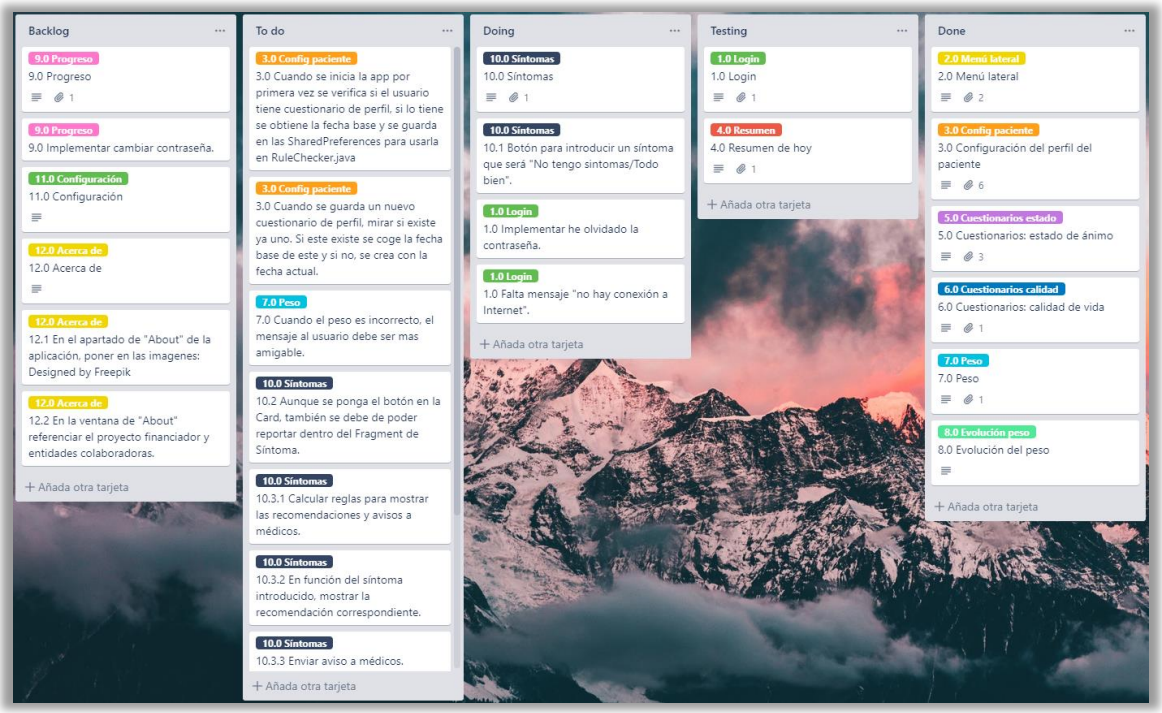

Ilustración 1. Tablero Kanban realizado en Trello.

Tratándose de un equipo de desarrollo formado por un grupo muy reducido, se ha optado por seguir un flujo de trabajo sencillo con las tres actividades básicas: *to do*, *doing*, *done* (pendiente, en proceso y terminado, respectivamente). A estas se le añade el backlog, un contenedor de trabajo donde se sitúa todos los ítems que se van a desarrollar, y el testing, donde permanecerán todos los ítems en fase de prueba.

1.4. Estructura.

Vamos a comentar la estructura seguida durante la realización del trabajo. En primer lugar, en el capítulo 2, se realiza una búsqueda de las aplicaciones del ámbito sanitario y se habla más detalladamente del grupo de investigación Itaca-Sabien, dando a conocer con profundidad el trabajo realizado por este. Acto seguido, en el capítulo 3, entramos en el análisis del problema mostrando los requisitos, casos de uso, prototipado y solución propuesta, donde se encuentra el proceso seguido para llegar a la solución. Una vez se expone el plan de trabajo en el capítulo anterior, se comenta el diseño de la solución, explicando la arquitectura del sistema y la estructura de la app, este se encuentra en el capítulo 4. En este mismo, también se habla de las herramientas y tecnologías utilizadas para el desarrollo del sistema.

A continuación, se expone el desarrollo de la solución donde se comenta el proceso de implementación de las funcionalidades y las ventanas del sistema, recopilado en el capítulo 5. Tras realizar el desarrollo, en el capítulo 6, se realizan las pruebas de validación de la app, verificando el correcto funcionamiento de esta. Para finalizar, se muestran las conclusiones obtenidas tras todo el proceso de creación del trabajo y los conocimientos aplicados para ello. Se añaden tanto un glosario de términos y abreviaturas como cuatro anexos que referencian a los cuestionarios sanitarios utilizados en la app, los síntomas que se reportan y la escala de heces Bristol utilizada.

2. Estado del arte.

Para empezar, el proyecto NuMielo se desarrolla junto con el grupo de investigación Itaca-Sabien de la Universitat Politècnica de València. El grupo cuenta con amplia experiencia previa obtenida en el desarrollo de proyectos software similares durante sus 20 años de vida.

Entre ellos se encuentra MyCyFAPP (6), que posee gran cantidad de puntos en común. Parte de la experiencia obtenida durante todo el tiempo desarrollando dichas aplicaciones se incorpora en este TFG.

Ilustración 2. Logotipo de MyCyFAPP.

Se trata de un proyecto europeo que se centra en promover y mantener comportamientos nutricionales adecuados promoviendo un papel activo del paciente. Este posee un conjunto de herramientas digitales creadas para los pacientes y sus cuidadores que apuntan a aumentar su conocimiento de la enfermedad ofreciéndoles herramientas para controlar la nutrición ellos mismos.

El proyecto se compone de dos aplicaciones, una móvil y otra web, y dos herramientas educacionales para niños. Estas proporcionan una serie de herramientas para que personas con fibrosis quística puedan seguir hábitos nutricionales saludables y, por otro lado, sus cuidadores conozcan mejor su enfermedad.

La aplicación web, destinada a los profesionales, permite seguir el avance de los pacientes. Por otro lado, la aplicación móvil de autogestión facilita a los pacientes la introducción de síntomas, consulta de registros de los platos o síntomas introducidos, programación de recordatorios, visualización de contenido educacional sobre la enfermedad y la realización de recetas para que la herramienta calcule la cantidad de enzimas pancreáticas que deben tomar.

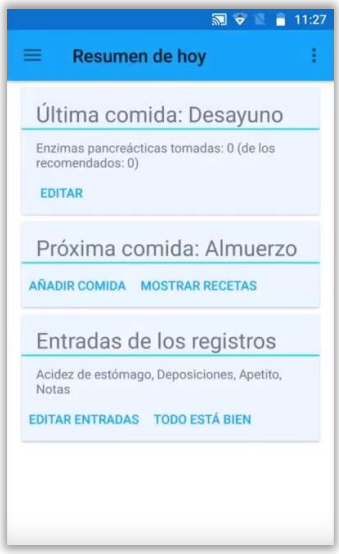

Ilustración 3. Ventana principal de MyCyFAPP.

En cuanto al apartado del diseño, esta utiliza las líneas de diseño *Material Design* de Google, obteniendo así una mayor organización, visibilidad y accesibilidad durante todo el uso de la app. Al tratarse de un proyecto

europeo, permite la traducción a distintos idiomas pudiendo ampliar el número de pacientes que pueden utilizar la aplicación.

El proyecto NuMielo, del cual forma parte este TFG, ya identifica por parte de los profesionales el vacío de una aplicación que ayude con el estudio, tratamiento y mejora de calidad de vida de pacientes con MM.

A modo de aumentar las competencias del autor sobre investigación, se realiza una búsqueda para encontrar aplicaciones o sistemas con funcionalidades similares.

En primer lugar, se realizan búsquedas sobre aplicaciones para el tratamiento de pacientes con MM. Al no encontrar ningún resultado sobre aplicaciones en Google Scholar[3], salvo artículos relacionados con la enfermedad, se efectúa una búsqueda sobre aplicaciones relacionadas con el tratamiento de enfermedades en la Play Store[4] de Google.

Mediante palabras clave como "seguimiento de pacientes", "apps de seguimiento" o "seguimiento de nutrición" se han encontrado las siguientes aplicaciones:

- Cita Sanitaria Madrid[5] o GVA +Salut[6] son aplicaciones orientadas a la gestión de consultas con los hospitales, no ofrecen funcionalidades similares a las requeridas en este TFG.
- Quirónsalud[7] es una aplicación dirigida al seguimiento del histórico personal médico. Esta también incluye la gestión de consultas que hemos mencionado en el punto anterior.
- MiSAOS[8] es una app enfocada al seguimiento del tratamiento de alguna enfermedad o trastorno concreto. Destinada a personas con un tratamiento mediante CPAP, esta es exclusiva para pacientes de OXIGEN salud, una empresa creada para suministrar oxigeno medicinal comprimido a domicilio.

En resumen, existe un vacío en lo que se refiere a un sistema para mejorar y seguir el tratamiento de personas con MM, sobre todo, en lo que a aplicaciones publicadas se refiere. Así, se propone una solución software que consiste en una aplicación Android para el seguimiento y recolección de datos sobre la enfermedad.

[3] https://scholar.google.com/
[4] https://play.google.com
[5] https://play.google.com/store/apps/details?id=org.madrid.citasanitaria
[6] https://play.google.com/store/apps/details?id=es.gva.mesSalut
[7] https://play.google.com/store/apps/details?id=com.divisait.quironsalud
[8] https://play.google.com/store/apps/details?id=myosa.www.oxigensalud.com.myosa

3. Análisis del problema.

En todo desarrollo software se encuentra una etapa llamada especificación de requisitos software. Esta etapa consiste en la descripción completa de los objetivos y comportamiento del sistema, proporcionando así una mayor comunicación entre los clientes, usuarios, analistas y diseñadores. También ayuda a reducir malinterpretaciones entre todas las entidades involucradas en el desarrollo de la aplicación.

En primer lugar, se debe distinguir entre requisitos funcionales y no funcionales.

3.1. Requisitos funcionales.

En este apartado vamos a incorporar los requisitos funcionales, de acuerdo con su definición, estos son características que describen la funcionalidad que se espera que el sistema proveerá.

1. La aplicación permitirá iniciar sesión al usuario para poder hacer uso de esta.
2. Se permitirá acceder a las distintas funcionalidades desde la misma ventana principal.
3. El usuario podrá realizar cuestionarios para evaluar su calidad de vida o su estado de ánimo.
4. Se podrán introducir valores para el peso.
5. La aplicación proporcionará una gráfica donde se mostrará la evolución del peso del usuario.
6. El usuario podrá reportar diversos síntomas.
7. La aplicación solicitará rellenar un cuestionario inicial tras utilizar la aplicación por primera vez.
8. Se permitirá el cambio de los datos personales del usuario tras su primer uso de la aplicación.
9. El sistema proporcionará un método para recuperar la contraseña en caso de ser olvidada.
10. Se permitirá el cambio de contraseña para iniciar sesión.
11. La aplicación mostrará notificaciones y alertas para recordar al usuario los datos que debe introducir.
12. El sistema permitirá observar el progreso del usuario durante el tratamiento.

13. El usuario podrá acceder al contenido educacional proporcionado por los profesionales sanitarios.

3.2. Requisitos no funcionales.

A continuación, vamos a indicar los requisitos no funcionales. Estos no se refieren a las funciones específicas del sistema, sino a las propiedades o atributos que debe cumplir la aplicación tales como la fiabilidad, eficiencia, integridad, etc. Mediante estos requisitos se definen restricciones del sistema.

1. La aplicación debe mostrar constantemente retroalimentación al usuario, informando de errores o confirmando acciones realizadas.
2. La aplicación debe ser fácilmente traducible a cualquier idioma.
3. Los tiempos de respuesta no deben ser superiores a 3 segundos.
4. La aplicación no debe bloquearse en ningún momento durante su uso.
5. La estructura del código debe estar estructurada, facilitando el posterior mantenimiento y mejora del software.
6. Se deben seguir las líneas de diseño establecidas por Google *Material Design* para conseguir un diseño más intuitivo y fácil de utilizar.
7. La aplicación debe dar soporte desde la versión 16 de la API Android y superiores, consiguiendo compatibilidad con prácticamente todos los dispositivos del mercado.

3.3. Casos de uso.

Una vez identificados los requisitos de la aplicación pasaremos a especificar el comportamiento deseado del sistema mediante los casos de uso. Estos son descripciones de las acciones del sistema que especifican la comunicación y el comportamiento del sistema mediante la interacción con los usuarios o sistemas, denominados actores.

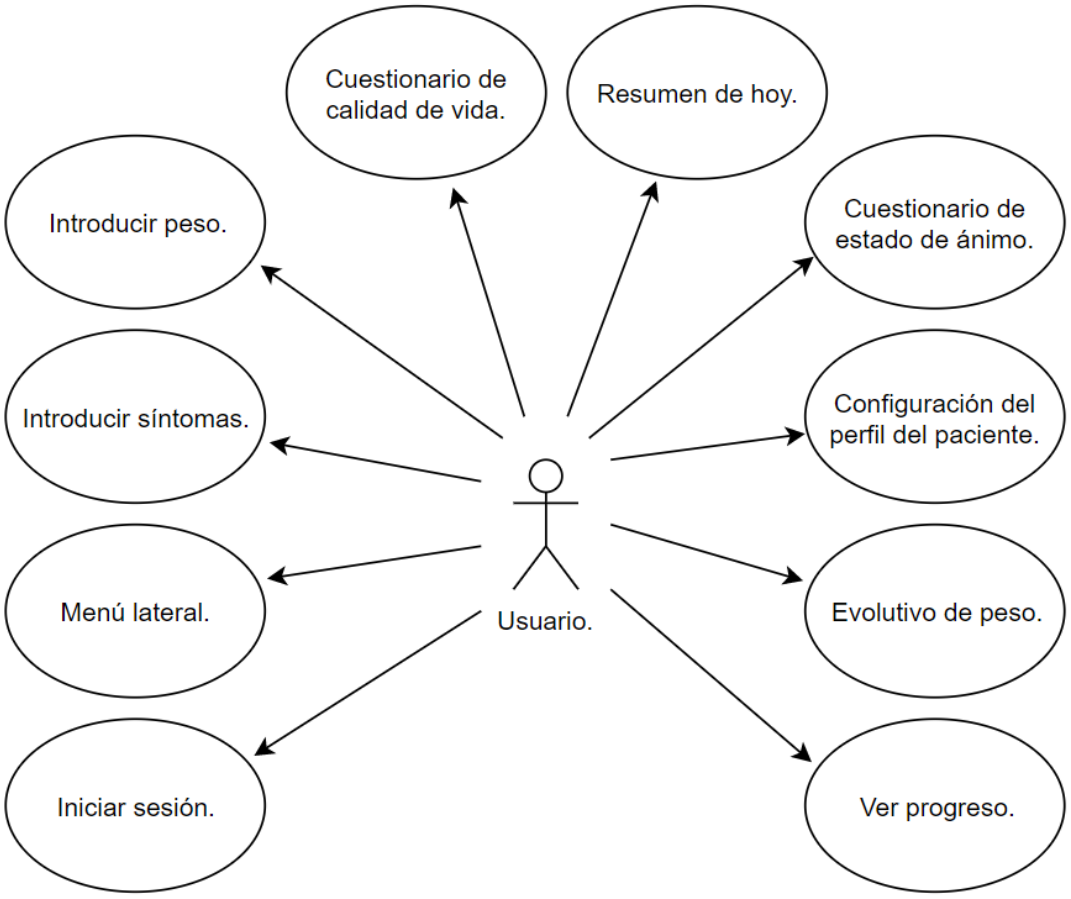

Ilustración 4. Diagrama de casos de uso.

