Grado en Ingeniería Informática

Trabajo Fin de Grado

App asistencial para niños prematuros

Cristian Manuel Suárez Vera

Tutor: **Alexis Quesada Arencibia**

Las Palmas de Gran Canaria
12 de julio de 2018

UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Escuela de Ingeniería Informática

SOLICITUD DE DEFENSA DE TRABAJO DE FIN DE

D/Dª Cristian Manuel Suárez Vera, autor del Trabajo de Fin de Título App Asistencial para niños prematuros,

correspondiente a la titulación Grado en Ingeniería Informática,
en colaboración con la empresa/proyecto (indicar en su caso) _____

S O L I C I T A

que se inicie el procedimiento de defensa del mismo, para lo que se adjunta la documentación requerida.

Asimismo, con respecto al registro de la propiedad intelectual/industrial del TFT, declara que:

[ ] Se ha iniciado o hay intención de iniciarlo (defensa no pública).
[X] No está previsto.

Y para que así conste firma la presente.

Las Palmas de Gran Canaria, a 13 de Julio de 2018.

El estudiante

Firmado por SUAREZ VERA, CRISTIAN MANUEL (FIRMA) el día 13/07/2018 con un

Fdo.: Cristian Suárez Vera

A rellenar y firmar **obligatoriamente** por el/los tutor/es

En relación a la presente solicitud, se informa:

[X] Positivamente

[ ] Negativamente
(la justificación en caso de informe negativo deberá incluirse en el TFT05)

QUESADA ARENCIBIA, FRANCISCO ALEXIS (FIRMA)

Firmado digitalmente por QUESADA ARENCIBIA, FRANCISCO ALEXIS (FIRMA)
Nombre de reconocimiento (DN): c=ES, serialNumber=42870168P, sn=QUESADA, givenName=FRANCISCO ALEXIS, cn=QUESADA ARENCIBIA, FRANCISCO ALEXIS (FIRMA)
Fecha: 2018.07.13 18:48:42 +01'00'

Fdo.: Alexis Quesada Arencibia

DIRECTOR DE LA ESCUELA DE INGENIERÍA INFORMÁTICA

1. Agradecimientos

A mi familia y amigos por haberme ayudado y soportado a llegar hasta aquí, en especial a mi padre que siempre está dispuesto a ayudarme a encontrar una solución.

A *StackOverFlow* por tener una gran comunidad siempre dispuesta a ayudar que ha sido de gran ayuda para solventar problemas técnicos específicos.

A mi tutor, Alexis Quesada Arencibia, por haberme dado la oportunidad de enfrentarme a este proyecto, ha sido todo un reto al que enfrentarse.

Resumen

Premature children present a unique characteristic and that is why It needs special care. One of the fields that they require is the physiotherapy, which will focus on encouraging, facilitating ..., the different stages of motor development. This assistance involves the realization of therapies that modification according to their age and their evolution. In addition, parents are entrusted continue with the therapy at home, performing the exercises that the professional advise This is a challenge of great difficulty for parents number of different exercises and postures that they must remember. In this project has been developed a multiplatform application prototype to support parents and physiotherapists to facilitate the monitoring of therapies settled down.

Resumen

Los niños prematuros presentan unas características únicas y es por ello que necesitan de unos cuidados especiales. Uno de los ámbitos que requieren es la fisioterapia, que irá enfocada a estimular, facilitar..., las diferentes etapas del desarrollo motor. Esta asistencia conlleva la realización de terapias que varían en función de su edad y su evolución. Además, se encomienda a los padres continuar con la terapia en casa, realizando los ejercicios que el profesional aconseja. Esto supone un reto de gran dificultad para los padres dada la cantidad de ejercicios y posturas diferentes que deben recordar. En este proyecto se ha desarrollado un prototipo de app multiplataforma de apoyo a padres y fisioterapeutas para facilitar el seguimiento de las terapias establecidas.

Índice

Índice de figuras

Índice de tablas

2. Introducción

El día 7 de abril de 2005 en la 58ª asamblea mundial de la salud la Organización Mundial de la Salud [13] trató la temática de la cibersalud (conocida también como e-Salud o e-Health) y su importancia. Indicando que esta çonsiste en el apoyo que la utilización costoeficaz y segura de las tecnologías de la información y las comunicaciones ofrece a la salud y a los ámbitos relacionados con ella, con inclusión de los servicios de atención de salud, la vigilancia y la documentación sanitaria, así como la educación, los conocimientos y las investigaciones en materia de salud"[5].

Dados los avances de las nuevas tecnologías y que estas están presentes en facetas de nuestra vida diaria como: educación, comunicación, producción industrial. ¿Porqué no en la medicina? Aplicaciones como BilliCam [3] con la que se puede sacar una foto al bebé con una tarjeta de calibración en el vientre. De esta forma, la aplicación distingue la luz y el tono de la piel del recién nacido, y con ello busca el diagnóstico de la ictericia neonatal. Como esta hay otras aplicaciones las cuales buscan solucionar problemas muy específicos.

Patient Portal by ConstantMD [14] es una aplicación para dispositivos móviles que permite almacenar citas en varios de centros y actividades. Además, permite tener ordenadas las citas y las reservas que se hayan realizado.

CoreFusion Pilates and Physio [8] tiene como objetivo objetivo mostrar a los usuarios el estado de las clases de Pilates para así poder saber cuales están libres y cuáles pueden ser reservadas.

Otras como *Connect Physiotherapy* [7] permiten al usuario indicar la zona que quiere rehabilitar y ver ejercicios relacionados con ella para fortalecer dicha zona. Esta tiene un objetivo similar a la propuesta, con la diferencia de que *Connect Physiotherapy* está centrada en pacientes de media y avanzada edad.

Estas aplicaciones no se acercan a la propuesta. *BilliCam* ayuda a los niños prematuros a solucionar un problema del diagnóstico de la icteria neonatal. *Patient Protal* y *CoreFusio* solucionan el problema de gestión de citas, un objetivo que a largo plazo se podría plantear en la propuesta. Y finalmente *Connect Physiotherapy* es la que más se acerca, ya que tiene un objetivo similar pero esta solamente contiene ejercicios enfocados para personas adultas. Es por ello que se considera que es necesaria esta aplicación, ya que trata de solucionar un problema que ninguna de ellas ha tenido en cuenta, los ejercicios para niños prematuros. Se podría pensar que esto se pude solucionar con una mejora de *Connect Physiotherapy* añadiendo nuevos ejercicios para niños prematuros. Pero esta se queda corta en otros apartados tales como el poder mandar vídeos de la realización del ejercicio para que el doctor pueda revisar la evolución del paciente.

En la actualidad el índice de niños prematuros ha aumentado considerablemente. Estos niños presentan unas características únicas y es por ello que necesitan de unos cuidados especiales. Uno de los ámbitos que requieren es la

fisioterapia, que irá enfocada a estimular, facilitar, etc. las diferentes etapas del desarrollo motor, favoreciendo su desarrollo y previniendo posibles complicaciones a lo largo de su crecimiento, sobretodo en su primer año de vida.

Para conseguir este fin, el Servicio Canario de Salud dispone de un servicio de fisioterapia para toda la isla que atiende y realiza un seguimiento de los casos en los que se requiere de una asistencia profesional.

Esta asistencia conlleva la realización de una serie de terapias, consistentes en la ejecución de una serie de ejercicios con el bebé que varían en función de su edad y su evolución. Además, se encomienda a los padres continuar con la terapia en casa, realizando los ejercicios que el profesional aconseja. Esto supone un reto de gran dificultad para los padres dada la cantidad de ejercicios y posturas diferentes que deben recordar. Esto hace que los padres busquen comunicarse con los profesionales para que estos comprueben si están realizando correctamente el ejercicio concreto mediante el envío de vídeos por plataformas externas. Además, dada la gran demanda del servicio, en ocasiones se ven desbordados y la frecuencia con la que pueden citar a los bebés es muy inferior a la deseada, no pudiendo realizar un seguimiento de forma óptima. Por ello esta propuesta en conjunto con con una alumna de fisioterapia será de gran ayuda tanto a pacientes como a doctores. Esta alumna se encargará de preparar el contenido visual de la propuesta, como pueden ser diagramas y vídeos explicativos de los ejercicios.

Este TFG se complementa con el proyecto realizado por Dña. Kenya Fiesening Álvaro y tutorizado por Dña. María del Mar Batista Guerra, de la Facultad de Ciencias de la Salud, concretamente para su obtención del título del Grado en Fisioterapia. En este proyecto, tal y como indica su título "Diseño del contenido de una App para facilitar la continuidad del tratamiento de fisioterapia en niños prematuros en el domicilio", se han realizado importantes aportaciones para definir los requisitos de la App y se ha trabajo en el diseño del contenido que debe facilitar la App.

2.1. Objetivos

El alcance de la **App asistencial para niños prematuros** tiene los siguientes objetivos:

1. Poner a disposición de los padres material gráfico y multimedia de apoyo que permita desarrollar los ejercicios indicados por el fisioterapeuta.

2. Grabar vídeos e instantáneas a los padres para que puedan registrar la evolución del bebé de forma que dicho material pueda ser supervisado por el fisioterapeuta.

3. La comunicación Fisioterapeuta – Padres.

4. Realizar una valoración de la evolución del bebé.

5. Evaluar la utilidad de los materiales por parte de los padres.

Dado el tiempo del que se dispone para la realización de esta propuesta, se ha decidido abarcar dos de los objetivos básicos, los cuales son el punto número 1 y el número 2. Con esto los padres podrán realizar los ejercicios indicados con mayor seguridad teniendo a su disposición material de apoyo y, por su parte el médico podrá ver la evolución del bebé para prevenir cualquier otro problema que pueda surgir.

3. Justificación de las competencias

3.1. Comunes a la Ingeniería informática

3.1.1. CII08

Çapacidad para analizar, diseñar, construir y mantener aplicaciones de forma robusta, segura y eficiente, eligiendo el paradigma y los lenguajes de programación más adecuados."

A lo largo del proyecto se han abarcado todas las fases de un proyecto: análisis, diseño y desarrollo del mismo. Además, se ha tenido que hacer un estudio de las tecnologías disponibles hasta el momento para el desarrollo y decidir cual sería la mejor.

3.1.2. CII012

Çonocimiento y aplicación de las características, funcionalidades y estructura de las bases de datos, que permitan su adecuado uso, y el diseño y el análisis e implementación de aplicaciones basadas en ellos."

