Universidad de Salamanca

Máster Universitario en Ingeniería Informática

Trabajo de Fin de Máster

PLATAFORMA PARA LA GESTIÓN ONLINE DE ACTIVIDAD FÍSICA DE CLIENTES

Septiembre de 2021

Autor

Mario González Morán

Tutor

Pablo Chamoso Santos

D. Pablo Chamoso Santos:

CERTIFICA

Que el trabajo titulado "Plataforma para la gestión online de actividad física de clientes" ha sido realizado por Mario González Moran y constituye la memoria del trabajo realizado para superar la asignatura Trabajo de Fin de Máster del Máster Universitario en Ingeniería Informática de la Universidad de Salamanca.

Y para que así conste a todos los efectos oportunos.

Salamanca, a 2 de septiembre de 2021

Fdo.: Pablo Chamoso Santos

Resumen

En el último año, los españoles han aumentado los malos hábitos debido al confinamiento y la pandemia. Estos malos hábitos se relacionan principalmente con la comida y el sedentarismo provocado por las malas noticias que afectan a la salud mental.

Esta reducción de la actividad física ha provocado que el Ministerio de Sanidad haya lanzado una serie de consejos para estar más activo durante el día, como hacer las tareas del hogar, bailar, hacer estiramientos, no estar más de dos horas al día seguidas sentado o salir a la calle en bici, andando o bajar por las escaleras en lugar de en ascensor [1].

Una propuesta muy interesante y cómoda para muchas personas, ya que por tiempo u otro motivo no tienen otra opción, puede ser la de contratar un entrenador personal "virtual". Esta persona podría llevar el seguimiento de varios usuarios de forma individual a través de una aplicación web pudiendo asignar tareas personalizadas, hábitos alimenticios saludables y un estricto control de las ejecuciones de estos y de los resultados.

Para muchas personas, disponer de un entrenador personal sin tener que salir de casa puede ser una gran ventaja, sobre todo para evitar el contacto con personas no convivientes en un gimnasio u otro lugar, o tener mejor disposición de horarios.

En este trabajo de final de máster se ha desarrollado una aplicación web que facilite la comunicación cliente-entrenador personal. La aplicación desarrollada engloba todas las ideas relacionadas con el entrenamiento personal y las guías para llevar a cabo una buena alimentación. Dispone de diferentes planes de pago que incluyen desde ejercicios personalizados según las características personales, videos o documentos explicativos, pautas de alimentación, seguimiento mediante imágenes o datos estadísticos y un chat personal entre el cliente y el entrenador personal al cargo.

Palabras clave: salud, actividad física, entrenador personal, tareas personalizadas, buena alimentación, control, seguimiento.

Abstract

In the last year, the Spanish have increased bad habits due to confinement and the pandemic. These bad habits are mainly related to food and sedentary lifestyles caused by bad news affecting mental health. This reduction in physical activity has prompted the Ministry of Health to launch a series of tips on how to be more active during the day, such as doing housework, dancing, stretching, not sitting for more than two hours a day, cycling, walking or taking the stairs instead of the lift [1].

A very interesting and convenient proposal for many people, who for time or other reasons have no other option, may be to hire a "virtual" personal trainer. This person could follow up several users individually through a web application and could assign personalised tasks, healthy eating habits and a strict control of their performance and results.

For many people, having a personal trainer at their disposal without having to leave home can be a great advantage, especially to avoid contact with other people who do not live with them in a gym or other place, or to have a better timetable.

In this Master's thesis, a web application has been developed to facilitate client-personal trainer communication. The application developed encompasses all the ideas related to personal training and guidelines for good nutrition.

It has different payment plans that include personalised exercises according to personal characteristics, videos or explanatory documents, dietary guidelines, monitoring through images or statistical data and a personal chat between the client and the personal trainer in charge.

Keywords: health, physical activity, personal trainer, personalised tasks, good nutrition, monitoring, follow up.

Índice general

Índice de figuras

Índice de tablas

1. Introducción

El presente documento recopila la memoria del Trabajo de Fin de Máster titulado **Plataforma para la gestión online de actividad física de clientes,** desarrollado por el alumno Mario González Morán entre mayo y agosto de 2021 para la obtención del título del Máster en Ingeniería Informática en la Universidad de Salamanca.

El principal objetivo de este Trabajo de Fin de Máster es el desarrollo de una aplicación web en la que pueda existir múltiples clientes y uno o varios entrenadores personales que ofrezcan sus servicios a través de diferentes planes personalizados. Esta aplicación permitirá el acceso a dichos clientes y entrenadores de forma individual y permitirá la gestión de los servicios disponibles.

Para obtener este objetivo se ha dividido el desarrollo del trabajo en dos partes:

- **API REST.** Permite la conexión entre la base de datos MySQL y la parte del Frontend de manera sencilla.
- **Frontend.** Muestra los datos obtenidos y permite la interacción con ellos a través de la API con el objetivo de que la experiencia de usuario sea realmente buena a partir de una aplicación web de una sola página.

Respecto al almacenamiento de la información con la que se trabaja en la aplicación web desarrollada se ha utilizado una base de datos MySQL. Las principales ventajas que ofrece este sistema de bases relacional son las siguientes [2]:

- **Flexibilidad.** Compatible con todas las versiones de Linux y Windows.
- **Alto rendimiento.** Proporciona una buena respuesta en sitios web en los que se reciben muchas consultas al día.
- **Gratuita.** De código abierto y sin coste.
- **Fácil de usar.** Se necesita conocer un número de comandos relativamente bajo para usar la base de datos.
- **Seguridad.** Permite almacenar contraseñas encriptadas y dar diferentes permisos de acceso y privilegios a los usuarios.

Para la implementación del chat entre clientes y entrenadores personales y las notificaciones dentro de la plataforma se ha utilizado la biblioteca de Socket.IO. Esta biblioteca permite la conexión en tiempo real de forma bidireccional y basada en eventos entre el servidor y el navegador.

Se ha seguido el Proceso Unificado como marco de trabajo para la obtención del producto software. Esta metodología esta caracterizada por ser dirigida por casos de uso, estar centrada en la arquitectura y ser iterativa e incremental.

El contenido del resto de la memoria queda estructurado de la siguiente forma:

- **Objetivos.** En este apartado se describen los objetivos software propuestos para la realización del trabajo y otros objetivos personales que me han llevado a su realización.
- **Conceptos teóricos.** En este apartado se realiza una descripción de los conocimientos teóricos adquiridos o utilizados en la realización del trabajo. Estos conocimientos se utilizan para facilitar la comprensión de las tecnologías utilizadas.
- **Herramientas y tecnologías.** Se describen las herramientas y tecnologías utilizadas durante el desarrollo del trabajo. Sirve como complemento de los conceptos teóricos del apartado anterior.

- **Análisis del sistema.** En este apartado se realiza una definición del sistema desarrollado con una descripción de la arquitectura. Se describe un flujo básico de datos dentro de la aplicación y se listan todos los posibles casos de uso que un usuario puede realizar.
- **Implementación y desarrollo.** En este apartado se realiza una descripción de la estructura de los proyectos desarrollados: proyecto de la API y proyecto del Frontend, y se describe el flujo de datos en el envío de mensajes al chat, como se realizan los pagos y otras funciones principales de la aplicación.
- **Resultados.** Se muestra el resultado final de la aplicación desarrollada y se explican las principales funcionalidades.
- **Conclusiones y líneas de trabajo futuras.** Descripción de las conclusiones obtenidas una vez realizado el trabajo. Se explica cómo se han conseguido los diferentes objetivos y las soluciones propuestas a los problemas surgidos. Para finalizar se detallan las posibles líneas de trabajo futuras.
- **Anexo I.** En este anexo se describe la planificación temporal del proyecto y la estimación del esfuerzo que provoca su realización.
- **Anexo II.** Es este anexo se recoge la especificación de requisitos y el análisis del sistema software.
- **Anexo III.** En este documento se realiza una descripción de la estructura de los proyectos desarrollados.
- **Anexo IV.** Este anexo constituye un manual de usuario detallando todas las funcionalidades del sistema.