Como podemos observar en la ilustración 4, en el sistema existe únicamente un rol de usuario al cual se le permite acceder a cualquier funcionalidad individualmente. Estas son independientes entre sí a excepción de “iniciar sesión” que es un requisito previo para el acceso al resto de funcionalidades, es decir, el usuario debe de iniciar sesión para poder utilizar el resto de la app.

A continuación, se van a describir más detalladamente los distintos casos de uso siguiendo su funcionalidad.

Tabla 1. Caso de uso: inicio de sesión.

Caso de uso.	**1- Inicio de sesión.**
Descripción.	Permite acceder a la aplicación y hacer uso de todas las funcionalidades.
Actor.	Usuario.

Precondición.	• No estar autenticado anteriormente • El usuario tiene que estar registrado desde la aplicación web.
Flujo básico.	• Introducir correo electrónico y contraseña. • Pulsar el botón “Aceptar”.
Postcondición.	Se accede a la ventana “Resumen de hoy”. Se guardan las credenciales para no volver a introducirlas durante el uso de la app.

Tabla 2. Caso de uso: menú lateral.

Caso de uso.	**2- Menú lateral.**
Descripción.	Se muestra un botón que accede a un menú donde se permite navegar a las demás funcionalidades. Este estará disponible en la ventana principal “resumen de hoy”. Siguiendo las políticas de buenas prácticas de *Material Design*, mientras se navegue por las distintas ventanas, el botón cambiará a una flecha para retroceder.
Actor.	Usuario.
Precondición.	Iniciar sesión.
Flujo básico.	• Pulsar el botón, situado en la esquina superior derecha. • Seleccionar un ítem de la lista para navegar a otra ventana, estas son: resumen, peso, evolución del peso, síntomas, cuestionarios, cuestionario de perfil, progreso, contenido educacional y configuración. • Si ya se ha accedido a una ventana, pulsar la flecha para retroceder.
Postcondición.	Muestra la ventana del ítem seleccionado.

Tabla 3. Caso de uso: configuración de perfil del paciente.

Caso de uso.	**3- Configuración de perfil del paciente.**

Descripción.	Cuestionario inicial donde se introduce el nombre, sexo y una serie de preguntas para el profesional sanitario. Este se debe de realizar obligatoriamente si es la primera vez que se usa la aplicación. Se permite modificar posteriormente las respuestas.
Actor.	Usuario.
Precondición.	Iniciar sesión.
Flujo básico.	• Tras iniciar sesión, se abre la ventana automáticamente y se introducen los datos necesarios para avanzar. • Pulsar el botón atrás para retroceder y modificar las respuestas. • Una vez se ha terminado el cuestionario, pulsar el ítem en el menú principal si se desea modificar las respuestas introducidas.
Postcondición.	Se abre la ventana “resumen de hoy” y se envían las respuestas al servidor.

Tabla 4. Caso de uso: resumen de hoy.

Caso de uso.	**4- Resumen de hoy.**
Descripción.	Es la ventana principal de la aplicación. En esta se visualizan las tareas pendientes a realizar, como la introducción de peso, síntomas o cuestionarios. Para el cálculo de las tareas pendientes se utilizan los días como unidad, de modo que el peso y el cuestionario de estado de ánimo se recuerdan cada 14 días, el cuestionario de calidad de vida cada 56 días y los síntomas diariamente.
Actor.	Usuario.
Precondición.	Iniciar sesión y, si es la primera vez utilizando la app, rellenar el cuestionario de configuración de perfil.
Flujo básico.	• Pulsar una de las tareas a realizar.

Postcondición.	Se abre la ventana correspondiente a la selección de la tarea. Si la tarea esta completada, se elimina la opción de la ventana.

Tabla 5. Caso de uso: cuestionario de estado de ánimo.

Caso de uso.	**5- Cuestionario de estado de ánimo.**
Descripción.	Muestra al usuario una serie de preguntas conforme a una escala estandarizada visualizando cada pregunta por pantalla con botones para volver atrás o avanzar. En concreto se usa la Escala de Ansiedad y Depresión Hospitalaria (HAD), es el instrumento de autoevaluación más usado para detectar malestar emocional (ansiedad y depresión) en poblaciones con enfermedad física (7). En el Anexo A podemos observar las preguntas utilizadas para este cuestionario.
Actor.	Usuario.
Precondición.	Iniciar sesión y tener pendiente realizar el cuestionario.
Flujo básico.	• Pulsar siguiente en la ventana de dialogo dónde se muestra al usuario las instrucciones sobre el cuestionario. • Pinchar en la tarea en la ventana *resumen de hoy* o en el menú lateral. • Seleccionar una respuesta, y pulsar el botón siguiente. • Pulsar el botón atrás para rectificar.
Postcondición.	Se abre la ventana “resumen de hoy” y se elimina la tarjeta de “cuestionario de estado de ánimo”. Se envían las respuestas al servidor.

Tabla 6. Caso de uso: cuestionario de calidad de vida.

Caso de uso.	**6- Cuestionario de calidad de vida.**

Descripción.	Muestra al usuario una serie de preguntas recopiladas en el cuestionario de salud EQ-5D-5L, es un instrumento genérico integrado por las cinco dimensiones consideradas más relevantes de la calidad de vida relacionada con la salud: movilidad, autocuidado, actividades habituales, dolor/malestar y ansiedad/depresión (8). En el Anexo B se detallan las preguntas que se utilizan en el cuestionario. Se visualiza cada pregunta por pantalla con botones para volver atrás o avanzar.
Actor.	Usuario.
Precondición.	Iniciar sesión y tener pendiente realizar el cuestionario.
Flujo básico.	• Pinchar en la tarea en la ventana “resumen de hoy” o en el menú lateral. • Seleccionar una respuesta, y pulsar el botón siguiente. • Pulsar el botón atrás para rectificar.
Postcondición.	Se abre la ventana “resumen de hoy” y se elimina la tarjeta de “cuestionario de calidad de vida”. Se envían las respuestas al servidor.

Tabla 7. Caso de uso: introducción de peso.

Caso de uso.	**7- Introducción de peso.**
Descripción.	El usuario se pesará en su báscula habitual e introducirá el valor.
Actor.	Usuario.
Precondición.	Iniciar sesión y tener pendiente introducir el peso.
Flujo básico.	• Pinchar en la tarea en la ventana “resumen de hoy” o en el menú lateral. • Introducir el valor del peso. • Pulsar el botón aceptar para guardar el valor introducido.

Postcondición.	Se abre la ventana “resumen de hoy” y se elimina la tarjeta de “introducción de peso”. Se envía el valor al servidor.

Tabla 8. Caso de uso: evolución del peso.

Caso de uso.	**8- Evolución del peso.**
Descripción.	Muestra una gráfica con todo el histórico de datos de peso, si existen. Permitiendo filtrar estos datos por 3 fechas: todo, 1 año o 6 meses de antigüedad.
Actor.	Usuario.
Precondición.	Iniciar sesión.
Flujo básico.	• Seleccionar el ítem en el menú lateral. • Elegir un filtro de fechas: todo, 1 año o 6 meses.
Postcondición.	Mostrar al usuario todos los datos según el filtro aplicado.

Tabla 9. Caso de uso: progreso del paciente.

Caso de uso.	**9- Progreso del paciente.**
Descripción.	Visualiza una ventana donde elegir entre calidad de vida, estado emocional o peso. Una vez seleccionada una de las opciones, se muestra el estado del usuario en esta opción y ofrece consejos pulsando el botón de ver consejos.
Actor.	Usuario.
Precondición.	Iniciar sesión.
Flujo básico.	• Pinchar en el ítem en el menú lateral. • Seleccionar una opción entre calidad de vida, estado emocional o peso.

	• Pulsar el botón ver consejos para visualizar los consejos según el resultado del progreso.
Postcondición.	Mostrar el resultado del progreso del usuario y visualizar consejos para mejorar estos resultados.

Tabla 10. Caso de uso: introducción de síntomas.

Caso de uso.	**10- Introducción de síntomas.**
Descripción.	Permite introducir síntomas, organizados según el tipo. Primeramente, se muestran todos los tipos, tras seleccionar uno se muestra el síntoma específico y finalmente, se deberá introducir un valor si el síntoma lo requiere. Los síntomas que se introducen han sido especificados por los profesionales y se han descrito en el Anexo C.
Actor.	Usuario.
Precondición.	Iniciar sesión.
Flujo básico.	• Pinchar en la tarea en la ventana “resumen de hoy” o en el menú lateral. • Seleccionar un tipo de síntoma. • Seleccionar el síntoma específico. • Introducir el valor si el síntoma lo requiere. • Pulsar el botón “aceptar” para finalizar.
Postcondición.	Se abre la ventana “resumen de hoy” y se elimina la tarjeta de “síntoma”. Se envía el resultado al servidor.

3.4. Mockups

En el desarrollo software es muy habitual el uso de mockups[9] o prototipado durante la fase de diseño, creando así las interfaces del sistema software sin la necesidad de implementar su funcionalidad. Estas interfaces pueden varias desde diseños simples a papel hasta unos diseños más realistas mediante programas destinados a ello.

Gracias al uso de los mockups, los diseñadores consiguen *feedback*[10] por parte de los *stakeholders*[11] reduciendo así los tiempos en las fases de implementación.

A continuación, se explican los prototipos de interfaz de la aplicación, comentando el flujo que existe entre ellos.

Ilustración 5. Mockup de la ventana de inicio de sesión.

Ventana de inicio de sesión.

En la ilustración 5 vemos la ventana básica que permite al usuario autenticarse en la aplicación y poder acceder al resto de funcionalidades del sistema. También facilita realizar el cambio de contraseña o solicitar una nueva si se ha olvidado. Esta es la primera ventana a la que se accede una vez se abre la aplicación por primera vez.

[9] Mockup: concepto de interfaz gráfica del sistema.
[10] Feedback: (del inglés) retroalimentación.
[11] Stakeholders: cualquier persona interesada en el proyecto.

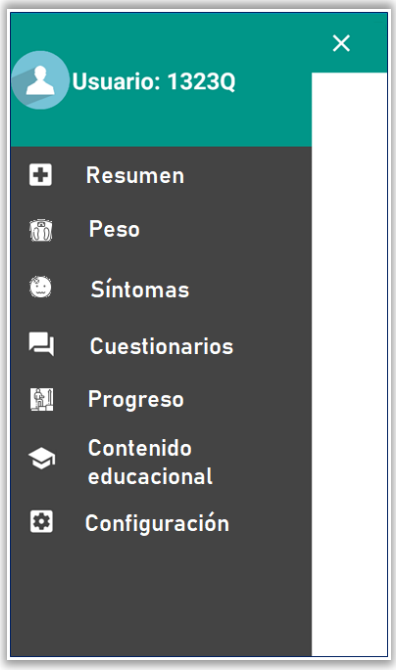

Ilustración 6. Mockup del menú lateral.

Menú lateral de navegación.

El menú lateral, que podemos observar en la ilustración 6, facilita al usuario un acceso rápido a cualquier funcionalidad de la aplicación, estando presente durante todo el uso de esta. Se puede hacer uso de este una vez el usuario es autenticado en la app.

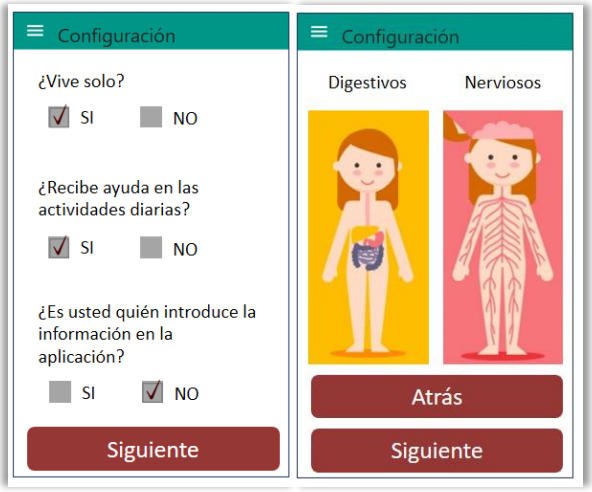

Ilustración 7. Mockup de la ventana de Configuración de perfil.

Ventana de configuración de perfil del paciente.

Una vez realizado el inicio de sesión y sea la primera vez que se accede a la aplicación, debe de rellenarse el cuestionario inicial de configuración de perfil que se muestra en la ilustración 7.

En esta ventana se presentan una serie de preguntas con respuesta afirmativa o negativa, que componen la primera parte, y una serie de imágenes seleccionables donde se indica el síntoma que tiene actualmente, que forma parte de la segunda parte del cuestionario. Una vez se ha rellenado completamente, la aplicación muestra la ventana *resumen de hoy*.

Ilustración 8. Mockup de la ventana de resumen de hoy.

Ventana de resumen de hoy.

Tras haber rellenado el cuestionario de perfil por primera vez, se muestra esta ventana donde se presentan las tres distintas tarjetas que recuerdan al usuario las tareas pendientes de realización. Esta pasa a ser la ventana principal de la aplicación, cada vez que el usuario termine una tarea o vuelva a entrar a la app, volverá a aparecer esta vista. Como se observa en la ilustración 8, las tarjetas se componen de un título, una descripción que indica que debe hacer el usuario y una imagen descriptiva relacionada con la tarea que ayuda a diferenciarlas.

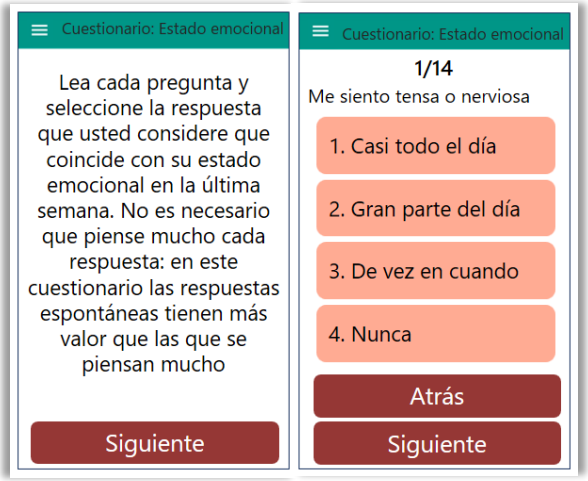

Ilustración 9. Mockup de la ventana de cuestionario de estado de ánimo.

Ventana de cuestionario de estado de ánimo.