Se ha decidido la estructura de la base de datos y el tipo de datos que se usaria en cada momento en base a las necesidades del proyecto. Para sacar el máximo partido a las funcionalidades que se han desarrollado.

3.1.3. CII016

Çonocimiento y aplicación de los principios, metodologías y ciclos de vida de la ingeniería de software."

Se ha utilizando una metodología basada en el desarrollo incremental, puesto que el objetivo inicial era que el cliente con cada entrega viera una nueva funcionalidad que probar. Con la que poder recibir *feedback* y realizar las correcciones o mejoras posibles.

3.2. Ingeniería del Software (IS)

3.2.1. IS01

Çapacidad para desarrollar, mantener y evaluar servicios y sistemas software que satisfagan todos los requisitos del usuario y se comporten de forma fiable y eficiente, sean asequibles de desarrollar y mantener y cumplan normas de calidad, aplicando las teorías, principios, métodos y prácticas de la ingeniería del software."

Durante el desarrollo del proyecto, se ha seguido una metodología y planificación adecuada al cliente de manera que se han cumplido los objetivos marcados. Entregado software funcional y de calidad, acordes con las necesidades del problema.

3.2.2. IS02

Çapacidad para valorar las necesidades del cliente y especificar los requisitos software para satisfacer estas necesidades, reconciliando objetivos en conflicto mediante la búsqueda de compromisos aceptables dentro de las limitaciones derivadas del coste, del tiempo, de la existencia de sistemas ya desarrollados y de las propias organizaciones."

En la primera fase del proyecto se le ha dado importancia a las necesidades del cliente para conocer cuales son las funcionalidades más relevantes y con esto poder establecer prioridades en las tareas y modificar algunas, las cuales, no encajaban con la visión del cliente.

Para ello se ha ido mostrando *mockups* de la aplicación y diagramas. Además, durante el desarrollo se ha cambiado ciertos objetivos, dado que inicialmente no se había visto alguna problemática que ha surgido durante este.

3.2.3. IS03

Çapacidad de dar solución a problemas de integración en función de las estrategias, estándares y tecnologías disponibles."

Dado que finalmente se ha decidido hacer la aplicación conectada con un proveedor de base de datos externo, ha generado dificultades de integración que se han tenido que solucionar.

3.2.4. IS04

Çapacidad de identificar y analizar problemas y diseñar, desarrollar, implementar, verificar y documentar soluciones software sobre la base de un conocimiento adecuado de las teorías, modelos y técnicas actuales."

Para el desarrollo del proyecto se ha puesto en práctica distintas técnicas, principios y modelos vistos durante el grado con el objetivo de obtener el mejor *software* posible. Como han sido los diagramas UML, diagramas de casos de uso, etc.

4. Aportaciones al entorno

4.1. Entorno socio-económico

El desarrollo de este proyecto ofrecerá tanto a padres como doctores un medio de comunicación y visualización de ejercicios rápido, sencillo y con el que evitar gasto de papel que conlleva el sistema actual. A día de hoy a los padres se le dan hojas con las ilustraciones referentes a los ejercicios que tienen que realizar.

El gasto masivo de papel es un tema que está a la orden del día, puesto que de él deriva la tala de árboles, la cual tiene varias consecuencias negativas:

1. Cambios de clima debido a la falta de árboles para la retención de humedad, con lo que aumenta la posibilidad de sequías.
2. Destrucción de ecosistemas y la perdida de biodiversidad que ello conlleva.
3. Disminución del medio de transformación de dióxido de carbono en oxigeno, aumentando el efecto invernadero y aumentando las enfermedades respiratorias de la población.

Además de reducir el impacto negativo en el medioambiente, reduce gastos en los centros que hagan uso de la aplicación. Eliminando el gasto referente al papel y reduciendo el tiempo que los doctores utilizan para solventar dudas del paciente debido a:

1. Mayor rapidez en la comunicación doctor-paciente, pues con la solución propuesta esta comunicación es instantánea.
2. Ahorro por parte de los pacientes en cuanto a desplazamiento en combustible y tiempo.
3. Mayor facilidad para recordar el ejercicio, dado que se provee de material en forma de vídeo.
4. Seguridad por parte del paciente, al tener material y medio de comunicación directo con su doctor que puede consultar en cualquier momento.
5. Facilidad para el doctor a la hora de realizar seguimientos.

Este proyecto tiene consecuencias positivas tanto en el medio ambiente, como para los doctores y familias con hijos prematuros.

4.2. Personal

A nivel personal, este proyecto resulta enriquecedor. Ha sido desarrollado desde cero pasando por todas y cada una de las fases de un proyecto informático desde el análisis hasta la implementación.

Gracias a ello se ha podido reforzar los conocimientos adquiridos durante la carrera referentes a cada una de estas fases de desarrollo, así como a las herramientas y tecnologías utilizadas. A nivel de implementación la aplicación móvil a servido para conocer nuevas tecnologías como *Ionic*, *TypeScript*, *SCSS*, *Firebase*.

5. Metodología de trabajo y planificación del proyecto

5.1. Metodología de trabajo

Inicialmente se planteó una metodología de trabajo inspirada en el modelo de ciclo de vida del software basado en prototipos. Siendo el prototipo una aplicación con al menos una funcionalidad implementada, que se le presentará al cliente con la que podrá ver los avances y como se resuelve el requisito planteado. Además este podrá dar su opinión al respecto permitiendo modificar la forma en la que se implementa si no es lo esperado o no se ajusta con lo que había previsto, con ello se esperaba obtener un software a medida que se ajustara de la mejor forma posible a las problemáticas.

Una vez realizada la primera reunión con el cliente y planteada la metodología prevista este lo consideró inviable, puesto que no tiene la disponibilidad necesaria para hacer reuniones con frecuencia y probar el resultado obtenido hasta ese momento. Por ello hubo que cambiar la metodología a una clásica en cascada, esta consiste en plantear las tareas de manera secuencial e ir solucionando cada una solamente después de haber terminado la anterior. Una vez se ha concluido con todas, ya tendremos nuestro proyecto finalizado y listo para presentar y validar.

Se ha decidido utilizar finalmente esta metodología al ser la que mejor se ajusta a las necesidades del cliente y por la fuerte dependencia entre las funcionalidades. Puesto que la mayoría necesitan de que exista previamente otra.

5.2. Planificación inicial del proyecto

Tras un estudio del estado del arte actual, se decidió en la primera fase del proyecto hacer un análisis de la problemática con el que decidir cuales son los actores que intervienen en el sistema, es decir, cuales serán los usuarios que interactuarán con él y que funcionalidades podrán usar cada uno. Estas funcionalidades también se decidirán en esta primera fase de análisis. Para todo ello se hará uso de de los diagramas de casos de uso [4] y las tablas de especificación de los casos de uso [17].

Una vez se decidió los actores que actúan en el sistema y los requisitos (funcionalidades) de cada uno se continuó con un prototipo de la interfaz, para poder tener una primera visualización de los objetivos que se tenía con esta aplicación.

Este prototipo fue de gran ayuda puesto que se solucionaron dudas que había en ese momento sobre las funcionalidades planteadas y las soluciones propuestas, además planteó otras dudas sobre otros requisitos que no se habían planteado en un primer momento. A continuación se comentarán el cambio más significativo.

Inicialmente en la primera pantalla de los pacientes, se usó una lista con una imagen del ejercicio y el título de este como se puede ver en la figura 1. Tras la

presentación de este primer prototipo, se decidió que estas ocupaban demasiado espacio en la pantalla y eran molestas. Es por ello que se pasó a mostrar los ejercicios solamente con el título.

Figura 1: Pantalla inicial paciente, prototipo

Con esa parte ya solventada se empezó con el aprendizaje de la tecnología a usar, *Ionic* [9]. Para el aprendizaje de este *framework* se hizo una pequeña aplicación de gestión de tareas sugerida en el propio libro de aprendizaje utilizado, *"Learning ionic : buid hybrid mobile applications with HTML5"* [2]. Para complementar el aprendizaje de este libro se realizó varios tutoriales *online*, con todo ellos se adquirió experiencia y fluidez en el uso del *framework*.

Una vez acabada la fase de aprendizaje se continuó con el desarrollo, este contemplaba las funcionalidades que se describirán a continuación:

Las comunes para todos los usuarios:

1. Registro e inicio de sesión
2. Vista del perfil personal

3. Editar el perfil
4. Enviar y reproducir vídeos
5. Ver ejercicios

Específicas del paciente:

1. Marcar ejercicio como hecho

Específicas del doctor:

1. Buscar paciente
2. Asignar ejercicio
3. Añadir observaciones al ejercicio que se va a asignar
4. Ver el perfil de un paciente
5. Ver la evolución de un paciente

5.3. Ajustes en la planificación del proyecto

En la tabla 1 se muestra de forma resumida la planificación inicial del proyecto, donde podemos ver las principales fases y tareas y una estimación de las horas de trabajo previstas para cada una de ellas. Sin embargo, la tarea de implementación (**Tarea 2.4**) se tuvo que retrasar según lo previsto inicialmente, puesto que la tarea de aprendizaje (**Tarea 2.3**) del *framework* llevó más de lo esperado. A ello hay que añadirle que se decidió incorporar una funcionalidad que no se esperaba realizar, la referente con la creación de un chat paciente - doctor y todo lo que ella conlleva, porque se consideró que esta sería la forma más intuitiva para el usuario de enviar vídeos. Esto hizo que además de retrasarse la implementación aumentase el tiempo que requiere. Es por todo ello que dentro de la fase de diseño e implementación la distribución de tiempo entre tareas no fue el esperado. Y finalmente la última funcionalidad referente al envío de vídeos no se pudo completar al cien por cien.

En la siguiente tabla 1 se muestran la planificación inicial del proyecto.

FASES (HORAS)	TAREAS
Análisis (70)	Tarea 1.1: Estudio del problema y soluciones actuales Tarea 1.2: Captura de los requisitos del sistema Tarea 1.3: Prototipo de la interfaz de usuario del sistema
Diseño (160)	Tarea 2.1: Diseño de la estructura del sistema Tarea 2.2: Diseño de la base de datos Tarea 2.3: Aprendizaje de tecnologías de implementación: Ionic 2, etc Tarea 2.4: Implementación de prototipos de la aplicación móvil
Pruebas (30)	Tarea 3.1: Pruebas de funcionamiento de la aplicación móvil Tarea 3.2: Pruebas de integración y rendimiento de la aplicación móvil
Documentación (40)	Tarea 4.1: Realización de la memoria Tarea 4.2: Realización de la presentación del TGF

Tabla 1: Tabla planificación inicial del proyecto.

