Escuela Politécnica Superior

Fitbot,

Aplicación móvil para mantenerte en forma con un asistente-compañero

Grado en Ingeniería Multimedia

Trabajo Fin de Grado

Autor:
Ramón Morcillo Cascales

Tutor/es:
José Vicente Berná Martínez

Universitat d'Alacant
Universidad de Alicante

Resumen

El siguiente proyecto trata de una aplicación móvil enfocada en llevar una vida sana realizando actividades físicas junto con el apoyo de un compañero.

Cada día más personas sufren de obesidad y otros problemas graves causados por el sedentarismo y por la falta de ejercicio físico en su vida. Fitbot pretende solventar este problema animando al usuario a realizar actividades físicas y a llevar un control de su peso y de su estado de salud. Además, cuenta con un chatbot conversacional integrado el cual cumple tres funciones:

- Asistir y ayudar al usuario en los procesos de la aplicación.
- Proponer actividades y ejercicios a realizar.
- Motivar al usuario a realizar deporte y llevar una vida sana.

Al inicio se realizó un estudio del estado del arte para evaluar el mercado actual de este tipo de proyectos, ver si existía algún proyecto similar y como estaba realizado. Después se estuvo investigando acerca de las tecnologías disponibles para llevar a cabo el objetivo que se había propuesto. Como resultado final el proyecto ha sido realizado con tecnologías pertenecientes al MEAN STACK (Mongo, Express, Angular y NodeJS). Para la parte del Front-End se ha realizado la aplicación mediante el framework de Ionic, el cual a su vez está basado en Angular.

Se ha diseñado un servidor con una base de datos mongoDB no relacional y una API REST en NodeJS con el entorno de desarrollo de Express que, mediante protocolos HTTP reparte la información entre el Front-end y el Back-end, incluyendo en este tanto la API-REST como la base de datos y las conexiones con el corpus conversacional del chatbot.

Para desarrollar el corpus conversacional del chatbot se ha usado el servicio de Google: Dialogflow. Y se ha estudiado la documentación de su API oficial para realizar las conexiones a la del proyecto y de este modo dirigir el flujo conversacional entre el usuario y el chatbot.

A lo largo del desarrollo se ha seguido una metodología basada en Sprints o iteraciones para realizar el proyecto, definiendo objetivos, analizando y estableciendo unos requisitos. Se ha de destacar el uso de la aplicación Trello para el seguimiento de las iteraciones puesto que su interfaz de tableros y tarjetas ha sido de gran ayuda a la hora de organizarme.

Tras obtener un mínimo producto viable y someterlo a pruebas se han ido realizando mejoras y resolviendo incidencias hasta que ha sido lanzada a producción y se encuentra actualmente en la Play Store. Se puede acceder a ella a través de este enlace:

https://play.google.com/store/apps/details?id=com.ramonmorcillo.fitbot01

El resultado final del proyecto ha sido una aplicación funcional en estado de producción, accesible al usuario corriente y que se ajusta a los objetivos y especificaciones definidos en un comienzo.

Motivación, justificación y objetivo general

Lo que me motivó a desarrollar Fitbot fue la idea de crear una herramienta para animar a la gente a realizar ejercicios y ponerse o mantenerse en forma. La idea se me ocurrió después de un tiempo yendo a gimnasios y diferentes actividades deportivas. Se me hacía tedioso tener que desplazarme hasta el gimnasio cada vez que quería entrenar y si quería asistir a alguna clase programada tenía que ajustar mi horario al de las clases del gimnasio. Además, muchas veces acababa yendo solo porque no lograba coincidir con mis amigos y echaba en falta esa motivación extra de saber que no estás solo mientras entrenas y te pones en forma.

A raíz de todo esto acababa dejando de ir al gimnasio o de realizar ejercicio por desmotivación y pereza. Comentándolo con otra gente me di cuenta de que no era el único que se encontraba en esta situación de desmotivación y unas ganas casi nulas de realizar ejercicio y estar en forma. Por ello decidí buscar una solución que se adaptara a la gran mayoría de usuarios con este problema. Pienso que una aplicación con un compañero o amigo de ejercicios que te acompañe y anime a dejar atrás el sedentarismo sería una buena respuesta que ayudaría a motivar a todo aquel que quisiera ponerse en forma sin depender directamente de gimnasios u otras personas.

La idea de que la aplicación se centre en dispositivos móviles es principalmente para potenciar su viabilidad ya que, al tenerla en el móvil, el usuario dispondría de ella en todo momento para hablar con su compañero por si tiene ganas de realizar deporte o necesita ánimos para llevar a cabo su dieta saludable. También haría que el chatbot interactuara más con el usuario al tenerlo más cerca y lo motivara a lo largo del día mediante alertas y mensajes motivadores.

El objetivo general de la aplicación sería ante todo animar al usuario a realizar ejercicio físico y mantenerse en forma. Dado que soy una persona activa, este tema me motiva bastante. La aplicación incluiría como integración principal un compañero de ejercicios que, mediante Procesamiento del lenguaje natural, interactuaría con el usuario animándole a hacer ejercicio y

lograr así una motivación extra. También se incluirá un apartado con datos del usuario que registren su progreso, actividades realizadas.

Al acabar la aplicación habré demostrado mis habilidades como ingeniero multimedia a la hora de desarrollar una aplicación móvil con un asistente virtual a modo de compañero de actividades, y la aplicación de muchas aptitudes aprendidas a lo largo de la carrera.

Agradecimientos

Agradecer en principio a mi familia, en especial a mi madre y a mi pareja que son las que más me han apoyado con este proyecto.

A mi tutor José Vicente Berná, el cual además de animarme me ha ayudado a encauzar el proyecto cuando me he atascado o he tenido que tomar una decisión importante.

Por último, agradecer a Fernando Herrera por los cursos que me han enseñado a usar las tecnologías para desarrollar este proyecto.

Citas

“Dude, sucking at something is the first step towards being sorta good at something.”

Ser malo en algo es el primer paso de ser bueno en algo

Jake el Perro

La enfermedad más difícil de curar es la costumbre

Carlos Ruiz Zafón. El laberinto de los espíritus

Índice de contenidos

Índice de figuras

Índice de tablas

1. Introducción

En las últimas décadas el sector de los móviles ha sufrido un avance sin precedentes. De literalmente no existir antes de la década de los 80 hasta hoy en día que algunos, por no decir la gran mayoría, son más potentes que los ordenadores personales de la gente de hace unos años. En mi opinión pienso que es el sector que más innova y avanza tecnológicamente porque es el dispositivo que más usamos a diario. Con solo comparar los smartphones más innovadores de hace 5 años, los cuales orgullosos decían incluir sensor de huella, con los de hoy en día que lo llevan integrado en medio de pantallas que ocupan el 100% del dispositivo.

Y es que el smartphone hace mucho que dejo de ser un simple dispositivo para realizar llamadas, actualmente es el dispositivo más usado para realizar fotografías, acceder a internet, comunicarse textual y oralmente. Incluso podemos pagar con él como si de una tarjeta se tratase.

Y con el aumento del número de mejoras de los smartphones también ha aumentado el número de aplicaciones para los mismos.

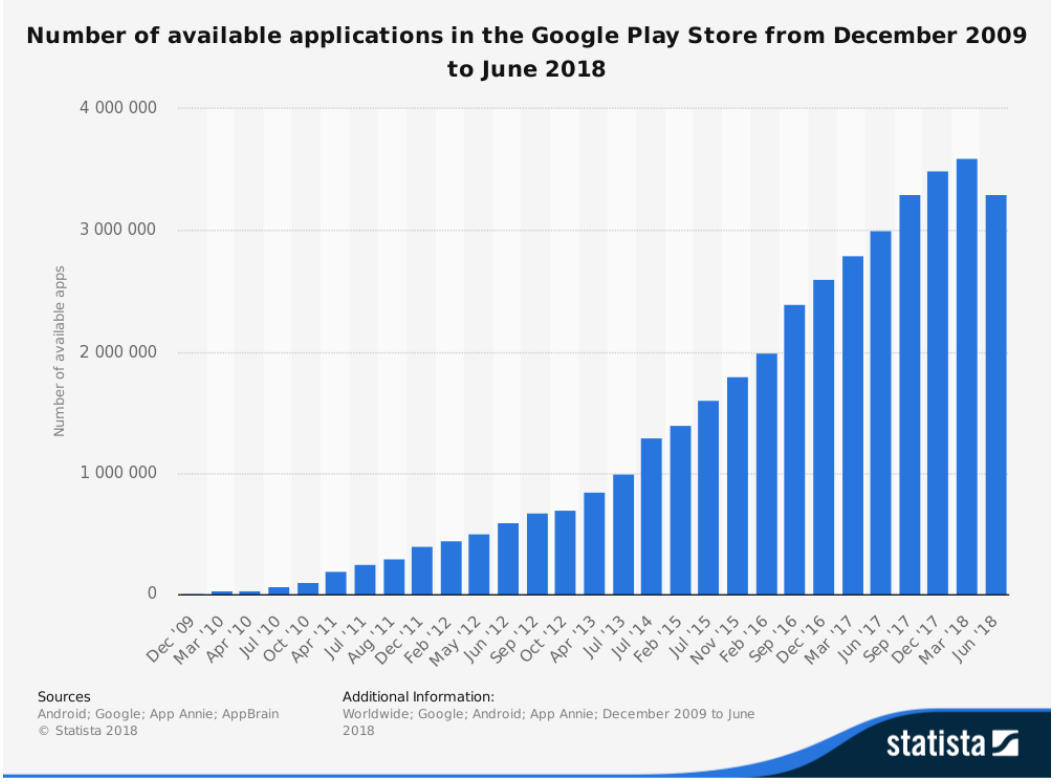

Figura 1. Gráfico del crecimiento de las aplicaciones.

Fuente: https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/)

Han avanzado tanto los Smartphones y hay tantas aplicaciones útiles que ya no podemos estar un día normal sin ellos. Se han convertido en nuestros compañeros para realizar casi cualquier tarea, desde la capacidad de sacar y almacenar fotos hasta el hecho de recordarnos cumpleaños o eventos importantes. A veces este número de eventos es tan grande que no tenemos tiempo de centrarnos y dedicar un poco de tiempo a nosotros mismos. Fitbot pretende solucionar este último problema, el del descuido personal de salud y buena forma física haciendo uso de nuestro compañero del día a día antes mencionado: El smartphone. Llevando encima un amigo que nos motivara a realizar ejercicio y a llevar unos hábitos de vida saludable conseguiríamos concienciarnos más en el problema y acabaríamos mejorando nuestra forma física.

Aparte de la función principal del compañero virtual, la aplicación dispondría de otras herramientas de ayuda a la hora de llevar una vida saludable. Ofrecería la posibilidad de llevar un registro del peso, el cual, junto con la altura del individuo, su edad y su sexo, permitirían calcular el índice de masa corporal para saber si el usuario se encuentra dentro de unos valores normales y sanos. En caso de no encontrarse el compañero motivaría y ayudaría al usuario a alcanzar IMC saludable. También tendría un apartado de frases y consejos motivacionales al que el usuario puede recurrir en caso de querer un extra de motivación e información.

Otra función de la aplicación sería un registro a modo de agenda-calendario en el que el usuario pudiera registrar los ejercicios realizados. También ofrecería la posibilidad de crear anotaciones personales si el usuario lo desea. Y por último estaría la parte de la gamificación de la aplicación, un apartado en el que el usuario pudiera ver los logros y la puntuación adquiridos al llevar unos hábitos de vida saludable, con la intención de que el usuario se motive en alcanzar dichos logros y reduzca las actividades no beneficiosas para su salud.

Como conclusión, Fitbot es una aplicación móvil que se centra en la idea de motivar y ayudar al usuario a llevar unos hábitos de vida saludables mediante un compañero virtual que acompañe al usuario a lo largo del día. Además, como complemento contiene otros apartados a modo de refuerzo motivacional.

2. Estudio de viabilidad

Antes de comenzar con el desarrollo del proyecto se ha considerado de importancia el análisis de los objetivos de este para saber si estos son persistentes o necesarios, en otras palabras, saber si es viable llevar a cabo el proyecto.

Para ello se hará uso de la metodología de análisis DAFO y también un análisis de los riesgos a los que podríamos enfrentarnos y diseñar un plan de contingencia en caso de afrontarlos.

2.1. Análisis DAFO

El análisis DAFO consiste en el estudio de la situación de un proyecto, analizando las características internas (Debilidades y Fortalezas) junto con las externas (Amenazas y oportunidades). Estos análisis se sitúan en una matriz cuadrada para una mejor visión de todas las características.

	POSITIVOS	NEGATIVOS
INTERNOS	**Fortalezas** - Conocimiento del desarrollo de aplicaciones. - Mente creativa. - Conocimiento del desarrollo de asistentes virtuales.	**Debilidades** - Falta de experiencia a la hora de lanzar aplicaciones móviles. - Es una idea nueva por lo que no hay otros proyectos para guiarme.
EXTERNOS	**Oportunidades** - Mejorar la salud de mucha gente - Crear un vínculo entre el usuario y el compañero de deporte	**Amenazas** - Mucha competencia en las tiendas de aplicaciones. - Necesidad del usuario de cambiar su rutina sedentaria. - Que el usuario no interactúe con el compañero

Figura 2. Esquema de un análisis DAFO.
(Fuente propia)

En primer lugar, tenemos las fortalezas, nuestros puntos fuertes. La primera es la del hecho de saber desarrollar aplicaciones web gracias a lo aprendido estos años en la carrera y es de suponer que este conocimiento mejorará durante el desarrollo de la aplicación.

Otra de nuestras principales fortalezas es que considero que soy un joven creativo, con mucha imaginación e ideas y una gran motivación. A la hora de crear un compañero virtual no solo es necesario saber cómo desarrollarlo, también hay que transmitirle un carácter amigable y simpático para que al usuario le guste interactuar con él.

La última fortaleza a destacar sería el hecho de haber trabajado previamente con el desarrollo de asistentes virtuales en las asignaturas de la carrera como e-Learning y en las prácticas de empresa. Esto ayudará a reducir la curva de aprendizaje.

En cuanto a las debilidades creo que la más importante es la falta de conocimiento a la hora de crear aplicaciones móviles. Esto supondrá una inversión extra de tiempo, aunque dada la importancia de este sector en el mundo actual el conocimiento adquirido al final del proyecto será de mucha utilidad en el mundo laboral.

La otra debilidad existente es el hecho de que al ser una idea innovadora no existen otros proyectos o aplicaciones que se centren en la idea de un compañero de ejercicios por lo que no dispondré de una base de la que tomar algunos ejemplos constructivos.

En tercer lugar, tenemos las oportunidades. Este proyecto brindará a muchos usuarios la oportunidad de mejorar sus hábitos de vida y estar en forma de un modo saludable. El hecho de pensar que con un asistente virtual en forma de compañero de ejercicios la vida de una persona puede mejorar en gran medida creo que es la oportunidad más importante.

La otra oportunidad es la de crear un vínculo entre el usuario y el compañero virtual para que el usuario no tenga que depender de terceros a la hora de llevar una vida saludable.

Por último, tenemos las amenazas. Aquellos factores externos al proyecto que pueden suponer un peligro para este. La primera de ellas es la competencia con otras aplicaciones ya que al existir tantas aplicaciones corremos el riesgo de que nuestra aplicación no alcance la visibilidad deseada o necesaria para llegar al público.

La segunda amenaza es que hoy en día el sedentarismo es tan habitual que algunos usuarios no querrán cambiar su rutina y hábitos alimentarios, por lo que acabarán abandonando la aplicación.

La última amenaza es la de que el carácter del compañero no se adecue al del usuario o que este último no se vea cómodo interactuando con su amigo virtual por otras causas y también acabe abandonado la aplicación.

2.2. Análisis de riesgos

El principal riesgo que se afrontará, como ya se ha comentado anteriormente en las debilidades, será la falta de experiencia a la hora de desarrollar aplicaciones móviles. Me expongo al riesgo de no adquirir los conocimientos necesarios de forma adecuada para desarrollar mi proyecto.

Otro riesgo real es el de no alcanzar el objetivo dentro del tiempo límite propuesto. Este riesgo puede depender directamente del anterior dado que la falta de conocimiento y experiencia supondrá dedicar un tiempo previo a la formación personal en esta materia.