2. Objetivo del proyecto

2.1. Objetivos software

Desde el estallido de la pandemia, la demanda de entrenadores personales está en continuo aumento debido al miedo a contraer el virus. En el último año, esta demanda ha aumentado considerablemente La opción de entrenar en solitario se está convirtiendo en la opción por la que se decantan la mayor parte de los usuarios.

Este crecimiento ha provocado que se busquen continuas soluciones y alternativas por parte de los entrenadores personales para ofrecer sus servicios. En el presente trabajo se ha desarrollado una aplicación que ofrece todos los servicios esenciales que puede ofrecer un entrenador personal a sus clientes a través de internet.

Esta aplicación ha sido desarrollada con tecnologías actuales y modernas que ofrecen una mayor fluidez y una mejor experiencia de usuario utilizando bibliotecas que suponen un menor coste de recursos al tener que ejecutarse la mayor parte de la lógica del lado del cliente.

A continuación, se detallan los objetivos software propuestos:

- **Estudio de las tecnologías utilizadas.** Estudio de los lenguajes de desarrollo utilizados y frameworks, así como de todas las bibliotecas o módulos disponibles que permitieran facilitar el desarrollo.
- **Creación de una aplicación web.** Se ha desarrollado una aplicación web en la que un entrenador personal ofrezca sus servicios a diferentes clientes.
- **Almacenamiento de información.** La información utilizada y creada se almacena de forma persistente y segura en una base de datos MySQL.
- **Visualización de la información.** La aplicación desarrollada dispone de diferentes vistas en las que se presenta la información de forma ordenada y adecuada dependiendo del rol que tenga el usuario.

2.2. Objetivos personales

La gran pandemia por la que está pasando toda la población mundial está obligando a cambiar por completo la forma de vida de las personas, no solo a nivel laborar sino también a nivel de ocio.

Actualmente existen gran cantidad de medidas preventivas en lugares públicos como limitación de aforo o distanciamiento social. Dentro de estos lugares, se encuentran los gimnasios. Los gimnasios son el lugar por excelencia para muchas personas en los que se realiza actividad física mediante máquinas u otras herramientas que no se suelen tener en casa. Debido a la pandemia muchas personas tienen miedo a relacionarse con otras personas no convivientes en espacios públicos.

Dado el problema anterior, y sumado a lo que se puede considerar una gran ventaja, como es tener tu propio entrenador personal, sin estar esclavo a un horario o disposición específica, a través de internet, surgió la idea de desarrollar una aplicación web que permitiera crear esta relación entrenador personal-cliente.

A nivel personal, el desarrollo de esta aplicación web utilizando tecnologías actuales que están en auge y proporcionan a los sitios web mayor dinamismo y experiencia de usuario puede ser muy importante para mi futuro laboral próximo.

Gracias a las tecnologías y herramientas utilizadas en el desarrollo he afianzado gran cantidad de conocimientos y aprendido muchos otros que eran desconocidos para mí. La implementación de la API en NodeJS y el desarrollo del Frontend en AngularJS han sido para mí la primera vez que he utilizado estas tecnologías para un proyecto de esta envergadura.

Otras herramientas utilizadas como la biblioteca de Socket.IO para la implementación del chat de mensajes y las notificaciones, y la utilización del módulo de Stripe para realizar los pagos han sido muy importantes para afianzar conocimientos y conocer características nuevas de estas herramientas.

Por último, para almacenar los proyectos realizados a lo largo del trabajo se ha utilizado un sistema de control de versiones Git. Todos los cambios que se han ido realizando se han registrado en un repositorio de GitHub.

3. Conceptos teóricos

3.1. SPA

SPA, o Single Page Application, es un tipo de aplicación web en la que todas las pantallas se muestran en una misma página sin recargar el navegador. El punto de entrada es único y suele ser un archivo *index.html* [3].

Este tipo de aplicaciones puede tener varias vistas, dentro de la misma página se van intercambiando distintas vistas. El efecto producido por este intercambio de vistas es que se realiza de forma muy rápida y las comunicaciones entre cliente y servidores son muy fluidas.

Los datos recibidos del servidor, que normalmente es una API REST, se suelen obtener en formato JSON. Este formato es ligero y puede ser soportado por múltiples lenguajes de desarrollo web.

Existen varios frameworks que se utilizan para el desarrollo de aplicaciones SPA como pueden ser AngularJS, EmberJS o React. En este trabajo se ha utilizado el framework AngularJS.

3.2. API REST

En términos generales, una API es una especificación que permite la conexión entre dos aplicaciones para cumplir con determinadas funciones. La principal ventaja es facilitar el trabajo a los desarrolladores y ahorrar tiempo y otros recursos [4].

Una API proporciona el acceso a aplicaciones de terceros para reutilizar servicios ya creados y se puede utilizar para realizar la comunicación con una base de datos, en este trabajo se realiza la conexión con una base de datos MySQL.

Actualmente el término API ha derivado en el concepto API REST. Este concepto engloba las siguientes características en cuanto arquitectura:

- **Protocolo cliente/servidor sin estado.** Arquitectura compuesta de clientes, servidores y recursos que gestiona solicitudes a través de HTTP. La comunicación entre cliente y servidor es sin estado, la información de cada solicitud no se almacena y es independiente de otras solicitudes.
- **Operaciones *POST*** (crear), ***GET*** (consulta), ***PUT*** (editar) y ***DELETE*** (eliminar).
- **Sistema de capas.** Se organiza en jerarquías invisibles para el cliente dentro de cada servidor (seguridad, equilibrio de carga, …).
- **URI.** Los objetos se manipulan a partir de una URI (Identificador de recursos uniforme). La URI es el elemento con identificador único de cada recurso del sistema REST. Facilita toda la información para realizar la operación específica sobre los datos.

Las principales ventajas de utilizar una API son las siguientes:

- **Mejora el rendimiento.** Los datos tratados suelen ser planos y no precisan de mucho tiempo de transferencia.
- **Cliente/Servidor.** Son sistemas independientes que se comunican con un lenguaje de intercambio, normalmente JSON.
- **Escalable.** La API podrá responder a todas las operaciones realizadas desde una o más aplicaciones web desde cualquier dispositivo.
- **Pocos recursos.** El procesamiento es menor en el servidor.

3.3. Socket.IO

Socket.IO es una biblioteca JavaScript basada en eventos y utilizada en aplicaciones web en tiempo real. Permite la conexión bidireccional entre cliente y servidor. Dispone de dos partes, una en el lado del servidor, a través de una biblioteca para NodeJS, y otra del lado del cliente, en este trabajo con una biblioteca para el framework AngularJS.

Las principales características son las siguientes:

- Programación en lenguaje Javascript.
- Multiplataforma.
- Centrado en la velocidad y asíncrono.
- Gestión de conexiones TCP y eventos.

4. Herramientas y tecnologías

En este apartado se realiza una descripción de las herramientas y tecnologías utilizadas para el desarrollo de la aplicación web. Se realiza una separación de las herramientas utilizadas entre la parte de la API y de la parte del Frontend.

4.1. API

4.1.1. NodeJS

NODEJS es un entorno de ejecución Javascript multiplataforma de código abierto construido con el motor de Javascript V8 de Chrome. Diseñado para crear aplicaciones web escalables y orientado a la ejecución de eventos asíncronos [5].

Las características más importantes son las siguientes:

- **Asíncrono y controlado por eventos.** No depende del tiempo si no de disparadores que responden a las interacciones del usuario.
- **Rápido.** La biblioteca de NodeJS es muy rápida ejecutando código.
- **Multiplataforma.** Puede ser ejecutado en sistemas Linux, MacOS o Windows.
-

4.1.2. Express

Express es un framework de aplicaciones web desarrolladas en NodeJS mínimo y muy flexibles, de código abierto y diseñado principalmente para la creación de APIs.