6. Tecnologías y herramientas utilizadas

6.1. Tecnologías

- **Ionic:** *framework* de desarrollo para aplicaciones móviles, este es de código abierto. Está basado en varias tecnologías entre ellas HTML, Typescrip, SCCS. La ventaja de este *framework* para el desarrollo multiplataforma es el tener que desarrollar solamente una versión de la aplicación y luego esta se puede exportar a IOS, Android o Windows Phone. Dada la importancia de esta tecnología, se expondrán las razones por las que se ha decidido usar próximamemnte en esta misma sección.

- **Ionic View App:** aplicación móvil con la que poder visualizar el proyecto que estás desarrollando, para poder hacer uso de ella tienes que subir a la web de ionic tu proyecto, una vez ahí te dan un código único identificador que introduces en tu teléfono tras esto te cargará la aplicación como si la tuvieses instalada y podrás probarla de manera nativa.

- **Angular:** *framework* de *JavaScript* de código abierto mantenido por Google usado principalmente para la creación de *SPA*[15] de manera más fácil a las existentes en la actualidad, además este te permite una creación de pruebas de forma más sencilla.

- **Cordova:** entorno de desarrollo para aplicaciones móviles el cual está basado en HTML, JavaScript y CSS, facilita el empaquetado de todo esto dependiendo del dispositivo y hace que no tengas que conocer la *API*[1] de cada plataforma por separado y hacer un tratamiento individual para cada una de ellas.

- **Typescript:** lenguaje de programación basado en JavaScript el cual mejora las desventajas o problemas que pudiera tener JavaScrip añadiendo un tipado estático y clases, entre otras mejoras. Esto hace que su uso sea más cómodo y al estar basado en JavaScript si hiciera falta escribir partes de código en JavaScrip puro se puede hacer sin problemas.

- **HTML5:** lenguaje básico de elaboración de páginas web. Es un estándar con el que estructurar las distintas páginas/ventanas de nuestra aplicación. Este se considera el lenguaje más importante para el crecimiento de la *World Wide Web* (WWW), y ha sido adoptado por todos los navegadores actuales para la visualización de páginas webs.

- **SCSS:** *Sassy CSS* al igual que con *TypeScrip* es una ampliación del CSS clásico con mejoras tales como la creación de variables de manera más cómoda e intuitiva. Al ser una ampliación de CSS se puede escribir CSS clásico en él y lo interpreta correctamente.

- **Firebase:** plataforma con múltiples funcionalidades para la creación de aplicaciones móviles, en ella podemos encontrar sistemas de autenticación,

base de datos, almacenamiento de datos, etc. Se ha decidido usar esta por su fácil uso y gran cantidad de documentación de calidad de la que dispone.

- **Git:** sistema de control de versiones con el que gestionar la evolución de la aplicación guardando los estados de todos los ficheros por los que va pasando durante el desarrollo, de esta manera si hubiese algún problema se puede deshacer los cambios hasta ese punto.

6.1.1. Ionic

Ionic es la base de esta propuesta. Es por ello que a continuación se expondrán algunas de las razones por la que se ha decidido usar este *framework*.

- El código se escribe una vez y se usa en todas las plataformas, con ionic solo se tiene que escribir una vez la lógica que va a seguir y esta puede ser compilada luego en la plataforma que se requiera.
- Los componentes por defecto que trae para el desarrollo de la interfaz dan un buen acabado, sin tener que modificar la mayoría de ellos. Lo que permite ahorrar tiempo en este apartado.
- Se puede comprobar el comportamiento de manera rápida y fácil ya que permite ejecutar la aplicación en un explorador como si se tratase de un dispositivo móvil. Además incluye un apartado donde se puede ver las tres versiones de compilación y comprobar el comportamiento en cada una de ellas por separado sin tener un dispositivo donde instalar la aplicación.
- Gran cantidad de documentación oficial y no oficial así como tutoriales escritos como en vídeo, de buena calidad.
- Lo apoya una gran comunidad y de calidad, lo que hace junto al punto anterior que la búsqueda de una solución a problemas que puedan surgir sea rápida y eficaz.
- Trabaja con tecnologías modernas, con lo que su arquitectura es limpia y robusta.
- Al estar basado en lenguajes básicos y estandarizados hace que su uso resulte en ciertos puntos intuitivo.
- Lleva varios años en el mercado funcionando y tiene cientos de aplicaciones hechas con él, por lo que no es un producto tan nuevo como para que pueda tener grandes fallos.

6.2. Herramientas

- **WebStorm:** IDE para trabajar con tecnolgías web, desarrollado por la empresa *JetBrains* [10]. Cuenta con editores para HTML, JavaScript, PHP, Typescript, SCSS entre otros además de autocompletado, funciones de refactorización automática y depurador.

- **StarUML:** programa de creación de diagramas, soporta distintos tipos de estos diagramas de clases, diagramas de casos de uso, etc.
- **Github:** portal web el cual soporta el sistema de control de versiones de git, donde poder subir tu proyecto gestionado con git y ver de manera visual todo lo que ello conlleva además el proyecto puede ser público, si se quiere, pudiendo cualquier persona que lo vea sugerir mejoras a este o posibles cambios.

6.3. Servidores

- **Firebase, base de datos en tiempo real:** para el guardado de datos se ha usado una base de datos no relacional que provee *firebase* por las ventajas que esta proporciona frente a una relacional. Una de las razones por la que esta se tuvo en cuenta ha sido que permite datos variables, es decir, puedan cambiar a lo largo del tiempo. Esto se le ha dado prioridad debido a que el cliente aún no tiene claro como quiere guardar los ejercicios.

Como ya se ha mencionado anteriormente se ha decidido usar una base de datos no relacional. A continuación se expondrán las razones para ello:

- Al tratarse de datos no homogéneos en referencia con los ejercicios, pues estos pueden tener o no varios pasos, pueden tener o no material visual en forma de imágenes o vídeos. Esto ocurre debido a que no están digitalizados de forma alguna a día de hoy y el propio cliente no sabe como lo hará.
- Siguiendo con el punto anterior los propios usuarios no son consistentes. Existen dos tipos de usuarios y cada uno de ellos tiene relaciones distintas con los ejercicios y con los propios usuarios, es por ello que las bases de datos no relacionales (NoSQL) presentan una mejora respecto de las relacionales.
- Otro punto a tener en cuenta ha sido el tener que trabajar con eventos de tiempo, dejar marcas temporales en el instante en el que se ha completado un ejercicio. Esto se soluciona de forma más fácil con este tipo de base de datos.
- Por otro lado esta aplicación no presenta complejidad en peticiones a la base de datos en un primer momento, esta se enfoca más en peticiones de "grandesçantidades de datos la mayor parte del tiempo, esto hace que no se pueda aprovechar la potencia de las peticiones SQL [16].
- Una de las funcionalidades que se pedía implementar en el proyecto es un sistema de mensajería interna (chat) por el que poder comunicarse doctores con pacientes y viceversa, esta funcionalidad es más sencilla solventarla en bases de datos no relacionales.

7. Normativa y legislación

7.1. Licencia Software

Una licencia software se define como un contrato entre el licenciante (autor/titular de los derechos de explotación/distribución) y el licenciatario (usuario consumidor, profesional o empresa) del programa informático, para utilizarlo cumpliendo una serie de términos y condiciones establecidas dentro de sus cláusulas, es decir, es un conjunto de permisos que un desarrollador le puede otorgar a un usuario en los que tiene la posibilidad de distribuir, usar o modificar el producto bajo una licencia determinada. Además se suelen definir los plazos de duración, el territorio donde se aplica la licencia (ya que la licencia se soporta en las leyes particulares de cada país o región), entre otros. [11]

7.2. Licencia pública general de GNU

Esta licencia [12] existe para garantizar la libertad de compartir y modificar el software que se encuentra bajo ella. De esta forma busca que que el software sea libre para todos los usuarios. Otras acciones distintas de su copia no están cubiertas por esta licencia, tales como ejecutar el programa. Dentro del software que se usa en esta propuesta, se encuentra bajo esta licencia: Git.

7.3. Licencia comercial

La licencia comercial también conocida como software propietario, es aquella que se vende bajo unas normas detalladas y condiciones de uso específicas. Estas licencias tienen como objetivo omitir el acceso de forma libre a su código fuente, el cual se encuentra a dispocición de su desarrollador. Además no se permite que sea modificado. Bajo esta licencia se encuentra StarUML y Firebase.

En el caso de firebase es gratuito su uso hasta cierta cantidad de peticiones mensuales, una vez se sobrepasa este límite tiene planes de uso según las necesidades del proyecto en cuestión.

7.4. MIT

Esta licencia es bastante flexible, por lo que tiene pocas limitaciones a la hora de su reutilización esto lo hace una licencia con una muy buena compatibilidad. La licencia MIT permite la reutilización dentro de software propietario y es compatible con GNU, por ejemplo. Con esta licencia nos encontramos en el proyecto a Ionic y Angular.

7.5. Licencia para educación JetBrains

La licencia de JetBrains para educación es de formato de suscripción mensual o anual, dependiendo de como el usuario quiera realizar el pago. Esta es gratuita para los miembros de la comunidad universitaria, sin embargo tiene algunas restricciones como puede ser el uso para propósito comercial, el uso para

ingeniería inversa, modificar o descubrir el código fuente de los productos. Con esta licencia se encuentra el entorno de desarrollo WebStorm.

7.6. Reglamento General de Protección de Datos

El RGPD es el nuevo reglamento que entró en vigor en mayo de 2016, de obligada aplicación en todas las empresas de la Unión Eurpea desde el 25 de mayo de 2018. Esta otorga mayor control y seguridad a los usuarios sobre su información personal ampliando sus derechos a decidir cómo quieren que sus datos sean tratados y cómo quieren recibir la informaciójn de las empresas.