Por último, se ha de tener en cuenta los riesgos que no dependen del proyecto tales como situaciones personales imprevistas, enfermedades o aumento del trabajo en mi puesto laboral. Riesgos que obstaculizarían los plazos previstos y generarían retrasos.

Analizados los riesgos se ha llegado a la conclusión de que se deben tomar ciertas medidas para evitar retrasos o en caso de que se den minimizarlos y contenerlos. La primera de ellas y la más importante es la planificación y organización del proyecto. Se establecerá una planificación objetiva de acuerdo con las capacidades y tiempo disponible. También se clasificarán los objetivos para abordar los más importantes primero.

3. Estado del arte.

Hoy en día el mundo del deporte y la actividad física se ha expandido y engloba aspectos nuevos adaptados a las tecnologías actuales como multitud de wearables que registran tu ritmo cardíaco o tus pasos y aplicaciones móviles deportivas. Este último campo, el de las aplicaciones deportivas ha crecido enormemente dado el interés que hay por ambas partes en él.

Una gran parte de los usuarios se ha unido a esta moda "fit" cuyo objetivo principal es cuidarse, estar en forma y hacer ejercicio para verse bien y mejorar su autoestima. Mucha de esta gente no tiene acceso a un gimnasio o entrenador deportivo ya sea por falta de recursos económicos, de tiempo o de otro tipo y recurren a aplicaciones móviles. Es por esto por lo que las empresas y desarrolladores independientes, dado este aumento de interés, han desarrollado multitud de aplicaciones que intentan solventar las necesidades de los usuarios en este ámbito.

Es entonces cuando los usuarios tienen que encontrar de entre todas estas aplicaciones la que más se ajuste a su problema y que les proporcione la mejor experiencia de usuario (UX), facilidad de uso, que sea más funcional y cómoda o que se adapte al dispositivo que emplea.

Para crear aplicaciones de la manera más sencilla, escalable y rápida, también han surgido una gran cantidad de herramientas y elegir la adecuada no es una tarea fácil a la hora de empezar un proyecto. La simple elección de la aplicación errónea puede suponer el retraso del proyecto o un aumento de la complejidad al desarrollarlo. Es habitual incluso requerir del uso de varias herramientas de desarrollo para nuestro proyecto si contiene elementos cliente-servidor.

Por tanto, se realizarán estudios y comparativas entre las diferentes aplicaciones existentes y tecnologías de desarrollo para evaluar de un modo detallado el estado del arte.

3.1. Sistemas similares

Cuando hablamos de sistemas similares nos referimos a aquellas aplicaciones que alcanzan un objetivo similar al proyecto propuesto. El núcleo de este proyecto es un compañero de deportes que ayude al usuario a realizar ejercicio y lleve un registro del ejercicio y actividades del usuario. Se van a analizar sistemas similares en cuanto a los dos aspectos. Por un lado, sistemas de realización y registro de ejercicios y por otro lado asistentes virtuales deportivos.

El resultado de este estudio será conocer los servicios ofrecidos por otras aplicaciones en el ámbito deportivo actual para que a la hora de crear el proyecto este sea capaz de competir con el resto ya sea innovando o mejorando lo existente. Para examinarlas se ha centrado el foco en las aplicaciones más populares y usadas dentro de varias categorías de las tiendas de aplicaciones, tales como: Deportes, Estilo de vida, Medicina y Salud y Bienestar.

3.1.1 Ejercicios en casa

Es una aplicación desarrollada con el fin de realizar ejercicio en casa sin necesidad de equipamiento. Es muy completa en su género dado que te permite centrarte en distintas partes del cuerpo por si quieres mejorar una parte completa. También incluye una monitorización semanal mediante alertas y notificaciones que te motiva a realizar ejercicio diario.

Como esta aplicación hay muchas otras en la Play Store de rutinas para realizar ejercicios. La interfaz es sencilla de usar, con muchas opciones y suele ser similar entre ellas.

Características:

- Aplicación tanto para Android como para iOS.
- Interfaz intuitiva y sencilla.
- Gran variedad de ejercicios donde elegir.
- Seguimiento de los ejercicios realizados por el usuario.

Defectos:

- Rutinas de ejercicios repetitivas.
- El sistema de motivación del usuario es muy básico.

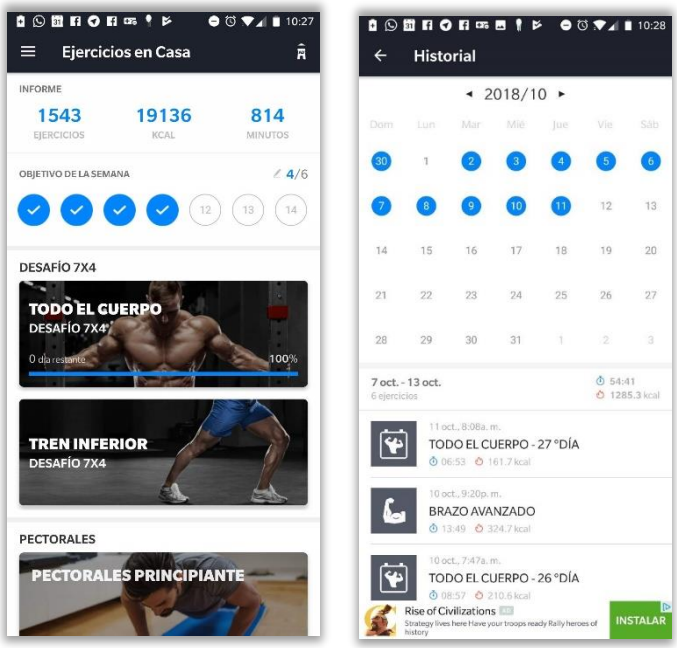

Figura 3. Aplicación Ejercicios en casa. Pantalla principal y pantalla del informe de seguimiento.

3.1.2. Google Fit

Desarrollada por Google LLC, es una aplicación centrada en el seguimiento de la actividad física y estado del usuario. El usuario puede introducir su altura y peso cuando quiera para registrarlo y la aplicación se encarga de mostrarte gráficas con los datos introducidos. También tiene registra datos como los pasos dados o las pulsaciones del usuario en un determinado momento. Para esta última se requiere de un dispositivo que registre dichas pulsaciones.

En la última versión se ha introducido un sistema de gamificación por medio de puntos para motivar al usuario a realizar ejercicio. Dicho sistema de puntos se adapta a las necesidades del individuo, aunque este también puede modificar los objetivos a su antojo.

Características:

- Aplicación tanto para Android como para iOS.
- Interfaz limpia, clara e intuitiva.
- Buen sistema de registro de datos.
- Sincronización de datos provenientes de terceros.

Defectos:

- Sistema de gamificación muy básico y poco motivador.

- No disponible en la AppStore de Apple.

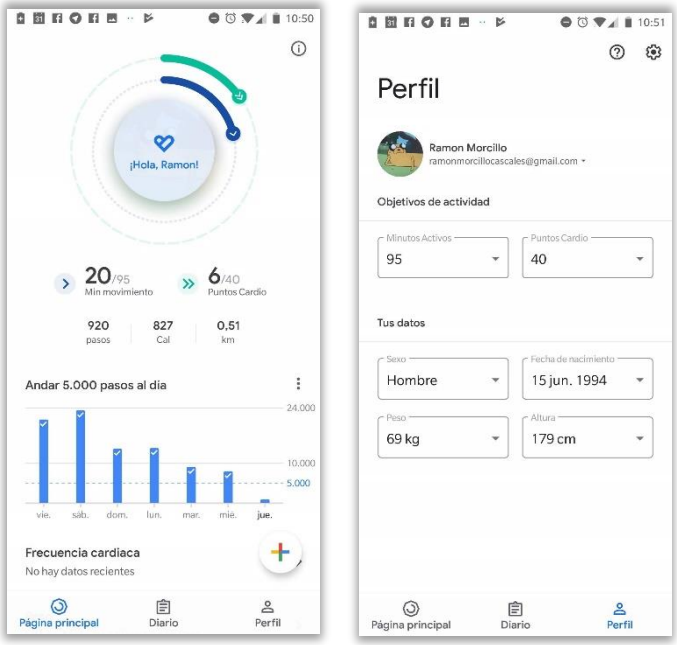

Figura 4. Aplicación Google Fit. Pantalla principal y pantalla del perfil del usuario.

3.1.3. GymBot

GymBot es un chatbot conversacional diseñado en para Facebook Messenger y cuya función es motivarte a realizar ejercicio y llevar un registro de estos. Mediante la aplicación de Facebook Messenger o la interfaz web, el usuario accede a conversar con el asistente y mediante comandos con el nombre y las series de los ejercicios realizados el asistente almacena los datos.

Además, si el usuario introduce el comando "stats" el asistente le mostrará unas estadísticas a partir de los ejercicios realizados. Dispone de una página web, aunque en esta solo se explica el funcionamiento del chatbot y se redirige a la aplicación de Facebook Messenger para dialogar con él.

Características:

- La aplicación dispone de un chatbot conversacional.
- Accesible desde cualquier navegador.

Defectos:

- La parte conversacional es muy débil o nula.

- No es muy intuitiva
- La mayoría de la comunicación se realiza por medio de comandos.
- Depende totalmente de Facebook Messenger para realizar su función.

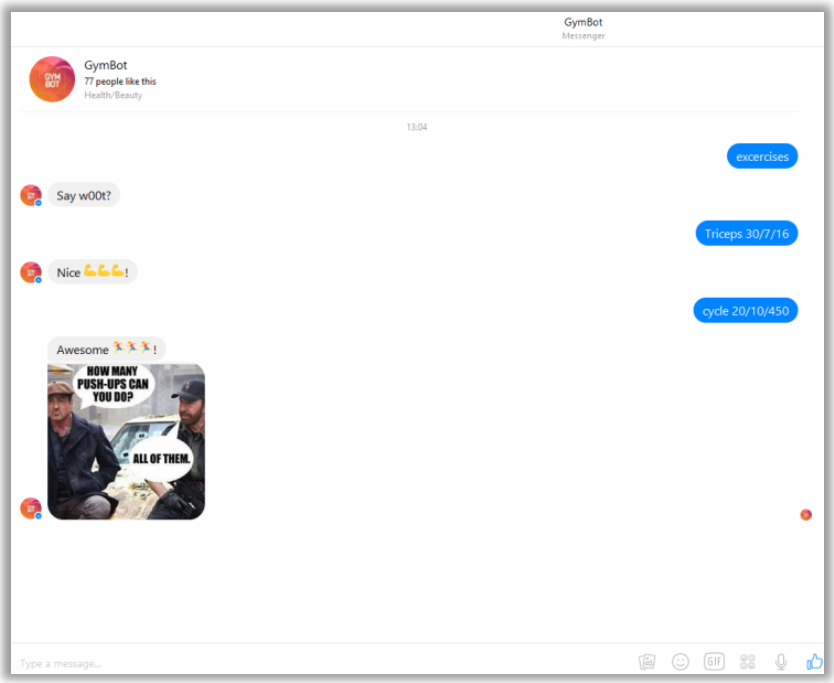

Figura 5. Gymbot. Ejemplo de conversación.

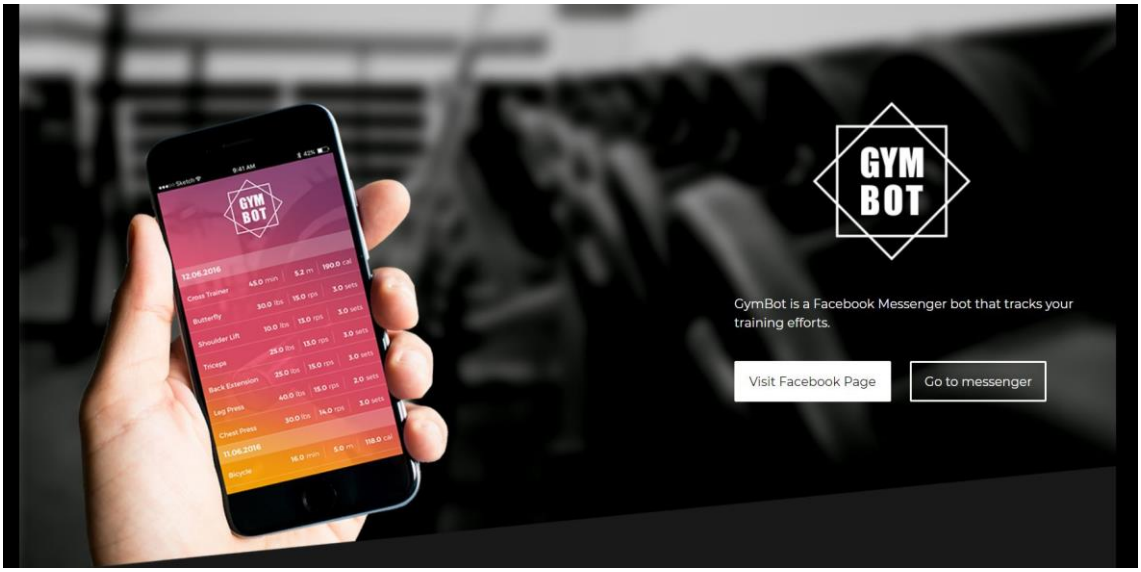

Figura 6. Gymbot. Página principal.

3.1.4. FitCircle

En este caso se trata también de un chatbot desarrollado para Facebook Messenger. El chatbot de FitCircle se llama Zi y se comunica contigo mediante conversación nativa y botones dirigidos. Puedes conversar con él de cosas básicas de la vida como si se tratase de una persona o compañero y luego mediante botones e imágenes te propone ejercicios a realizar o dietas a llevar a cabo.

Al igual que GymBot, FitCircle también dispone de una web principal en la que poder informarnos más del chatbot y acceder a este.

Características:

- La aplicación dispone de un chatbot conversacional.
- El asistente ofrece distintas rutinas de ejercicios para llevar a cabo.
- Dispone de dietas a realizar en caso de necesitar una.

Defectos:

- Depende totalmente de Facebook Messenger para realizar su función.
- No dispone de un seguimiento de las actividades realizadas.
- No dispone de aplicación propia para móvil.

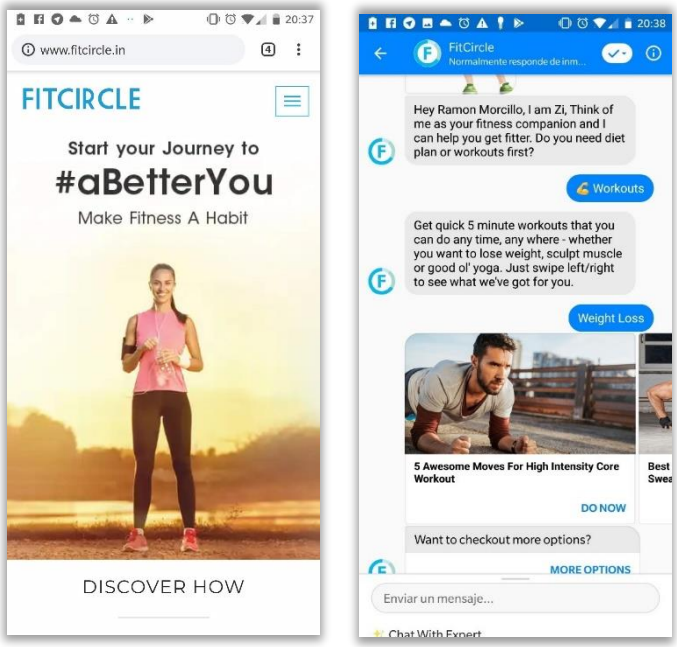

Figura 7 FitCircle. Página principal y ejemplo de conversación.

3.2. Limitaciones de los sistemas

- Ejercicios En Casa.
    - No dispone de un compañero de ejercicios.
    - Está limitada a únicamente realizar ejercicios de entrenamiento dentro de casa y no incluye realizar actividades para estar en forma pero que se realicen en el exterior o que se puedan adaptar al día a día.
- Google Fit
    - Tampoco dispone de un compañero que motive a estar en forma.
    - Únicamente sirve como registro de actividad.
- GymBot
    - Solo está disponible en Facebook Messenger.
    - Es necesario conocer bien el uso de los comandos para interactuar con el asistente y registrar nuestra actividad.
- FitCircle
    - Únicamente podemos interactuar con el asistente a través de Facebook Messenger.
    - No dispone de un registro de las actividades realizadas.