A esta ventana se accede desde el menú lateral o desde la ventana *resumen de hoy* una vez se ha seleccionado la tarjeta correspondiente. Nada más acceder a ella, se muestra una ventana modal que indica al usuario cómo debe proceder con el cuestionario. Una vez pulsa siguiente se muestra la pregunta con cuatro opciones, el número de pregunta en el que se encuentra y los botones de navegación, tal y como se muestra en la ilustración 9. Una vez completado el cuestionario se vuelve a la ventana *resumen de hoy*.

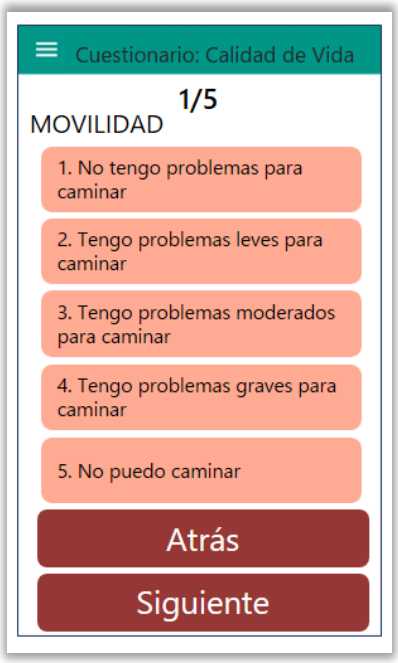

Ilustración 10. Mockup de la ventana de cuestionario de calidad de vida.

Ventana de cuestionario de calidad de vida.

En la ilustración 10 se muestra el cuestionario de calidad de vida, pudiendo acceder a este desde el menú lateral o seleccionando la tarjeta correspondiente en la ventana *resumen de hoy*. En esta se exponen las preguntas con sus cinco respuestas de selección única, la numeración de la pregunta y los botones que permiten la navegación entre ellas. Cuando se completa el cuestionario se muestra la ventana resumen de hoy.

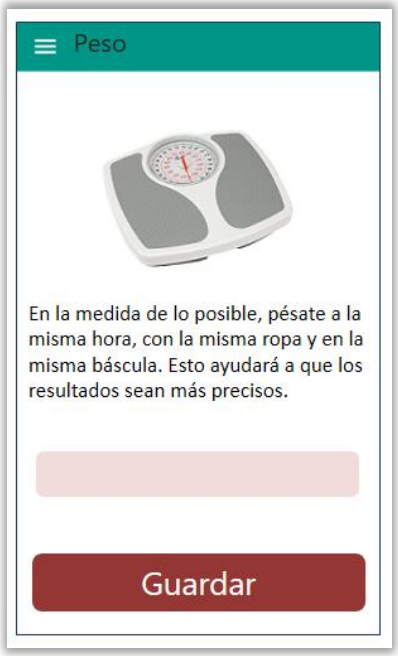

Ilustración 11. Mockup de la ventana de introducción de peso.

Ventana de introducción de peso.

Para acceder a esta ventana el usuario debe seleccionar la opción en el menú lateral o la tarjeta correspondiente en la ventana *resumen de hoy*. Tras acceder a esta, el usuario debe de leer el mensaje informativo antes de introducir el valor y finalmente pulsar el botón aceptar, tal y como se muestra en la ilustración 11. Una vez introducido el valor correctamente se regresa a la ventana *resumen de hoy*.

Ilustración 12. Mockup de la ventana de evolución del peso.

Ventana de evolución del peso.

Seleccionando la opción desde el menú lateral accedemos a la ventana de evolución del peso, que podemos observar en la ilustración 12. En esta ventana se muestra una gráfica de líneas donde se contemplan los datos de los pesos introducidos. También se incluye un pequeño menú en la parte superior donde se selecciona el filtro a realizar a los datos ya sea de 1 año, 6 meses o todos.

Ilustración 13. Mockup de la ventana de progreso del paciente.

Ventana de progreso del paciente.

A esta ventana se accede desde su opción en el menú lateral. Como se puede ver en la ilustración 13, está compuesta de dos ventanas diferentes. La primera muestra tres tarjetas seleccionables que nos permiten acceder a la siguiente ventana según el peso, estado emocional o calidad de vida. En la última se observa el estado actual de la selección anterior, mediante el uso de una cara que varía de color, una pequeña descripción del progreso y un botón que mostrará al usuario diferentes consejos para mejorar el estado actual.

Ilustración 14. Mockup de la ventana de introducción de síntomas.

Ventana de introducción de síntomas.

Una vez seleccionada la tarjeta en la ventana *resumen de hoy* o la opción en el menú lateral, accedemos a la ventana que observamos en la ilustración 14. Esta ventana se compone de otras dos pequeñas ventanas. La primera muestra una lista de los tipos de síntomas que se pueden reportar. Una vez seleccionado uno nos lleva a la siguiente ventana donde aparece otra lista con los síntomas específicos del tipo de síntoma seleccionado. Finalmente, tras seleccionar el síntoma aparece una nueva ventana en la que, dependiendo del síntoma, se deben de introducir datos adicionales, como la temperatura o una pequeña descripción. Tras reportar el síntoma completamente se regresa a la ventana *resumen de hoy*.

3.5. Solución propuesta.

La solución que se propone para alcanzar el objetivo del proyecto NuMielo está dividida en dos subsistemas:

- El subsistema móvil para pacientes: consiste en una aplicación móvil Android orientada a los pacientes que permite introducir los datos necesarios para hacer el seguimiento durante su tratamiento y proporcionar consejos a medida que se avanza en este.
- El subsistema web para profesionales, administración y recogida de datos: una aplicación web destinada a los médicos que facilita la gestión y el seguimiento de los pacientes pudiendo crear, modificar o eliminar los perfiles de usuario, recibir alertas, ver el progreso y los síntomas

reportados. Para la recogida de datos se hace uso de una API REST con la que se comunica la aplicación móvil.

El autor del TFG desarrolla el subsistema de la aplicación móvil, pero se ve afectado por las decisiones del subsistema web ya que envía los datos a la API desarrollada por este. Dado que el sistema web necesita recopilar datos de la aplicación móvil, se necesita una coordinación entre ambos grupos para negociar tanto un modelo de datos como las tecnologías a utilizar.

Del desarrollo del subsistema web y la API REST se encarga la empresa Bengesdat[12]. En la aplicación web se visualizan los datos recopilados desde la aplicación móvil haciendo uso de la API desarrollada por esta empresa.

Cabe destacar que el proyecto NuMielo es una iniciativa que parte del Hospital Universitari i Politècnic La Fe, este promueve la cooperación entre la UPV y el hospital facilitando futuros proyectos en conjunto.

3.6. Plan de trabajo.

Al inicio del TFG se decide realizar el siguiente plan de trabajo, dividido en cuatro fases:

- Documentación sobre las herramientas y tecnologías a utilizar.
- Diseño de la arquitectura del sistema.
- Implementación de la aplicación.
- Validación y verificación del sistema.

Para la primera fase de documentación se establece una duración aproximada de cuatro semanas debido al estudio y aprendizaje de las nuevas tecnologías, teniendo que realizar pequeños proyectos de prueba de las funcionalidades de estas herramientas.

En lo que respecta a la segunda fase de diseño de la arquitectura se estiman dos semanas de trabajo. Por otro lado, para la implementación y desarrollo de la aplicación se eleva el tiempo necesario a seis semanas. Para terminar, la etapa de validación y verificación del sistema es de dos semanas asumiendo cambios a realizar por fallos tanto del sistema como de interfaz.

[12] http://www.bengesdat.es/index.php

Tras definir el plan de trabajo inicial y, como hemos comentado en el apartado de la solución propuesta, sabiendo que se coordina este trabajo con Bengesdat, la empresa encargada del subsistema web y administración, se han tenido una serie de contratiempos que han provocado un retraso en la entrega del TFG.

Durante la implementación del proyecto, Bengesdat solicita un cambio de tecnología en el acceso al servidor de datos, generando un aumento del tiempo necesario tanto para la documentación de la nueva tecnología como para la implementación del proyecto. La fase de diseño no se ve realmente afectada, ya que como el autor continúa con el desarrollo de la app, reutiliza el modelo de datos ya establecido realizando pequeñas modificaciones para adaptarlo al nuevo diseño que necesita la tecnología.

El progreso y el contenido educacional son dos funcionalidades que finalmente no se han incorporado porque, en el momento de la escritura de la memoria, los profesionales sanitarios están generando el contenido necesario para llevar a cabo su implementación.

Finalmente, el tiempo inicial del plan de trabajo se ve aumentado dos semanas y la fase de validación no se ve alterada manteniendo su duración inicial de dos semanas.

4. Diseño de la solución.

4.1. Arquitectura.

4.1.1. Componentes del sistema.

A continuación, se diseña la arquitectura de la primera fase del TFG. En la ilustración 15 se presentan los componentes principales de la aplicación móvil.

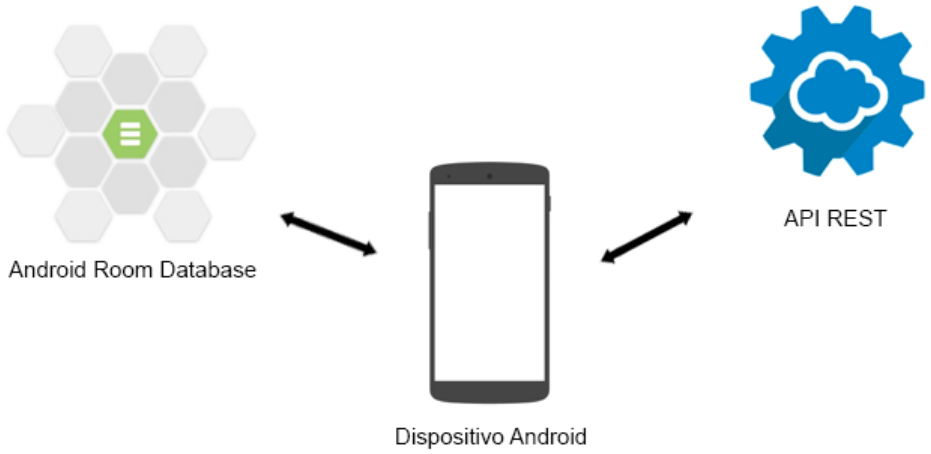

Ilustración 15. Esquema de componentes del sistema con API REST.

La app está compuesta por el propio dispositivo móvil, ejerciendo el rol de cliente, que se comunica con la API REST donde también se sitúa la aplicación web y una base de datos *Android Room* local, permitiendo trabajar cuando no exista conexión a internet o el servidor no responda.

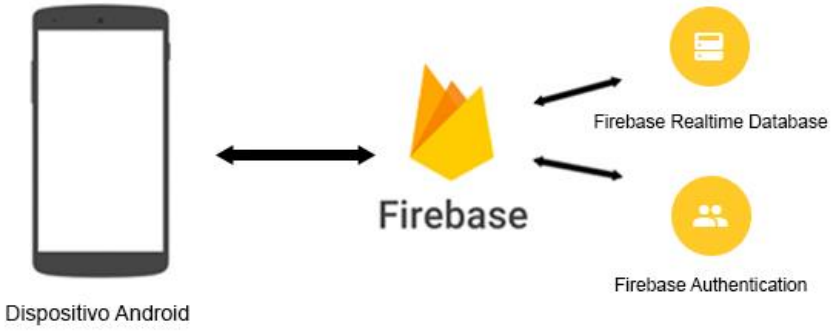

Ilustración 16. Esquema de componentes del sistema con Firebase.

Respecto a la segunda fase del TFG, se sustituye la base de datos local *Android Room* y la API REST por una base de datos *Firebase*. Como observamos en la ilustración 16, el dispositivo móvil sigue ejerciendo el rol de cliente, pero esta vez se comunica con la nube de *Firebase* para enviar los datos recogidos. Se hace uso de

Firebase Realtime Database como base de datos y *Firebase Authentication* como librería para la gestión de las credenciales de los usuarios.

4.1.2. Arquitectura de la aplicación móvil.

Se ha elegido seguir el modelo de arquitectura de tres capas para este proyecto. Esta arquitectura está formada por un conjunto ordenado de subsistemas en el que cada nivel está construido en relación a componentes de la capa o nivel inferior, y proporciona la base para las capas superiores.

En la ilustración 17 se muestran las tres capas que conforman la arquitectura por capas distinguiéndose las siguientes:

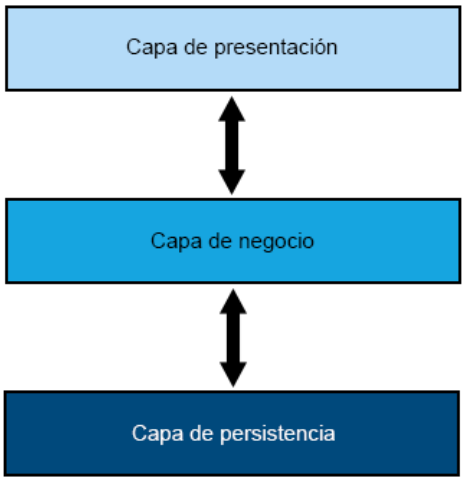

Ilustración 17. Esquema de arquitectura de tres capas.

- La capa de presentación es la encargada de mostrar los resultados de computación al usuario y recoger los datos necesarios para realizar dichos cómputos. Se comunica con el usuario mediante las interfaces de la aplicación.
- La capa de negocio, también llamada capa de lógica, proporciona la funcionalidad de la aplicación. Es decir, es la encargada de realizar las llamadas a la capa de persistencia mediante los datos proporcionados por la capa de presentación, haciendo de intermediaria entre ambas capas.
- La capa de persistencia o datos es donde se almacenan los datos necesarios para el correcto funcionamiento de la aplicación. Proporciona métodos de obtención y almacenamiento de datos para la capa de negocio.

Esta arquitectura se utiliza generalmente en el desarrollo de sistemas software porque aporta una serie de ventajas como el aislamiento de la lógica de la app en diferentes componentes separados lo que posibilita el desarrollo y la creación de test en paralelo. Además, proporciona una distribución en diferentes máquinas o procesos incrementando tanto la dedicación de recursos para cada capa como la reutilización, pero resulta más caro de implementar y ejecutar. Asimismo, aporta flexibilidad a la hora de añadir, modificar o eliminar algún componente en cualquier capa. De esta forma, no es necesaria la modificación de las otras capas ya que son independientes.

4.1.3. Arquitectura MVVM.

Se ha optado por seguir el patrón arquitectónico MVVM debido a que encaja perfectamente con las necesidades del desarrollo de una aplicación que sigue una estructura de cliente-servidor. Además, los desarrolladores de Google han desarrollado un conjunto de librerías para facilitar el proceso de implementación de aplicaciones con esa estructura. Hablaremos de estas librerías en el apartado de tecnología utilizada.