Entre los puntos de este reglamento se encuentra el derecho al olvido. Este derecho lo pueden solicitar los usuarios y una vez se aplica los datos personales deben de ser suprimidos cuando ya no sean necesarios para el fin con el que fueron reunidos. Otro de los puntos es el derecho a la portabilidad. Este implica que los datos de los que dispone el sistema tienen que ser extraibles de este por el usuario para que permita el trasnlado a otro si así lo quisiera. El Reglamento pide que el consentimiento, con carácter general, sea libre, informado, específico e inequívoco. Las empresas deberán revisar la forma en la que obtienen y guardan el consentimiento. Se exige que el consentimiento tenga que ser "manifiesto" en determinados casos, como puede ser para autorizar el tratamiento de datos sensibles. Por tanto, el consentimiento tiene que ser verificable y quienes recopilen datos personales deben poder probar que el afectado les concedió su consentimiento. Es por todo esto que para que la aplicación cumpla con la ley se debe asegurar los datos personales tales como: nombre, apellidos, teléfono, etc. Estos datos se encuentran en la base de datos protegidos por una contraseña a la cual sólo tendrá acceso el administrador de la misma.

8. Análisis

8.1. Actores

Se han identificado dos actores principales los cuales se expondrán a continuación.

8.1.1. Doctor

Personal del centro médico el cual tendrá pacientes a su cargo, sus funciones serán las siguientes:

- Buscar pacientes entre los registrados en el sistema.
- Asignar ejercicios a un paciente.
- Añadir observaciones a los ejercicios que se van a asignar.
- Ver perfil de un paciente.
- Ver evolución de un paciente.

Toda funcionalidad tienen como objetivo que el doctor pueda indicarle a sus pacientes los ejercicios que este tiene que realizar y si en su caso concreto tendrá que tener en cuenta alguna variación del original dejarlo indicado en las observaciones. Además tendrá que poder ver el perfil del paciente para poder consultar sus datos, en caso de que le fueran necesarios.

8.1.2. Paciente

Otro actor que se ha identificado es el paciente con la siguiente funcionalidad:

- Ver ejercicio.
- Marcar ejercicio como hecho.

Con esta funcionalidad se busca que el paciente pueda llevar un control de los ejercicios que ha realizado y esto pueda ser consultado por el doctor en cualquier momento.

Además estos actores principales tendrán funcionalidades **comunes:**

- Registro en la aplicación.
- Inicio de sesión.
- Ver perfil.
- Editar perfil.
- Iniciar chat.
- Enviar y visualizar vídeos por el chat.

Estas funciones son necesarias que estén disponibles en ambos actores. Debido a que el chat es tanto doctor con paciente, como a la inversa. Además, los usuarios tienen que poder darse de alta en el sistema.

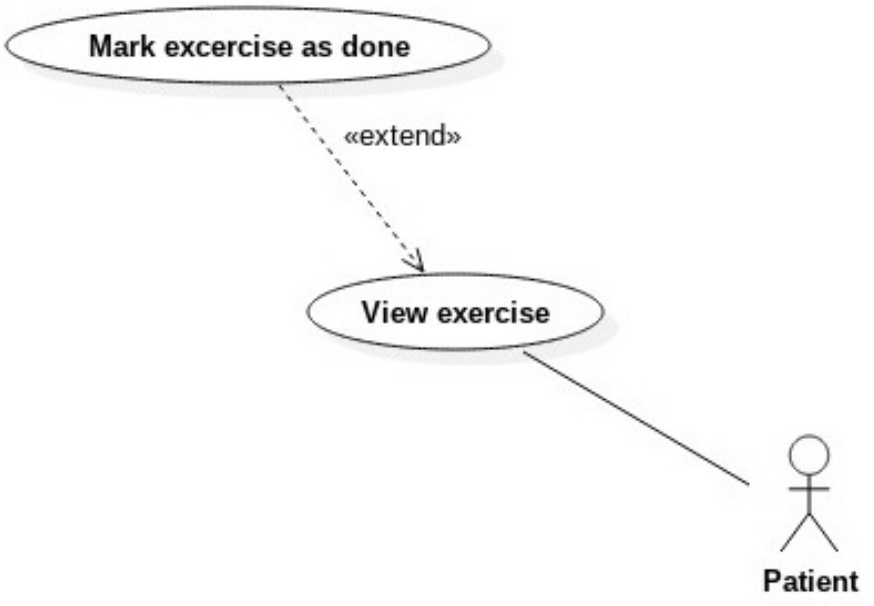

Figura 2: Diagrama casos de uso paciente.

9. Requisitos

Para representar los requisitos de la aplicación se ha usado UML [18] por varios motivos que se expondrán a continuación. El primero de ellos por estar familiarizados con este, al haber sido usado durante el grado en múltiples ocasiones en asignaturas distintas. Por otra parte este lenguaje de modelado es un estándar aprobado por la ISO lo que hace que cualquier diagrama creado pueda ser interpretado de igual forma por diferentes personas que conozcan el estándar.

A continuación se muestran los diagramas UML con los que se han representado los requisitos de los distintos actores de la aplicación. Solo se ha querido representar los requisitos que se quieren implementar en esta fase para no cargar con más información de la necesaria hasta el momento. En el apartado mejoras futuras se nombrarán el resto de ellos.

En la figura 2 se muestran las funcionalidades que el paciente tiene disponible. Como se puede ver este dispondrá de la posibilidad de ver los ejercicios y una vez complete estos podrá indicar que se han completado.

Como podemos observar en la figura 3 el doctor tendrá a su disposición la posibilidad de realizar una búsqueda entre los pacientes, ver su perfil, asignar ejercicios y de manera opcional añadirle observaciones a los ejercicios que asigna. Por último, podrá evaluar la evolución del paciente.

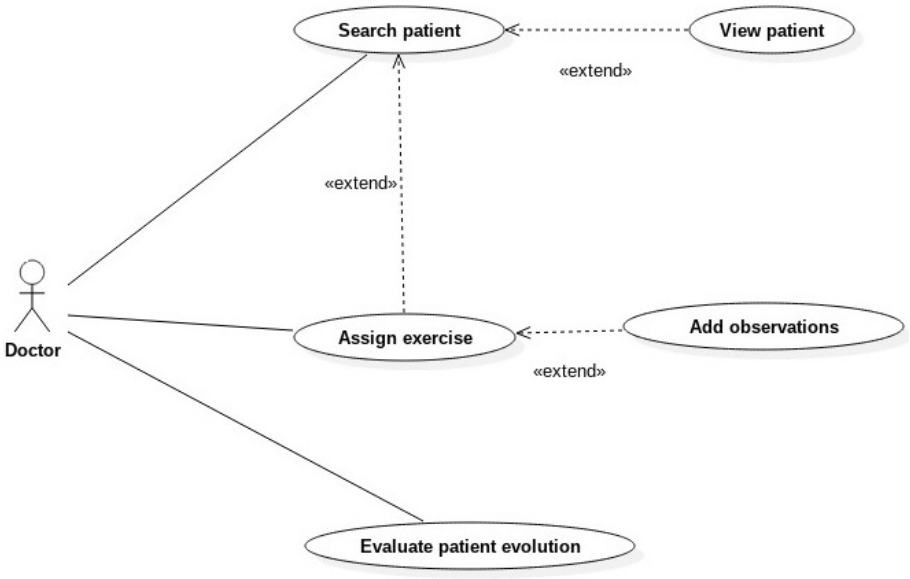

Figura 3: Diagrama casos de uso doctor.

Tal como se muestra en la figura 4 de manera genérica todos los usuarios podrán darse de alta e iniciar sesión en el sistema y ver su propio perfil. Además de esto, podrán iniciar un chat, enviar mensajes por este, ver los mensajes que se han recibido y enviar vídeos.

En la siguiente tabla 2 se muestran los casos de uso anteriores con una breve descripción.

ID	ACTOR	DESCRIPCIÓN
1	Usuario	Puede registrarse en la aplicación
2	Usuario	Puede iniciar sesión en la aplicaicón
3	Usuario	Puede cerrar la sesión
4	Usuario	Puede crear un chat con otra persona
5	Usuario	Puede ver el chat ya creado o que otro usuario ha creado con él
6	Usuario	Puede enviar mensajes por el chat que ha creado u otro usuario ha creado co él
7	Usuario	Puede enviar un vídeo en formato mp4 por el chat
8	Usuario	Puede ver el vídeo que le han enviado
9	Doctor	Buscar pacientes en el sistema para asignarle ejercicios o consultar su información
10	Doctor	Puede ver el perfil de un paciente que ha buscado
11	Doctor	Puede asignar un ejercicio a un paciente para que este le quede reflejadoq que tiene que hacerlo
12	Doctor	En el momento de asignar un ejercicio puede añadir observaciones si lo considera el doctor
13	Doctor	Puede consultar la evolución del paciente mirando los ejercicios que ha realizado y con que frecuencia
14	Paciente	Puede ver el ejercicio que se le ha asignado para consultar las observaciones o el ejercicio en sí
15	Paciente	Puede marcar como completado un ejercicio que le ha sido asignado

Tabla 2: Tabla descriptiva de los distintos requisitos.

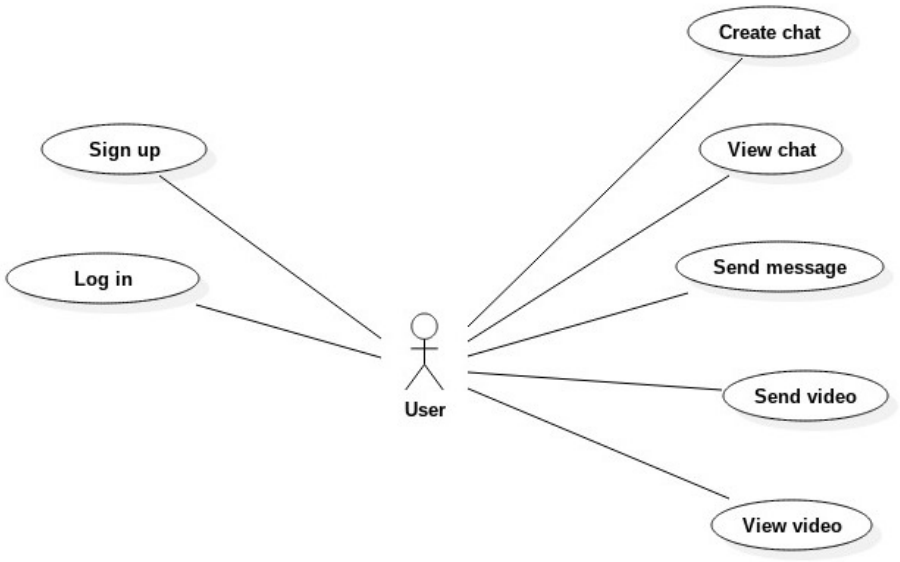

Figura 4: Diagrama casos de uso usuarios.