3.2.1. Conclusión personal de los sistemas similares y sus limitaciones.

Como conclusión al estudio de los sistemas similares y de las limitaciones que presentan no se detecta ni se encuentra a simple vista ni tras una búsqueda ninguna aplicación que englobe de un modo óptimo y sencillo un compañero motivador deportivo junto con un registro de ejercicios y actividades. Las aplicaciones que se centran en estar en forma y realizar ejercicio no disponen de un compañero que anime al usuario y los asistentes virtuales que pretenden motivar o ayudar al usuario a realizar ejercicio están muy limitados en cuanto a la plataforma mediante la que interactúan y no ofrecen un sistema de registro de actividades como las aplicaciones que se centran en ello.

Por tanto, si un usuario necesita de ambas cosas en una misma aplicación ahora mismo no puede solventar esta necesidad y tendría que disponer de una aplicación para cada necesidad.

3.3. Modelos de negocio

En esta sección quiero hacer un inciso en los modelos de negocio que presentan las distintas aplicaciones antes mencionadas. Google Fit es la única que no obtiene una fuente de ingresos a partir de su aplicación y que su servicio es puramente gratuito. Tampoco dispone de ninguna opción avanzada de pago.

GymBot no ofrece ninguna opción de pago ni muestra publicidad en sus aplicaciones por lo que no generaría ingresos directos a partir del chatbot.

De las dos restantes una obtiene beneficios de forma indirecta y la otra de una forma más directa. FitCircle no muestra anuncios, pero sí que indirectamente, a la hora de querer realizar dietas, promociona en su web principal una tienda online de productos con proteínas orgánicas.

El último modelo de negocio es el de Ejercicios en casa. Esta aplicación como otras muchas de rutinas de ejercicios sí que muestran publicidad emergente durante el uso de la aplicación no siempre demasiado intrusiva. Esta publicidad sí que genera unos ingresos directos a partir de la aplicación.

Como conclusión de los modelos de negocio, en el caso de Fitbot, el modelo de negocio deseado, si hubiera uno, sería similar al último mencionado. Una publicidad no muy intrusiva principalmente. Además de dar la opción al usuario de pagar por una versión 'pro' en la que no hubiera publicidad de ningún tipo.

3.4. Tecnologías para el desarrollo

A la hora de desarrollar Fitbot es muy importante decidir qué entorno de programación vamos a usar dado que esto afectará de un modo importante al desarrollo del proyecto. Fitbot está pensada en un principio únicamente como aplicación móvil y por tanto se enfocará el estudio de tecnologías a este campo.

Además de la aplicación en sí, se debe programar el elemento principal de la misma, el compañero de ejercicios. Para el cual se han de investigar las mejores tecnologías para desarrollar chatbots conversacionales y decidirse por la que mejor se adapte al proyecto.

3.4.1. Tecnologías para el desarrollo de aplicaciones

Actualmente para desarrollar una aplicación móvil hay dos tendencias principales. Desarrollarla de manera nativa utilizando el lenguaje de programación de la plataforma en la que se quiere lanzar: iOS o Android. Se ha de mencionar que no se enfocará la aplicación al tercer sistema operativo Windows Phone dado que ya no recibe soporte por parte de Microsoft y está obsoleto.

La otra opción es desarrollar una aplicación híbrida, esto quiere decir que mediante el mismo código fuente, el entorno de desarrollo genere aplicaciones tanto para iOS como para Android.

3.4.1.1. Xamarin

Xamarin es un entorno de desarrollo adquirido por Microsoft en el que se pueden desarrollar apps nativas tanto para Android como para Windows e iOS. Al ser adquirida por Microsoft se encuentra integrada totalmente en el editor de texto de este: Visual Studio. El lenguaje de desarrollo que utiliza es .NET.

El modelo de pago se basa en que puedes usar de forma gratuita Visual Studio y Xamarin *community edition* gratis. Esta opción no es muy bien recibida por la comunidad que afirma tener muchos errores. También hay dos opciones de pago sin apenas errores que son Xamarin Professional (999$ al año) y Xamarin Enterprise (1899$ al año). Por último, hay que mencionar que Xamarin Professional ofrece un periodo de prueba gratuito de un mes.

3.4.1.2. Apache Cordova

Apache Cordova es un entorno de desarrollo de código abierto basado en código HTML, CSS y JavaScript para programar aplicaciones para múltiples sistemas operativos.

Posee una gran comunidad que mejora el propio entorno mediante numerosos plugins que amplifican las capacidades de desarrollo del propio Apache Cordova. Dada la cantidad de plugins, su comunidad y que es de código abierto, es integrado en otros entornos de desarrollo de aplicaciones como Adobe PhoneGap o Ionic. Su uso es totalmente gratuito.

3.4.1.3. Ionic Framework

Creado en 2013, Ionic es un entorno de desarrollo híbrido y de código abierto para aplicaciones móviles. Está basado principalmente en Angular y Apache Cordova. Utiliza tecnologías de desarrollo web como HTML, CSS, TypeScript, JavaScript y Sass. A su vez también hace uso de otras herramientas propias como ionic-cli: una interfaz de comandos para consola que simplifica tareas como crear la aplicación o añadir páginas.

Su uso básico es gratuito y en su página principal ofrecen recursos de aprendizaje a modo de tutoriales, así como una gran documentación bien detallada de todo lo que incluye el propio entorno de desarrollo. También se pueden adquirir servicios mensuales que mejoran la experiencia y añaden funcionalidades de desarrollo extra; estos servicios son Ionic Developer por 29$ al mes y Ionic Team por 49$ al mes.

3.4.2. Tecnologías para el desarrollo de chatbots y asistentes virtuales.

Hoy en día los asistentes virtuales han cobrado una gran importancia en nuestras vidas. Millones de dispositivos desde smartphones hasta neveras los incluyen, incluso se han creado dispositivos solo para ellos como el Google Home o el Amazon Echo.

Para desarrollar el compañero de ejercicios se necesita una tecnología que genere un asistente amigable y que motive al usuario a interactuar con él y ponerse en forma. Por ello se va a realizar un estudio de las tecnologías más usadas actualmente.

3.4.2.1. Chatfuel

Chatfuel es una tecnología de desarrollo de chatbots conversacionales creada en 2015. La orientación principal de la compañía estuvo dirigida a chatbots para Telegram, aunque ahora el foco principal es la creación de manera sencilla de chatbots para Facebook Messenger.

Las tecnologías en las que se basa y su entorno seria Facebook Messenger

En cuanto al precio de la tecnología, para su uso se debe contratar un servicio de pago mensual de 15$ al mes llamado Chatfuel PRO.

Como puntos a favor destaca su facilidad de aprendizaje y uso de esta a la hora de desarrollar chatbots para Facebook Messenger, además del coste gratuito de uso. Aunque por otro lado su principal contra seria que solo está disponible para chatbots en Facebook Messenger y no para otras interfaces como web o aplicaciones sin depender de Facebook Messenger.

3.4.2.2. Telegram Bot Api

Telegram es una aplicación multiplataforma de mensajería instantánea y envío de archivos y contenido multimedia fundada en 2013. En lo que se refiere a su API para bots, es una interfaz con un entorno basado en llamadas http creada para aquellos desarrolladores que quieren construir bots para Telegram. El coste de uso de esta tecnología es gratuito.

Por tanto, como aspectos positivos y negativos del uso de esta tecnología para desarrollar mi proyecto destaco que es fácil de aprender y los múltiples servicios ofrecidos por parte de Telegram para integrar en los chatbots de esta plataforma. Pero al igual que Chatfuel el principal problema de esta tecnología es que solo está diseñada para el desarrollo de chatbots en Telegram y por tanto no permite su uso en otras plataformas.

3.4.2.3. DialogFlow

Inicialmente conocida como API.AI, fue comprada por Google en septiembre de 2016 y un año más tarde, en octubre de 2017, se la renombro como Dialogflow. Permite crear asistentes virtuales o bots que se comunican mediante interfaces conversacionales de texto o voz haciendo uso de Inteligencia Artificial.

Su coste de uso inicialmente es gratuito, aunque dispone de una Enterprise Edition orientada a empresas que buscan escalar fácilmente el soporte ofrecido a los usuarios en función de las demandas de estos.

Como pros de esta tecnología se destaca su implementación multiplataforma ya que permite exportar el asistente desarrollado a múltiples plataformas distintas para interactuar con él, entre ellas destaco interfaces web, Facebook Messenger, Telegram y Whatsapp.

Y como contras el principal sería la curva de aprendizaje de la tecnología ya que al comienzo es complicado entender el uso de esta y la forma en la que interactúa el asistente virtual.

3.4.3. Conclusión y tecnologías seleccionadas

De entre todas las tecnologías para el desarrollo de aplicaciones se ha seleccionado Ionic y las razones que han llevado a esta decisión son las siguientes:

- Es un entorno híbrido lo que supone que si se quiere distribuir la aplicación en distintos sistemas operativos no se requerirá de programar en cada lenguaje nativo, sino que con programar la aplicación una vez será suficiente.
- Está basada en el framework de Front-end Angular y tecnologías como HTML, CSS y Javascript. Y tanto el framework como las tecnologías se han usado y practicado a lo largo de la carrera por lo que la curva de aprendizaje será menos que si se tuviera que basar en tecnologías .NET como Xamarin.
- Apache Cordova ya se encuentra integrado y adaptado al framework lo cual permite disponer de múltiples plugins desarrollados por los creadores y por la comunidad, todos ellos visibles en la documentación.
- El precio. La tecnología de Ionic es gratuita y ofrece la cantidad de servicios necesarios para desarrollar mi aplicación.

En cuanto a las tecnologías de desarrollo de asistentes virtuales para el compañero se ha elegido Dialogflow. Los motivos de la elección han sido los siguientes:

- Desarrollo multiplataforma. Mientras que las otras tecnologías solo permitían el desarrollo de chatbots para una única plataforma como Telegram o Facebook Messenger, Dialogflow permite que un mismo asistente pueda ser exportado e integrado en las plataformas más populares y sobre todo en una aplicación como la que se desarrollará en el proyecto.
- El precio. En comparación con otras de pago como Chatfuel, la Edición Estándar gratuita de Dialogflow proporciona las herramientas necesarias para desarrollar el asistente sin tener que realizar una inversión extra.

4. Objetivos

Desde un principio el objetivo principal del proyecto ha sido la creación de una aplicación que motive al usuario a realizar ejercicio y estar en forma. Para alcanzar este objetivo han surgido

muchas ideas de las cuales bastantes han sido descartadas y otras han adquirido una importancia principal en el proyecto. Los objetivos que van a ayudar a realizar el principal son:

Objetivos por parte de la **aplicación:**

- **Crear una aplicación fácil de usar e intuitiva con un diseño sencillo y moderno.** Si el usuario no se adapta al diseño y se le hace tedioso el uso de la aplicación acabara por no acceder a la misma. Además, el diseño de esta estará inspirado por estilos minimalistas actuales como Material Design.
- **Diseñar un asistente conversacional que motive al usuario.** Aunque la idea principal de la aplicación no contenía un asistente virtual, este último ha acabado adquiriendo el papel principal del proyecto dado que su existencia ayuda en mayor medida al objetivo principal de motivar al usuario a realizar ejercicio.
- **Diseñar una interfaz de chat mediante la cual interactuar con el asistente deportivo.** Es muy importante que la comunicación entre el usuario y el compañero deportivo se realice de la forma más sencilla e intuitiva posible y una interfaz de chat es el mejor modo para alcanzar ese objetivo.
- **Crear un sistema de registro de actividades o ejercicios realizados.** Para aumentar la motivación del usuario es importante que el mismo pueda acceder a un registro en el que anotar su progreso personal.

Objetivos **personales:**

- **Aprender a diseñar y crear aplicaciones para smartphones.** Considero que al ser el smartphone el dispositivo más usado habitualmente, aprender a desarrollar aplicaciones para este es una buena apuesta de formación personal.
- **Mejorar la forma de planificación personal para aprovechar el tiempo y los recursos disponibles.** Para llevar a cabo este proyecto y proyectos futuros considero que la planificación es uno de los objetivos más importantes y por ello con este proyecto se quiere mejorar la planificación personal.
- **Aprender a desarrollar asistentes conversacionales**. El campo de los asistentes virtuales está en pleno auge y muchas compañías y empresas ya han invertido en la investigación e incorporación de uno o varios. Por ello creo que ampliar los conocimientos propios en este campo puede ser de ayuda en un futuro más allá de este proyecto.

Objetivos **opcionales:**

- **Investigar y crear un sistema de gamificación para aumentar la motivación y sensación de progreso.** Añadir un apartado de gamificación a una aplicación potencia el uso de esta por parte del usuario, ya sea mediante un sistema de puntos u otros métodos.
- **Crear un método de monetización de la aplicación por la cual obtener ingresos.** Aprender a monetizar y obtener un beneficio económico del proyecto tanto personal como para la mantención de este.

5. Metodología

La metodología de un proyecto no es más que el cómo se va a desarrollar el mismo. Existen múltiples tipos de metodologías para el desarrollo de software y aplicaciones, para este proyecto se usará una metodología de desarrollo por fases junto con la metodología SCRUM. Las fases del proyecto serán las mismas que la mayoría de los proyectos de software:

- **Análisis:** Se comenzará realizando un estudio del mercado actual en lo referente al sector de la aplicación. Una vez se hayan analizado los resultados se procederá a especificar los requisitos del proyecto.
- **Diseño:** Una vez se tenga una idea de los requisitos del proyecto, se comenzará a estructurarlos y situarlos gráficamente a modo conceptual en un diseño completo de la aplicación.
- **Implementación:** En esta fase se iniciará el desarrollo de todo el contenido gráfico y las funcionalidades de la aplicación en base a los diseños creados en la fase anterior.
- **Despliegue:** Esta es la fase final en la que se comprueba que se haya implementado todo de un modo correcto y se corrigen los posibles fallos que se puedan encontrar. Una vez realizadas estas subtareas se pone en producción y se evalúa el realizar una nueva iteración.

En cuanto a la metodología SCRUM, la cual está basada en desarrollar las funciones principales de una forma ágil y flexible y en los principios de continua detección de errores y posibles mejoras, por lo cual se adapta a los imprevistos y se autogestiona de una manera fácil y equilibrada. Algunas de las ventajas de esta metodología para el desarrollo de la aplicación frente a otras serían:

- **Alta flexibilidad ante cambios:** Dada la inexperiencia personal a la hora de afrontar un proyecto de esta magnitud se puede dar una mala contemplación inicial de los requerimientos de este. La metodología SCRUM posee una gran reacción ante posibles cambios en los requerimientos de una aplicación.
- **Reducción del tiempo de mercado:** Mediante esta metodología la aplicación alcanzaría un estado con las funcionalidades básicas lo antes posible lo cual permitiría un mayor periodo posterior de testeo y mejora.

- **Mayor calidad del software producido:** El requerimiento de implementar y mejorar las funcionalidades a través de cada iteración consigue que se depure y mejore el software iteración a iteración.
- **Mejor control del tiempo:** Al dividir el proceso de implementación de funcionalidades en iteraciones denominadas también sprints se puede conocer el tiempo medio en llevar a cabo cada iteración y a su vez estimar cuando se implementará una determinada funcionalidad.
- **Menor número de riesgos:** Gracias a la anterior ventaja de controlar mejor el tiempo de desarrollo y a la prioridad de implementación de las funciones principales se pueden prever posibles retrasos y riesgos futuros en el desarrollo y afrontarlos de una manera adecuada.

También se va a hacer uso de herramientas para realizar el desarrollo del proyecto de un modo más eficiente, gestionando el tiempo, planificando objetivos y contabilizando las tareas realizadas y por hacer. Las herramientas en cuestión van a ser principalmente Clockify y Trello.