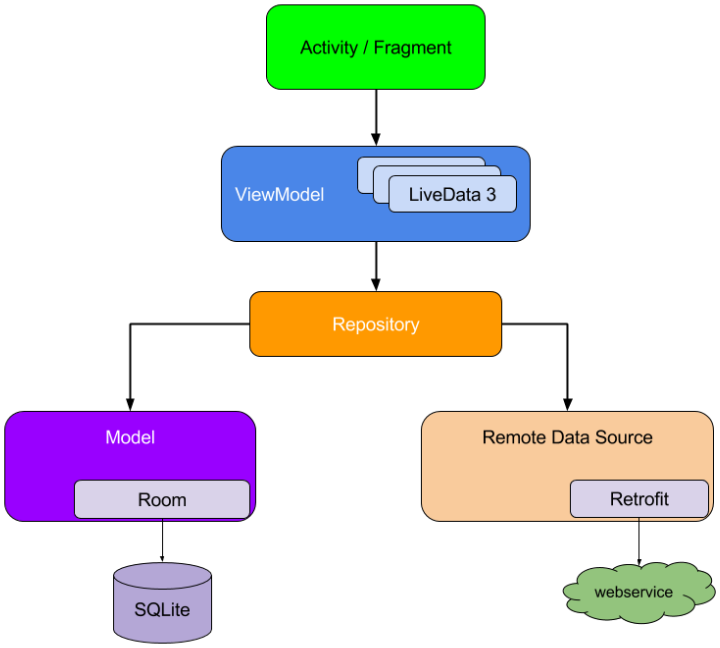

Ilustración 18. Esquema de arquitectura MVVM.

Como muestra la ilustración 18, cada componente depende sólo del que se encuentra un nivel más abajo. Los *fragments* y *activities* se sitúan en la capa de presentación, nombrada en el apartado de arquitectura del sistema, el *view model* se encuentra en la capa de lógica y en la capa de persistencia se sitúa la base de datos *Room*. Entre la capa de lógica y persistencia se encuentra un componente llamado *repository* que añade otra capa de abstracción entre las distintas fuentes de datos y el resto de la aplicación.

Este diseño ayuda a crear una experiencia de usuario más consistente y agradable. De este modo, si el usuario sale de la aplicación y vuelve a entrar inmediatamente o pasados unos días, puede observar los datos tal y como se encontraban, ya que estos persisten de manera local. Si los datos están inactivos, el *repository* actualiza los datos en segundo plano.

Como se ha comentado en el apartado de plan de trabajo, debido a una serie de problemas internos por parte de Bengesdat, se produce un retraso en la coordinación de los equipos obligando a cambiar la arquitectura y la tecnología. Este cambio afecta a la base de datos y la API REST, la cual es sustituida por una base de datos *Firebase,* una plataforma para el desarrollo de aplicaciones móviles y web propietaria de Google.

El cambio de tecnologías ha resultado más sencillo que en un diseño monolítico gracias a la modularidad proporcionada por el modelo de arquitectura por capas. Así, solamente se ha visto afectada la capa de persistencia de datos, sin verse afectadas por estos cambios el resto de capas del modelo.

4.2. Diseño detallado.

4.2.1. Estructura del proyecto Android.

Como hemos comentado anteriormente en la arquitectura, se diseña una primera base de datos *Room* y una API REST mientras que en la segunda fase del TFG se sustituye tanto *Room* como la API por una base de datos *Firebase*. A continuación, se van a detallar las dos estructuras diseñadas para el desarrollo de este TFG.

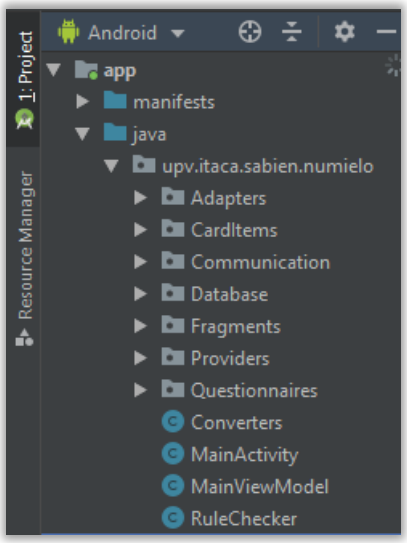

Ilustración 19. Estructura del proyecto con API REST.

La primera estructura utilizada en el desarrollo de la app es la que se muestra en la ilustración 19. Observamos los distintos paquetes en los que se divide el proyecto.

El paquete *Database* contiene las clases que representan la capa de persistencia, donde se encuentra la clase *Repository* y las entidades de la base de datos *Room*. El paquete *Communication* contiene las clases con las llamadas a la API REST.

Finalmente, el resto de paquetes: *Adapters*, *CardItems*, *Fragments*, *Providers* o *Questionnaires* contienen las clases necesarias para el correcto funcionamiento de la app, como los fragmentos que utilizan las actividades Android, estos los veremos de manera más detallada en el apartado desarrollo de la solución propuesta.

Para la segunda fase del TFG se realiza el cambio de arquitectura y se migra la base de datos a *Firebase*, la estructura del proyecto queda tal y como se ve en la ilustración 20.

La principal diferencia respecto a la estructura anterior al cambio es la eliminación del paquete *Communication*, ya que no necesitamos las llamadas a la API REST.

Otro cambio se realiza en el paquete *Database*, donde las clases que contenían las entidades para la base de datos *Room* han sido eliminadas debido a que ya no van a ser utilizadas.

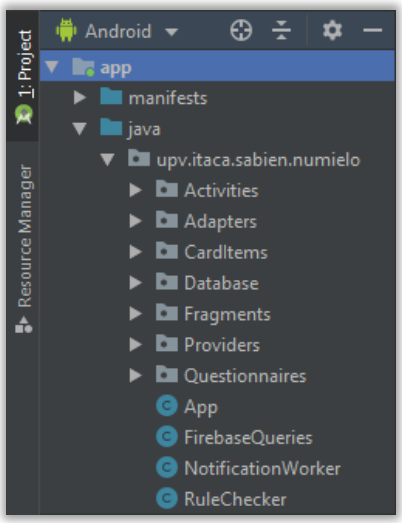

Ilustración 20. Estructura del proyecto con Firebase Database.

A parte de todos los cambios mencionados anteriormente cabe destacar las siguientes modificaciones. Se ha añadido la clase *FirebaseQueries* donde se encuentran los métodos necesarios para realizar las consultas a la base de datos *Firebase*. Al no disponer de la base de datos *Room*, se ha eliminado la clase *Converters* encargada de realizar transformaciones sobre las fechas e introducirlas en la base de datos. Finalmente, se han añadido tanto la clase *App* como *NotificationWorker* las cuales añaden funcionalidades del sistema necesarias para su correcto funcionamiento.

4.2.2. Modelo de datos.

Con el objetivo de visualizar la estructura de la base de datos del sistema se hace uso de un modelo de datos que muestra una abstracción de la misma. Su función es describir los elementos reales que intervienen con el problema y como se relacionan.

La estructura que sigue nuestra aplicación contiene seis entidades que forman las tablas de la base de datos. Estas tablas tienen en común los siguientes atributos:

- id: código identificador de la entidad.
- id_usuario: es la clave ajena que relaciona las tablas con la entidad Usuario.
- fecha_realizacion: almacena la fecha en la que se realizó la inserción del dato en la base de datos.
- sincronizado: indica si el dato está sincronizado con el servidor o no.

- fecha_sinc_exito: almacena la fecha del momento en el que se sincronizó el dato con el servidor.

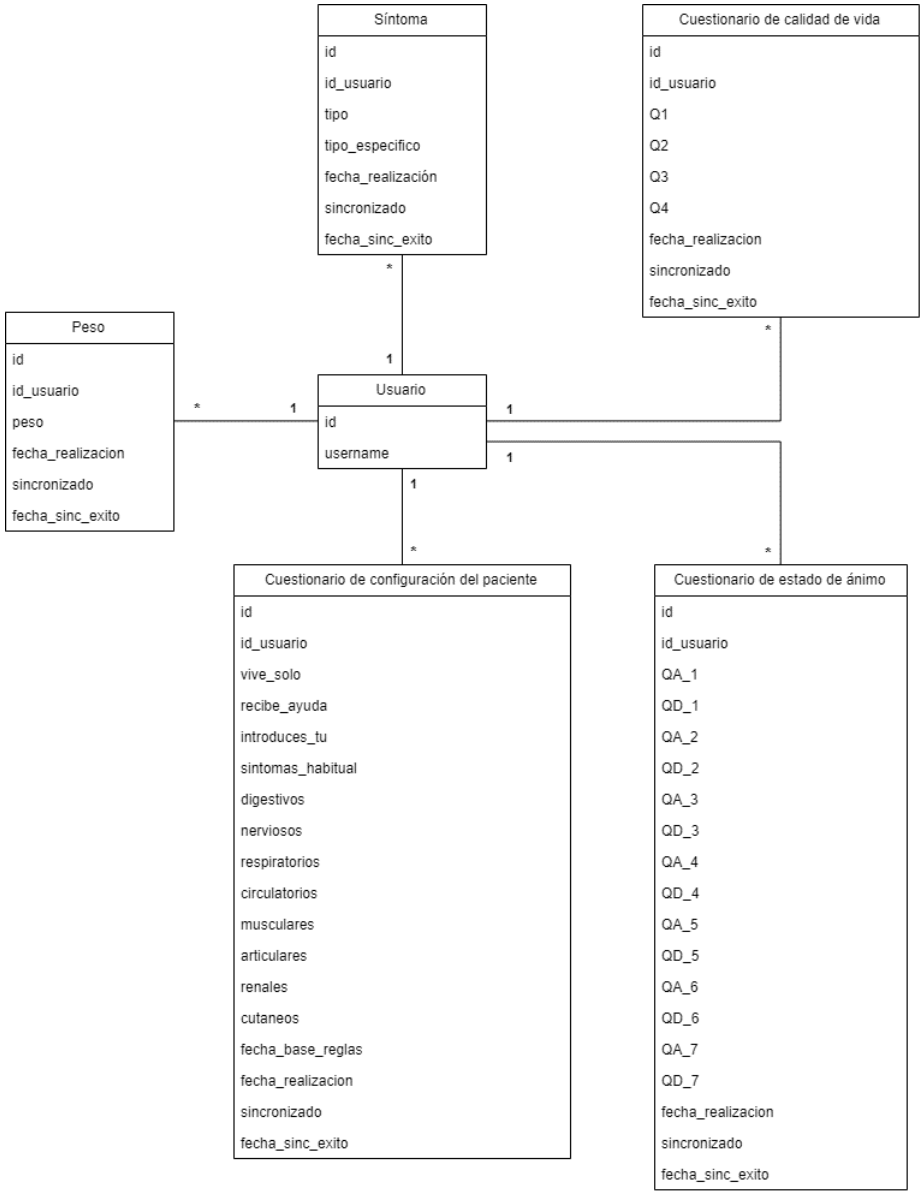

Ilustración 21. Modelo de datos.

Como vemos en la ilustración 21, las relaciones son uno a muchos ya que únicamente existe un usuario que introduce muchos pesos, síntomas o rellena cuestionarios.

El resto de atributos de las tablas sólo almacenan el dato a sincronizar, por ejemplo, el peso, los síntomas introducidos o las respuestas de las preguntas de los cuestionarios.

Cabe destacar que tanto el cuestionario de calidad de vida como el de estado de ánimo tienen una codificación para las preguntas y respuestas. En el cuestionario de

calidad de vida, es simplemente el numero de la pregunta, pero en el cuestionario de estado de ánimo se emplea una codificación diferente. Esta codificación sigue las preguntas del Anexo A. Cada pregunta tiene un código como, por ejemplo, A.1. que se codifica en la base de datos como QA_1.

Esta base de datos nos permite almacenar de manera local todos los datos introducidos por el usuario, pudiendo trabajar con estos si es necesario, hasta que se disponga de conexión a internet y se estos se envíen al servidor para poder ser observados por los profesionales.

Tal y como hemos comentado en el apartado de arquitectura, el desarrollo de la app consta de una segunda fase donde se sustituye esta base de datos *Room* por una base de datos *Firebase Realtime Database*.

La base de datos *Firebase* utiliza una estructura JSON para almacenar los datos. Por esta razón, es necesaria una transformación del modelo utilizado en *Room* como vamos a comentar a continuación.

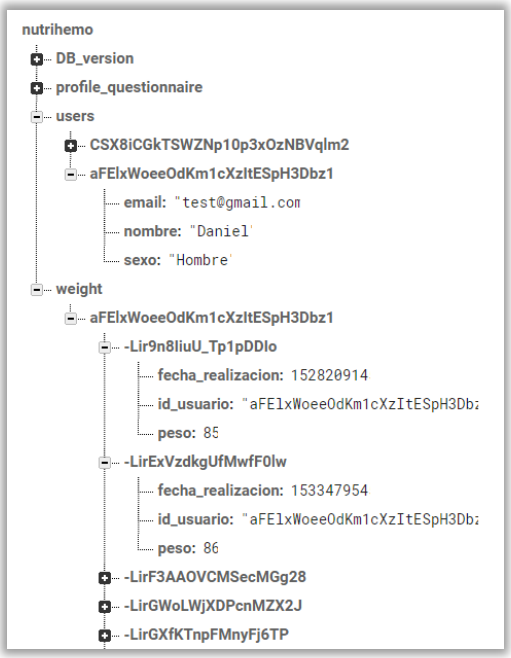

Ilustración 22. Base de datos Firebase.

La base de datos *Firebase* se estructura mediante pares clave-valor. Las tablas utilizadas en el modelo relacional se corresponden con las claves del primer nivel de la nueva base de datos. Por ejemplo, como vemos en la ilustración 22, las claves "profile_questionnaire", "users" o "weight" corresponden con las tablas "Cuestionario de configuración del paciente", "Usuario" o "Peso". Como se ha indicado anteriormente, para cada una de estas claves almacenaremos un valor. El valor asignado serán los datos a guardar. Con el objeto de agrupar los datos de cada usuario dentro de estas claves, los datos a almacenar están formados por un par clave-valor donde la clave es el identificador de usuario obtenido por *Firebase Authenticator* al crear las credenciales y el valor son los datos introducidos.

Tras este código identificador del usuario para cada nuevo dato almacenado *Firebase* genera una clave única similar a como funcionan las claves primarias autonuméricas. A modo de ejemplo podemos observar la clave "weight" de la ilustración 22. En el primer nivel se almacena el código del usuario, seguidamente el código

autogenerado del dato insertado y, finalmente, los pares clave-valor de los atributos de la tabla relacional de “Peso”.

Cabe destacar la ausencia del atributo “sincronizado” y “fecha_sinc_exito”. Estos datos no son necesarios ya que *Firebase* realiza la sincronización automáticamente. Una vez se habilita la persistencia *offline* de los datos, Firebase Realtime Database puede almacenar en el dispositivo hasta 10MB de datos por defecto, incluso si el dispositivo se reinicia. Este tamaño es suficiente para la mayoría de casos, permitiendo el uso de los datos no sincronizados.

4.3. Herramientas y tecnología utilizada.

En este apartado se describen las herramientas y tecnologías utilizadas para el desarrollo completo del proyecto, desde la especificación hasta la implementación de este.

4.3.1. Entorno de desarrollo.

Ilustración 23. Icono de Android Studio.