4.2. FRONTEND

4.2.1. AngularJS

AngularJS es un framework Javascript basado en el esquema MVC (Modelo Vista Controlador) desarrollado por Google para el desarrollo de páginas web SPA (Single Page Applications). Una SPA es un tipo de aplicación web en la que todas las pantallas se muestran en la misma página sin recargar el navegador [6].

Las características principales son las siguientes:

- Desarrollo en Typescript.
- Reutilización de código mediante componentes.
- Código abierto.
- Gran soporte, recursos y herramientas disponibles.

4.2.2. HTML

HTML, o Lenguaje de Marcas de Hipertexto, es un lenguaje informático que forma parte de la mayor parte de páginas o aplicaciones web. Las marcas hacen referencia a la estructura y al estilo del documento que forma la página web y el hipertexto hace referencia al enlace que puede haber entre unos textos y otros [7].

Entre las características principales se encuentran las siguientes [8]:

- Puede ser creado/editado en cualquier editor de textos.
- Puede ser visualizado en cualquier navegador con diferentes dispositivos.
- No diferencia entre mayúsculas y minúsculas.
- Utiliza etiquetas o marcas para definir la estructura. Cada elemento tiene una etiqueta de comienzo y otra de fin.
- Permite enlazar hojas de estilos CSS y scripts JS.

4.2.3. CSS

CSS, o hojas de estilo en cascada, es un lenguaje de diseño gráfico que se aplica generalmente a los documentos HTML para dotarlos de estilos (colores, márgenes, …). El estilo se aplica de arriba abajo utilizando las herencias [9].

La principal causa de utilización de los archivos CSS es separar la presentación de los datos con el contenido. De esta forma los archivos HTML incluirán la información y los datos, y los archivos CSS el estilo de cómo se presentan.

4.2.4. BOOTSTRAP

Bootstrap es una biblioteca de código abierto y multiplataforma utilizada para el desarrollo de páginas y aplicaciones web. Cuenta con gran multitud de plantillas con botones, formularios, tablas, menús de navegación y otros elementos basados en HTML, CSS y JS.

4.2.5. FONT AWESOME

Font Awesome es un framework de iconos vectoriales y estilos css. Sustituye iconos comunes por gráficos vectoriales convertidos en fuentes, para ello utiliza una librería de iconos transformadas en fuentes. Una de las grandes ventajas que proporciona es que no hay problema con la resolución de los iconos al realizar un escalado al tratarse de iconos vectoriales.

4.3. Otras herramientas y tecnologías

4.3.1. VISUAL STUDIO CODE

Visual Studio Code es un editor de código gratuito que proporciona soporte para todas las operaciones de desarrollo como son la depuración, la ejecución de tareas y el control de versiones. El objetivo principal es proporcionar las herramientas necesarias para que un desarrollador realice de forma eficaz las tareas de desarrollo, compilación y depuración de código [10].

Esta herramienta dispone de múltiples extensiones que permiten agregar distintos idiomas, cambiar temas y otros muchos servicios. Estas extensiones se ejecutan de forma independiente, en procesos separados, lo que hace que el rendimiento del editor no se vea afectado [11].

Las características principales son las siguientes:

- Incluye soporte de depuración.
- Control de Git.
- Finalización de código inteligente.
- Personalizable: permite cambiar el tema, atajos del teclado, abrir consolas o distintas ventanas.
- Gratuito y de código abierto.
- Permite su uso con distintos lenguajes de programación.
- Buen soporte técnico: existe mucha documentación en foros o sitios relacionados.

4.3.2. XAMMP

XAMMP es una herramienta gratuita fácil de instalar y usar que proporciona un sistema gestor de bases de datos MySQL, un servidor Apache e intérpretes para lenguajes PHP y Perl (Figura 4.1).

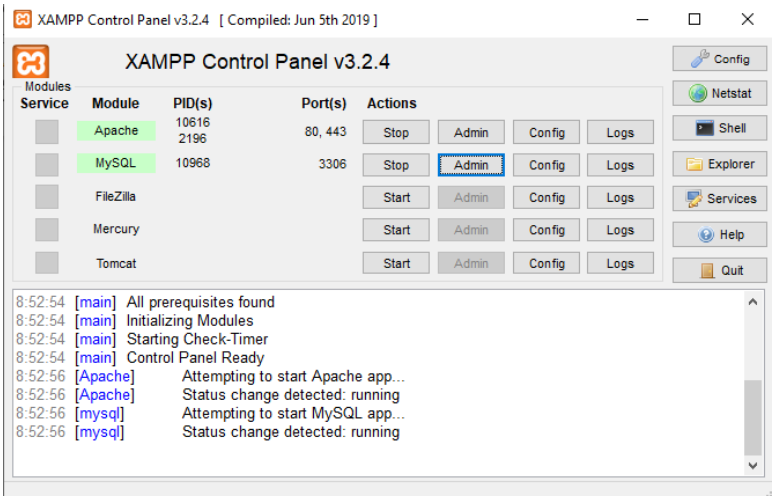

Figura 4.1. Interfaz de XAMMP.

4.3.3. Git Hub Desktop

Git Hub Desktop es una aplicación que permite conectarse con Git Hub usando una aplicación de escritorio en lugar de un navegador o la línea de comandos [12]. Se puede utilizar para realizar todas las operaciones básicas como añadir, clonar o crear repositorios, hacer commit de cambios sobre repositorios, cambiar entre las diferentes ramas de un proyecto, mergear diferentes ramas, … (Figura 4.2).

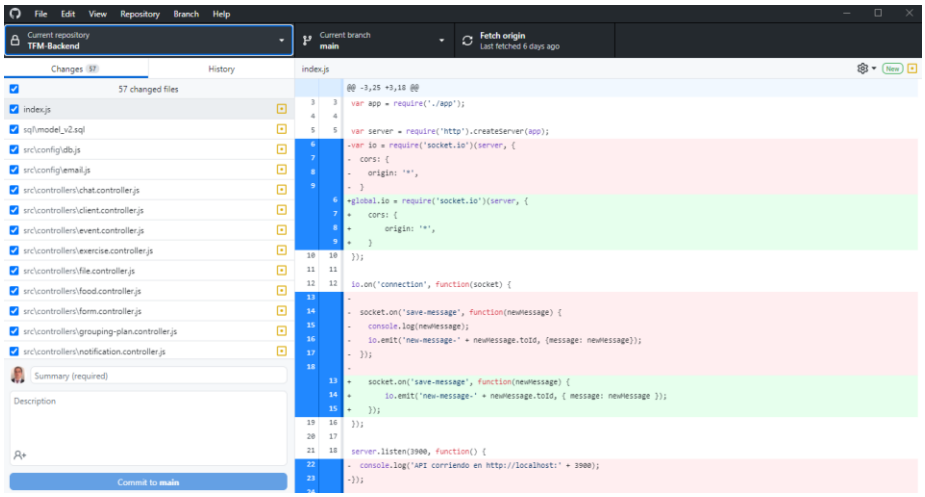

Figura 4.2. Interfaz de Bit Hub Desktop.

4.3.4. NPM

NPM es un sistema de gestión de paquetes para NodeJS. Permite la instalación de aplicaciones disponibles en el repositorio que facilitan o realizan una determinada tarea o función [13].

4.3.5. MySQL

MySQL es un sistema de gestión de bases de datos relacionales. Las bases de datos relacionales utilizan múltiples tablas conectadas entre sí que almacenan la información y la organizan de forma correcta.

Las principales características son las siguientes [14]:

- **Código abierto.**
- **Arquitectura cliente-servidor.** Clientes y servidores se comunican de forma diferenciada para mejorar el rendimiento. Los clientes pueden realizar consultar para añadir, modificar o eliminar datos.
- **Compatible con el lenguaje SQL.**
- **Permite automatizar ciertas tareas.** A partir de ciertos eventos se pueden actualizar los datos automáticamente para mejorar el rendimiento.