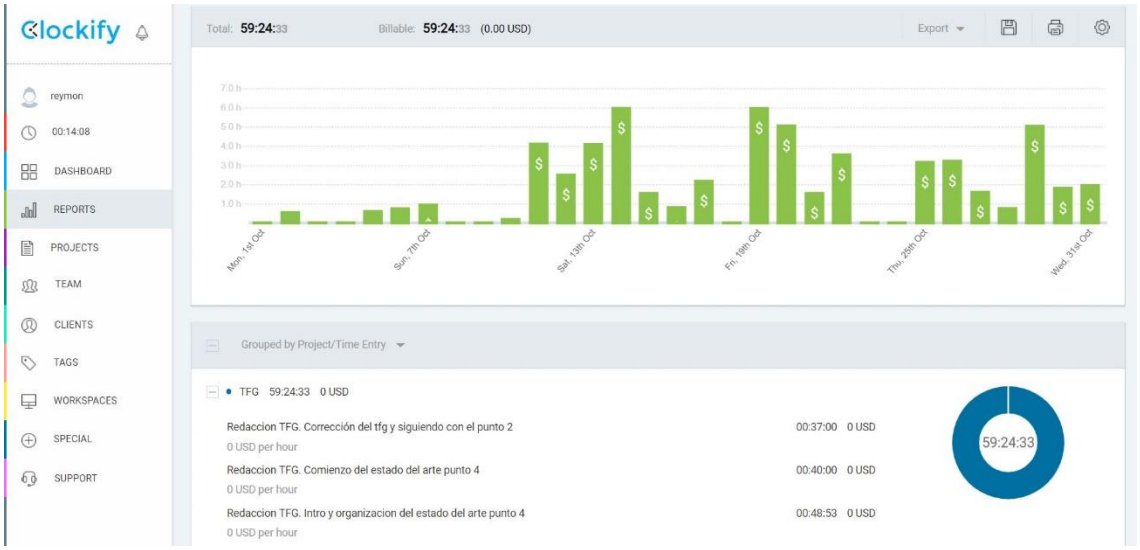

Figura 8. Interfaz de reportes de Clockify filtrada en mi proyecto.

Clockify es una aplicación web y móvil cuya finalidad es la de contabilizar el tiempo de las tareas de un proyecto de un modo preciso y simple. Además de la herramienta principal de temporizador, Clockify posee un historial de tareas realizadas y un sistema de etiquetas y proyectos en los que agrupar los registros. Lo cual hace muy útil la diferenciación de tareas a la hora de trabajar en varios apartados de un mismo proyecto. Como funcionalidad extra incluye la generación de informes detallados de los tiempos de trabajo realizados por el usuario.

Trello es una herramienta también con interfaz web y aplicación móvil cuyo objetivo es el de organizar y gestionar el trabajo y las tareas a llevar a cabo. La interfaz principal se compone de un tablero en el cual por medio de listas agrupar las tareas (tarjetas). En concreto agrupare las tareas en tres listas: tareas por hacer, en proceso y realizadas. También se usará un sistema de etiquetas de colores para diferenciar los ámbitos de las tareas.

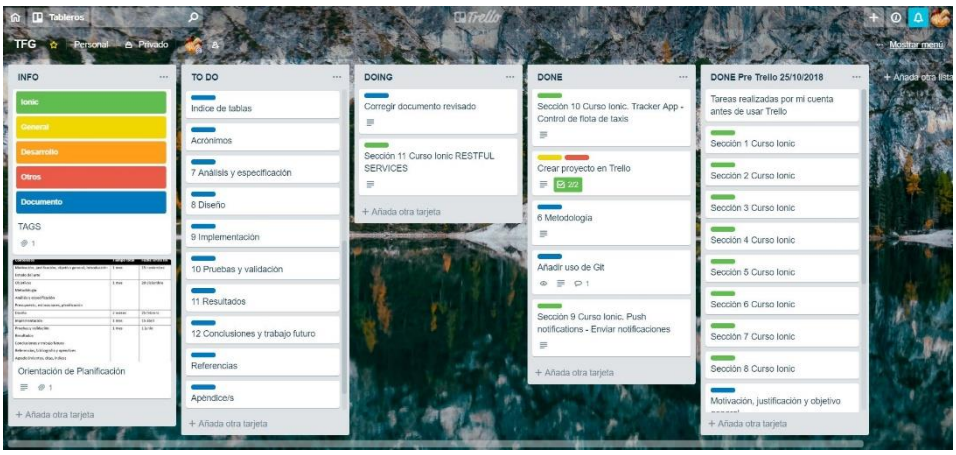

Figura 9. Tablero de Trello del Proyecto.

Por último, a la hora de desarrollar el proyecto en cuanto a producir el código necesario y adecuado para realizar las funcionalidades de la aplicación, se usarán principalmente dos herramientas:

- Atom, como editor de texto. Atom es un editor de texto gratuito que presume de ser de código abierto por lo que posee numerosas herramientas y plugin creados por la comunidad. La decisión de usar este editor se ha tomado porque de todos los usados en el pasado, es con Atom con el que mejor se ha trabajado
- Git y GitHub. Se usará este sistema para almacenar el código en un repositorio y así tener varias versiones de este por seguridad ante la pérdida del código del ordenador físico y por la posibilidad de revertir el estado actual del proyecto a una versión anterior.

6. Análisis y especificación

Esta sección se centra en las principales funcionalidades y problemas que queremos solventar con nuestra aplicación. Para ello se realizará un análisis y especificación de todos los requisitos del proyecto. Estos requisitos serán lo más concretos posibles dado que se usarán como guía a la hora de desarrollar la aplicación y si la fase de desarrollo es avanzada el coste de solventar un requisito para solucionar un problema será mayor.

Para realizar el análisis y la especificación se tomará como base el estándar IEEE 830 para el ERS (Especificación de Requerimientos de Software) cuyas recomendaciones y modo de proceder tienen como producto final una clara definición sistemática de los requisitos necesarios a la hora de diseñar una solución de software para cada problema.

Para realizar un análisis lo más específico posible se han dividido estos requisitos en requisitos funcionales y requisitos no funcionales. Cada requisito estará identificado con un ID único y también dispondrá de un nombre y una prioridad.

La prioridad del requisito estará clasificada en:

- Alta, cuando sea esencial cumplir con el requisito para el correcto desarrollo de la funcionalidad.
- Media, cuando solamente se desee cumplir con dicho requisito, pero no sea esencial para el desarrollo de la aplicación.
- Baja, cuando la implementación del requisito sea meramente opcional.

Por último, antes de comenzar a analizar los requisitos se analizará el usuario para el cual deben cumplirse dichos requisitos. La aplicación, al no tener roles de usuarios ni una distinción de usuarios por categorías de pago, dado que es gratuita, solo tendrá un tipo de usuario. Sus características como la edad o el sexo podrán variar ampliamente ya que la aplicación contempla su uso en usuarios de cualquier edad y sin importar si son hombres o mujeres.

Este tipo de usuario será uno cuyos intereses estarán centrados en llevar una vida sana y cuidar su salud. El usuario al que se enfocará la aplicación no deberá poseer un conocimiento extremo del manejo de aplicaciones ni de ninguna materia en particular.

6.1. Requisitos Funcionales.

Los requisitos funcionales comprenden las principales funcionalidades de la aplicación y los comportamientos en interacciones que el sistema realizará para afrontarlos.

Identificador	RF-01
Nombre	Usuario inicia conversación con compañero
Prioridad	Alta
Descripción	El usuario debe poder hablar con el compañero al acceder a la aplicación en cualquier momento que lo desee. Podrá hablar de hacer algo hoy, buscar ánimo y motivación o de temas ajenos a la aplicación.

Tabla 1. Requisito funcional 1.

Identificador	RF-02
Nombre	Control de actividades
Prioridad	Alta
Descripción	El usuario podrá llevar un registro donde anotar las actividades que realiza con el compañero deportivo o individualmente. También podrá editarlas y eliminarlas

Tabla 2. Requisito funcional 2.

Identificador	RF-03
Nombre	Estado de salud del usuario
Prioridad	Media
Descripción	El usuario podrá anotar y llevar un registro de sus datos físicos como el peso o la estatura además del estado en el que se encuentra tanto física como anímicamente.

Tabla 3. Requisito funcional 3.

Identificador	RF-04
Nombre	Cálculo del IMC del usuario
Prioridad	Media

Descripción	Mediante la introducción de una serie de datos como el peso, la edad y la estatura del usuario el compañero o la aplicación deberá ofrecer Información acerca del IMC del usuario y el estado de salud de este.

Tabla 4. Requisito funcional 4.

Identificador	RF-05
Nombre	Loguin
Prioridad	Media
Descripción	El usuario podrá registrarse y autentificarse en la aplicación lo cual le aportará una serie de ventajas a la hora de usar la aplicación.

Tabla 5. Requisito funcional 5.

Identificador	RF-06
Nombre	Explicación previa de la aplicación
Prioridad	Media
Descripción	El usuario deberá recibir una información previa de cómo usar la aplicación previa al uso de esta.

Tabla 6. Requisito funcional 6.

Identificador	RF-07
Nombre	Personalización
Prioridad	Baja
Descripción	El usuario podrá adaptar algunos aspectos de la aplicación a su gusto.

Tabla 7. Requisito funcional 7.

6.2. Requisitos No Funcionales

Los requisitos no funcionales son aquellos que describen aspectos del sistema pero que no describen el modo de actuar ni las funcionalidades de este. Algunos de estos requerimientos deberían estar presentes desde el inicio en cualquier sistema ya que están relacionados con aspectos como la seguridad o el control de errores.

Identificador	RNF-01
Nombre	Disponibilidad
Prioridad	Alta
Descripción	El compañero debe estar disponible en cualquier momento del día para que el usuario pueda interactuar con él. Además, el tiempo de respuesta del compañero deberá ser lo más breve posible.

Tabla 8. Requisito no funcional 1.

Identificador	RNF-02
Nombre	Seguridad
Prioridad	Alta
Descripción	Todos los datos que el usuario introduzca dentro de la aplicación deben cumplir las normas de seguridad y protección de datos.

Tabla 9. Requisito no funcional 2.

Identificador	RNF-03
Nombre	Multiplataforma móvil
Prioridad	Media
Descripción	La aplicación debe ser multiplataforma móvil y funcionar en los distintos sistemas operativos móviles actuales: Android e iOS

Tabla 10. Requisito no funcional 3.

Identificador	RNF-04
Nombre	Control de errores
Prioridad	Baja
Descripción	Aportar mecanismos que permitan informar al usuario de los errores que se hayan producido a cualquier nivel.

Tabla 11. Requisito no funcional 4.

7. Diseño

El apartado del diseño del Trabajo de Fin de Grado es sin duda el más importante ya que será donde se le dará respuesta a los requisitos funcionales y no funcionales anteriormente planteados, se hará uso de los identificadores de los requisitos para relacionar el diseño de la solución de cada uno.

El diseño de estas soluciones es una tarea compleja en la que intervienen aspectos a distintos niveles o vistas del proyecto. Por tanto, se agrupará el diseño en distintos apartados en función de la materia o el aspecto que se está diseñando, de este modo cada apartado contendrá todo lo relativo a ese contexto. Y la suma de estos diseños conformarán la aplicación en sí.

7.1. Diseño arquitectura conceptual

La arquitectura conceptual de un proyecto son los módulos y bloques funcionales que contendrá la solución. En el caso de este proyecto se distinguirán 4 módulos principales a groso modo: Cliente, APIREST, Dialogflow y Persistencia. Los dos últimos estarían incluidos en el servidor de la aplicación.

- La parte del **Cliente:** En esta parte se encuentran las vistas de la aplicación móvil con las que van a interactuar los usuarios.
- La parte de la **API REST**: Esta es la parte encargada de los servicios y controladores de la aplicación. En esta parte el usuario no puede acceder, es la encargada de manejar la información y los datos entre la parte del Cliente, donde los usuarios ingresan o piden la información, y la base de datos donde se almacena dicha información.
- La parte de **Dialogflow** es donde se encuentra el corpus del asistente conversacional, el compañero de la aplicación.
- La parte de la **Persistencia:** Parte que se encarga de la persistencia de los datos, de su almacenamiento y distribución de un modo correcto en una base de datos.

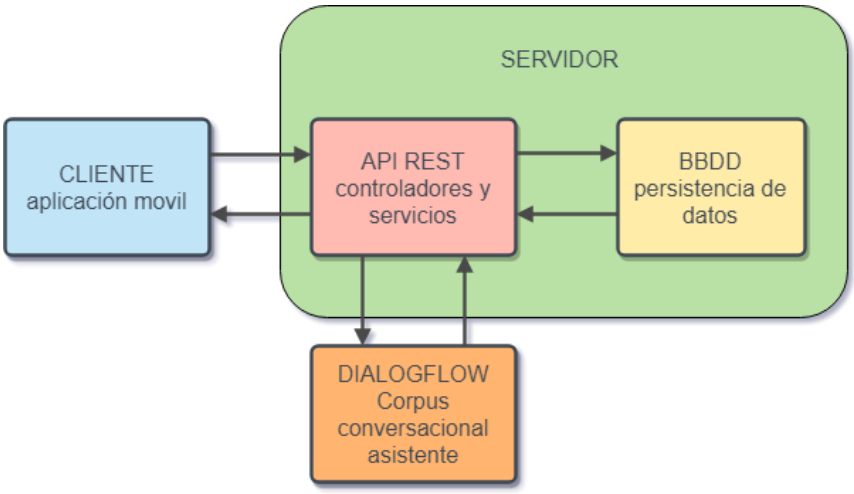

Figura 10. Diseño arquitectura conceptual.

7.2. Diseño de la persistencia

Cuando se habla de persistencia se hace referencia de lo concerniente al almacenamiento de datos de la aplicación. Analizando los requisitos del proyecto se ha llegado a la conclusión de que se necesitará almacenar datos del usuario para acceder a la aplicación y para llevar un registro de su físico y datos de las interacciones del usuario con el compañero, principalmente las conversaciones mantenidas. El tipo de base de datos será una SQL dinámica, la cual irá creciendo conforme accedan más usuarios e interactúen con el compañero.

Como mecanismos de seguridad de la base de datos solo los usuarios registrados podrán interactuar con los datos de la base de datos. Para almacenar los datos de la aplicación se hará uso de una base de datos formada por 4 tablas.

- Usuarios (users): Tabla principal de la base de datos que contiene la información de los usuarios registrados en nuestra aplicación. Los datos que se almacenarán en esta tabla serán el nombre y los apellidos del usuario, por separado para que más adelante el compañero se dirija únicamente al nombre. También se guardará su cumpleaños, email, contraseña, la cual se encriptará, y otros datos.
    - Campos: id, name, email, password, token, birthday, height, age, weightActual, dateCreated, gender, img.
    - Clave primaria: email.

- Actividades (activities): Tabla que almacenará las actividades que anote el usuario. Las actividades tendrán un título, una descripción y otros datos más específicos como el número de personas con las que el usuario la ha realizado, si la ha realizado en el exterior o el nivel de intensidad de la actividad.
    - Campos: id, user_id, title, description, company, outside, intensity, calories, duration, date_start, date_end.
    - Clave primaria: id.
    - Clave ajena: user_id > Usuarios.
- Pesos (weights): Tabla cuya función será la de almacenar un registro a modo de historial de las fluctuaciones del peso del usuario. Contendrá los pesos introducidos por el usuario y la fecha en la que se introdujeron.
    - Campos: id, user_id, measure, dateCreated.
    - Clave primaria: id.
    - Clave ajena: user_id > Usuarios.
- Mensajes (messages): Tabla que almacenará los mensajes tanto del usuario como del compañero distinguiendo la procedencia de estos si son de uno o del otro además se guardará la fecha de creación del mensaje.
    - Campos: id, user_id, text, type, transmitter, dateCreated.
    - Clave primaria: id.
    - Clave ajena: user_id > Usuarios.

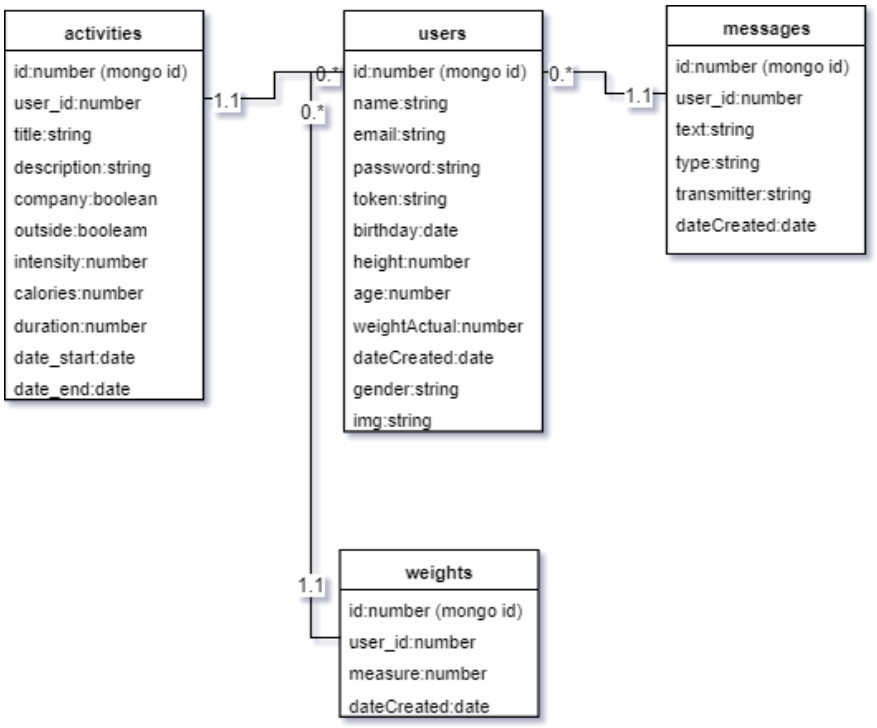

Figura 11. Esquema de la base de datos.