10. Diseño

10.1. Diseño de la arquitectura del sistema

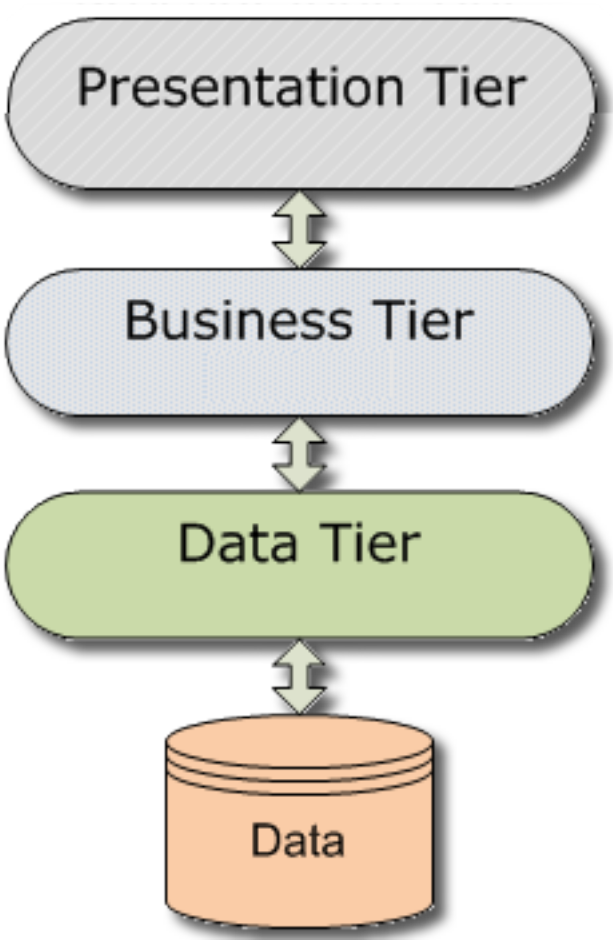

Figura 5: Arquitectura multinivel

Se ha usado una arquitectura multinivel partiendo de la base de datos que nos provee *firebase* a la cual se conecta la aplicación a través de su API. Esta API se consume desde la aplicación con los proveedores de los que dispone Ionic, para posteriormente mostrar la información resultante en la capa de presentación. El problema ha sido a la hora de añadir nueva lógica referente a la aplicación. Esta API no permite ser modificada y se ha tenido que incluir cierta lógica de negocio

en la propia aplicación. Es por ello que para el futuro si se quiere implementar este prototipo y que sea escalable, se tendría que desarrollar una API propia donde alojar toda la lógica de negocio.

Con ello se aprovecharía de mejor forma las ventajas de esta arquitectura. Como puede ser la fácil escalabilidad debido a que pueden haber varios clientes consumiendo de la misma lógica. Y en caso de que la lógica cambie, no tenga un mantenimiento complejo debido a que se encuentra toda en un lugar centralizado. Por otro lado disponer de esta API supondría un ahorro de recursos en el cliente, porque este no tendría que hacer cálculos referentes a la lógica del negocio.

10.2. Diseño de la base de datos

Para el registro de usuarios se ha usado el módulo de autenticación del que dispone esta misma plataforma, el cual permite el registro de usuarios por múltiples vías si así lo quisiéramos, guardan de forma segura las contraseñas e incorpora un sistema de verificación de correo en el momento del registro. Es por ello que se ha tenido que crear un objeto en la base de datos, a parte del existente, para incluir datos personales que usando este módulo no nos permitían guardar. Para ello se ha creado un grupo llamado *users* donde crear objetos de nombre, el identificador único, generados en el momento del registro de cada usuario y dentro de este el resto de datos que nos han sido necesarios.

Como se puede ver en la **figura 6**, los usuarios comparten parámetros comunes de los cuales solamente los que se explicarán a continuación pueden generar duda de su utilidad o uso que se le ha podido dar. Es el caso de *city*, hace referencia a la ciudad de nacimiento del usuario y por otro lado el trío *day-month-year* los cuales forman la fecha de nacimiento del usuario en cuestión. En este caso el usuario tiene asignados cinco ejercicios dentro del objeto *exercise* a los cuales se les referencia"guardando como clave su ID. Dentro de este objeto cada ejercicio tiene un campo observaciones (*observatios*) y un campo completado (*done*) donde se guarda la fecha y hora en la que se completó un ejercicio.

En caso de ser un usuario de tipo doctor este tendrá un objeto en el que guarda los identificadores de sus pacientes asignados. Como se puede ver en la **figura 7** estos pacientes tienen asignados una fecha y hora, correspondiente a la última hora en la que se visitó su perfil.

Para no hacer grandes cantidades de peticiones sin necesidad se ha creado un objeto *chats*, el cual se puede ver en la **figura 8**. Para explicar mejor como están estructurados estos datos se pondrá un ejemplo. Si un usuario **A** tiene un chat iniciado con un usuario **B** en la base de datos en la parte de chat aparecerá la siguiente información:

chats/ ID-usuario-A/ [ID-usuario-B: ID-chat]

chats/ ID-usuario-B/ [ID-usuario-A: ID-chat]

Figura 6: Estructura en la base de datos del objeto: usuarios (paciente)

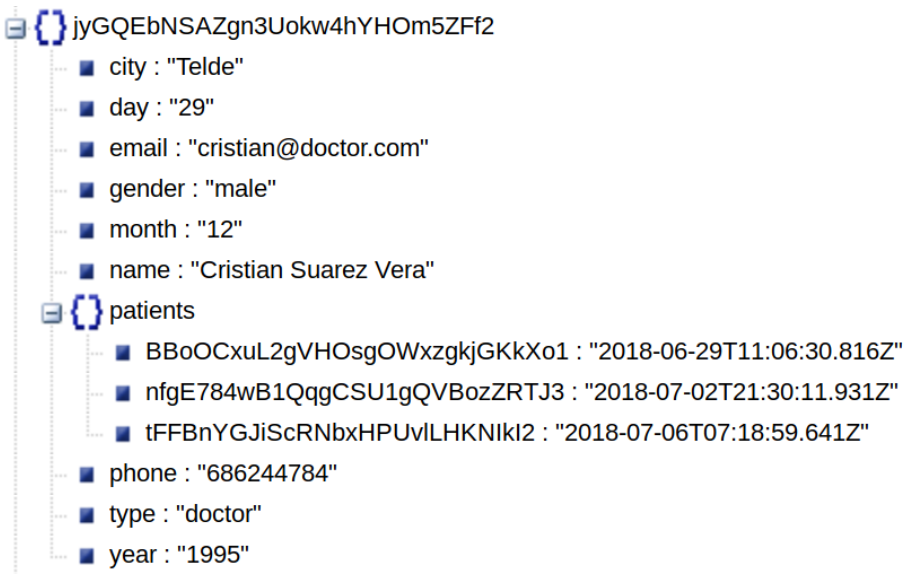

Figura 7: Estructura en la base de datos del objeto: usuarios (doctor)

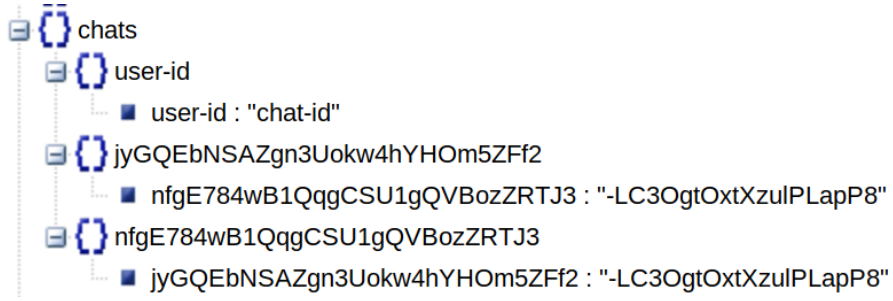

Figura 8: Estructura en la base de datos del objeto: chats

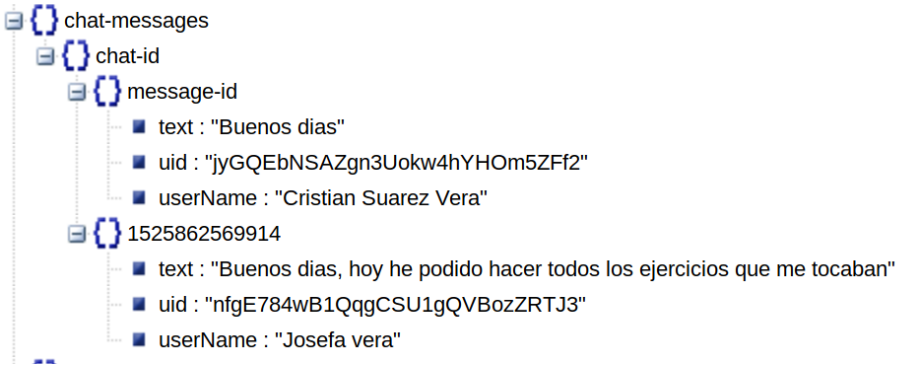

Figura 9: Estructura en la base de datos del objeto: mensajes del chat

En caso de que un tercer usuario **C** inicie un chat con **A** aparecerá una nueva entrada y quedará de la siguiente forma la base de datos:

chats/ ID-usuario-A/ [ID-usuario-B: ID-chat], [ID-usuario-C: ID-chat]

chats/ID-usuario-B/[ID-usuario-A: ID-chat]

chats/ID-usuario-C/[ID-usuario-A: ID-chat]

Como se ha dicho esto se ha realizado de esta forma para ahorrar tener que traer todos los mensajes del chat en caso de que un usuario solo quiera visualizar con que personas tiene un chat abierto, de esta forma solo hará una petición al objeto con su identificador y este le devolverá los ID de los usuarios con los que mantiene la conversación y el ID del chat en cuestión si quiere obtener los mensajes.