4.3.6. Stripe

Stripe es un sistema de pagos que permite vincular una cuenta bancaria o cuenta de Stripe y recibir pagos a través de internet a cambio de tus servicios o productos en una aplicación web propia.

Las principales ventajas son las siguientes [15]:

- **Alta gratuita.** El proceso de alta es totalmente gratuito, únicamente se paga una pequeña comisión por cada pago recibido.
- **Integración.** El pago está integrado en la propia aplicación.
- **Versatilidad.** Acepta cualquier tipo de tarjeta bancaria y es compatible con diferentes aplicaciones móviles.
- **Seguridad.** Los datos de los clientes se encuentran almacenados en un sistema muy seguro.

4.3.7. UML

El Lenguaje Unificado de Modelado, o UML, es un lenguaje utilizado para describir la arquitectura, el diseño y la implementación de sistemas softwares. Mediante UML se pueden generar diagramas de casos de uso, diagramas de clases, diagramas de secuencia, diagramas de colaboración, diagramas de estado y otros muchos más.

4.3.8. Visual Paradigm

Visual Paradigm es una herramienta utilizada para el modelado UML. Proporciona todas las herramientas necesarias para elaborar los diferentes tipos de diagramas.

4.3.9. Microsoft Project

Microsoft Project es una herramienta utilizada para la administración de proyectos. Permite a los administradores de los proyectos asignar recursos a las tareas, seguir el progreso y administrar el presupuesto o las cargas de trabajo.

5. Análisis del sistema

5.1. Definición del sistema

Esta aplicación pondrá a servicio de entrenadores personales y clientes todas las funcionalidades básicas que se pueden llegar a dar como realizar un control de la nutrición o asignar tareas personalizadas y otras muchas funcionalidades más complejas que pueden surgir en dicha relación cliente-entrenador personal como compartir videos, chat personalizado o creación de eventos virtuales.

La finalidad de este trabajo es desarrollar una aplicación web mediante el framework AngularJS que conectado a una API REST implementada en NodeJS y que utiliza el framework ExpressJS proporcione todas las funciones propuestas.

Para realizar el almacenamiento de la información se ha utilizado una base de datos relacional MySQL.

5.2. Arquitectura

La aplicación web desarrollada sigue una arquitectura basada en MySQL, ExpressJS, AngularJS y NodeJS.

Figura 5.1. Arquitectura MySQL, ExpressJS, AngularJS, NodeJS.

Cada una de estas tecnologías tiene una funcionalidad:

- **MySQL.** Es una base de datos relacional que permitirá el almacenamiento de la información.
- **ExpressJS.** Es un framework de Javascript utilizado en el servidor para el desarrollo de la API.
- **AngularJS.** Es una framework de Javascript utilizado del lado del cliente para el desarrollado del frontend de la aplicación.
- **NodeJS.** Desarrollado en Javascript, permite lanzar un servidor que realiza la conexión entre los datos almacenados y la parte del frontend.

Las tecnologías utilizadas permiten el desarrollo de aplicaciones escalables en tiempo real que ofrecen un gran rendimiento y una buena experiencia de usuario. En la Figura 2 se muestra la conexión de las diferentes tecnologías utilizadas en el desarrollo de la aplicación web.

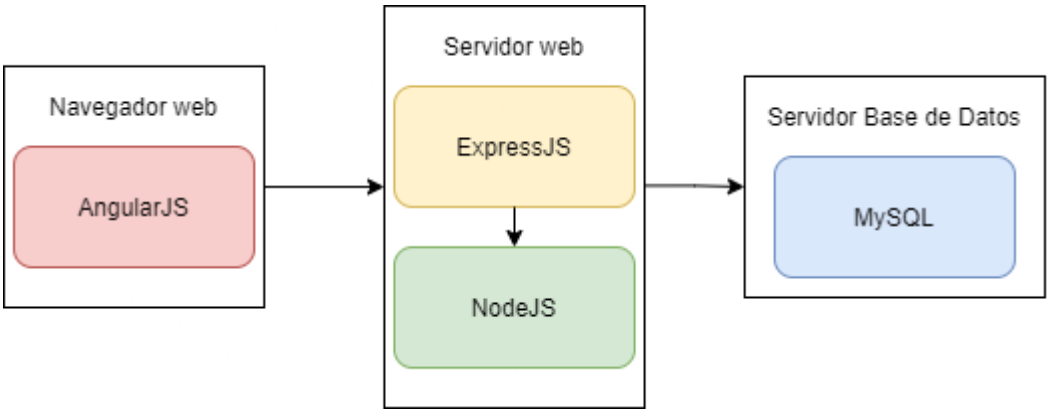

Figura 5.2. Arquitectura de la aplicación.

5.3. Flujo de la aplicación

En el siguiente diagrama se muestra el flujo de llamadas y de datos que se siguen en la aplicación web desarrollada.

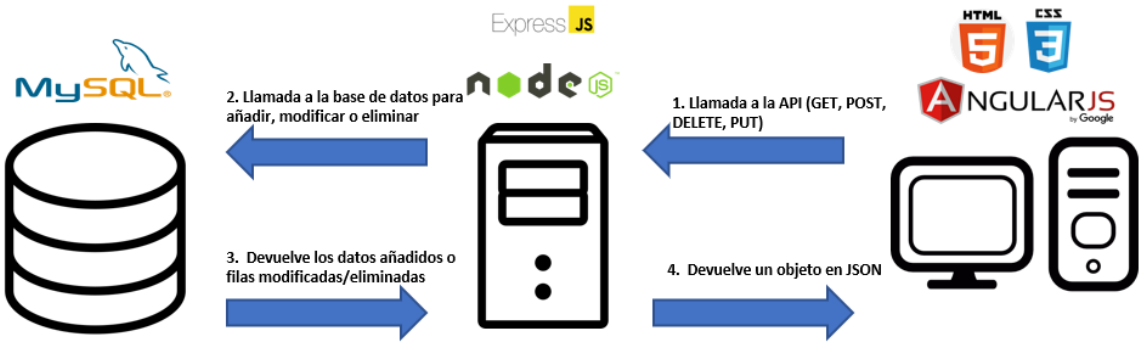

Figura 5.3. Flujo de la aplicación.

5.4. Plan de proyecto

En esta sección se detallan las técnicas utilizas para realizar la gestión y el control del proyecto.

5.4.1. Estimación del esfuerzo

La estimación de costes se utiliza para predecir el esfuerzo probable que se requiere para la construcción de un sistema software. El esfuerzo está basado en una relación entre el tiempo y el personal necesarios para el desarrollo de un proyecto.

El método utilizado para calcular la estimación del esfuerzo es el UCP, o Puntos de Caso de Uso. Este método considera los actores, escenarios y los factores técnicos y del entorno para evaluar la funcionalidad del sistema en forma de casos de uso.

Requiere el cálculo de tres variables:

- **UUCP** (Puntos de caso de uso sin ajustar)
- **TCF** (Factores de complejidad técnica)
- **ECF** (Factores de complejidad del entorno)

Para obtener los Puntos de Caso de Uso se utiliza la siguiente fórmula:

UCP = UUCP * TCF * ECF

Para calcular la estimación del esfuerzo se utilizará la siguiente fórmula:

Esfuerzo = UCP * F

Siendo F un factor de conversión que representa el número de horas de persona por UCP. Por defecto el valor de F es 20.

Para calcular la estimación del esfuerzo se utilizará la siguiente fórmula:

Esfuerzo = UCP * F

Siendo F un factor de conversión que representa el número de horas de persona por UCP. Por defecto el valor de F es 20.