7.3. Diseño API REST

La parte de la APIREST (Representational State Transfer o Transferencia de Estado Representacional) se encontrará en el servidor de la aplicación y será la encargada de manejar los datos mediante llamadas basadas en el protocolo HTTP entre la parte del cliente y la base de datos. A continuación, se muestra una lista con los principales routings de la aplicación.

Nombre	Descripción	Método HTTP	URI	Cabecera	Cuerpo
Login: login	Autentificación para acceder a la aplicación.	POST	/login	Content-Type	email, password
Usuario: Registrar usuario	Crea un nuevo usuario.	POST	/register	Content-Type	name, email, password
Usuario: Comprobar token	Comprueba si el token del usuario es válido	POST	/token-verification	Authorization: userToken	

Usuario: Leer usuario	Devuelve la información de un usuario	GET	/user/userID	Authorization: userToken	
Usuario: Actualizar usuario	Actualiza la información del usuario	PUT	/user/userID	Content-Type, Authorization: userToken	Cualquier dato del usuario que se quiera actualizar: name, email, password, birthday, weight_actual, gender, img
Usuario: Eliminar usuario	Actualiza el estado del usuario a false.	DELETE	/user/userID	Authorization: userToken	
Actividad: Crear actividad	Crea una nueva actividad.	POST	/activity	Content-Type: application/json, Authorization: userToken	title, description, company, outside, intensity, calories, duration, date_start, date_end
Actividad: Leer actividad	Recibe la información de una actividad	GET	/activity/activityID	Authorization: userToken	
Actividad: Leer actividades	Recibe la información de varias actividades de forma paginada	GET	/activity?from=desde&limit=limite	Authorization: userToken	from, limit

Actividad: Actualizar actividad	Actualiza la información de una actividad	PUT	/activity/activityID	Content-Type: application/json, Authorization: userToken	Cualquier dato de la actividad que se quiera actualizar: title, description, company, outside, intensity, calories, duration, date_start, date_end
Actividad: Eliminar actividad	Elimina la actividad de la base de datos	DELETE	/activity/activityID	Authorization: userToken	
Actividad: Obtener totales	Devuelve el total de actividades, tiempo y calorías quemadas	GET	/activity-totals	Authorization: userToken	
Peso: Añadir peso	Añade un peso nuevo a la base de datos	POST	/weight	Content-Type: application/json, Authorization: userToken	measure, date_created
Peso: Actualizar peso	Actualiza la medida de un peso concreto	PUT	/weight/weightID	Content-Type: application/json, Authorization: userToken	Measure
Peso: Eliminar peso	Elimina el peso de la base de datos	DELETE	/weight/weightID	Authorization: userToken	

Peso: Obtener pesos gráfica	Obtiene los datos de los pesos para la gráfica de inicio	GET	/chart-weights	Authorization: userToken	
Mensaje: Enviar mensaje	Envía un mensaje al agente de dialogflow y recibe la respuesta	POST	/message	Content-Type: application/json, Authorization: userToken	dateCreated, text
Mensaje: Obtener mensajes.	Obtiene los mensajes de un usuario con el chatbot paginados	GET	/message?from=desde&limit=limite	Authorization: userToken	from, limit

Tabla 12. Api REST.

7.4. Diseño de la arquitectura tecnológica Front/Back-end

En este apartado se indica la arquitectura tecnológica que se ha definido y construido para el proyecto. Las tecnologías han sido escogidas en función de los requerimientos y necesidades de este.

7.4.1. Arquitectura tecnológica Front-end:

Para la parte del Front-end dado que el producto final del proyecto es una aplicación móvil la tecnología usada ha sido Ionic para el desarrollo de esta.

Una vez la aplicación ha sido generada para Android se ha decidido usar como alojamiento Play Store de Google para su posterior distribución.

7.4.2. Arquitectura tecnológica Back-end:

En cuanto a las tecnologías usadas en el desarrollo del Back-end del proyecto puedo dividirlas en las tecnologías usadas en la API REST, para la persistencia y por último para la parte de la inteligencia conversacional del chatbot.

Para el desarrollo de la API REST se ha utilizado NodeJS como entorno de ejecución de esta. Dentro de la API REST se ha usado Express para servir las peticiones requeridas desde y para la aplicación móvil y a modo de control de seguridad se ha usado Json Web Token en las peticiones para el manejo de los códigos de autentificación (tokens) de los usuarios.

Como tecnología para el alojamiento de la API REST se ha elegido Heroku que es una plataforma como servicio de computación en la Nube que soporta distintos lenguajes de programación.

En cuanto a la persistencia de la aplicación se ha usado MongoDB como gestor de bases de datos no relacional. Para alojar la base de datos, mLab ha sido el seleccionado, el cual es un servicio de base de datos en la nube totalmente administrado y que hospeda bases de datos MongoDB.

Por último, para el desarrollo del chatbot y de su inteligencia conversacional se ha usado como tecnología Dialogflow el cual está basado en tecnologías de procesamiento del lenguaje natural.

La siguiente figura muestra a modo de esquema la arquitectura del proyecto con el conjunto de tecnologías implementadas.

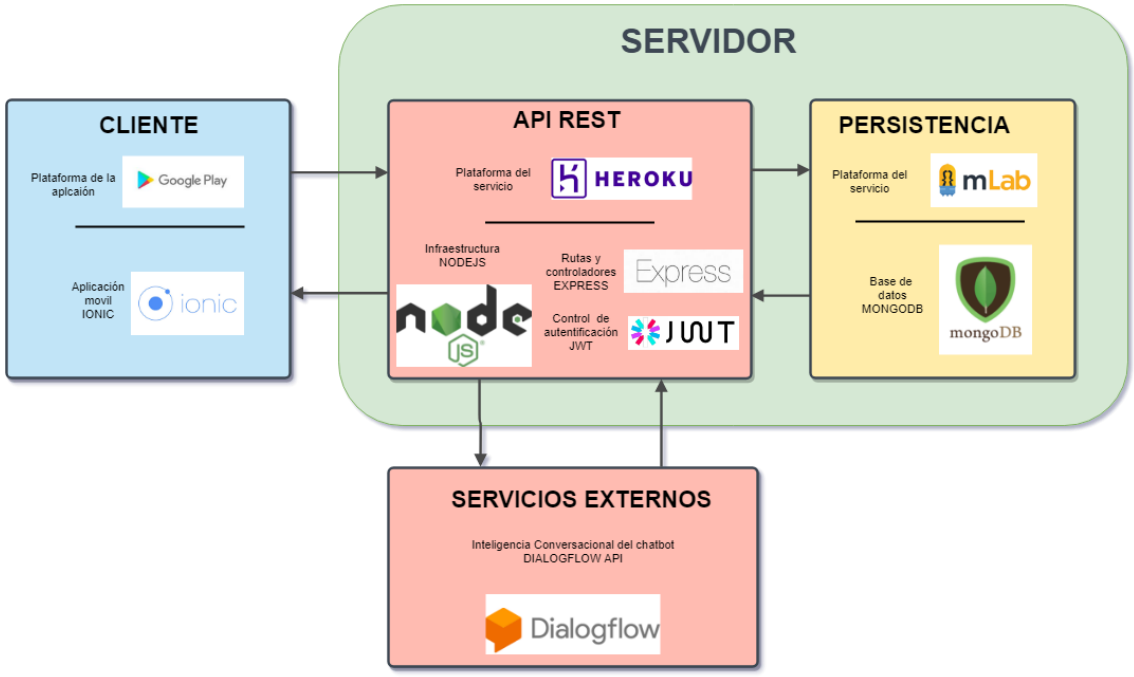

Figura 12. Stack tecnológico del proyecto.
(Fuente propia)

7.5. Diseño Interfaces

Antes de comenzar a realizar la aplicación se realizará el diseño de las interfaces para la toma de decisiones en etapas de diseño y no en desarrollo, lo cual agilizará esta última fase. En esos diseños se reflejarán los requisitos funcionales de la herramienta previamente descritos. Hay que tener en cuenta que cuanto mayor exactitud tenga el diseño de interfaces ahora, menos decisiones y desarrollos habrá que llevar a cabo después.

A continuación, se expondrán los diseños previos realizados explicando los requisitos aplicados en cada uno.

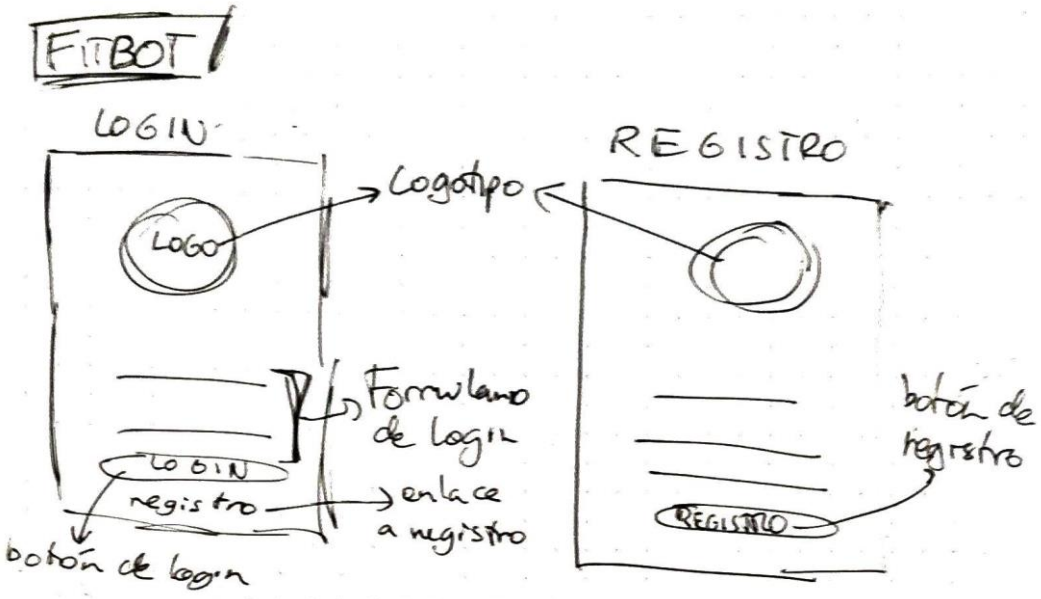

Figura 13. Mockup de Login y Registro.

La primera interfaz tras la pantalla de carga, la cual aparecerá al iniciar la aplicación mientras se cargan los recursos, será el login donde el usuario iniciará sesión para acceder a la aplicación. En caso de no disponer de cuenta propia con la que acceder habrá un enlace a la pantalla de registro donde podrá crear su propia cuenta mediante el nombre, correo electrónico y contraseña. Aquí se encontraría el requisito funcional RF05 en el cual el usuario podría iniciar sesión en la aplicación o registrarse previamente en caso de no disponer de una cuenta.

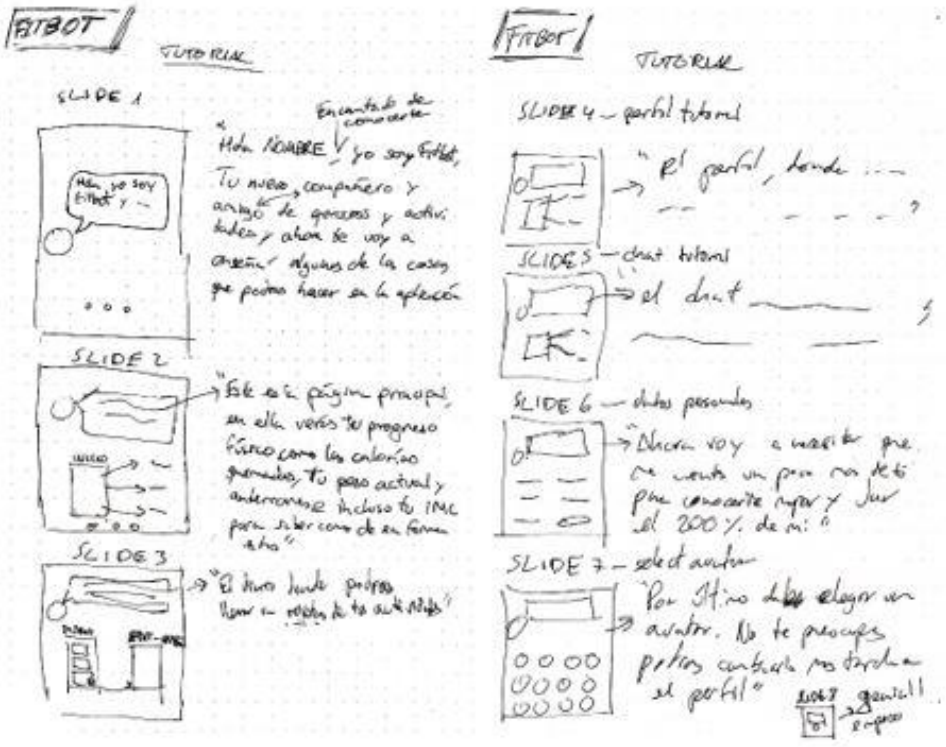

Figura 14. Mockups del tutorial.

Posteriormente al inicio de sesión, si es la primera vez que el usuario ha entrado en la aplicación accederá al tutorial. El cual consistirá en un conjunto de diapositivas donde el usuario recibirá información acerca del uso de las interfaces y además introducirá sus datos físicos para que la aplicación y el chatbot se adapten al usuario y su correcto funcionamiento. En esta interfaz se ven reflejados los requisitos funcionales RF03 y RF06 ya que el usuario a través de un tutorial dispondrá de una explicación del funcionamiento de la aplicación previa a su uso. Además, el usuario podrá registrar sus datos físicos para luego conocer su estado de salud.

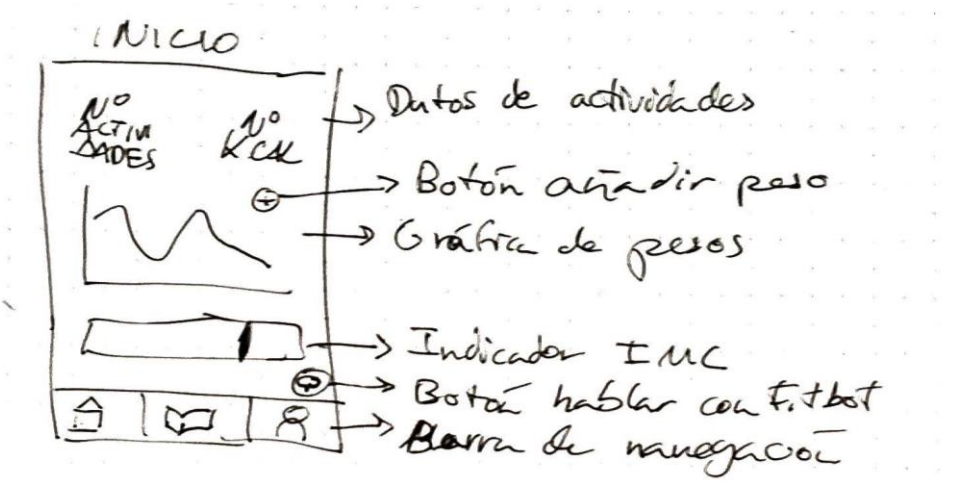

Figura 15. Mockup de Inicio.