A raíz de la implementación anterior se ha tenido que crear el objeto que muestra la **figura 9**, este tiene como objetivo almacenar todos los mensajes de los chats entre los distintos usuarios. Cada chat tiene su identificador único con el que se crea el objeto y dentro de este un identificador de mensaje el cual internamente tiene el mensaje que ha sido enviado, el identificador del usuario que ha enviado el mensaje y el nombre completo del usuario que ha enviado el mensaje. Este último parámetro puede parecer redundante pero nos ahorra posteriormente, a la hora de mostrar los mensajes, tener que hacer una petición al servidor para obtener este dato.

Por último en la **figura 10** se ve como se ha creado el objeto para los ejercicios. En él se almacena la información referente a cada ejercicio como puede ser la edad recomendada del ejercicio, la descripción del mismo, el tiempo medio que se tarda en realizarlo y el título que este tiene. Además todo ello está bajo un objeto que tiene como nombre un identificador único para el ejercicio.

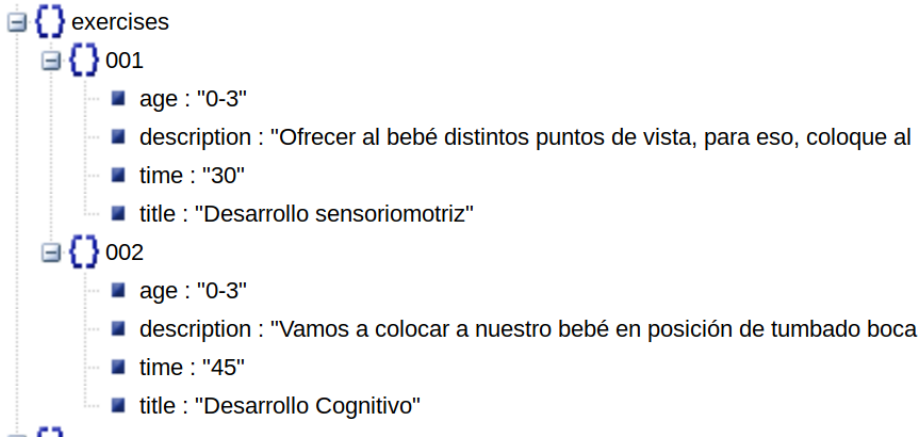

Figura 10: Estructura en la base de datos del objeto: ejercicios

10.3. Almacenamiento de contenido multimedia

Para almacenar todo el contenido multimedia referente a los ejercicios se ha decidido usar *firebsae storage*. Un servicio que nos permite guardar todas las ilustraciones y vídeos referentes al prototipo y acceder a ellos en todo momento. Este almacenamiento se ha organizado de tal forma que disponemos de un directorio principal y dentro de este existen directorios con el nombre del ejercicio al que corresponde el contenido. Cada uno de ellos tiene en su interior todo el contenido multimedia referente al ejercicio.

10.4. Diseño arquitectónico

En este apartado se ha usado el diseño propio del que dispone Ionic. Este consta de vista y proveedor. Los proveedores acceden a la información disponible en el servido, le dan formato y la envían a la vista. En este caso en estos proveedores contienen algo de lógica del negocio. Esto ocurre porque no se ha podido modificar la API que ofrece firebase. La vista recibirá la información preparada por los proveedores y la mostrará al usuario para que haga uso de ella.

```
constructor(platform: Platform, statusBar: StatusBar,splashScreen: SplashScreen,
            private auth: UserProvider) {
  platform.ready().then( onfulfilled: () => {
    this.checkSessionAndSetRootPage();
    statusBar.styleDefault();
    splashScreen.hide();
  });
}

private checkSessionAndSetRootPage() {
  this.auth.session.subscribe( next: session => {
    if (session) {
      this.rootPage = TabsPage;
    } else {
      this.rootPage = LoginPage;
    }
  });
}
```

Figura 11: Fichero app.components.ts, control de sesión.

11. Desarrollo

11.1. Estructura del proyecto

Uno de los puntos buenos que tiene este *framework* es poseer una estructura de directorios bien conocida y definida, por ello será fácil navegar por ella incluso para nuevos desarrolladores que se incorporasen al proyecto.

Los principales directorios son los siguientes:

- /src/app/: ficheros de configuración de la aplicación.
- /src/assets/: ficheros multimedia y recursos estáticos.
- /src/enviroments/: ficheros de configuración para los distintos entornos (desarrollo o producción)
- /src/pages/: directorio donde van ubicados los directorios de cada pandalla o componente de la aplicación.
- /src/pages/page1/: ficheros referentes a una pantalla en concreto en el van el HTML, TS, SCSS (si fuese necesario)
- /src/providers/: ficheros de acceso a los datos.
- /src/theme/: fichero de estilo genérico en el cual se basa la aplicación.

Como se ha indicado anteriormente en el proyecto existen distintos tipos de ficheros según su objetivo a cumplir, obtener información, dar formato a los datos para mostrar los mismos, configuración, etc. A continuación se pasará a mostrar algunos extractos que se han considerado relevantes.

```
haveAChat(userId, userId2) {
  return new Promise( executor: resolve => {
    this.chats.child(userId)
      .on('value', (chatList) => {
        if (this.isNotDefined(chatList)) {
          resolve(false);
        } else {
          resolve(this.isUserInChatList(chatList, userId2));
        }
      });
  });
}

private isNotDefined(snapshot) {
  return snapshot.val() == null;
}

private isUserInChatList(snapshot, userId2) {
  return snapshot.val()[userId2] != undefined;
}
```

Figura 12: Fichero chats.ts, función haveAChat.

En la **figura 11** se puede ver como se gestiona el control de sesiones. Con ello la aplicación no permite el uso de ninguna funcionalidad si el usuario no ha iniciado sesión. Como se muestra si el usuario no ha iniciado la sesión se le enviará a la *LoginPage*. En esta se permite el inicio de sesión y da acceso al registro para el caso de usuarios que no se han dado de alta en el sistema. En caso de que si que exista una sesión por parte del usuario a este se le enviará a la página principal de la aplicación *TabsPage*. En ella se encuentran las distintas ventanas disponibles separadas por una barra con pestañas.

En esta **figura 10** se quiere mostrar la importancia que tienen los nombres en las funciones y la importancia que se le ha dado a estos durante todo el desarrollo. Se puede ver como con un simple vistazo cual es el objetivo de esta función, saber si un usuario tiene un chat abierto con otro. Una vez dentro de esta se puede casi leer como si de una frase se tratase, .^{es}te chat tiene un hijo con este id de usuario en la lista de chats". Si se continúa leyendo a partir de la línea cinco se puede leer "si no está definido se resuelve que no tiene un chat y en caso contrario se comprueba si el usuario está en esta lista y se transmite"

Gracias a que se le ha dado buenos nombres a las funciones y parámetros. Y tratarse de funciones cortas hace que este código sea más legible y entendible sin tener que realizar grandes esfuerzos. Estas son algunas de las reglas de código limpio [6] que se han tratado de seguir en este proyecto. Para que el código sea mantenible de sin tener que utilizar mucho tiempo en comprender lo que se hace en cada parte. Estas reglas se han considerado de vital importancia pues

Figura 13: Fichero view-chat.ts, función onViewDidEnter.

se quiere una aplicación longeva que sea fácilmente sostenible y ampliable, esta tiene que ser en un primer momento, al menos, entendible por los desarrolladores que se encarguen del mantenimiento de dicho código.

Otra de las leyes o normas que se cumple como consecuencia de usar buenos nombres, consiste en usar el mínimo posible de comentarios. Al estar bien nombradas estas funciones, no es necesario añadirles un comentario explicativo sobre la misma. Es por ello que los comentarios pierden valor.

Por último nombrar un principio más que se puede ver en este código. Se trata del principio de responsabilidad única, el cual dice que una función debe cumplir con un único objetivo. Por ejemplo, no debe llamarse *tieneUnChat* e internamente comprobar si tiene un chat y crear uno en caso de que no lo tenga. Si fuera así estaría mintiendo con el nombre y se podría hacer un mal uso de esta función.

Para mejorar la experiencia del usuario se ha tenido en cuenta la carga de datos. Para ello se ha hecho uso de las funciones que nos da ionic, como son *ionViewDidEnter* y *ionViewWillLeave*. Las cuales se ejecutan cuando el usuario abre une nueva ventana y cuando sale de ella, respectivamente. Es por ello que se realiza la carga de datos por parte del servidor mientras la pantalla se muestra. De esta manera al no hacerlo en el momento que la pantalla ya está visible por el usuario, este no tendrá que esperar por los datos en una pantalla en blanco. Debido a que los datos han sido solicitados al servidor en el momento que se empieza a preparar la pantalla para mostrarse. De igual forma este no tendrá que esperar a que se cierre la conexión con la base de datos para cerrar una pantalla. Dado que en el momento que se comienza la "destrucción"de la misma el sistema laza la desconexión de manera asíncrona.

12. Conclusiones y trabajos futuros

12.1. Conclusiones

En términos generales los objetivos con los que se inició el proyecto han sido cumplidos, logrando una primera aproximación funcional para la problemática planteada. Teniendo en cuenta el poco margen de tiempo del que se dispone en este tipo de proyectos, se ha logrado una buena primera versión que esperamos que pueda seguir evolucionando y mejorando con el tiempo.

La participación y realización de este proyecto ha resultado una experiencia enriquecedora y formativa. Se ha podido reforzar los conocimientos adquiridos en la carrera tanto genéricos como específicos de la Ingeniería del Software. Además de obtener nuevos referentes a las tecnologías que se han decidido usar como Ionic, TypeScript, Firebase, entre otras.

Al tratarse de un proyecto que se ha iniciado desde cero hasta el final del mismo ha sido una experiencia nueva, al no haber hecho antes tal proceso al completo. Si es cierto que se conocían todas sus fases por separado, pero el poder aplicarlas juntas ha resultado una experiencia muy positiva.

Además, haber buscado una solución a un problema existente y haber sido capaz de aprender una tecnología nueva, me ha hecho ver que estos años de universidad no solamente se han adquirido conocimientos técnicos. También ha hecho que adquiera una forma de pensar con la que poder buscar soluciones a problemas y una mentalidad abierta para aprender nuevas cosas ya sean tecnologías o metodologías de forma más o menos rápida y fácil.