Para realizar el cálculo de los Puntos de Caso de Uso sin Ajustar es necesario considerar el número y complejidad de los casos de uso y el número y complejidad de los actores. La fórmula quedaría de la siguiente manera:

UUCP = UUCW + UAW

Siendo UUCW el factor de peso de los Casos de Uso sin Ajustar y UAW el factor de peso de los Actores sin Ajustar.

Para realizar el cálculo de los factores de complejidad técnica se debe considerar un peso (W) acorde a su impacto y una complejidad percibida (F) que se basa en un valor dentro de una escala del 0 (irrelevante) al 5 (esencial).

Para realizar el cálculo de los factores de complejidad del entorno se debe considerar un peso (W) acorde a su impacto y una complejidad percibida (F) que se basa en un valor dentro de una escala del 0 (impacto negativo) al 5 (impacto positivo).

Mediante el uso de la herramienta EZEstimate se han obtenido los resultados de la Figura 5.4.

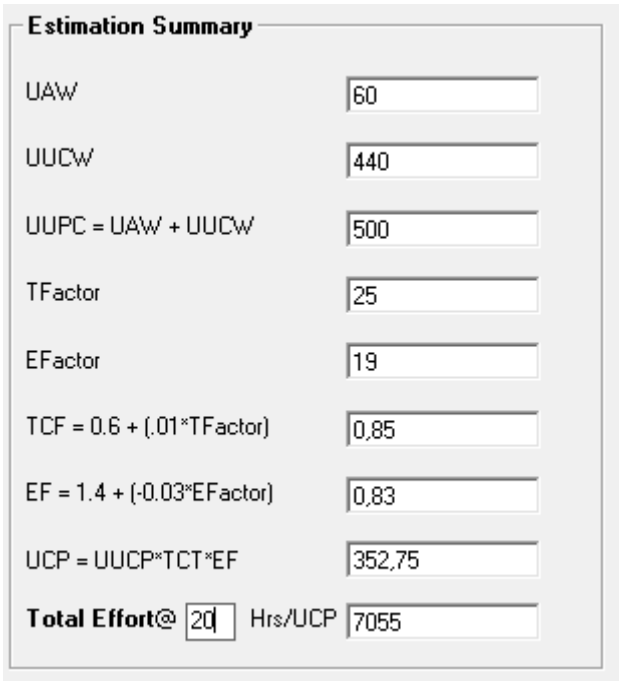

Figura 5.4.

Como se puede observar sería necesario un total de 7055 horas para realizar el proyecto. Si se trabaja en jornadas de 8 horas al día, aproximadamente 2 años.

Este valor obtenido no se corresponde con la realidad, pues pueden influir numerosos aspectos que no se han tenido en cuenta como la utilización de módulos en el proyecto que faciliten y ahorren trabajo. Como ejemplo, se han utilizado módulos para la subida y tratamiento de archivos en el backend y módulos para mostrar las tablas de datos en el frontend.

5.4.2. Planificación temporal

Mediante la planificación temporal se identifican las tareas necesarias para realizar con éxito el proyecto y los plazos previstos para su realización. En la Figura 5.5 podemos ver las tareas de la fase de inicio.

Proyecto TFM	**158 días**	**vie 16/04/21**	**mar 23/11/21**
Inicio	**24 días**	**vie 16/04/21**	**mié 19/05/21**
Iteracción 1	**13 días**	**vie 16/04/21**	**mar 04/05/21**
Modelado de negocio	**3 días**	**vie 16/04/21**	**mar 20/04/21**
Reunión inicial con el tutor	1 día	vie 16/04/21	vie 16/04/21
Identificación de los objetivos del trabajo	1 día	lun 19/04/21	lun 19/04/21
Comprobación de viabilidad del proyecto	1 día	mar 20/04/21	mar 20/04/21
Requisitos	**10 días**	**mié 21/04/21**	**mar 04/05/21**
Definición de los objetivos del sistema	1 día	mié 21/04/21	mié 21/04/21
Casos de uso del sistema	4 días	jue 22/04/21	mar 27/04/21
Especificación inicial del documento de requisitos	3 días	mié 28/04/21	vie 30/04/21
Identificación de actores	2 días	lun 03/05/21	mar 04/05/21
Identificación de escenarios	2 días	lun 03/05/21	mar 04/05/21
Hito final de iteracción	0 días	mar 04/05/21	mar 04/05/21
Iteracción 2	**11 días**	**mié 05/05/21**	**mié 19/05/21**
Modelado de negocio	**5 días**	**mié 05/05/21**	**mar 11/05/21**
Estudio de las herramientas existentes	2 días	lun 10/05/21	mar 11/05/21
Estudio de los entornos de desarrollo	2 días	mié 05/05/21	jue 06/05/21
Requisitos	**6 días**	**mié 12/05/21**	**mié 19/05/21**
Reunión con tutor para revisar requisitos	1 día	mié 12/05/21	mié 12/05/21
Definición formal de los requisitos	1 día	jue 13/05/21	jue 13/05/21
Estimación del esfuerzo	2 días	vie 14/05/21	lun 17/05/21
Documentación de especificación de requisitos	2 días	mar 18/05/21	mié 19/05/21
Hito de especificación de requisitos	0 días	mié 19/05/21	mié 19/05/21
Hito fin de inicio	0 días	mié 19/05/21	mié 19/05/21

Figura 5.5. Fase de inicio

5.5. Especificación de requisitos

En esta sección se ha realizado una especificación de requisitos siguiendo la metodología de Durán y Bernárdez y obteniendo la información de la propuesta del proyecto. El análisis del modelo se ha realizado siguiendo el paradigma del análisis orientado a objetos.

En el Anexo II se puede consultar la especificación de requisitos al detalle.

5.5.1. Descripción del grupo de trabajo

Entre los participantes que forman el grupo de trabajo de desarrollo del proyecto se han identificado al tutor del trabajo y al jefe del proyecto, que se trata de la misma persona que el desarrollador. En la siguiente tabla se puede ver el jefe del proyecto:

Participante	**Mario González Morán**
Organización	Universidad de Salamanca
Rol	Jefe del proyecto
Es desarrollador	Sí
Es cliente	No
Es usuario	No
Comentarios	Ninguno

Tabla 5.1. Participante Mario González Morán

5.5.2. Requisitos por objetivos

Los requisitos objetivos del sistema son los siguientes:

- Gestión de los usuarios
- Gestión de las tareas
- Gestión de la librería
- Gestión de los planes
- Gestión del equipo
- Gestión de los clientes

En la siguiente tabla se puede ver uno de estos objetivos:

OBJ-0001	**Gestión de los usuarios**
Versión	1.0 (15/06/2021)
Autores	Mario González Morán
Fuentes	Propuesta TFM
Descripción	El sistema deberá *gestionar a los usuarios registrados en la plataforma*
Subobjetivos	Ninguno
Importancia	vital
Urgencia	inmediatamente
Estado	validado
Estabilidad	alta
Comentarios	Ninguno

Tabla 5.2. Requisito por objetivo.

5.5.3. Requisitos de información

En esta sección se muestran los requisitos de almacenamiento de información identificados en el sistema. Los requisitos identificados son los siguientes:

- Información de los usuarios
- Información de las notificaciones
- Información de los planes de pago
- Información de los mensajes del chat
- Información de los eventos
- Información de las sesiones
- Información de las tareas de los clientes
- Información del equipo de colaboradores
- Información de los clientes
- Información de los ejercicios
- Información de los workouts
- Información de las sesiones de video
- Información de los planes nutricionales
- Información de las recetas
- Información de los alimentos
- Información de las agrupaciones de planes
- Información de los documentos
- Información de los formularios
- Información de los programas

En la siguiente tabla podemos ver uno de los requisitos de información:

IRQ-0006	**Información de las sesiones**
Versión	1.0 (15/06/2021)
Autores	Mario González Morán
Fuentes	Propuesta TFM
Dependencias	
Descripción	El sistema deberá almacenar la información correspondiente a *las sesiones creadas.* En concreto:
Datos específicos	- Nombre - Tipo - Organizador - Descripción - Fecha - Hora - Duración - URL - Sesión de video

Tabla 5.3. Requisito de información

5.5.4. Requisitos funcionales

Los requisitos funcionales son aquellos que describen el comportamiento o función que debe de realizar el sistema software cuando se cumplan ciertas condiciones. En la siguiente tabla podemos ver un requisito funcional del sistema.