Una vez haya iniciado sesión y completado el tutorial (en caso de ser la primera vez que accede a la aplicación) el usuario accederá a la interfaz de inicio de la aplicación donde se mostraran sus datos personales en cuanto a las actividades realizadas y los pesos registrados. También podrá ver mediante un indicador visual su Índice de Masa Corporal para evaluar su salud de un modo sencillo. En esta interfaz se aplican los requisitos funcionales RF03 y RF04 que hacen referencia al estado de salud del usuario y cálculo del IMC. Mediante esta interfaz el usuario puede ver su estado de salud en cuanto a calorías quemadas, historial de pesos e índice de masa corporal. Y al añadir un nuevo peso a la gráfica se procedería a calcular el nuevo IMC.

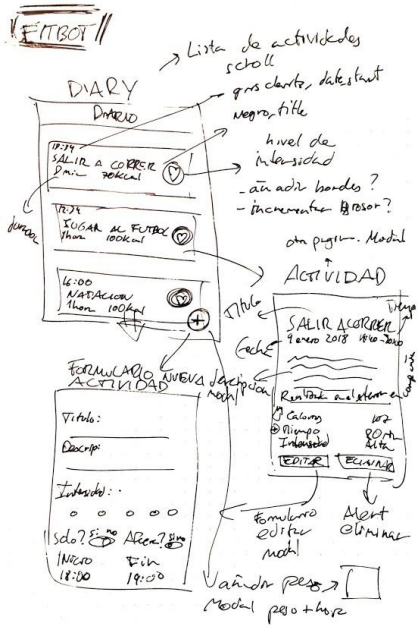

Figura 16. Mockup de Diario, Actividad y registro de actividad.

En la interfaz de diario es donde el usuario podrá ver el historial de las actividades realizadas y la información más relevante de las mismas como el título, duración, calorías, etc. Mediante un sistema de desplazamiento vertical podrá visualizar el total de sus actividades. Desde el botón inferior accederá a la interfaz de crear actividad desde la que podrá añadir una nueva interfaz al historial de estas. Y si presiona alguna de las actividades del diario accederá a la interfaz de la vista de la actividad.

En esta última ya se muestra la información total de la misma y dos botones, uno para editar la información en caso de querer corregir algún dato mal introducido y otro para eliminarla completamente. A lo largo de estas interfaces se lleva a cabo el requisito funcional RF02 el cual consiste en el control de actividades.

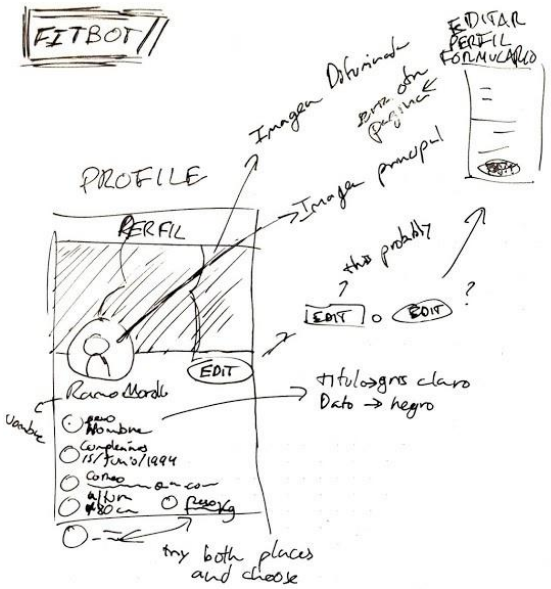

Figura 17. Mockup de Perfil y editar perfil.

En estas interfaces el usuario podrá ver y modificar sus datos personales desde los físicos como el peso o la altura a los no físicos como el nombre o el correo. También podrá cerrar la sesión y salir de la aplicación si lo desea. Los requisitos funcionales RF03 y RF04 están presentes en esta aplicación mediante la modificación de los datos ya que los datos físicos actúan en los cálculos del consumo de calorías por actividad y también en el cálculo del índice de masa corporal. También actúa el requisito funcional RF07 ya que el usuario podrá modificar aspectos como la imagen del perfil.

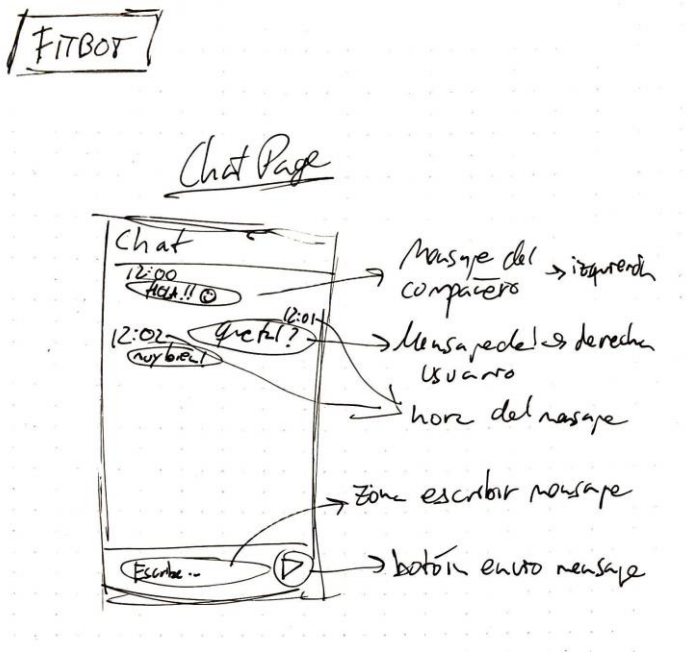

Figura 18. Mockup de chat.

A esta interfaz se accede a través del botón inferior con el icono de un mensaje para hablar con el compañero, en esta interfaz se representará el RF01, ya que es donde el usuario podrá interactuar con el compañero de actividades. Se mostrarán los mensajes en la parte superior de la aplicación con su correspondiente hora de envío mientras que en la parte inferior se enviarán los mensajes mediante un cuadro para introducir texto y un botón de envío.

7.6. Guías de estilos

Una guía de estilos es el apartado del proyecto donde se definen los estilos y formatos de la aplicación ya que las distintas interfaces deben tener una guía estructurada para no cometer errores a la hora de realizar el producto.

Colores

Dado que la aplicación tiene como objetivo realizar actividades físicas se ha decidido basar el estilo en una paleta de colores enérgicos como es el caso del naranja. Las tonalidades que se ha escogido provienen de la paleta de colores del naranja que viene definida por la guía de estilos de Material Design.

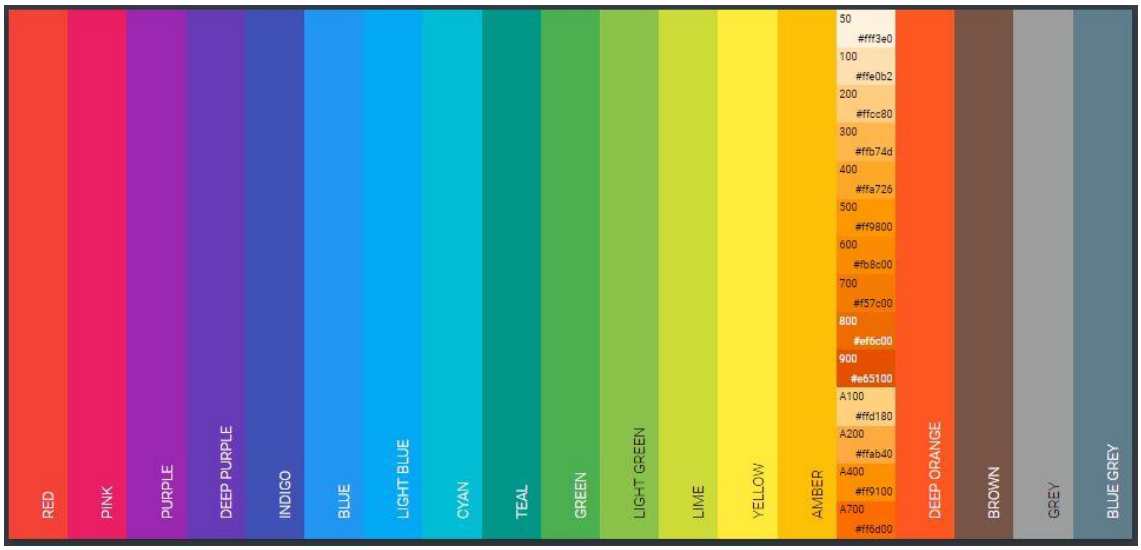

Figura 19. Paleta de colores de Material Design con las tonalidades del naranja resaltadas.
(Fuente https://www.materialpalette.com/colors)

Tipografía

Como familia de fuentes de texto para la aplicación se ha usado “Roboto” la cual es sans-serif y fue desarrollada por Google bajo la descripción de “fuente moderna, pero a la vez accesible y emocional”

Figura 20. Muestra de la familia de fuentes Roboto.
(Fuente https://es.wikipedia.org/wiki/Roboto)

Logotipo

Por último, para desarrollar el logotipo de la aplicación se han juntado los dos componentes principales de la aplicación que son el chatbot como compañero deportivo y la salud que es el objetivo de la realización de actividades físicas. Por parte de la salud se ha elegido un corazón y por el compañero una cara impersonal.

Figura 21. Logotipo de Fitbot.
(Fuente propia)

Para realizarlo se ha usado la herramienta gratuita y de código abierto Inkscape la cual te permite crear y editar gráficos vectoriales, diagramas o logotipos.

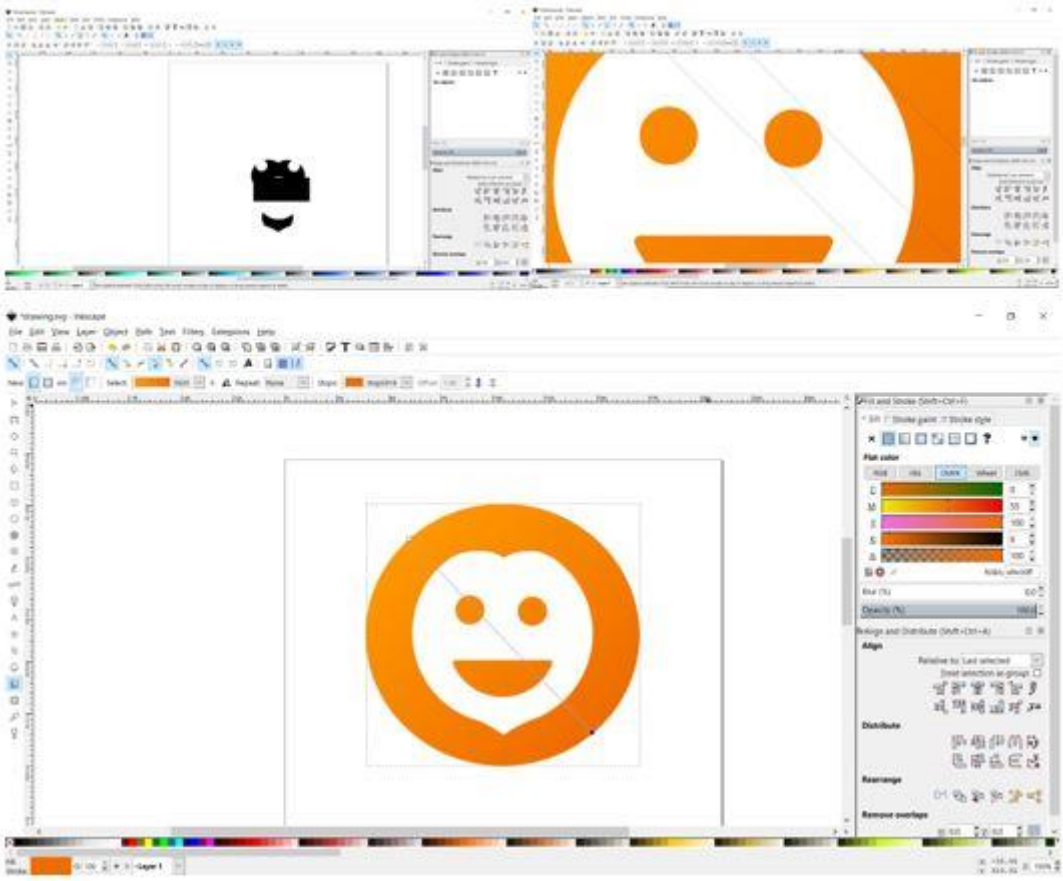

Figura 22. Conjunto de pasos del proceso de creación del logo.
(Fuente propia)

8. Implementación

En este apartado se relata el proceso de desarrollo que se ha llevado a cabo para realizar el proyecto una vez acabadas las fases de anteriores y teniendo un diseño claro en el que basarse. Se va a dividir la implementación en las etapas y partes en las que se han ido realizando.

8.1. Entorno de desarrollo.

Como entorno de desarrollo de la aplicación se ha usado un ordenador personal el cual tiene como sistema operativo Windows 10. Para el editor de texto se ha usado tanto Atom como Visual Studio Code cada uno con plugins añadidos para facilitar la creación y edición de código.

Durante la fase de desarrollo se ha utilizado el mismo ordenador como servidor para la APIREST y la base de datos. El navegador Google Chrome ha servido para visualizar la aplicación en desarrollo y Android Studio para generar una versión instalable y así visualizarla en un móvil.

También se ha de destacar el uso de Git y GitHub para el manejo de las versiones del código tanto de la API como de la parte del cliente.

8.2. Implementación de la Base de datos.

Para la base de datos se ha instalado en un ordenador MongoDB y se ha creado una base de datos en un servidor local para el desarrollo de la aplicación. Se ha usado Robo3T, una herramienta diseñada para el manejo de datos de la base de datos NoSQL de mongo.

8.3. Implementación de la APIrest

Para comenzar se ha instalado NodeJS el cual viene con un instalador de dependencias propio denominado NPM (Node Package Manager). Después de las configuraciones iniciales se han instalado las dependencias de NodeJS necesarias para el desarrollo de la APIrest.

- **Express**: Framework para crear servicios RESTful en NodeJS mediante el uso de peticiones HTTP. Se usará para crear las rutas de las peticiones y respuestas.
- **Body-parser:** Se ha usado para el manejo de los parámetros en las peticiones http.
- **Bcrypt:** Librería para el uso de la encriptación de la contraseña del usuario.

- **Dialogflow:** Librería oficial de Dialogflow. Se ha usado para conectar la API con el agente y su corpus conversacional.
- **Jsonwebtoken:** Se ha hecho uso de ella a la hora de tratar con los tokens del usuario en las peticiones y también en la comprobación del inicio de sesión.
- **Mongoose:** Pluguin que ha sido de ayuda a la hora de manejar la base de datos de Mongo desde Node y ha aportado métodos de gran utilidad a la hora de crear los modelos.
- **Nodemon:** Extensión de desarrollo que recarga automáticamente el script cuando sucede algún cambio.

Se ha estructurado la API en subdirectorios para una mejor organización del código y entendimiento posterior de la traza de las peticiones.

- **Configuración:** Es el directorio donde se guarda el archivo con las configuraciones generales de la API como el puerto donde sirve la API, la conexión a la base de datos o las credenciales para la conexión con Dialogflow.
- **Rutas:** Aquí es donde se ejecuta toda la lógica de la Api ya que es donde se encuentran las peticiones que se realizan y la lógica y formación de la respuesta de cada una.
- **Dialogflow:** Aquí se incluyen las peticiones de la API contra la API de Dialogflow como el envió de mensaje y su flujo de intents.
- **Middlewares:** Los middlewares son los scripts que son llamados como punto intermedio en algunas peticiones como por ejemplo la comprobación del token de un usuario válido en la mayoría de las rutas de la API.
- **Modelos:** Aquí están definidos los modelos de mongo y se usan a la hora de hacer peticiones CRUD como por ejemplo a la hora de crear un usuario. Es importante que la construcción de los modelos sea lo más precisa posible ya que luego evitará código extra como la comprobación de si los campos introducidos en un modelo son correctos.

8.4. Implementación de la aplicación

Para realizar la aplicación se ha comenzado instalando el cliente de instalación de Ionic desde el manejador de paquetes de Node. También se han instalado dependencias como la librería de gráficas de Google para Angular: Angular 2 Google Charts, ya que Ionic está basado en Angular.

La aplicación se ha estructurado en varios directorios que engloban funcionalidades y objetivos similares.