En términos generales este proyecto ha sido una experiencia muy gratificante. A pesar de sus cambios en medio del desarrollo, sus horas de aprendiendizaje de nuevas herramienta en las que parecía que no se avanzaba con el proyecto, las entrevistas con el cliente y con todo lo ocurrido se puede decir que ha sido una muy buena experiencia y se ha logrado obtener un resultado cercano al esperado.

12.2. Mejoras futuras

A continuación se describirán mejoras a la funcionalidad implementada y nuevas funcionalidades que se podrían incorporar, de cara a completar este proyecto.

- Enviar vídeos por el chat, funcionalidad que se tuvo que dejar por falta de tiempo y es necesaria para el producto mínimo viable.
- Permitir usuarios de tipo administrador, el cual pueda añadir ejercicios a la base de datos, modificar o eliminar los mismos.
- Añadir evaluación de la evolución del paciente, a día de hoy solo se puede ver cuando ha realizado el ejercicio.

- Enviar esta evaluación por el chat al paciente para que sea notificado.
- Permitir marcar ejercicio como hecho sin tener que entrar en la vista del ejercicio.
- Implementar un sistema de notificaciones si un ejercicio se ha marcado con una periodicidad diaria y hace más de 20 horas que no se ha indicado que se ha realizado.
- En el momento del registro permitir más localidades de nacimiento, no solamente municipios de Gran Canaria.
- Añadir un servicio de gestión de citas, donde el médico pueda indicar que tiene una cita con un paciente y este pueda ver en su teléfono cuando será la siguiente cita.
- Poder notificar tanto paciente como doctor que no podrá asistir a la cita.
- Notificación cuando la fecha de la cita prevista se acerca.
- Clasificar ejercicios por categorías, para ayudar a la búsqueda una vez se tenga una gran cantidad de ellos.
- Mejorar la estética de la interfaz de usuario, como puede ser, añadir un logotipo de la aplicación.

Referencias

[1] *Application Programing Interface o interfaz de programación de aplicaciones*. 2018. URL: https://es.wikipedia.org/wiki/Interfaz_de_programaci%C3%B3n_de_aplicaciones.

[2] Ravulavaru Arvind. *Learning ionic : buid hybrid mobile applications with HTML5, SCSS, and Angular*. 2017. ISBN: 978-1-78646-605-1.

[3] *BilliCam, app móvil para el diagnóstico de la ictericia neonata*. URL: https://homes.cs.washington.edu/~mernst/advice/poster-examples/deGreef-billicam.pdf.

[4] *Casos de uso*. Mayo de 2018. URL: https://es.wikipedia.org/wiki/Caso_de_uso.

[5] *Cibersalud, informe de la Secretaría*. Abr. de 2005. URL: http://apps.who.int/gb/archive/pdf_files/WHA58/A58_21-sp.pdf?ua=1.

[6] *Clean code o Código Limpio, filosofía de programación*. URL: https://www.genbetadev.com/metodologias-de-programacion/12-ideas-de-la-filosofia-clean-que-no-pueden-faltar-en-tu-codigo.

[7] *Connect Physiotherapy, the largest specialist provider of musculoskeletal services in the UK*. URL: https://www.connecthealth.co.uk/.

[8] *Core Fusion Pilates and Physiotherapy*.

[9] *Ionic framework, build amazing apps in one codebase, for any platform, with the web*. 2018. URL: https://ionicframework.com/.

[10] *JetBrains, the drive to develop Keep Evolving*. URL: www.jetbrains.com.

[11] *Licencia de software*.

[12] *Licencia pública general reducida de GNU*.

[13] *Organización Mundial de la Salud*. Feb. de 2018. URL: http://www.who.int/es.

[14] *Secure Patient Portal by ConstantMD by eMedical Companion Inc.*

[15] *SPA, Single Page Aplication o Aplicación de Página Única*. 2018. URL: https://es.wikipedia.org/wiki/Single-page_application.

[16] *Structured Query Language o lenguaje de consulta estructurada*. URL: https://es.wikipedia.org/wiki/SQL.

[17] *Tabla de especificación de casos de uso*. URL: http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/416.

[18] *Unified Modeling Language o lenguaje unificado de modelado*. 2018. URL: https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado.

13. Manual de usuario

13.1. Introducción

Este manual ha sido escrito con la finalidad de explicar clara y sencillamente las funcionalidades disponibles en la aplicación. Para ello se hará uso de capturas de pantalla de la propia aplicación acompañadas siempre de un breve texto descriptivo.

13.2. Registro

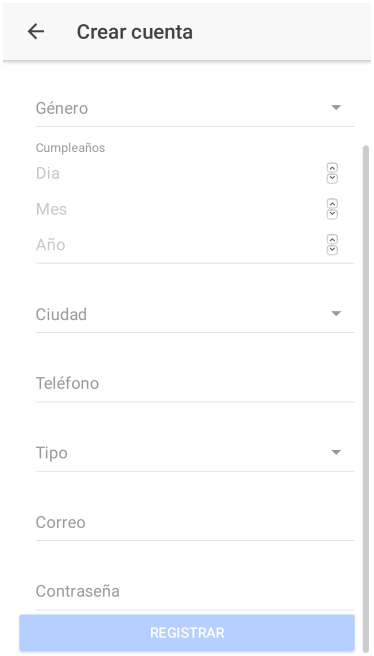

Figura 14: Formulario de registro

Antes de hacer uso de las funcionalidades de las que dispone la aplicación los usuarios han de darse de alta. Para ello existe una ventana en la aplicación en la que completar los datos del registro, como se puede ver en la figura 14.

13.3. Inicio de sesión

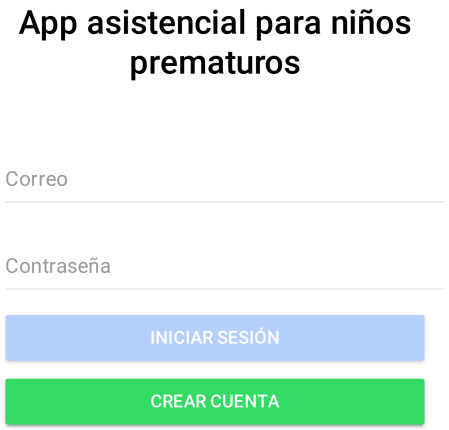

Figura 15: Formulario inicio de sesión

Para poder acceder a la aplicación el usuario tiene que haber iniciado su sesión previamente. Es por ello que la primera ventana que se muestra, en caso de no estar en una sesión activa, es el formulario de inicio de sesión que se puede ver en la figura 15.

13.4. Barra de navegación

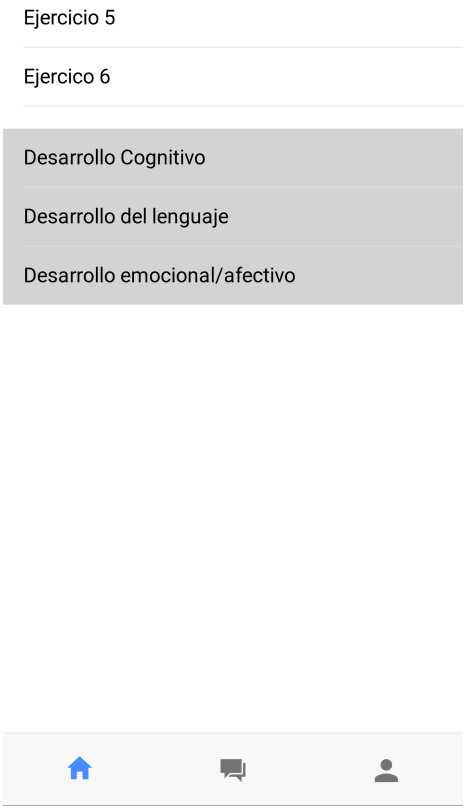

Figura 16: Pantalla principal, paciente

En la figura 16 se puede ver en la parte inferior una barra de navegación. Esta barra está disponible en todas las vistas de la aplicación. Con ella el usuario podrá navegar entre las distintas opciones disponibles. Estas están indicadas con iconos representativos de las mismas. Como son: página principal (*home*), chats y mi perfil, respectivamente. En el caso de los doctores, estos tienen una pestaña adicional con el icono de una lupa donde podrán buscar entre los pacientes registrados en la aplicación.

Además en esta pantalla el usuario pacietne podrá ver los ejercicios que tiene asignados. Estos, estarán ordenados según estén completados o no. Los ejercicios que ya se han marcado como completados estarán al final de la lista en un color gris. Si el usuario pulsa sobre cualquiera de los ejercicios podrá ver el material disponible para dicho ejercicio.

13.5. Vista ejercicios, paciente

Figura 17: Vista ejercicio con observaciones, paciente

Los pacientes al entrar a ver uno de los ejercicios que tiene asignados verá la pantalla que se muestra en la figura 17. En esta podrá ver las especificaciones del ejercicio tales como la descripción, las observaciones que el doctor puede haberle dejado indicadas y el material visual que se encuentre disponible. Como pueden ser imágenes o vídeos. Además de ello encontrará al final de esta un botón con el que podrá indicar si el ejercicio ha sido completado.

13.6. Perfil de usuario

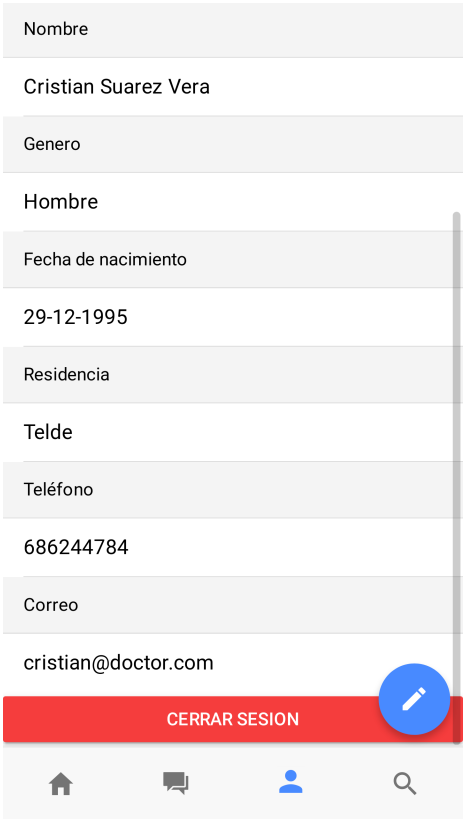

Figura 18: Perfil de usuario

Todos los usuarios tendrán disponible una vista como se muestra en la figura 18 en la que poder ver sus datos personales. Esta vista además al final incluye un botón con el que pueden cerrar la sesión, si así lo quisieran.