FRQ-0001	**Gestión de usuario**
Versión	1.0 (15/06/2021)
Autores	Mario González Morán
Fuentes	Propuesta TFM
Dependencias	
Descripción	El sistema deberá *proporcionar la funcionalidad necesaria para que los usuarios puedan registrarse y autenticarse en la plataforma.*
Importancia	vital
Urgencia	inmediatamente
Estado	validado
Estabilidad	media
Comentarios	Ninguno

Tabla 5.4. Requisito funcional.

5.6. Modelo de análisis

5.6.1. Modelo del dominio

Para realizar la representación del modelo del dominio se ha empleado el lenguaje UML y se ha creado un diagrama de clases en el que se muestran los diferentes modelos con sus atributos.

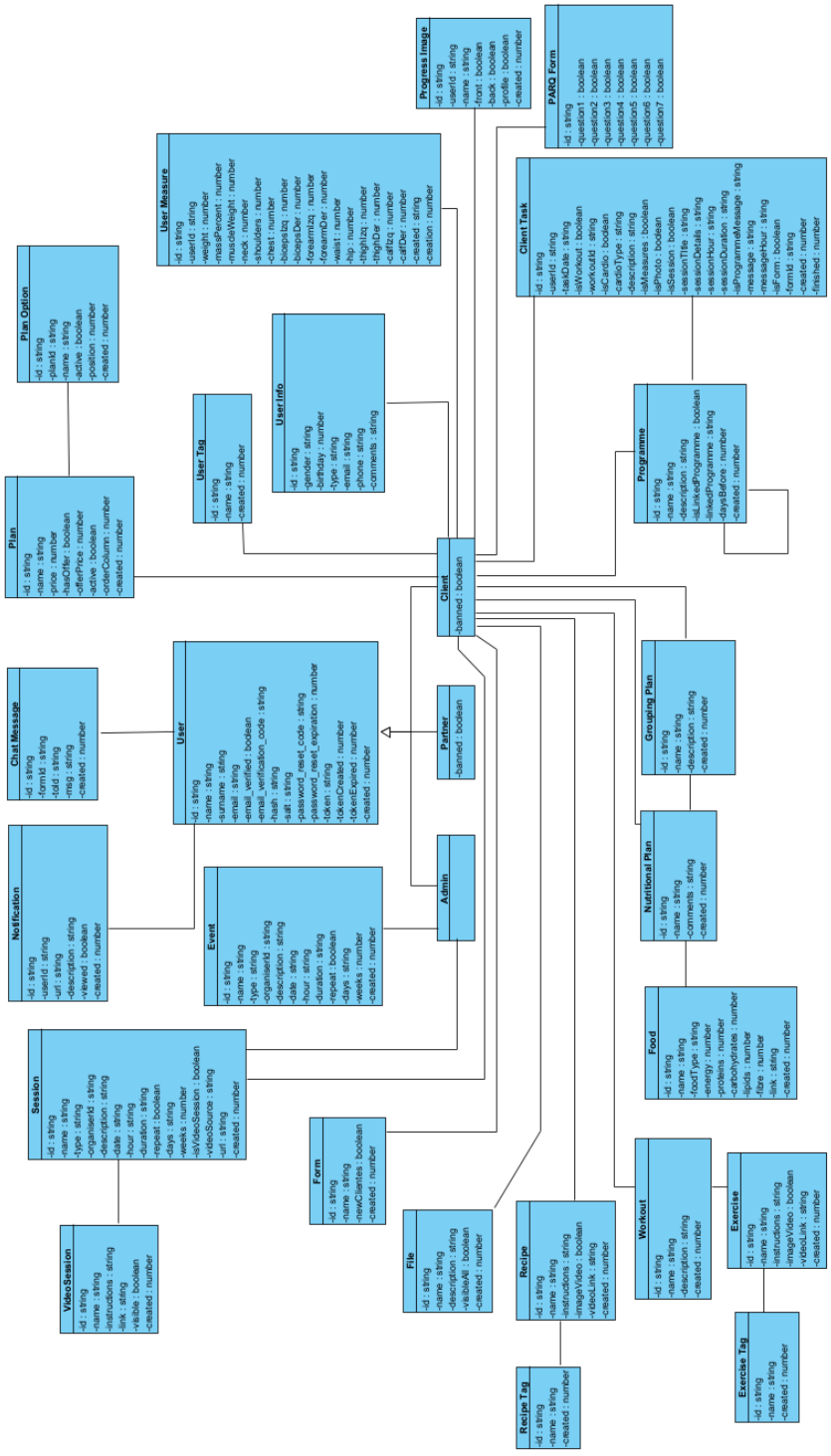

Figura 5.5. Diagrama de clases

5.6.2. Casos de uso

En este apartado se recogen las diferentes situaciones en las que puede encontrarse un usuario que utiliza la aplicación web. Un usuario puede tener tres roles diferentes:

- **Administrador.** Es el usuario con todos los permisos dentro de la aplicación (entrenador personal "jefe"). Se trata del responsable de gestionar toda la información dentro de la aplicación.
- **Colaborador.** Un usuario con este rol es invitado por el administrador de la aplicación y desempeña la función de entrenador personal con los clientes que tenga asignados.
- **Cliente.** Cualquier usuario que se registra en la aplicación y espera un servicio a cambio.

5.6.3. Administrador

Los casos de uso de un usuario con el rol de administrador son los siguientes:

Planificación

- **Añadir evento.** Se puede añadir un evento con tu equipo o clientes para realizar un seguimiento indicando el nombre, el tipo, el organizador, la fecha, la hora y la duración, y una pequeña descripción.
- **Añadir sesión.** Crear una sesión dirigida a uno o más clientes indicando el nombre, el tipo, el organizador, la fecha, la hora y la duración, una pequeña descripción y un video disponible en *Sesiones de video* o un enlace externo que lleve a una plataforma de videoconferencias.
- **Listar mis próximas tareas.** Se muestra una lista con los próximos eventos y sesiones que tiene el administrador.
- **Listar actividades recientes de tus clientes.** Se muestra una lista con las actividades recientes completadas por los clientes.
- **Listar clientes con tareas sin completar.** Se muestra una lista con clientes que tienen aún tareas sin completar.
- **Listar clientes sin próximas tareas.** Se muestra una lista con los clientes que no tienen tareas asignadas.
- **Listar usuarios con suscripciones a punto de finalizar.** Se muestra una lista con los clientes cuya suscripción está a punto de finalizar.
- **Seleccionar calendario propio o de cualquier otro usuario.** Se puede seleccionar el calendario personal de cada usuario disponible en la plataforma para poder ver que tareas, sesiones o eventos tiene asignados.
- **Ver todos los eventos y sesiones en modo lista.** Listar todos los eventos y sesiones en una tabla. Es una vista alternativa al calendario.
- **Ver información de un evento/sesión.** Seleccionar un evento o sesión del calendario para ver la información en un modal.

Clientes

- **Listar clientes.** Listar todos los clientes disponibles en la aplicación.
- **Banear a un cliente.** Banear a un cliente registrado en la aplicación para prohibirle el acceso a los servicios ofrecidos.

- **Añadir cliente.** Añadir un nuevo cliente enviándole un email o compartiendo un enlace personalizado con él.
- **Añadir etiquetas de clientes.** Añadir etiquetas para asignárselas a los clientes y que puedan ser filtrados o seleccionados por ellas.
- **Añadir grupos de clientes.** Agrupar varios clientes para compartir servicios comunes entre ellos.