Android Studio[13] es un entorno desarrollado por *Google* y *JetBrains*, construido sobre *IntelliJ IDEA* y diseñado específicamente para el desarrollo de aplicaciones Android. Es habitual hacer uso de este entorno cuando se está desarrollando una aplicación para el sistema Android, debido a su facilidad de uso, el gran soporte y documentación por parte de *Google*. Además, ofrece una gran cantidad de herramientas ya integradas en el propio IDE, entre ellas:

- Editor de diseño: permite compilar diseños fácil y rápidamente, tanto de manera visual, arrastrando los elementos de la interfaz, o escribiendo directamente sobre el XML.
- Emulador: mediante el AVD manager, podemos crear dispositivos virtuales en cualquier versión de Android para realizar pruebas directamente desde el ordenador.
- Perfiles en tiempo real: estos perfiles muestran datos en tiempo real relacionados con la CPU, la memoria y la actividad de red de la app, ayudando a depurar con mayor facilidad.

[13] https://developer.android.com/studio

- Control de versiones: permite realizar interacciones con los distintos tipos de gestión de versiones, como *Git*, *Mercurial* o *Subversion* entre otros.

4.3.2. Lenguaje de programación.

Ilustración 24. Icono de Java.

Java es un lenguaje de programación orientado a objetos. Es el lenguaje más utilizado gracias a su facilidad de uso y aprendizaje. Otra de las razones por la cual se escoge este lenguaje es por ser independiente de la plataforma, es decir, permite compilar el software en una máquina y ejecutarlo en cualquier otra.

Google dispone una gran cantidad de documentación para desarrollar aplicaciones Android con *Java*, esto acostumbra a ser un factor importante a la hora de elegir el lenguaje con el que se va a trabajar.

4.3.3. Herramientas.

Ilustración 25. Icono de Bitbucket.

Para el control de versiones se ha utilizado *Bitbucket*[14] de *Atlassian*. Es un servicio de alojamiento web para código fuente y proyectos de desarrollo que usa tanto *Mercurial* como *Git* para el control de versiones, en nuestro caso hemos usado *Git*. Permite crear repositorios privados de manera ilimitada y gratuita para equipos de menos de 5 integrantes. También integra *CI/CD* y ofrece una configuración sencilla para desarrollar y probar el proyecto de manera automática.

Ilustración 26. Icono de Trello.

Trello [15] es una herramienta que facilita el trabajo colaborativo de una manera flexible y sencilla. Mediante la creación de tableros se puede seguir la metodología que se desee, por ejemplo, la metodología ágil utilizada en este proyecto. Una de las ventajas que tiene es que te permite acceder a la herramienta desde cualquier navegador web y dispone de app móvil, añadiendo así mayor portabilidad. Además, incluye un gran número de complementos para añadir al tablero y ofrecer más funcionalidades.

[14] https://bitbucket.org/product/
[15] https://trello.com/

Ilustración 27. Icono de Balsamiq.

Se ha utilizado *Balsamiq*[16] para el diseño de las interfaces. Esta herramienta permite crear diseños para cualquier tipo de producto software desde páginas web hasta dispositivos Android o iOS. Sigue los principios de simplicidad y facilidad de uso, esto lo consigue permitiendo realizar los prototipos con tan solo arrastrar el componente que se desee. También facilita la exportación a distintos tipos de ficheros, como PNG o PDF.

Ilustración 28. Icono de Firebase.

Firebase [17] es una plataforma de desarrollo para aplicaciones móviles y web desarrollada por *Firebase Inc*. y adquirida por *Google* en 2014. Su finalidad es ayudar a crear aplicaciones rápidamente y sin gestionar la infraestructura dando funcionalidades como el análisis de las apps, bases de datos, mensajes o reportes de fallos. Al estar construida sobre la infraestructura de *Google* se tiene tanto soporte como documentación directamente de la misma empresa.

En este proyecto se usa la base de datos *Firebase Realtime Database*, esta sincroniza automáticamente cualquier cambio en la base de datos con el servidor de *Firebase*. También posee la capacidad de almacenar datos localmente si la sincronización no se ha completado.

También se ha utilizado *Firebase Authenticator* para la gestión del inicio de sesión de la aplicación. Este paquete ofrece las operaciones CRUD necesarias para crear o modificar los usuarios de la aplicación directamente desde la consola de *Firebase*.

16 https://balsamiq.com/
17 https://firebase.google.com/

4.3.4. Librerías.

Ilustración 29. Icono de Gson.

Gson[18], también conocida como *Google Gson*, es una librería open-source para serializar objetos *Java* a *JSON* y deserializarlos. Ofrece métodos para realizar la conversión de manera práctica y sencilla.

Ilustración 30. Icono de Android Jetpack.

Android Jetpack [19] es una serie de colecciones de componentes de software Android que permiten acelerar el desarrollo de manera que elimina el código estándar, administrando automáticamente actividades como el ciclo de vida de los componentes o las tareas en segundo plano. Estas librerías fueron diseñadas para funcionar en conjunto, pero es posible utilizarlas de manera individual con versiones anteriores. También ofrece una mayor frecuencia de actualización de forma que siempre se tiene acceso a las nuevas y mejores versiones de los componentes.

MPAndroidChart [20] es una librería gratuita creada por Philipp Jahoda para realizar gráficas para Android, también ha desarrollado *Charts* como una versión para *iOS*. Su finalidad es ayudar a crear gráficas potentes pero fáciles de utilizar, permite realizar gráficos de líneas, barras, tartas, burbujas, combinaciones de estos, etc.

Finalmente, ***Android Room*** [21] es un componente incluido en la librería *Android Jetpack* que proporciona una capa de abstracción sobre *SQLite*, construyendo un acceso a la base de datos más robusta. Comparte el mismo objetivo que las librerías de *Jetpack*, conseguir crear bases de datos de una forma más sencilla, permitiendo la persistencia de los datos en la aplicación incluso sin conexión a internet.

---

18 https://github.com/google/gson
19 https://developer.android.com/jetpack
20 https://github.com/PhilJay/MPAndroidChart
21 https://developer.android.com/topic/libraries/architecture/room

5. Desarrollo de la solución propuesta.

En este apartado se va a comentar el desarrollo de las funcionalidades de la aplicación y como se ha llegado a implementar de este modo. Como se han desarrollado dos arquitecturas diferentes, se van a explicar por separado para diferenciar claramente entre ellas.

5.1. Desarrollo con la API REST.

Una decisión muy importante a la hora de crear un proyecto Android es la selección del nivel de API Android. En nuestro caso necesitamos cubrir la totalidad de los dispositivos del mercado, ya que la aplicación está dirigida a personas de la tercera edad. Por consiguiente, debemos de elegir un nivel suficientemente bajo, pero cubriendo también a las versiones más actuales del SO. Se establece el nivel mínimo de API 16 (versión de Android 4.1) cubriendo el 99% de los dispositivos del mercado.

En lo que respecta a la arquitectura con API REST, se necesita almacenar los datos localmente en el dispositivo para posteriormente, enviarlos al servidor. Para ello, hacemos uso de la librería *Room*.

Primeramente, se crean las entidades necesarias para el funcionamiento de la base de datos, estas representan las tablas de SQLite. Para dar las propiedades a la tabla se usa una notación específica encima de la línea de código que lo necesite.

```
@Entity(tableName = "peso_table",
        foreignKeys = @ForeignKey(entity = Usuario.class,
                parentColumns = "id",
                childColumns = "id_usuario",
                onDelete = CASCADE))
public class Peso {

    @PrimaryKey(autoGenerate = true)
    private int id;

    private int id_usuario;

    private double peso;

    private Date fecha_realizacion;
```

Ilustración 31. Clase Peso.

Por ejemplo, a la clase Peso se le atribuye la etiqueta *@Entity* junto con el nombre de la tabla SQLite y la clave ajena, mediante *@ForeignKey*. Para indicar la clave primaria en el atributo "*id*" se establece la etiqueta *@PrimaryKey*, como observamos en la ilustración 30.

Una vez terminadas las entidades, continuamos creando una interfaz DAO, en esta se crean los métodos que usará *Room* para acceder a los datos. Del mismo modo que las entidades, los métodos tienen etiquetas asociadas que *Room* interpreta en las diferentes consultas SQLite. En la ilustración 31 vemos un ejemplo de los métodos de la clase Peso etiquetados con @Insert, @Update, @Delete o @Query, estos tienen como argumento el objeto que posteriormente insertará en la base de datos.

```
@androidx.room.Dao
public interface NumieloDao {

    @Insert
    void insert(Peso peso);

    @Update
    void update(Peso peso);

    @Delete
    void delete(Peso peso);

    @Query("DELETE FROM peso_table")
    void deleteAllPeso();

    @Query("SELECT * FROM peso_table ORDER BY id ASC")
    LiveData<List<Peso>> getAllPeso();
```

Ilustración 32. Clase NumieloDao.

```
10  @androidx.room.Database(entities = {CuestionarioCalidad.class, CuestionarioEstado.class,
11                                      CuestionarioPerfil.class, Peso.class,
12                                      Sintoma.class, Usuario.class}, version = 1)
13  @TypeConverters({Converters.class})
14  public abstract class NumieloDB extends RoomDatabase {
15
16      private static NumieloDB instance; //Singleton pattern
17
18      public abstract NumieloDao numieloDao();
19
20      public static synchronized NumieloDB getInstance(Context context) {
```

Ilustración 33. Clase NumieloDB.

Para terminar la base de datos, se crea una última clase llamada “NumieloDB” donde mediante la etiqueta *@Database* y extendiendo de *RoomDatabase*, *Room* se encarga de crear la base de datos junto a las entidades creadas anteriormente. Se aplica el patrón Singleton consiguiendo que sólo exista una única instancia de la base de datos, como puede verse en la línea de código 16 de la ilustración 32.

Finalmente, se aplica el patrón repositorio en la clase denominada “NumieloRepository” proporcionando otra capa de abstracción entre las distintas fuentes de datos y el resto de la aplicación. Esta oculta las diferentes operaciones a la base de datos como las consultas SQLite y proporciona una API al ViewModel.

5.2. Desarrollo con Firebase.

La empresa Bengesdat, encargada de realizar la API REST, notificó un cambio en la arquitectura eliminando dicha API y sustituyendola por los servicios ofrecidos por *Firebase*. Como hemos hablado en el apartado de arquitectura, se elimina la base de datos Room y con ella, el paquete *Communication*, donde se encontraban las llamadas a la API.

De ahora en adelante, cada vez que se requiere una consulta a la base de datos, se realiza mediante una llamada a un método de la clase “FirebaseQueries”. En la ilustración 33, se observa un ejemplo de como enviar un objeto al servidor de *Firebase*. Si no hay conexión a internet, este permanece localmente hasta poder realizar la sincronización de forma que no hay pérdidas de datos ni inconsistencias en el sistema.

```
private void savePeso(Peso peso){
        DatabaseReference mDatabase = FirebaseDatabase.getInstance().getReference();
        String userid = FirebaseAuth.getInstance().getCurrentUser().getUid();
        mDatabase.child("weight").child(userid).push().setValue(peso);
}
```

Ilustración 34. Método para guardar peso de la clase FirebaseQueries.

Los datos se almacenan en el servidor de *Firebase* siguiendo una estructura JSON de clave-valor, como hemos comentado en el apartado del diseño detallado. A modo de ejemplo, seguiremos la última línea de código de la ilustración 33. Una vez obtenida la referencia a la base de datos en la variable “mDatabase”, accedemos a la primera clave de esta, mediante “child(“weight”)”. Seguidamente, con “child(userid)” accedemos al valor de la siguiente clave, el codigo de usuario único. Finalmente, utilizando “push().setValue(peso)” se añade el valor del dato a la lista de pesos.

Como hemos comentado en los objetivos, un aspecto importante es el soporte de varios idiomas. Para ello se hace uso de los recursos de Android creando un fichero XML por idioma, donde se incluyen todos los *strings* que utiliza la aplicación para mostrar texto. Estos ficheros se agrupan en la carpeta de strings del proyecto, tal y como se muestra en la ilustración 34.

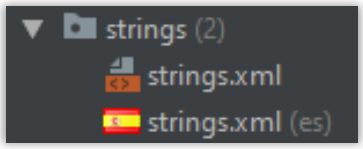

Ilustración 35. Carpeta de idiomas en los recursos de Android.

Se crean los ficheros siguiendo una nomenclatura que posteriormente interpreta Android. Por ejemplo, cuando el fichero se crea en español, se le asigna la etiqueta “es” y cuando se trata del inglés se utiliza “en”. De esta forma, Android selecciona el fichero XML según el idioma del dispotivo.

Para hacer uso de los ficheros es necesario referenciar los strings en el código. Para ello, se sustituye el texto plano de la variable o etiqueta de texto por una referencia al string del fichero. Por ejemplo, se le asigna el valor “R.strings.menu_resumen” a la variable que muestra el texto del ítem de resumen del menú lateral. De este modo,

cuando Android selecciona el fichero adecuado, el string a mostrar cambia dinámicamente.

```
<resources>
    <string name="app_name">Numielo</string>
    <string name="title_activity_tipo_sintoma">Tipo de síntoma</string>
    <string name="title_activity_sintoma">Síntomas</string>
    <string name="title_activity_sintoma_otro">Otro tipo de sintomas</string>
    <string name="menu_resumen">Resumen</string>
    <string name="menu_peso">Peso</string>
    <string name="menu_evolucion_peso">Evolucion del peso</string>
    <string name="menu_contenido">Contenido educacional</string>
    <string name="menu_configuracion">Configuración</string>
    <string name="email">Email</string>
    <string name="cuestionario_calidad">Cuestionario de calidad de vida</string>
    <string name="cuestionario_estado">Cuestionario de estado de ánimo</string>
    <string name="pista_et_contraseña">Contraseña</string>
    <string name="resumen_peso_text">Pulse aqui para introducir el peso.</string>
```

Ilustración 36. Fichero strings.xml

5.3. Ventanas y funcionalidades.

La capa de presentación de la app esta compuesta por una actividad o *activity,* Google la define como un componente de una aplicación que contiene una pantalla con la que los usuarios interactúan. Junto con esta actividad principal se usan fragmentos o *fragments*, definidos como comportamientos o partes de una interfaz de usuario dentro de la actividad. Mediante el uso de actividades y fragmentos se consiguen diseños de interfaces más dinámicos y flexibles.

Cabe destacar el uso de una paleta de colores que favorece tanto el estilo del proyecto como la visibilidad de los elementos, tratándose de un punto muy importante para conseguir accesibilidad y una mayor legibilidad del texto.

A continuación, se comentan las ventanas de la aplicación junto con las funcionalidades que engloban.

Ilustración 37. Pantalla de inicio de sesión.

Pantalla de inicio de sesión.

La pantalla mostrada en la ilustración 36 contiene la actividad y el fragmento, indicado en el rectángulo rojo. Esta engloba el caso de uso de inicio de sesión.

Nada más instalar la aplicación, si el usuario está dado de alta por el personal sanitario, podrá acceder desde esta ventana con el correo electrónico y la contraseña proporcionada.

Debajo del botón “iniciar sesión” se encuentra el botón que muestra una pequeña ventana de diálogo, que permite al usuario, mediante la introducción del correo electrónico, realizar un cambio de contraseña en caso de olvido. Ambos botones tienen una diferencia de tamaño y forma, destacando la opción principal al usuario.