13.7. Pantalla principal, doctor

Figura 19: Pantalla principal, doctor

El doctor, cuando inicia la aplicación verá los pacientes que tiene asignados. Estos tendrán una marca a la derecha de su nombre si han completado ejercicios desde la última vez que el doctor ha revisado su historial. De esta forma podrá saber si hay novedades entre sus pacientes, como se puede ver en la figura 19.

13.8. Vista ejercicios, doctor

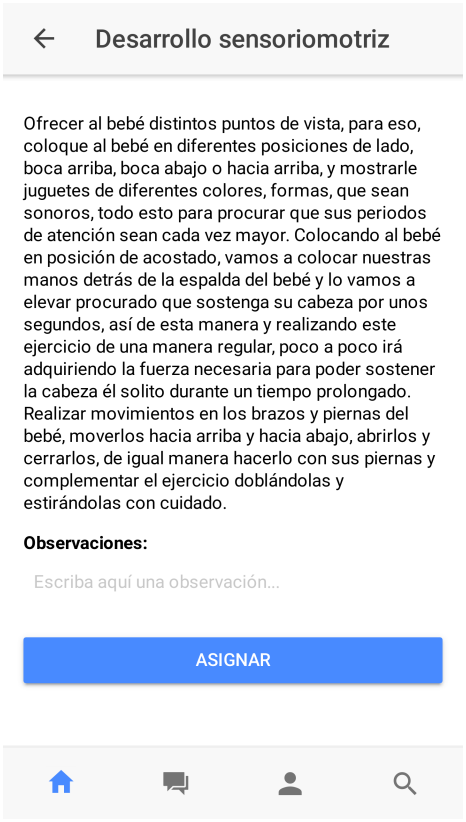

Figura 20: Asignar ejercicio con observaciones, doctor

El doctor, una vez entra en la vista del ejercicio (figura 20) que quiere asignar a un paciente tendrá a su disposición un campo que puede completar o no con las observaciones que quiera añadir. Para completar la asignación tendrá que pulsar en el botón situado al final de la pantalla.

13.9. Vista perfil paciente

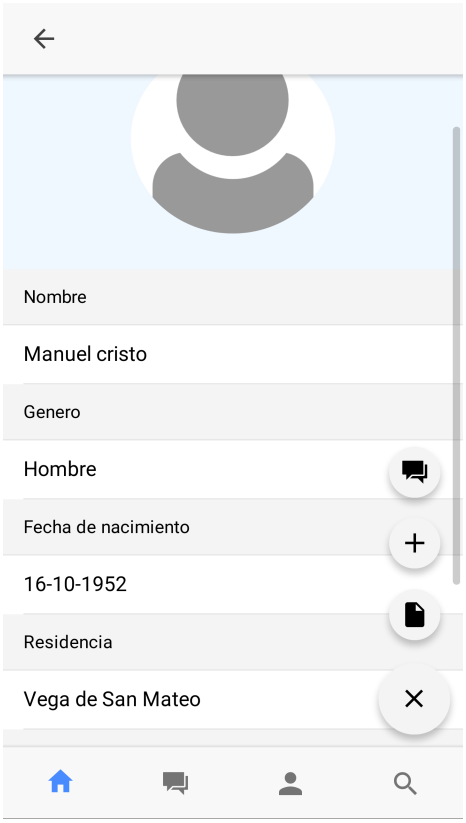

Figura 21: Vista perfil paciente

Si el doctor entra en el perfil de un paciente podrá pulsar sobre el botón -"situado en la parte inferior derecha en todo momento. Al pulsar sobre ella verá las opciones que se muestran en la figura 21. Entre las opciones se encuentran: iniciar un chat, asignar ejercicios y ver el historial.

13.10. Ver historial del paciente

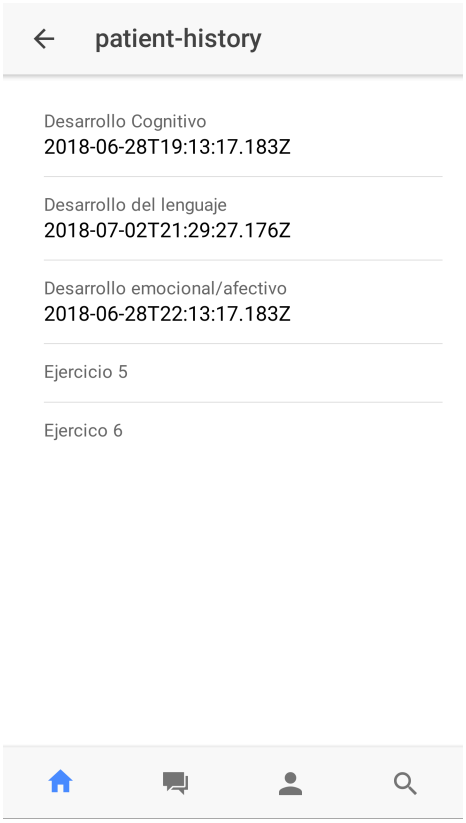

Figura 22: Historial del paciente

En la figura 24 se puede ver el historial de ejercicios de un paciente con la fecha y hora en la que se ha completado cada ejercicio.

13.11. Asignar ejercicios

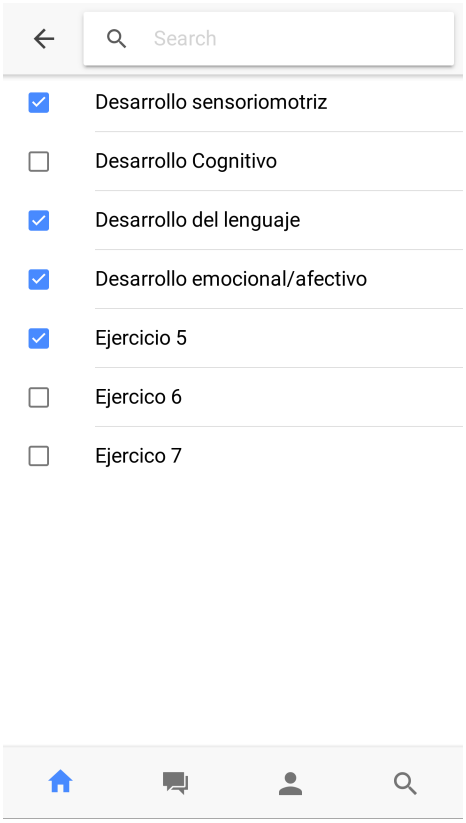

Figura 23: Asignar ejercicios

En la vista que se muestra en la figura 24 el doctor podrá ver de un vistazo los ejecicios que tiene el paciente asignados o no. Si quiere asignar un ejercicio sin observaciones podrá pulsar sobre el *checkbox*, de igual forma para desasignarlo. Si en lugar de ello pulsa sobre el ejercicio verá todo el material disponible del ejercicio y un campo observaciones que podrá completar antes de asignar el ejercicio.

13.12. Lista de chats

Figura 24: Chats abiertos

Todos los usuarios tendrán disponible una pantalla donde poder ver los chats que tiene activos en ese momento, tal como se muestra en la figura 24.

13.13. Mensajes de chat

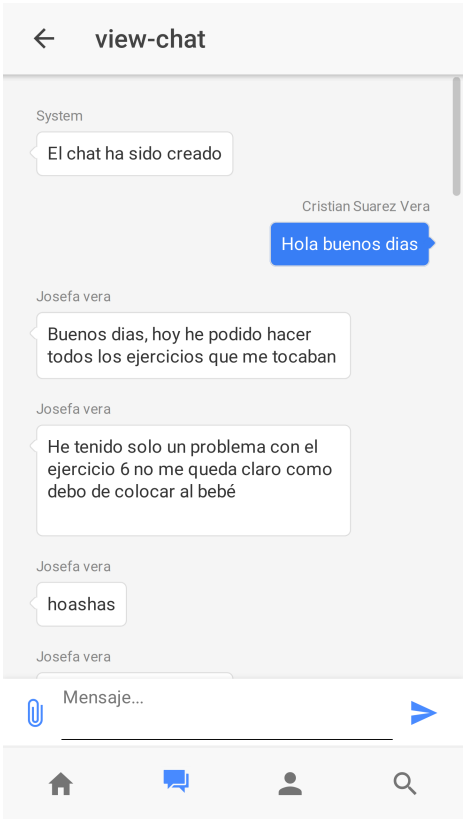

Figura 25: Mensajes de un chat

En la pantalla para el visionado de los mensajes del chat (figura 25) los usuarios podrán ver la conversación que mantiene con otro usuario. Además, se le permite enviar mensajes a este completando el recuadro del final de la pantalla y pulsando sobre el botón con forma de triángulo tumbado a la derecha.

13.14. Busqueda paciente

Como se muestra en la figura 26 el doctor podrá buscar pacientes. Para ello tendrá que escribir en la barra de búsqueda superior su nombre y la lista mostrará los disponibles.

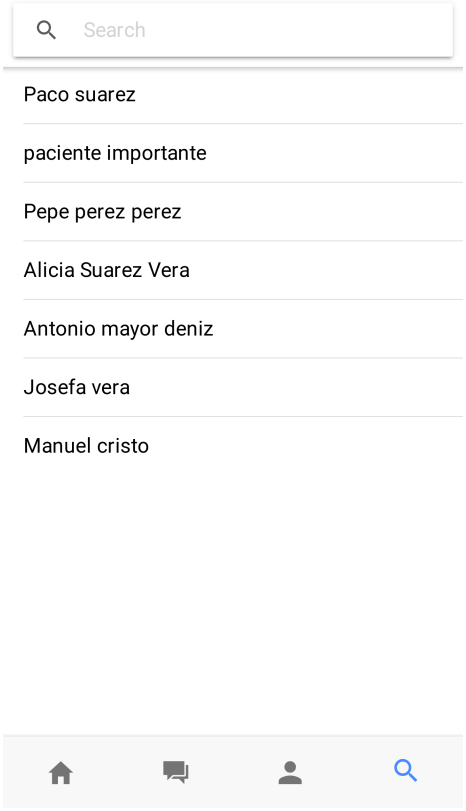

Figura 26: Buscar paciente