Librería

- **Añadir ejercicio.** Añade un nuevo ejercicio a la librería de ejercicios disponibles introduciendo un nombre y unas instrucciones, e indicando el enlace al video en el que se muestra cómo realizar el ejercicio o unas imágenes descriptivas si se desea. Además, se pueden añadir etiquetas para filtrar los ejercicios.
- **Editar ejercicio.** Edita un ejercicio añadido previamente.
- **Elimina ejercicio.** Elimina un ejercicio de la librería de ejercicios.
- **Listar ejercicios.** Lista los ejercicios disponibles en la librería de ejercicios.
- **Añadir workout.** Añade un nuevo workout a la librería de workouts disponibles introduciendo un nombre, una descripción y creando bloques, añadiendo descansos y ejercicios.
- **Editar workout.** Edita un workout añadido previamente.
- **Eliminar workout.** Elimina un workout de la librería de workout.
- **Listar workouts.** Lista los workouts disponibles en la librería de workouts.
- **Añadir sesión de video.** Añade una nueva sesión de video a la librería de sesiones de video disponibles introduciendo un nombre, unas instrucciones y en enlace externo a un vídeo. El video se podrá además etiquetas e indicar que sea visible por todos los clientes o solo por los clientes que dispongan de ciertas etiquetas.
- **Editar sesión de video.** Edita una sesión de video añadida previamente.
- **Eliminar sesión de video.** Elimina una sesión de video de la librería de sesiones de video.
- **Listar sesiones de video.** Lista las sesiones de video disponibles en la librería de sesiones de video.
- **Añadir plan nutricional.** Añade un nuevo plan nutricional a la librería de planes nutricionales disponibles introduciendo un nombre, unos comentarios y añadiendo diferentes comidas compuestas por diferentes alimentos.
- **Editar plan nutricional.** Edita un plan nutricional añadido previamente.
- **Eliminar plan nutricional.** Elimina un plan nutricional de la librería de planes nutricionales.
- **Listas planes nutricionales.** Lista los planes nutricionales disponibles en la librería de planes nutricionales.
- **Añadir receta.** Añade una nueva receta a la librería de recetas disponibles introduciendo un nombre y unas instrucciones. Se podrá añadir un vídeo o imágenes explicativas si se considera necesario y diferentes etiquetas para filtrarlas.
- **Editar receta.** Edita una receta añadida previamente.
- **Eliminar receta.** Elimina una receta de la librería de recetas.
- **Listas recetas.** Lista las recetas disponibles en la librería de recetas.
- **Añadir alimento.** Añade un nuevo alimento a la biblioteca de alimentos disponibles introduciendo el nombre, el tipo y los valores nutricionales para 100 gramos de ese alimento. Además, se podrá añadir un enlace para ir a una página web en la que se muestre la información de dicho alimento.
- **Editar alimento.** Edita un alimento añadido previamente.

- **Eliminar alimento.** Elimina un alimento de la librería de alimentos.
- **Listas alimentos.** Lista los alimentos disponibles en la librería de alimentos.
- **Añadir agrupación de planes.** Añade una nueva agrupación de planes a la librería de agrupaciones de planes disponibles seleccionando varios planes nutricionales y asignándoles un nombre y una pequeña descripción.
- **Editar agrupación de planes.** Edita una agrupación de planes añadida previamente.
- **Eliminar agrupación de planes.** Elimina una agrupación de planes de la librería de agrupaciones de planes.
- **Listas agrupaciones de planes.** Lista las agrupaciones de planes disponibles en la librería de agrupaciones de planes.
- **Añadir documento.** Añade un nuevo documento a la librería de documentos disponibles introduciendo un nombre y una descripción y subiendo un archivo en formato *.pdf*. Se podrá seleccionar la visibilidad del archivo.
- **Editar documento.** Edita un documento añadido previamente.
- **Eliminar documento.** Elimina un documento de la librería de documentos.
- **Listar documentos.** Lista los documentos disponibles en la librería de documentos.
- **Añadir formulario.** Añade un nuevo formulario a la librería de formularios disponibles introduciendo un nombre y añadiendo diferentes preguntas. Se podrá seleccionar una opción para enviar esos formularios a los nuevos clientes registrados en la plataforma.
- **Editar formulario.** Edita un formulario añadido previamente.
- **Eliminar formulario.** Elimina un formulario de la librería de formulario.
- **Listar formularios.** Lista los formularios disponibles en la librería de formularios.
- **Añadir programa.** Añade un nuevo programa a la librería de programas disponibles introduciendo un nombre y una descripción y añadiendo tareas a los diferentes días de la semana que lo componen.
- **Editar programa.** Edita un programa añadido previamente.
- **Eliminar programa.** Elimina un programa de la librería de programas.
- **Listar programas.** Lista los programas disponibles en la librería de programas.

Equipo

- **Añadir miembro al equipo.** Añadir un miembro al equipo de entrenadores personales. Hay que introducir el nombre, los apellidos y un correo válido para que el nuevo miembro reciba los datos de acceso a la plataforma vía email.
- **Eliminar miembro del equipo.** Elimina un miembro del equipo.
- **Listar equipo.** Lista todos los miembros del equipo disponibles.

Planes y pagos

- **Añadir plan de pago.** Añadir un nuevo plan de pago introduciendo su nombre y un precio. Se dispone de una opción para añadir una oferta dentro de cada plan y otra para añadir las características disponibles
- **Editar plan de pago.** Editar un plan de pago creado anteriormente.
- **Eliminar plan de pago.** Eliminar un plan de pago.

- **Ordenar planes de pago.** Ordenar los planes de pago para que se muestran en la pantalla de *Registro*.

Vista específica de un cliente

- **Añadir nueva tarea a cliente.** El administrador podrá asignar una nueva tarea a un cliente un día específico.
- **Ver información de una tarea de un cliente.** El administrador podrá ver la información relacionada con una tarea asignada al cliente.
- **Modificar información personal de un cliente.** El administrador podrá modificar la información personal de un cliente.
- **Asignar/Eliminar entrenador personal.** El administrador podrá asignar o eliminar la asignación de un entrenador personal de un cliente.
- **Enviar formulario a cliente.** El administrador podrá enviar un formulario a un cliente.
- **Ver respuestas de formulario de cliente.** El administrador podrá ver las respuestas de un cliente a un formulario específico.
- **Eliminar formulario asignado a cliente.** El administrador podrá eliminar un formulario enviado a un cliente.
- **Compartir plan nutricional con cliente.** El administrador podrá compartir un plan nutricional con un cliente.
- **Cambiar visibilidad de plan nutricional compartido con un cliente.** El administrador podrá cambiar la visibilidad de un plan nutricional compartido con un cliente.
- **Eliminar plan nutricional compartido con cliente.** El administrador podrá dejar de compartir un plan nutricional con un cliente.
- **Compartir receta con cliente.** El administrador podrá compartir una receta con un cliente.
- **Eliminar receta compartida con cliente.** El administrador podrá eliminar una receta compartida con un cliente.
- **Compartir documento con cliente.** El administrador podrá compartir un documento con un cliente.
- **Eliminar documento compartido con cliente.** El administrador podrá eliminar un documento compartido con un cliente.
- **Registrar medidas de un cliente.** El administrador podrá registrar las medidas de un cliente.
- **Consultar medidas de cliente.** Consultar el listado de medidas subidas por un cliente.
- **Añadir imágenes de cliente.** El administrador podrá subir las imágenes de un cliente.
- **Consultar imágenes de un cliente.** Consultar el listado de imágenes subidas por un cliente.

5.6.4. Colaborador

Los casos de uso de un usuario con el rol de colaborador son los siguientes:

Planificación

- **Listar próximas actividades.** El colaborador podrá ver en el calendario o en una lista todos los eventos y sesiones en los que tendrá que participar.