Gracias al uso de *Firebase Authenticator*, una vez se ha iniciado sesión correctamente, este guarda las credenciales de modo que el usuario no tiene que volver a introducirlas hasta que cierre la sesión o desinstale la app. Por lo que, el usuario obtiene una experiencia más agradable ya que no debe introducir las credenciales constantemente.

Pantalla de configuración del perfil del paciente.

Esta pantalla se muestra si no existe ningún cuestionario de perfil en la base de datos o se selecciona manualmente en el menú lateral. No se permite el acceso a ninguna otra funcionalidad del sistema si este cuestionario se encuentra vacío. Cuando alguna de las preguntas no tiene respuesta, se muestra un mensaje al usuario indicando que no puede avanzar sin contestar todas las preguntas. Si el usuario no termina el cuestionario, la aplicación guarda las respuestas tanto en la base de datos como en las *SharedPreferences* del dispositivo, este método de almacenamiento utiliza un XML que queda registrado en el espacio compartido de la aplicación, de modo que los datos persisten mientras la app siga instalada.

Al pulsar el botón siguiente se inicia una transacción para navegar al siguiente fragmento y añadirlo a una pila de retroceso de transacciones. Así, tras pulsar tanto el botón atrás de la aplicación como el botón físico del dispositivo es posible retroceder al fragmento anterior. El flujo de estos fragmentos se puede ver en la ilustración 37, desplazándose de izquierda a derecha.

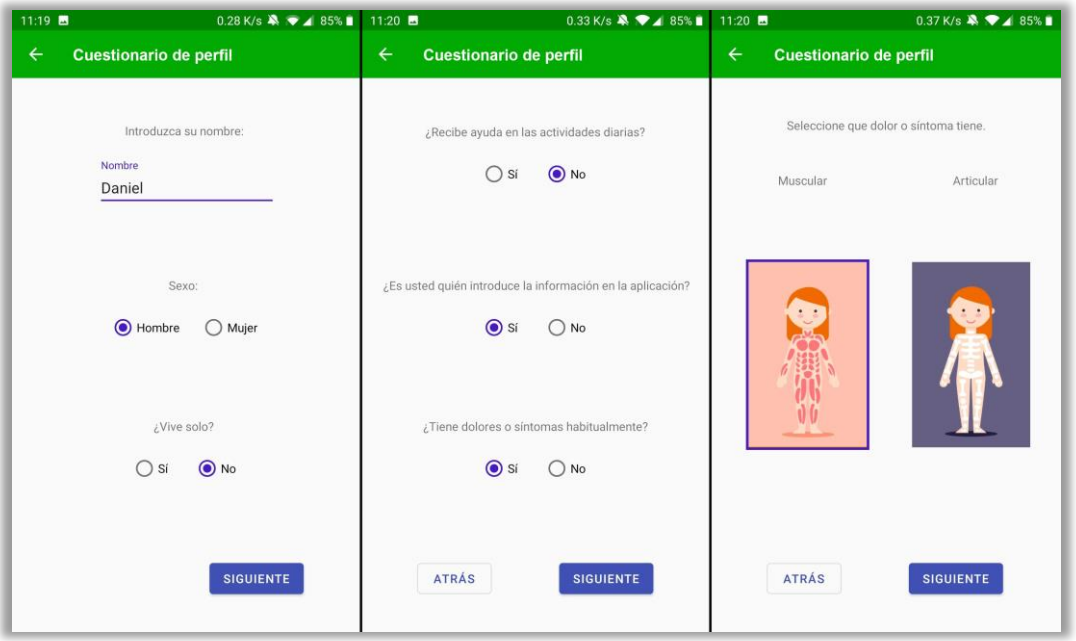

Ilustración 38. Pantallas de configuración de perfil del paciente.

Ilustración 39. Pantalla de resumen de hoy.

Pantalla de resumen de hoy.

Esta es la pantalla principal de la aplicación, consta de un fragmento que contiene una lista de tarjetas o *cards* [22] . Estas tarjetas aparecen o desaparecen cuando se han completado las tareas asociadas. Por ejemplo, si el peso no se ha introducido en 14 días, la tarjeta "peso" se muestra en la lista como una tarea pendiente de realizar.

Cuando se pulsa sobre una tarjeta, se inicia la transacción del fragmento asociado a la tarea. Una vez finalizada, se elimina la tarjeta de la lista y se vuelve a esta ventana. Por lo tanto, el usuario sabe en todo momento que tareas restan por completar.

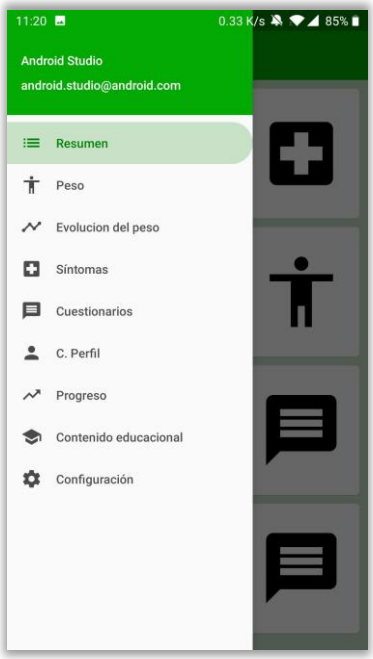

Ilustración 40. Pantalla del menú lateral.

Menú lateral.

Inicialmente el menú principal sería accesible desde cualquier parte de la aplicación, permitiendo el acceso a todas las pantallas desde cualquier otra. En cambio, sólo es accesible desde la pantalla de resumen de hoy siguiendo las líneas de diseño de Google. El botón para acceder al menú principal, que podemos observar en la esquina superior izquierda de la ilustración 38, cambia la forma a una flecha para poder regresar a la pantalla anterior. Así, proporciona una mayor accesibilidad y navegación por la app.

En la ilustración 39 vemos el diseño final del menú lateral donde se destaca con un color de fondo la selección actual, de modo que el usuario sabe dónde se encuentra. En la parte superior muestra el nombre y el correo electrónico del usuario.

[22] https://material.io/design/components/cards.html

Ilustración 41. Pantalla de introducción de peso.

Pantalla introducción de peso.

En esta ventana, el único cambio a resaltar es la posición del botón, finalmente situado en la misma línea que el campo de entrada de datos. De esta forma, si la pantalla del usuario es más pequeña, cuando se ajuste al tamaño el botón siempre permanece visible. Para facilitar la introducción de los datos, una vez se accede a la ventana, se abre el teclado numérico y se mantiene el foco en el campo de introducción de texto.

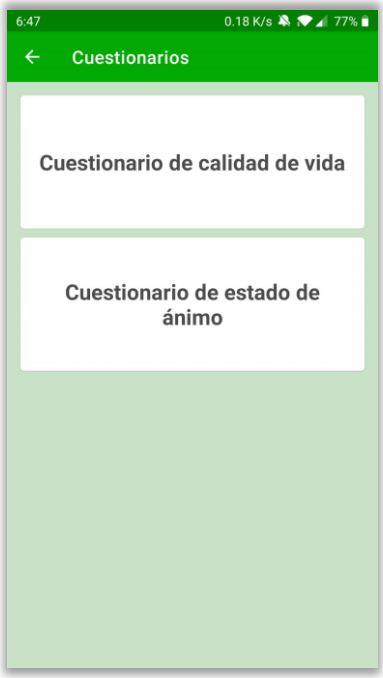

Ilustración 42. Pantalla de selección de cuestionarios.

Pantalla de selección de cuestionarios.

Una vez se pulsa la opción “Cuestionarios” en el menú lateral se accede a la ventana que vemos en la ilustración 41. Su función es mostrar dos tarjetas fijas que permiten acceder a los dos tipos de cuestionarios. Cuando el usuario no tiene la tarea pendiente de introducción de algún cuestionario, pero sí necesita volver a rellenarlo, puede acceder desde el menú y elegir manualmente el que desee introducir.

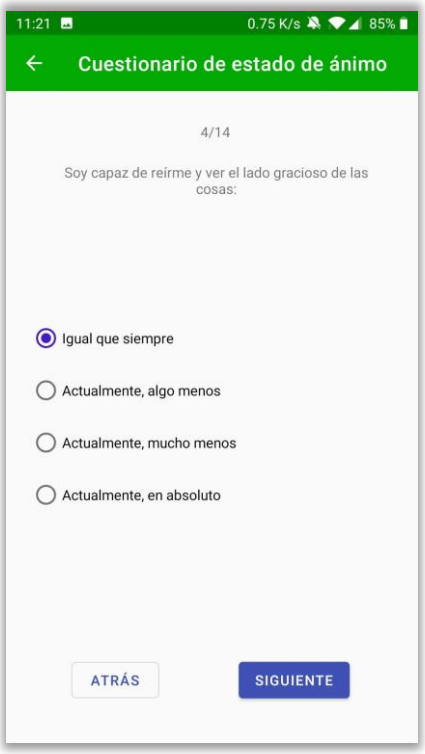

Ilustración 43. Pantalla de cuestionario de estado de ánimo.

Pantalla de cuestionario de estado de ánimo y calidad de vida.

La separación que vemos en la ilustración 42 entre los textos de la pregunta, las respuestas y los botones de navegación es amplia debido a la necesidad de poder escalar los elementos al tamaño de la pantalla. Además, si el usuario establece un tamaño de letra o pantalla más elevado, no se solapan los elementos entre sí, aumentando la visibilidad.

Como se ha visto en el mockup, se necesita mostrar un consejo en la pantalla antes de realizar el cuestionario de estado de ánimo. Se ha optado por usar una ventana de diálogo modal, como el que vemos en la ilustración 43, que no permite seguir adelante sin pulsar el botón aceptar. Por consiguiente, se crea una mayor atención al aviso por parte del usuario.

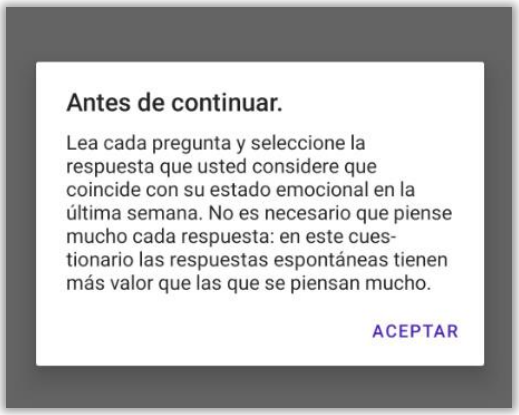

Ilustración 44. Ventana de diálogo del cuestionario de estado de ánimo.

Ilustración 45. Pantalla de selección de síntoma.

Pantalla de síntomas.

Se ha decidido mantener un código de colores en la selección de tipos de síntomas para aportar una mayor distinción entre ellos. Los colores se seleccionan siguiendo la guía de legibilidad de texto de *Material Design*, evitando que alguna tarjeta no sea identificable ni afecte negativamente a la vista del usuario, como se observa en la ilustración 44.

Esta pantalla consta de una lista de tarjetas organizada por un código de colores. Cuando se selecciona una tarjeta de un tipo de síntoma se accede al siguiente fragmento donde se muestra otra lista de tarjetas. Esta última lista muestra los síntomas específicos del síntoma seleccionado, en este caso, a modo de ayuda visual, se mantiene el mismo color que el síntoma seleccionado anteriormente. Por ejemplo, si seleccionamos la tarjeta “digestivo”, se muestra una lista de síntomas digestivos con todas las tarjetas amarillas.

Además, en la barra superior se indica que síntoma se ha seleccionado para así, evitar que el usuario se descuide o despiste y sepa en todo momento en que punto está de la tarea.

Hay ciertos síntomas que requieren introducir datos extra como la temperatura o comentarios. Por ejemplo, al seleccionar otro tipo de síntoma se necesita una descripción del síntoma antes de continuar con el reporte.

Ilustración 46. Pantalla de introducción de otro tipo de síntomas.

Pantalla de evolución de peso.

Para poder visualizar correctamente la gráfica del historial de pesos, se fuerza la rotación de la pantalla en horizontal. Haciendo uso de la librería MPAndroidChart se crea la gráfica que vemos en la ilustración 46. En el eje Y se muestra el rango de pesos en intervalos de diez y en el eje X las fechas en función del filtro seleccionado, ya sea seis meses, un año o todo el registro de datos.

Los datos son recogidos del servidor de Firebase, de modo que si no se dispone de conexión a internet se notifica al usuario indicando que es necesaria dicha conexión para funcionar correctamente. Para obtener el peso exacto de una fecha, basta con pulsar sobre el punto en la gráfica. De este modo, se muestra una etiqueta con la fecha y el peso registrado como vemos indicado en la ilustración 46.

Ilustración 47. Pantalla de evolución del peso.

6. Verificación.

A continuación, se describen las pruebas realizadas para comprobar el correcto funcionamiento de la app. Estas son una serie de pruebas funcionales realizadas manualmente y siguiendo un guion. Así, verificamos de manera exhaustiva el comportamiento de las funcionalidades de la aplicación, es decir, comprobamos si la aplicación hace lo que debe y, sobre todo, sigue la especificación previa.

Las pruebas diseñadas por el autor se transfieren a cuatro miembros del grupo de investigación, que deberán seguir una serie de pautas y reportar cualquier problema detectado. El guion seguido se confecciona según el caso de uso y para cada uno se realiza la secuencia de pruebas a realizar.

Por ejemplo, para el caso de uso “Inicio de sesión” se define el estado inicial en el que se encuentra el usuario de prueba, los pasos a realizar y el estado final que se debe obtener, como vemos en la tabla 11:

Tabla 11. Secuencia 1.1 de pruebas funcionales.

Nombre de prueba.	**1.1 Inicio de sesión incorrecto.**
Caso de uso.	Inicio de sesión.
Estado inicial.	Pantalla principal del dispositivo.
Secuencia.	1. Abrir aplicación. 2. Introducir correo electrónico erróneo. 3. Introducir contraseña errónea. 4. Pulsar el botón “iniciar sesión”.
Estado final.	Mensaje de error indicando que las credenciales son incorrectas.
Observaciones.	Al tratarse del mismo estado final, se engloban las combinaciones de correo y contraseña errónea, correo erróneo únicamente y contraseña errónea únicamente.

Además, se indica el nombre de la prueba, el caso de uso al que pertenece y, en caso de necesitarlo, una serie de observaciones.

El resto de guiones de pruebas de los casos de uso se describen a continuación:

Tabla 12. Secuencia 1.2 de pruebas funcionales.

Nombre de prueba.	**1.2 Inicio de sesión correcto.**
Caso de uso.	Inicio de sesión.
Estado inicial.	Pantalla principal del dispositivo.
Secuencia.	1. Abrir aplicación. 2. Introducir correo electrónico correcto. 3. Introducir contraseña correcta. 4. Pulsar el botón “iniciar sesión”.
Estado final.	Accede a la aplicación mostrando la ventana de “resumen de hoy”.
Observaciones.	-

Tabla 13. Secuencia 1.3 de pruebas funcionales.