- **Configuración:** Directorio en el cual se define información referente a la configuración global de la aplicación como la URL de la APIREST.
- **Modelos:** Aquí están definidos los modelos de los objetos y clases usados en la aplicación como el usuario o sus actividades.
- **Páginas:** Probablemente el directorio más importante de la aplicación ya que contiene a su vez los directorios de las páginas de la aplicación. Dentro del directorio de una página como la del login o el perfil se encuentran los archivos referentes a las vistas de estas y la lógica computacional.
- **Servicios:** En este directorio se programan las peticiones a la API y otros servicios externos en archivos que luego se llaman en las páginas necesarias.
- **Assets:** En esta localización se encuentran archivos de diseño como los avatares de los usuarios o imágenes varias usadas a lo largo de la aplicación.

A la hora de calcular las calorías de una actividad se ha tomado como base el MET (Metabolic Equivalent of Task), que es la unidad de medida del índice metabólico y se define como la cantidad de calor emitido por una persona en posición sedente por metro cuadrado de piel. Esta relación por metro cuadrado de piel permite una mayor aproximación a la media, a mayor tamaño mayor metabolismo basal.

Por tanto, se ha establecido una media de mets para cada nivel de intensidad para así posteriormente calcular en número de calorías en base a los mets, el peso y los minutos de actividad del usuario.

- Intensidad baja 2 mets.
- Intensidad media 4.5 mets.
- Intensidad alta 8 mets.

De esta fórmula se obtienen los mililitros de O2 que posteriormente se pasarán a litros y serán multiplicados por 4,56 kcal, quedando como resultado final la siguiente fórmula:

*Energía en Kcalorías = ((nº METs x 3.5 (ml O2) x minutos de AF x Peso del sujeto) / 1000) * 4.56*

8.5. Implementación del compañero

Para desarrollar e implementar el asistente en la aplicación se ha usado Dialogflow. En esta herramienta se ha creado un nuevo proyecto de Google y un nuevo asistente de dialogflow.

No se ha tomado ningún modelo base de los que proporciona Dialogflow y se ha decidido crear el corpus conversacional desde un asistente en blanco. El corpus conversacional está compuesto por intents, los cuales se podrían explicar cómo intenciones del usuario con el asistente. Se han dividido los intents de Fitbot en 5 contextos:

- **Generales.** Estos intents componen conversaciones de carácter coloquial y sin una funcionalidad explicita como preguntar como estas, el nombre, etc.
- **Actividad.** Intents relacionados con las actividades como crear una actividad en la aplicación o pedir al compañero que te recomiende alguna actividad a realizar.
- **Peso.** Relacionados con la funcionalidad de la aplicación de añadir o modificar el peso del usuario.
- **Ajustes.** Intents relacionados con ajustes de la aplicación como cambiar el avatar del usuario por uno distinto.
- **Por defecto**. Estos intents son los fundamentales en el corpus del chatbot, entre ellos se encuentran el intent de saludo el cual se llama al iniciar la conversación y el Default Falback, este último es el que aparece cuando el compañero no entiende lo que el usuario le ha dicho.

Mediante la detección de ciertos parámetros en estos intents, se realizan ciertas funciones en la aplicación y en la API como la selección de un nuevo avatar o la recomendación de actividades autocompletadas.

8.6. Entorno de producción

Una vez acabada la fase de desarrollo se ha generado un ejecutable de la aplicación poder desplegarlo en la Play Store de Google, se ha priorizado el despliegue en Android antes que en iOS debido a las ayudas que proporciona el primero a los desarrolladores, desde herramientas hasta el mismo precio para añadir tu aplicación a la tienda siendo de 20 euros en la Play Store de Android frente a los 100 de la App Store de iOS.

En cuanto a la persistencia de la aplicación se ha subido la base de datos de mongoDB a mLab el cual es un servicio de base de datos en la nube totalmente administrado que sirve para hospedar bases de datos de MongoDB. MLab te permite escoger el proveedor de nube en el cual quieres ejecutarla siendo las opciones Amazon, Google y Microsoft Azure.

Por la parte de la APIrest, para que la aplicación funcionase correctamente una vez se encuentre en la Play Store de Google, se ha decidido subir a Heroku; una plataforma que funciona como

servicio computacional en la nube y soporta distintos lenguajes de programación, entre ellos soporta NodeJS. Una vez la Api ha sido subida a Heroku se ha apuntado la aplicación a la URL proporcionada por el servicio.

9. Pruebas y validación

Una vez se ha acabado la aplicación y ha sido lanzada a producción se ha comenzado la fase de pruebas para revisarla y medir su funcionamiento con usuarios reales. Tras una primera fase de pruebas unitarias las siguientes pruebas han sido pruebas de integración con usuarios reales. Estos tras la presentación de la aplicación y el tutorial se han desplazado por las principales interfaces, han creado actividades e interactuado con el asistente.

Aquí se exponen algunos ejemplos de los problemas que han surgido de las pruebas de usuarios y la solución que se ha llevado a cabo para solventarlos.

- El botón para acceder a hablar con el chatbot se encuentra solo en la pantalla de inicio. Esto generaba una mala experiencia de usuario ya que para interactuar con el compañero había que volver a la pantalla de inicio estuvieses donde estuvieses. Se tomó la decisión de implementar el botón en el resto de las interfaces principales como el perfil y el diario y el resultado fue favorable ya que los usuarios interactuaban más con el asistente y se pudo mejorar el corpus conversacional más rápido, aparte de mejorar también la experiencia de usuario.
- Otro elemento que empeoraba la experiencia de usuario era un mensaje que aparecía la primera vez que accedías a la aplicación cuando el usuario aún no había realizado ninguna actividad que te sugería de añadir una. En principio esta acción pretendía mejorar la experiencia de usuario mostrándole primero la interfaz para crear una actividad y guiarle para crearla. El problema era que si el usuario rechazaba la sugerencia cada vez que volvía a abrir la aplicación o volvía a la interfaz principal y aún no había creado una actividad el mensaje le aparecía convirtiéndose en un elemento molesto. La solución que se tomó fue almacenar la decisión tomada y en el caso de rechazar la sugerencia ya no volvía a aparecer.
- Otro problema que encontré fue que en algunos usuarios la gráfica no se mostraba y generaba un mensaje de error que no se había advertido durante la fase de desarrollo y primeras pruebas. Entonces se procedió a investigar cómo solucionarlo y una vez se hubo resuelto se generó una nueva versión de la aplicación con el error resuelto.

También mencionar que a través de los mensajes guardados en la base de datos de las conversaciones que mantenían los usuarios con el asistente se ha podido ver en qué momentos

el compañero no respondía correctamente a las preguntas de los usuarios y se han añadido respuestas a esas preguntas para entrenar la parte conversacional y mejorar el corpus del asistente.

Que me recomiendaa hacer hoy?	normal	USER	2019-03-28 ...
Ups, no he entendido a que te refieres.	normal	BOT	2019-03-28 ...
Que me recomiendas hacer?	normal	USER	2019-03-28 ...
Ups, no he entendido a que te refieres.	normal	BOT	2019-03-28 ...

Figura 23. Ejemplo de mensajes sin una respuesta apropiada guardados en la base de datos. (Fuente propia)

10. Resultados

El resultado final del proyecto ha sido una aplicación lanzada a producción en la Play Store que consta de una serie de interfaces que hacen que funcione adecuadamente y cumpla los requisitos propuestos.

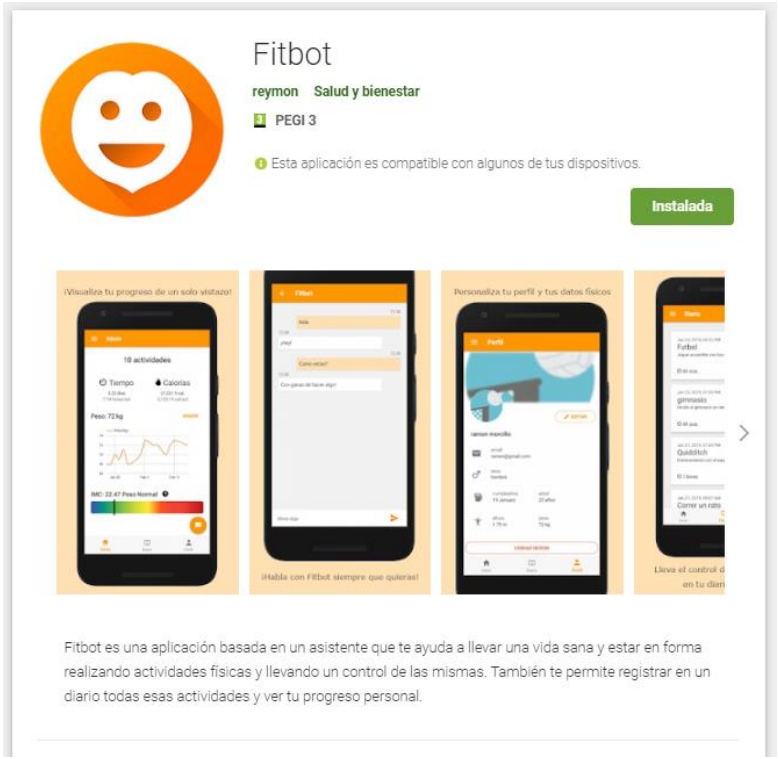

Figura 24. Fitbot en la Play store. (Fuente https://play.google.com/store/apps/details?id=com.ramonmorcillo.fitbot01)

En los siguientes apartados se describirán las principales interfaces de la aplicación.

10.1. Inicio de la aplicación

En este apartado se han englobado aquellas interfaces que el usuario ve por primera vez como la pantalla de carga, la de iniciar sesión y la de registro. Las tres contienen el logo principal de la aplicación y las de Login y registro contienen un formulario para iniciar sesión y registrarse respectivamente.

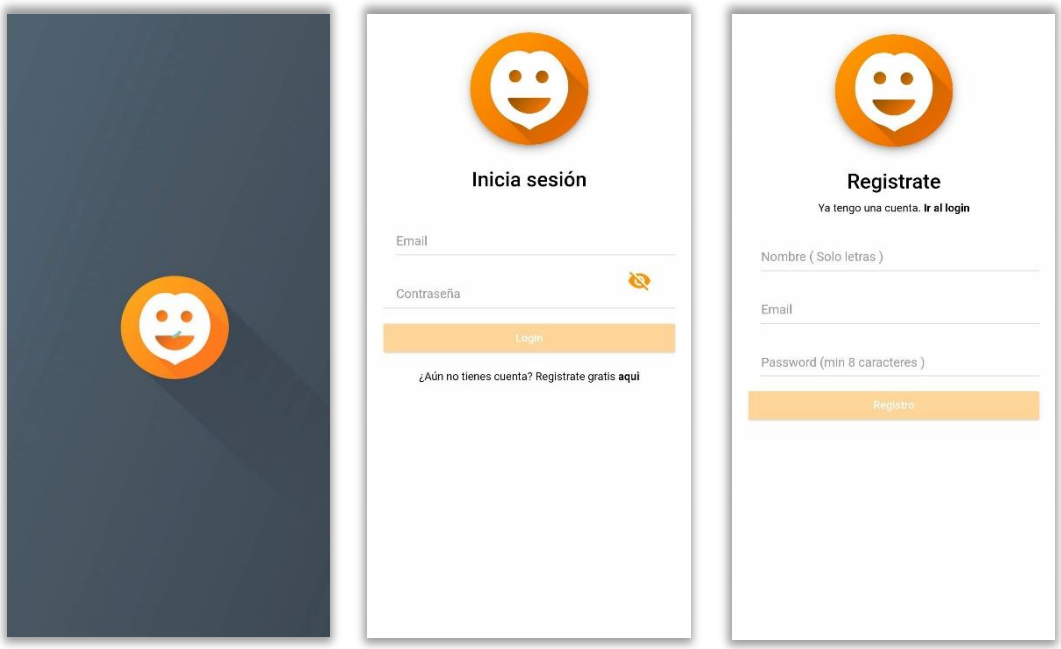

Figura 25. Pantalla de carga, Login y Registro.
(Fuente propia)

Lo siguiente que se encuentra el usuario una vez inicia sesión por primera vez en nuestra aplicación es el tutorial de esta. Este cuenta con 7 diapositivas explicativas e interactivas en las que el asistente guía de un modo cercano y coloquial al usuario por las principales interfaces de la aplicación.

Figura 26. Diapositivas 1, 2 y 3 del tutorial.
(Fuente propia)

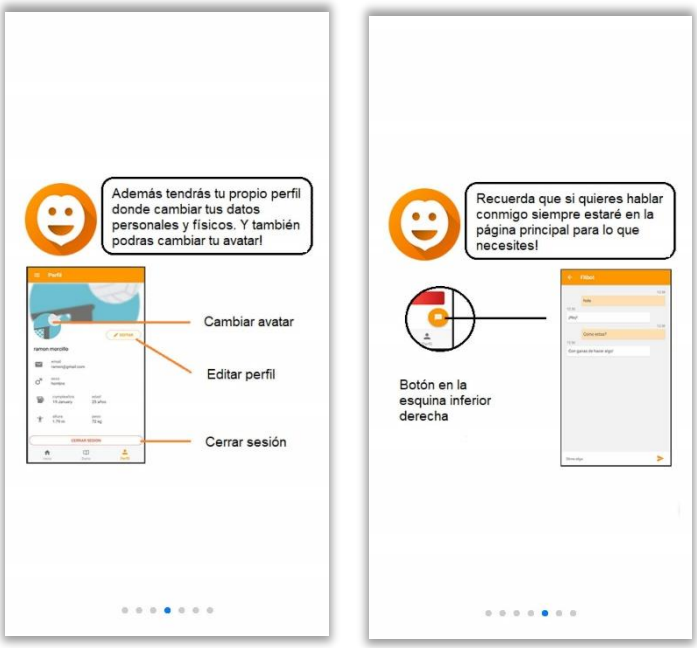

Figura 27. Diapositivas 4 y 5 del tutorial.
(Fuente propia)

Con esto se consigue que el usuario tenga una idea general de las funcionalidades de la aplicación y de lo que puede hacer con ella previo a su uso. Para finalizar el tutorial, en las últimas dos diapositivas se le pide que introduzca el resto de los datos necesarios y que seleccione un avatar personal.

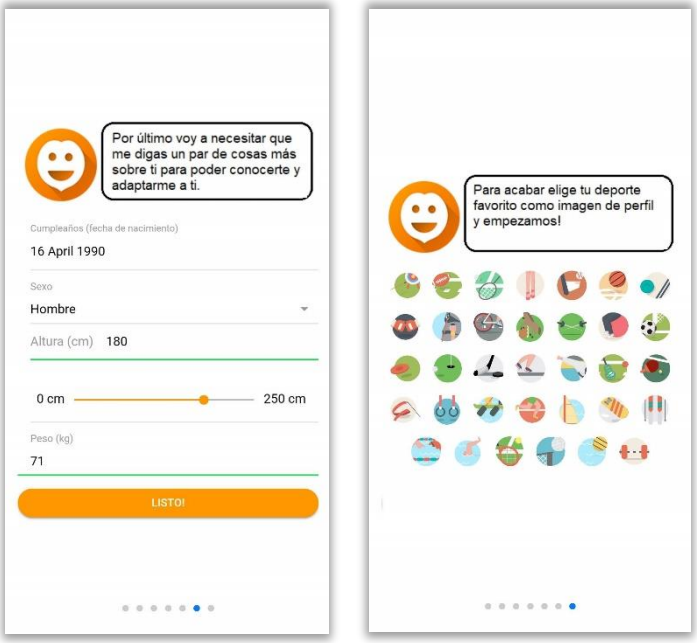

Figura 28. Diapositivas 6 y 7 del tutorial.
(Fuente propia)

10.2. Pantalla de Inicio

Una vez el usuario haya iniciado sesión de manera correcta y acabado el tutorial, se encontrará con la interfaz principal de Inicio. Esta interfaz se adecua mucho a los mockups realizados en la fase de diseño y su función principal sigue siendo la de poder ver nuestro progreso y estado físico de un modo compacto y resumido.