Nombre de prueba.	**1.3 Inicio de sesión sin conexión.**
Caso de uso.	Inicio de sesión.
Estado inicial.	Pantalla principal del dispositivo.
Secuencia.	1. Abrir aplicación. 2. Introducir correo electrónico. 3. Introducir contraseña. 4. Pulsar el botón “iniciar sesión”.
Estado final.	Muestra un mensaje de error indicando que la autenticación ha fallado.
Observaciones.	Es indiferente comprobar el inicio de sesión sin conexión con credenciales correctas o incorrectas.

Tabla 14. Secuencia 1.4 de pruebas funcionales.

Nombre de prueba.	**1.4 Cambio de contraseña.**
Caso de uso.	Inicio de sesión.
Estado inicial.	Pantalla principal del dispositivo.
Secuencia.	1. Abrir aplicación. 2. Pulsar el botón “he olvidado la contraseña”. 3. En la ventana de diálogo, introducir el correo electrónico. 4. Pulsar el botón “aceptar”.
Estado final.	Muestra un mensaje indicando que si el correo electrónico introducido se encuentra dado de alta se envía un correo a la dirección introducida.
Observaciones.	-

Tabla 15. Secuencia 1.5 de pruebas funcionales.

Nombre de prueba.	**1.5 Cambio de contraseña vacío.**
Caso de uso.	Inicio de sesión.
Estado inicial.	Pantalla principal del dispositivo.
Secuencia.	1. Abrir aplicación. 2. Pulsar el botón “he olvidado la contraseña”. 3. No introducir el correo electrónico en la ventana de diálogo. 4. Pulsar el botón “aceptar”.
Estado final.	Muestra un mensaje indicando que se debe de introducir un correo electrónico para realizar el cambio de contraseña.

Observaciones.	-

Tabla 16. Secuencia 2.1 de pruebas funcionales.

Nombre de prueba.	**2.1 Acceso a funcionalidades desde el menú lateral.**
Caso de uso.	Menú lateral.
Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Pulsar botón de menú, situado en la esquina superior izquierda. 2. Pulsar cualquier ítem del menú.
Estado final.	Se accede a la ventana del ítem seleccionado.
Observaciones.	Se realiza la secuencia indicada para cada ítem del menú principal.

Tabla 17. Secuencia 2.2 de pruebas funcionales.

Nombre de prueba.	**2.2 Cambio del botón del menú lateral.**
Caso de uso.	Menú lateral.
Estado inicial.	Ventana de “resumen de hoy”.
Secuencia.	1. Pulsar el botón del menú lateral. 2. Seleccionar un ítem del menú. 3. Pulsar nuevamente el botón del menú.
Estado final.	Se vuelve a la ventana anterior.
Observaciones.	Se realiza la secuencia para cada ítem del menú principal comprobando que el botón cambia de forma a una flecha indicando que se va a retroceder.

Tabla 18. Secuencia 3.1 de pruebas funcionales.

Nombre de prueba.	**3.1 Introducción obligatoria del cuestionario de configuración del perfil del paciente.**
Caso de uso.	Configuración de perfil del paciente.
Estado inicial.	Ventana de “inicio de sesión”.
Secuencia.	1. Introducir las credenciales correctas. 2. Pulsar el botón de “aceptar”.
Estado final.	Muestra automáticamente el cuestionario de configuración de perfil.
Observaciones.	Se obliga al usuario a rellenar el cuestionario antes de poder acceder a cualquier otra funcionalidad.

Tabla 19. Secuencia 3.2 de pruebas funcionales.

Nombre de prueba.	**3.2 Introducción errónea del cuestionario de configuración de perfil del paciente.**
Caso de uso.	Configuración de perfil del paciente.
Estado inicial.	Ventana de “resumen de hoy” o tras inicio de sesión.
Secuencia.	1. Seleccionar el ítem de cuestionario de configuración de perfil. 2. No contestar ninguna pregunta. 3. Pulsar “aceptar”.
Estado final.	Muestra un mensaje indicando que faltan datos por introducir.
Observaciones.	Se realiza la secuencia anterior para todas las preguntas del cuestionario de modo que no se puede avanzar si no se completan todas las preguntas.

Tabla 20. Secuencia 3.3 de pruebas funcionales.

Nombre de prueba.	**3.3 Introducción correcta del cuestionario de configuración de perfil del paciente.**
Caso de uso.	Configuración de perfil del paciente.
Estado inicial.	Ventana de “resumen de hoy” o tras inicio de sesión.
Secuencia.	1. Seleccionar el ítem de cuestionario de configuración de perfil. 2. Contestar todas las preguntas. 3. Pulsar “aceptar”.
Estado final.	Muestra un mensaje indicando que el cuestionario se ha guardado correctamente y se regresa a la ventana de “resumen de hoy”.
Observaciones.	-

Tabla 21. Secuencia 3.4 de pruebas funcionales.

Nombre de prueba.	**3.4 Botones de navegación en el cuestionario de configuración de perfil del paciente.**
Caso de uso.	Configuración de perfil del paciente.
Estado inicial.	Ventana de “cuestionario de configuración del perfil del paciente”.
Secuencia.	1. Introducir los datos correctamente. 2. Pulsar “siguiente”. 3. Una vez se ha avanzado al siguiente conjunto de preguntas, pulsar “atrás” o el botón de retroceso del dispositivo.
Estado final.	Muestra las siguientes o las preguntas anteriores según si se ha pulsado el botón “siguiente” o “atrás” respectivamente.

Observaciones.	-

Tabla 22. Secuencia 4.1 de pruebas funcionales.

Nombre de prueba.	**4.1 Actualización de tarjetas en el resumen de hoy.**
Caso de uso.	Resumen de hoy.
Estado inicial.	Ventana de “resumen de hoy”.
Secuencia.	1. Seleccionar una de las tareas. 2. Realizar la tarea, ya sea la introducción de síntomas, peso o cuestionarios.
Estado final.	Se vuelve a la ventana “resumen de hoy” y se elimina la tarjeta según la tarea realizada.
Observaciones.	Se repite la secuencia para todas las tareas comprobando que una vez se ha completado una, es eliminada.

Tabla 23. Secuencia 5.1 de pruebas funcionales.

Nombre de prueba.	**5.1 Dialogo informativo mostrado.**
Caso de uso.	Cuestionario de estado de ánimo.
Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Seleccionar la tarea de cuestionario de estado de ánimo o el ítem en el menú lateral. 2. Pulsar el botón “aceptar” para cerrar la ventana de diálogo.
Estado final.	Pantalla de “cuestionario de estado de ánimo”.
Observaciones.	Se comprueba si pulsando fuera de la pantalla o con el botón físico del dispositivo es posible cerrar el diálogo. Sólo se cierra si se pulsa el botón “aceptar”.

Tabla 24. Secuencia 5.2 de pruebas funcionales.

Nombre de prueba.	**5.2 Botones de navegación del cuestionario de estado de ánimo.**
Caso de uso.	Cuestionario de estado de ánimo.
Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Seleccionar la tarea de cuestionario de estado de ánimo o el ítem en el menú lateral. 2. Pulsar el botón “aceptar” para cerrar la ventana de diálogo. 3. Seleccionar una respuesta a la pregunta. 4. Pulsar el botón “siguiente”. 5. En la siguiente pregunta, pulsar el botón “atrás”.
Estado final.	Se avanza una pregunta y retrocede a la anterior correctamente.
Observaciones.	-

Tabla 25. Secuencia 5.3 de pruebas funcionales.

Nombre de prueba.	**5.3 Introducción errónea del cuestionario de estado de ánimo.**
Caso de uso.	Cuestionario de estado de ánimo.
Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Seleccionar la tarea de cuestionario de estado de ánimo o el ítem en el menú lateral. 2. Pulsar el botón “aceptar” para cerrar la ventana de diálogo. 3. No seleccionar ninguna respuesta. 4. Pulsar el botón “siguiente”.

Estado final.	Muestra un mensaje indicando que se debe seleccionar una respuesta.
Observaciones.	Se realiza la secuencia anterior con todas las preguntas del cuestionario de modo que no se puede avanzar si no se completan todas las preguntas.

Tabla 26. Secuencia 5.4 de pruebas funcionales.

Nombre de prueba.	**5.4 Introducción correcta del cuestionario de estado de ánimo.**
Caso de uso.	Cuestionario de estado de ánimo.
Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Seleccionar la tarea de cuestionario de estado de ánimo o el ítem en el menú lateral. 2. Pulsar el botón “aceptar” para cerrar la ventana de diálogo. 3. Responder a todas las preguntas. 4. Pulsar el botón “siguiente”.
Estado final.	Muestra un mensaje indicando que el cuestionario se ha guardado correctamente y se regresa a la ventana de “resumen de hoy”.
Observaciones.	-

Tabla 27. Secuencia 6.1 de pruebas funcionales.

Nombre de prueba.	**6.1 Botones de navegación del cuestionario de calidad de vida.**
Caso de uso.	Cuestionario de calidad de vida.
Estado inicial.	Pantalla de “resumen de hoy”.

Secuencia.	1. Seleccionar la tarea de cuestionario de calidad de vida o el ítem en el menú lateral. 2. Seleccionar una respuesta a la pregunta. 3. Pulsar el botón “siguiente”. 4. En la siguiente pregunta, pulsar el botón “atrás”.
Estado final.	Se avanza una pregunta y retrocede a la anterior correctamente.
Observaciones.	-

Tabla 28. Secuencia 6.2 de pruebas funcionales.

Nombre de prueba.	**6.2 Introducción errónea del cuestionario de calidad de vida.**
Caso de uso.	Cuestionario de calidad de vida.
Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Seleccionar la tarea de cuestionario de calidad de vida o el ítem en el menú lateral. 2. No seleccionar ninguna respuesta. 3. Pulsar el botón “siguiente”.
Estado final.	Muestra un mensaje indicando que se debe seleccionar una respuesta.
Observaciones.	Se realiza la secuencia anterior con todas las preguntas del cuestionario de modo que no se puede avanzar si no se completan todas las preguntas.

Tabla 29. Secuencia 6.3 de pruebas funcionales.

Nombre de prueba.	**6.3 Introducción correcta del cuestionario de calidad de vida.**
Caso de uso.	Cuestionario de calidad de vida.

Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Seleccionar la tarea de cuestionario de calidad de vida o el ítem en el menú lateral. 2. Responder a todas las preguntas. 3. Pulsar el botón “siguiente”.
Estado final.	Muestra un mensaje indicando que el cuestionario se ha guardado correctamente y se regresa a la ventana de “resumen de hoy”.
Observaciones.	-

Tabla 30. Secuencia 7.1 de pruebas funcionales.

Nombre de prueba.	**7.1 Introducción errónea de peso.**
Caso de uso.	Introducción de peso.
Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Seleccionar la tarea de introducción de peso o el ítem en el menú lateral. 2. Introducir un valor no numérico. 3. Pulsar el botón “guardar”.
Estado final.	Muestra un mensaje indicando que se debe de introducir un valor numérico.
Observaciones.	Se comprueba que el mensaje también se muestra cuando no se introduce ningún valor.

Tabla 31. Secuencia 7.2 de pruebas funcionales.

Nombre de prueba.	**7.2 Introducción correcta de peso.**
Caso de uso.	Introducción de peso.

Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Seleccionar la tarea de introducción de peso o el ítem en el menú lateral. 2. Introducir un valor numérico. 3. Pulsar el botón “guardar”.
Estado final.	Muestra un mensaje indicando que se ha guardado correctamente y se regresa a la ventana de “resumen de hoy”.
Observaciones.	-

Tabla 32. Secuencia 8.1 de pruebas funcionales.

Nombre de prueba.	**8.1 Gráfica actualizada correctamente.**
Caso de uso.	Evolución de peso.
Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Seleccionar la tarea de introducción de peso o el ítem en el menú lateral. 2. Introducir un valor numérico. 3. Pulsar el botón “guardar”. 4. Abrir el menú lateral y pulsar en el ítem de “evolución de peso”. 5. Seleccionar el punto más a la derecha de la gráfica.
Estado final.	El valor del último punto de la gráfica es igual al valor introducido anteriormente.
Observaciones.	-

Tabla 33. Secuencia 8.2 de pruebas funcionales.

Nombre de prueba.	**8.2 Grafica filtra correctamente.**

Caso de uso.	Evolución de peso.
Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Abrir el menú lateral y pulsar en el ítem de “evolución de peso”. 2. Seleccionar uno de los tres filtros.
Estado final.	La gráfica filtra únicamente el rango de tiempo seleccionado.
Observaciones.	Se realiza la secuencia anterior cambiando entre los tres filtros, de este modo se observa si el filtrado es correcto.

Tabla 34. Secuencia 9.1 de pruebas funcionales.

Nombre de prueba.	**9.1 Colores cambian conforme con el tipo de síntoma.**
Caso de uso.	Introducción de síntomas.
Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Seleccionar la tarea de introducción de síntomas o el ítem en el menú lateral. 2. Seleccionar un tipo de síntoma.
Estado final.	El color de las tarjetas del síntoma especifico es el mismo que el tipo de síntoma seleccionado.
Observaciones.	Se realiza la secuencia anterior con todos los tipos de síntomas comprobando que el color de cada tipo de síntomas es el mismo que el de los síntomas específicos.

Tabla 35. Secuencia 9.2 de pruebas funcionales.

Nombre de prueba.	**9.2 Síntoma introducido correctamente.**
Caso de uso.	Introducción de síntomas.

Estado inicial.	Pantalla de “resumen de hoy”.
Secuencia.	1. Seleccionar la tarea de introducción de síntomas o el ítem en el menú lateral. 2. Seleccionar un tipo de síntoma. 3. Seleccionar el síntoma específico. 4. Introducir valor si es necesario.
Estado final.	Muestra un mensaje indicando que el síntoma se ha guardado correctamente y se regresa a la ventana de “resumen de hoy.
Observaciones.	Se realiza la secuencia anterior con todos los tipos de síntomas.

Una vez se finaliza una prueba, el usuario reporta si hay un error y se realizan las pruebas de regresión siguiendo el mismo guion utilizado anteriormente. Así, se comprueba si el error reportado ha sido solucionado.

En lo que respecta a las fases del proyecto NuMielo, este se divide en una parte de desarrollo y otra de pilotaje. Esta última fase tiene lugar de principios de octubre a finales de diciembre donde la aplicación se prueba con usuarios reales, los cuales han firmado el consentimiento informado proporcionado por los profesionales sanitarios.

El objetivo de esta fase es valorar los aspectos de usabilidad, diseño e interacción, funcionalidad, rendimiento y estabilidad. Tras recopilar todos los datos proporcionados por la fase de pilotaje se plantea si realizar mejoras o añadir nuevas funcionalidades a la aplicación.

7. Conclusiones.

Como se ha comentado en el apartado de introducción y objetivos, este TFG forma parte de un proyecto de mayor magnitud. Formado por dos subsistemas, NuMielo tiene como objetivo la creación de un sistema que permite realizar un seguimiento a los pacientes para mejorar su tratamiento y su calidad de vida. Este sistema recoge datos del proceso de la enfermedad para, posteriormente, realizar mejoras en el proceso de tratamiento y conocer mejor la enfermedad.

El proyecto NuMielo está formado por un subsistema de aplicación web para profesionales, donde estos realizan el seguimiento de la enfermedad y una aplicación móvil para los pacientes.

El objetivo de este TFG ha sido desarrollar el subsistema de la aplicación móvil, una herramienta para los pacientes con Mieloma Múltiple. Esta permite realizar un seguimiento y recolectar los datos que finalmente visualizan los profesionales sanitarios.

Entre los sub-objetivos se plantea obtener una interfaz amigable, atractiva y fácil de utilizar; facilitar la traducción de la aplicación a cualquier idioma permitiendo a más personas el uso del software y seguir un conjunto de buenas prácticas para facilitar el mantenimiento del proyecto.

Para alcanzar estos objetivos se han seguido las líneas de diseño de *Material Design* obteniendo una interfaz amigable, atractiva y fácil de utilizar. La facilidad de traducción del idioma de la app se ha conseguido haciendo uso de los recursos Android proporcionados por el propio SO. Finalmente, las buenas prácticas seguidas para obtener la una buena estructura, proporcionar robustez y mantenibilidad del proyecto, se han seguido una serie de patrones arquitectónicos y documentación del código.

Se han aplicado conocimientos adquiridos durante el grado de Ingeniería Informática durante todo el desarrollo del TFG. La elección de la metodología ágil utilizada se ha estudiado en la asignatura de Proyecto de Ingeniería del Software. Por otro lado, asignaturas como Ingeniería del Software o Análisis y Especificación de Requisitos han ayudado con la realización de los requisitos software y la elección del modelo de arquitectura por capas. Finalmente, el lenguaje utilizado para la programación de esta app, Java, ha sido introducido desde el inicio de la carrera en asignaturas como Introducción a la Programación, Estructuras de Datos.

El autor ha dedicado tiempo al estudio y aprendizaje de *Android Room* como primera solución de implementación de la base de datos, ya que no es una tecnología

vista en cursos anteriores. Una vez realizado el cambio de tecnología, migrando la base de datos *Android Room* a *Firebase*, también se realiza un aprendizaje de esta nueva tecnología. Aunque, *Firebase* ha supuesto una ventaja a la hora de implementar la comunicación entre la app y la nueva base de datos, simplificando el código y las llamadas necesarias para realizar las operaciones de lectura y escritura.

Durante el desarrollo de la app se han realizado cambios en la tecnología utilizada para el almacenamiento de datos, obligando a cambiar también la arquitectura del sistema. Dos de las funcionalidades parcialmente descritas en el trabajo, el progreso del paciente y el contenido educacional, no son implementadas debido a que los profesionales sanitarios continúan generando los datos y documentación necesaria para terminar de especificar e implementarlas. Aunque este cambio de tecnología haya podido afectar en el desarrollo, debido a la elección del modelo de arquitectura por capas, se ha conseguido reducir este impacto consiguiendo aislar el resto de componentes sin verse afectados.

Cabe destacar ciertas competencias transversales puestas en práctica durante la realización de este TFG. Una de las más empleadas ha sido la competencia de diseño y proyecto, estando presente durante todo el desarrollo de la app y realizando un amplio esfuerzo en este. También se ha puesto en práctica el análisis y resolución de problemas, ya que, partiendo de una idea proporcionada por los profesionales de la salud, se ha impuesto un esfuerzo inicial a analizar dicha idea y ofrecer la solución más correcta y mejor que mejor se adapta a los requisitos. Además, la planificación y gestión del tiempo ha ayudado a planificar el proyecto y cumplir con las fechas de entrega, incluso con los contratiempos de los cuales hemos hablado anteriormente. Para finalizar, tratándose de un pequeño grupo de desarrollo también se ha desarrollado la competencia de trabajo en equipo y liderazgo.

En resumen, se ha enfrentado a un problema real ofreciendo la solución óptima para este. Los problemas de coordinación han estado presentes en una parte del proceso de desarrollo software, pero gracias al modelo de arquitectura elegido se ha podido tratar con estos de una manera más sencilla reduciendo el tiempo necesario para realizar los cambios. Además, el proyecto NuMielo continúa tanto su desarrollo como el uso real de la app después de la realización de este TFG. La app va a ser probada y mantenida, por esta razón se ha desarrollado un código estructurado y fácil de mantener, ya que son características decisivas en un sistema software duradero. Hablaremos más detalladamente en el apartado de trabajos futuros.

8. Trabajos futuros.

Aunque los objetivos del TFG se han alcanzado, el proyecto NuMielo aún no ha terminado. La fase de desarrollo continúa, finalizando aquellas funcionalidades no abordadas fuera del alcance del trabajo para, después, realizar la fase de pilotaje en un entorno con usuarios reales. Esta última fase de validación se hace respecto a la versión inicial del sistema, obtenido en este TFG. A partir de esta fase, el proyecto será continuado por otro desarrollador, al cual el autor delega y explica el proceso y las pautas a seguir para continuar con dicho proyecto.

Como hemos mencionado anteriormente, el proyecto continúa su desarrollo y, en consecuencia, la app. En base a esta primera versión de la aplicación, se introducirán mejoras gracias al *feedback* o retroalimentación ofrecida por los usuarios reales de prueba. También se podrán introducir nuevas funcionalidades según las opiniones de estos usuarios.

Las funcionalidades de progreso y contenido educacional se han descrito parcialmente en este trabajo y se implementarán una vez los profesionales generen la documentación y se pueda finalizar la especificación.

9. Referencias.

1. MERINO, MARÍA; IVANOVA, YOANA; MARAVILLA, PAULINA; HIDALGO-VEGA, Álvaro. Valor social de la mejora del abordaje del mieloma múltiple en el sistema nacional de salud. En : . 2019.
2. RODRÍGUEZ-LAGOS, F. ANDREA; SORLÍ, JOSÉ V.; CALVIÑO-NAVEIRA, MICHELLE C.; CAPELL, Nuria Estañ. Papel del médico de familia en el diagnóstico concomitante de mieloma y amiloidosis primaria en una misma paciente. . 2018. DOI 0034-9887.
3. RIBELLES, LUCIA CARRASCO; ROJO-AGUSTI, ANA; ANTEQUERA-SANCHEZ, CARLOS; IBÁÑEZ SÁNCHEZ, GEMA; PERLA, AURORA; DOMÍNGUEZ, EVA; JARQUE, Isidro. *Systematic Review on Support Systems for Cancer Patient: Nutritional Assessment and Educative Strategy Improve Patients QOL*. 2018. ISBN 978-84-09-07373-3.
4. LETELIER, Patricio. Agility at work, Modelos de proceso para desarrollo ágil. [en línea]. 2014. Disponible en: http://agilismoatwork.blogspot.com/2014/10/modelos-de-proceso-para-desarrollo-agil.html
5. LETELIER, Patricio. Agility at work, Tableros kanban. [en línea]. 2015. Disponible en: http://agilismoatwork.blogspot.com/2015/06/visualizacion-agil-del-trabajo-del.html
6. EUROPEAN COMMISSION. MyCyFAPP. [en línea]. 2018. Disponible en: https://www.mycyfapp.eu/index.php/en/
7. TEROL-CANTERO, M. CARMEN; CABRERA-PERONA, VÍCTOR; MARTÍN-ARAGÓN, Maite. *Revisión de estudios de la Escala de Ansiedad y Depresión Hospitalaria (HAD) en muestras españolas.* 2015.
8. CABASÉS, Juan M. *El EQ-5D como medida de resultados en salud*. 2015.

10. Glosario de términos.

A

API: Application programming interface (interfaz de programación de aplicaciones) – Es un conjunto de comandos, funciones y protocolos informáticos que permiten a los desarrolladores crear programas específicos para ciertos sistemas operativos.

AVD: Android Virtual Device, del inglés, dispositivo virtual Android.

C

CI/CD: integración continua y distribución continua.

CPAP: Continuous Positive Airway Pressure (Presión positiva continua en la vía aérea) – Es un dispositivo utilizado para tratar la interrupción temporal de la respiración durante el sueño enviando una presión continua de aire por las vías aéreas.

I

IDE: Integrated development environment, del inglés, entorno de desarrollo integrado.

M

MVVM: Model–view–viewmodel (Modelo-vista–modelo de vista) – Patrón arquitectónico software que permite separar la interfaz de usuario de la lógica de la aplicación.

R

REST: Representational State Transfer (Transferencia de Estado Representacional) – Estilo de arquitectura software destinado al desarrollo de servicios web.

S

SO: abreviatura de sistema operativo.

11. Anexos.

11.1. Anexo A.

Escala de Ansiedad y Depresión Hospitalaria (HAD) (Zigmond y Snaith, 1983)

Lea cada pregunta y subraye la respuesta que usted considere que coincide con su propio estado emocional en la última semana. No es preciso que preste atención a los números que aparecen a la izquierda.

No es necesario que piense mucho tiempo cada respuesta: en este cuestionario las respuestas espontáneas tienen más valor que las que se piensan mucho.

A.1. Me siento tenso/a o nervioso/a:

3. Casi todo el día.
2. Gran parte del día.
1. De vez en cuando.
0. Nunca.

D.1. Sigo disfrutando de las cosas como siempre:

0. Ciertamente, igual que antes.
1. No tanto como antes.
2. Solamente un poco.
3. Ya no disfruto con nada.

A.2. Siento una especie de temor como si algo malo fuera a suceder:

3. Sí, y muy intenso.
2. Sí, pero no muy intenso.
1. Sí, pero no me preocupa.
0. No siento nada de eso.

D.2. Soy capaz de reírme y ver el lado gracioso de las cosas:

0. Igual que siempre.
1. Actualmente, algo menos.
2. Actualmente, mucho menos.
3. Actualmente, en absoluto.

A.3. Tengo la cabeza llena de preocupaciones:

3. Casi todo el día.

2. Gran parte del día.

1. De vez en cuando.

0. Nunca.

D.3. Me siento alegre:

3. Nunca.

2. Muy pocas veces.

1. En algunas ocasiones.

0. Gran parte del día.

A.4. Soy capaz de permanecer sentado/a tranquilo/a y relajado/a:

0. Siempre.

1. A menudo.

2. Raras veces.

3. Nunca.

D.4. Me siento lento/a y torpe:

3. Gran parte del día.

2. A menudo.

1. A veces.

0. Nunca.

A.5. Experimento una desagradable sensación de "nervios y hormigueos" en el estómago:

0. Nunca.

1. Sólo en algunas ocasiones.

2. A menudo.

3. Muy a menudo.

D.5. He perdido el interés por mi aspecto personal:

3. Completamente.

2. No me cuido como debería hacerlo.

1. Es posible que no me cuide como debiera.

0. Me cuido como siempre lo he hecho.

A.6. Me siento inquieto/a como si no pudiera parar de moverme:

3. Realmente mucho.

2. Bastante.

1. No mucho.

0. Nunca.

D.6. Espero las cosas con ilusión:

0. Como siempre.

1. Algo menos que antes.

2. Mucho menos que antes.

3. En absoluto.

A.7. Experimento de repente sensaciones de gran angustia o temor:

3. Muy a menudo.

2. Con cierta frecuencia.

1. Raramente.

0. Nunca.

D.7. Soy capaz de disfrutar con un buen libro o con un buen programa de radio o televisión:

0. A menudo.

1. Algunas veces.

2. Pocas veces.

3. Casi nunca.

11.2. Anexo B.

Cuestionario de Salud (EQ-5D-5L)

Debajo de cada enunciado, marque UNA casilla, la que mejor describe su salud HOY.

1. MOVILIDAD.

□1.1. No tengo problemas para caminar.

□1.2. Tengo problemas leves para caminar.

□1.3. Tengo problemas moderados para caminar.

□1.4. Tengo problemas graves para caminar.

□1.5. No puedo caminar.

2. AUTO-CUIDADO.

□2.1. No tengo problemas para lavarme o vestirme.

□2.2. Tengo problemas leves para lavarme o vestirme.

□2.3. Tengo problemas moderados para lavarme o vestirme.

□2.4. Tengo problemas graves para lavarme o vestirme.

□2.5. No puedo lavarme o vestirme.

3. ACTIVIDADES COTIDIANAS (Ej. Trabajar, estudiar, hacer las tareas domésticas, actividades familiares o actividades durante el tiempo libre).

□3.1. No tengo problemas para realizar mis actividades cotidianas.

□3.2. Tengo problemas leves para realizar mis actividades cotidianas.

□3.3. Tengo problemas moderados para realizar mis actividades cotidianas.

□3.4. Tengo problemas graves para realizar mis actividades cotidianas.

□3.5. No puedo realizar mis actividades cotidianas.

4. DOLOR/ MALESTAR.

□4.1. No tengo dolor ni malestar.

□4.2. Estoy levemente ansioso o deprimido.

□4.3. Estoy moderadamente ansioso o deprimido.

□4.4. Estoy muy ansioso o deprimido.

□4.5. Estoy extremadamente ansioso o deprimido.

11.3. Anexo C.

Tipo.	Síntoma.	Valor.	Condición alerta.	Aviso.	Recomendación.
Infección.	Fiebre.	Temperatura.	>38º	Sí.	Sí.
Digestiva.	Náuseas.	-	-	No.	Sí.
	Vómitos.	-	3 episodios en 24 h	Sí.	Sí.
	Estreñimiento.	-	>48 h	No.	Sí.
	Diarrea.	Escala Bristol (ver Anexo D)	>5 deposiciones tipo 6 o 7 al día.	Sí.	Sí.

	Boca seca.	-	-	No.	Sí.
	Alteración del sabor.	-	-	No.	Sí.
	Falta de apetito (anorexia, pérdida de peso involuntaria).	-	-	No.	Sí.
	Llagas en la boca.	-	-	No.	Sí.
	Dolor abdominal.	-	-	No.	Sí.
Respiratorios.	Dificultad para respirar.	-	-	Sí.	Sí.
	Tos con esputo amarillo o verde.	-	-	Sí.	Sí.
	Dolor en pecho al toser.	-	-	Sí.	Sí.
Sistema nervioso.	Calambres, adormecimiento en manos y/o pies.	-	-	Sí.	Sí.
	Insomnio.	-	-	No.	Sí.
	Depresión.	-	-	No.	Sí.
	Cansancio no explicado	-	-	Sí.	Sí.

	(sensación de ahogo, fatiga).				
	Temblores.	Temperatura.	>38º	Sí.	Sí.
	Hiperactividad.	-	-	No.	Sí.
	Rechazo a la actividad física.	-	-	No.	Sí.
Sistema circulatorio.	Hipertensión arterial.	Sangrado de encías y/o nasal?	Respuesta positiva.	Sí.	Sí.
	Hematomas.	-	-	Sí.	Sí.
	Hinchazón en brazos y piernas.	-	-	Sí.	Sí.
	Anomalía del ritmo cardíaco (taquicardia).	-	-	No.	Sí.
Renal.	Dificultad para orinar.	¿Siente dolor?	Respuesta positiva.	Sí.	Sí.
	Sangre al orinar.	-	-	Sí.	Sí.
Cutánea.	Sarpullido, erupciones o picor en la piel.	-	-	Sí.	Sí.
Articulares.	Dolor en manos y/o pies.	-	-	Sí.	Sí.

11.4. Anexo D.

Ilustración 48. Escala de heces Bristol.