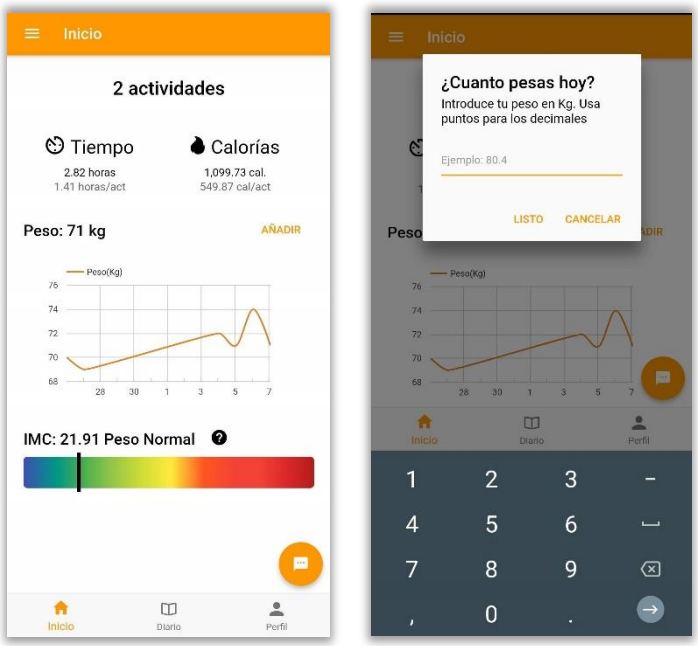

Figura 29. Interfaz de Inicio y Formulario de añadir un nuevo peso. (Fuente propia)

10.3. Gestión de actividades

Las interfaces agrupadas en esta sección son las que forman parte del registro de actividades, edición e historial de las mismas. La interfaz principal sería la del diario en la cual el usuario puede ver las actividades que ha realizado, así como una breve información de cada una. Si el usuario presiona el botón de con el símbolo de suma podrá acceder a la interfaz de añadir una nueva actividad al diario. En un principio en los mockups el botón estaba situado en la zona inferior derecha, no central, de la interfaz; pero tras las primeras pruebas con usuarios reales se llegó a la conclusión de que mejoraría la comunicación con el compañero el poder hablar con él a través de un botón en todas las interfaces, no solo la de inicio. Es por ello por lo que el lugar previo del botón de añadir actividad fue ocupado por el de hablar con el compañero para situarlo en el mismo sitio en todas las interfaces principales.

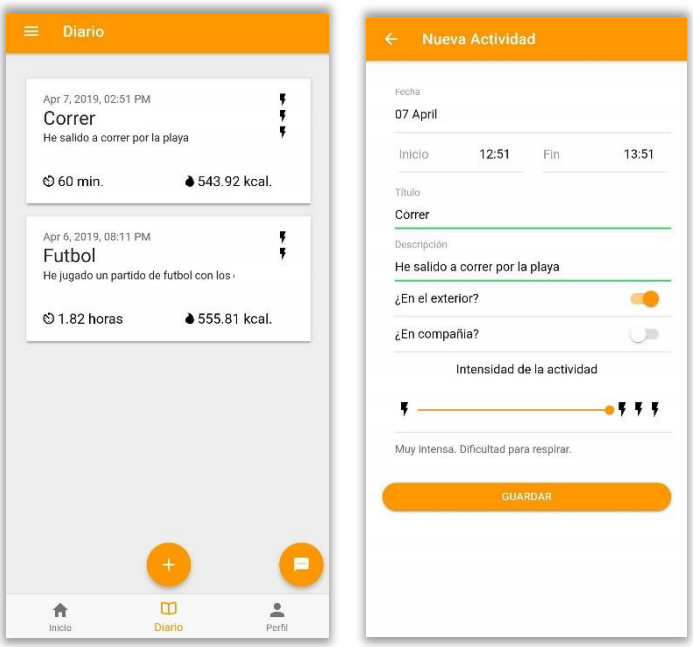

Figura 30. Interfaz de Diario e interfaz de Añadir nueva actividad.
(Fuente propia)

El usuario también puede acceder a la Interfaz de la actividad con la información de esta presionándola en el diario. Dentro de esta interfaz, además de ver los datos de esta, podrá editarla si ha introducido algún dato erróneo o eliminarla si lo considera necesario.

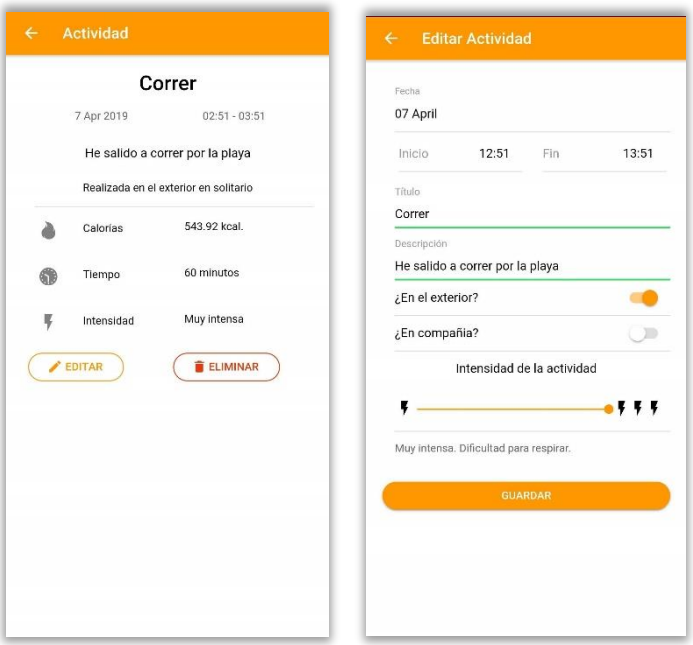

Figura 31. Interfaz de Actividad e interfaz de editar actividad.
(Fuente propia)

10.4. Perfil de usuario

En esta interfaz el usuario puede ver sus datos personales como el nombre, correo electrónico o edad entre otros. Además, pulsando el botón de editar podrá acceder a la interfaz de editar perfil donde podrá cambiar los datos que había introducido. También podrá cerrar sesión en la aplicación si lo desea presionando el botón de cerrar sesión situado en la zona inferior. Por último, presionando su imagen de perfil accederá a la interfaz de selección de avatar para seleccionar uno nuevo si lo desea. Al igual que en el diario a esta interfaz se le añadió un botón en la esquina inferior izquierda para acceder a la interfaz de conversación con el chatbot.

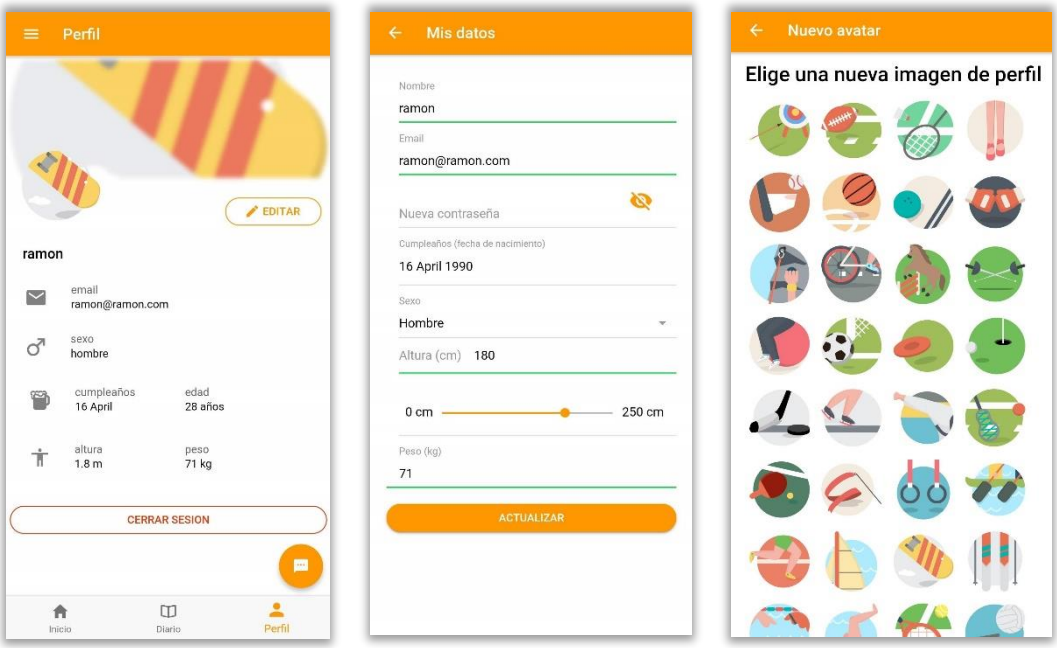

Figura 32. Interfaz de Perfil, Editar perfil y Seleccionar avatar.
(Fuente propia)

10.5. Conversación con el compañero

A esta interfaz en un principio se accedía solamente desde la pantalla de inicio, pero por motivos previamente explicados que mejoraban la usabilidad y la experiencia del usuario se decidió crear un acceso desde las tres interfaces principales: Inicio, Diario y Perfil. Una vez el usuario presiona el botón y accede a esta interfaz el compañero le envía un mensaje para iniciar la conversación con el usuario. Desde esta interfaz se pueden acceder a otras como por ejemplo si se le pregunta al compañero temas como la creación de una actividad, el compañero guiará al usuario a la interfaz de crear actividad.

Figura 33. Interfaz de Conversación con el compañero.
(Fuente propia)

10.6. Información de la aplicación

Esta interfaz no tenía previsto ser implementada en un principio, pero luego se decidió que estaría bien una interfaz con información acerca de la aplicación y la versión actual de la misma, además de las versiones anteriores.

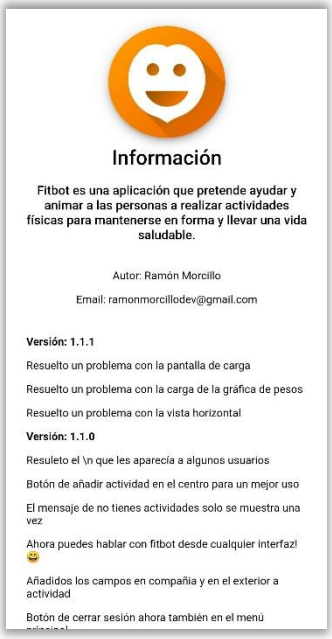

Figura 34. Interfaz de Información de la aplicación.
(Fuente propia)

11. Conclusiones y trabajo futuro

A modo de conclusión del trabajo de fin de grado he de decir que estoy satisfecho con el resultado final ya que he logrado alcanzar los objetivos que se habían propuesto inicialmente.

La aplicación cumple el objetivo principal de gestionar y motivar al usuario a realizar actividades físicas con un asistente que este contigo en cualquier momento a modo de compañero. El usuario puede guardar sus datos personales, ver su historial de pesos a modo de gráfica, su estado de salud, su historial de actividades donde puede añadir o editar las actividades que va realizando y por supuesto hablar con el compañero de actividades. Me hubiera gustado añadir una parte de gamificación, pero por temas de falta de recursos temporales al final no se ha logrado implementar.

La API REST es totalmente funcional y sirve las peticiones que recibe desde la aplicación sin problema. Además, se ha conseguido crear y mantener la base de datos de MongoDB correctamente. Y en cuanto al compañero de actividades se ha logrado crear y entrenar un asistente virtual totalmente conversacional para ayudar y animar al usuario.

Con esta aplicación se ha aprendido el valor que tiene la enseñanza online y el ser autodidacta ya que el conocimiento acerca de las tecnologías usadas para realizar el proyecto lo adquirí a base de cursos online tanto para hacer la aplicación en Ionic, la APIREST en NodeJS, la base de datos en MongoDB y el corpus conversacional de Dialogflow.

Pero no todo el conocimiento lo he adquirido por mi cuenta, las bases de este las he adquirido a lo largo de la carrera en asignaturas como Usabilidad y accesibilidad para temas de diseño e interfaces de usuario. También en Programación del Cliente Web donde aprendí a programar en JavaScript y a realizar peticiones y otras funcionalidades. Y por supuesto en Servicios Multimedia Avanzados donde aprendí sobre el desarrollo de aplicaciones web tanto la parte de Front-end como la de Back-end pertinente a la APIREST.

11.1. Trabajo futuro

Ningún trabajo o proyecto se finaliza al cien por cien y este no es la excepción. Aunque se ha alcanzado una fase de producción funcional del producto, que en un principio no era seguro si llegaría por falta de conocimiento y recursos como el tiempo, aún se le podrían aplicar muchas

mejoras en un futuro. Si bien es cierto que no se pretende obtener beneficio económico alguno mediante la aplicación, existen ciertos apartados que sería interesante implementar.

- Interfaz Web. El siguiente paso o mejora funcional de la aplicación sería la creación de una aplicación web para la plataforma. La aplicación tendría las mismas funcionalidades que la versión móvil pero el usuario podría acceder a ella desde el navegador.
- Acceso a un registro de las conversaciones. El usuario podrá ver las conversaciones mantenidas con el asistente y así poder releer consejos útiles que el asistente le dio en un pasado.
- Mejorar la interactividad del asistente. Añadir funcionalidades al asistente como la de iniciar una conversación con el usuario si llevan tiempo sin usar la aplicación o sin hablar con el compañero. De este modo fomentaría el uso de la aplicación y que la tasa de desuso y desinstalaciones no bajara drásticamente.

12. Bibliografía y Referencias

En esta sección se incluyen las obras y materiales consultados y empleados en la elaboración de la memoria.

1. *Angular.* Herramienta para el desarrollo de aplicaciones. Disponible en: https://angular.io/
2. *Bcrypt.* Librería de JavaScript para la encriptación de contraseñas. Disponible en: https://www.npmjs.com/package/bcrypt
3. *Clockify.* Herramienta para registrar el tiempo y la productividad. Disponible en: https://clockify.me
4. *Dialogflow.* Desarrollador de lenguaje natural a través de machine learning. Disponible en: https://dialogflow.com/
5. *Ejercicios en casa.* Aplicación Android. Disponible en: https://play.google.com/store/apps/details?id=homeworkout.homeworkouts.noequipment
6. Virtualgym. *Equivalente metabólico*. Disponible en: https://virtuagym.zendesk.com/hc/es/articles/201185502-Equivalente-Metab%C3%B3lico
7. *Fitbot.* Disponible en: https://play.google.com/store/apps/details?id=com.ramonmorcillo.fitbot01
8. *FitCircle*. Aplicación Android. Disponible en: http://www.fitcircle.in
9. Statista. *Number of available applications in the Google Play Store from December 2009 to March 2019.* Gráfico del consumo de aplicaciones móviles. Disponible en: https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/
10. *Google fit.* Aplicación Android. Disponible en: https://play.google.com/store/apps/details?id=com.google.android.apps.fitness
11. *GymBot.* Aplicación Android. Disponible en: http//gymbot.io
12. *Ionic.* Herramienta para el desarrollo de aplicaciones móviles. Disponible en: https://ionicframework.com/
13. *Json web token.* Decodifica, verifica y genera JWT. Disponible en: https://jwt.io

14. *Color Sport Elements.* Pack de iconos usado para los avatares. Disponible en: https://www.flaticon.es/packs/color-sport-elements
15. *MongoDB.* Sistema de base de datos NoSQL. Disponible en: https://www.mongodb.com
16. Verywellfit. *Using Metabolic Equivalent for Task (MET) for Exercises*. Como usar el met en los ejercicios. Disponible en: https://www.verywellfit.com/met-the-standard-metabolic-equivalent-3120356
17. *Metabolic equivalent of task (MET).* Wikipedia. Disponible en: https://en.wikipedia.org/wiki/Metabolic_equivalent
18. *Metodología scrum.* Softeng. Disponible en: https://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum.html
19. *mLab.* Servicio de base de datos MongoDB en la nube. Disponible en: https://mlab.com
20. *Moment.js*. librería de JavaScript para el manejo de fechas y tiempos. Disponible en: https://www.npmjs.com/package/moment
21. *Mongoose.* Modelado de objetos de Mongodb para Node.js. Disponible en: https://mongoosejs.com
22. *Ng2-google-charts*. Módulo de gráficos de Google para Angular. Disponible en: https://www.npmjs.com/package/ng2-google-charts
23. *NodeJS*. Entorno de ejecución para JavaScript Disponible en: https://nodejs.org
24. *Npm*. Manejador de paquetes por defecto para Node.js. Disponible en: https://www.npmjs.com
25. *Postman*. Herramienta para análisis de API REST. En forma de plugin para Chrome y aplicación. Disponible en: https://www.getpostman.com
26. Verywellfit. *Calculating how many calories you burn during exercise.* Cálculo de la quema de calorías durante ejercicios. Disponible en: https://www.verywellfit.com/how-many-calories-you-burn-during-exercise-4111064
27. *Trello*. Software de administración de proyectos. Disponible en: https://trello.com
28. *Underscore.js.* Librería de JavaScript que proporciona funcionalidades extra al lenguaje. Disponible en: https://underscorejs.org/