- **Listar próximas actividades de clientes asignados.** El colaborador podrá ver los calendarios de sus clientes asignados.
- **Ver actividades recientes de clientes asignados.** El colaborador podrá ver una lista de las actividades recientes completadas por sus clientes asignados.
- **Ver clientes asignados con tareas sin completar.** El colaborador podrá ver una lista de sus clientes asignados sin tareas asignadas.
- **Ver clientes asignados sin próximas tareas.** El colaborador podrá ver un listado de sus clientes sin próximas tareas asignadas.

Clientes

- **Listar clientes asignados.** El colaborador podrá ver una lista de los clientes asignados.

Vista específica de un cliente (*mismos que el administrador)

- **Añadir nueva tarea a cliente.**
- **Ver información de una tarea de un cliente.**
- **Modificar información personal de un cliente.**
- **Asignar/Eliminar entrenador personal.**
- **Enviar formulario a cliente.**
- **Ver respuestas de formulario de cliente.**
- **Eliminar formulario asignado a cliente.**
- **Compartir plan nutricional con cliente.**
- **Cambiar visibilidad de plan nutricional compartido con un cliente.**
- **Eliminar plan nutricional compartido con cliente.**
- **Compartir receta con cliente.**
- **Eliminar receta compartida con cliente.**
- **Compartir documento con cliente.**
- **Eliminar documento compartido con cliente.**
- **Registrar medidas de un cliente.**
- **Consultar medidas de cliente.**
- **Añadir imágenes de cliente.**
- **Consultar imágenes de un cliente.**

5.6.5. Cliente

Los casos de uso de un usuario con el rol de cliente son los siguientes:

Planificación

- **Ver información de tarea.** Selecciona en el calendario una tarea asignada y ver la información correspondiente.
- **Marcar tarea como finalizada.** Marcar como finalizada una tarea asignada.
- **Listar tareas**. Ver todas las tareas asignadas.

Información

- **Ver información del usuario.** Ver la información relacionada con el cliente.

- **Contestar cuestionario PAR-Q.** Contestar el cuestionario PAR-Q.
- **Ver entrenadores personales asignados.** Ver los entrenadores personales que el usuario tiene asignados.
- **Ver formularios recibidos.** Lista de formularios recibidos para ser respondidos.
- **Responder formulario.** Responder a las preguntas de un formulario recibido.

Nutrición

- **Listar planes nutricionales asignados.** Ver los planes nutricionales asignados con las comidas disponibles y los alimentos asignados a cada comida con sus valores nutricionales.

Documentos

- **Ver documento compartido.** Ver un documento compartido.

Evolución

- **Introducir medidas personales.** Introduce las medidas corporales o los valores de peso corporal o muscular o porcentaje de grasa.
- **Consultar evolución de métricas.** Ver en un gráfico o en una tabla los registros introducidos previamente.
- **Subir imágenes de progreso.** Sube imágenes de frente, de espaldas y de perfil introduciendo la fecha en la que fueron tomadas.
- **Consultar imágenes de progreso subidas.** Lista con las imágenes subidas anteriormente.

5.6.6. Casos de uso en común

Perfil

- **Actualizar imagen de perfil.** Sube o actualiza la imagen del perfil.
- **Eliminar imagen de perfil.** Elimina la imagen del perfil.
- **Modificar nombre, apellidos o email.** Actualiza el nombre, los apellidos o el email del usuario.
- **Cambiar contraseña.** Actualiza la contraseña de inicio de sesión introduciendo la antigua contraseña y la nueva contraseña que cumpla los requisitos.

Chat

- **Enviar mensaje.** Enviar un mensaje a otro usuario por el chat.
- **Seleccionar chat.** Selecciona al usuario para enviarle un mensaje.

Notificaciones

- **Listar notificaciones.** Muestra un listado con las notificaciones disponibles.

6. Implementación y desarrollo

En este apartado se va a realizar una descripción detalladas del proceso de implementación y de desarrollo de la aplicación web. Para comenzar se va a realizar una descripción de los proyectos creados: uno para la implementación de la API y otro para la implementación del Frontend. A continuación, se entrará en detalle en el proceso de desarrollo de los aspectos más relevantes que forman parte del trabajo.

6.1. Estructura del proyecto

Para realizar una explicación de la estructura del proyecto que forma este trabajo habrá que dividir el proceso en dos, ya que los proyectos que forman la API y el Frontend son independientes.

6.1.1. Frontend

En la Figura 6.1 se muestra la estructura del proyecto que forma el Frontend desarrollado mediante el framework AngularJS.

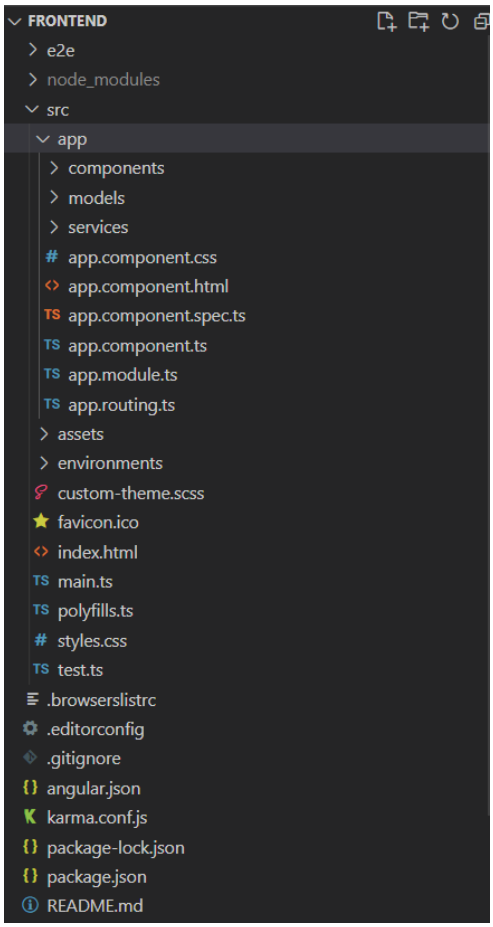

Figura 6.1. Estructura del Frontend.

El proyecto se llama *FRONTEND*, y cuenta con varios directorios importantes organizados de la siguiente manera:

- ***node_modules:*** módulos adicionales utilizados en el desarrollo.
- ***/src/app/components:*** contiene los diferentes componentes utilizados en las vistas que forman la aplicación web. Cada componente cuenta con diferentes métodos que permiten la interacción con los datos.
- ***/src/app/models:*** contiene los archivos con los diferentes modelos utilizados.
- ***/src/app/services:*** contiene diferentes servicios en los que se implementan las operaciones de comunicación con la API y otras operaciones como la manipulación de cookies.
- ***/src/assets:*** contiene un directorio con las imágenes utilizadas en el diseño de la aplicación u otras imágenes mostradas por defecto en ciertos componentes.
- ***/src/environments:*** contiene los archivos de configuración. En ellos se especifican la URL de la API y la propia URL del servidor en el que se desplegará el Frontend.

En el archivo *app.routing* del directorio *src/app* se especifican las rutas de la aplicación como se puede ver en la Figura 6.2.

```
const APP_ROUTES: Routes = [
    {path: '', component: HomeComponent},
    {path: 'home', component: HomeComponent},
    {path: 'login', component: LoginComponent},
    {path: 'signup/:planId', component: SignupComponent},
    {path: 'access', component: CardPricesComponent},
    {path: 'about-me', component: AboutMeComponent},
    {path: 'terms-and-conditions', component: TermsAndConditionsComponent},
    {path: 'not-found', component: NotFoundComponent},

    {path: 'admin', component: HomeAdminComponent},
    {path: 'admin/home', component: HomeAdminComponent},
    {path: 'admin/all-events-and-sessions', component: AllEventsAndSessionsCompone
    {path: 'admin/clients', component: ClientsComponent},
    {path: 'admin/tags', component: ClientTagsComponent},
    {path: 'admin/groups', component: ClientGroupsComponent},
    {path: 'admin/exercises', component: ExercisesComponent},
    {path: 'admin/workouts', component: WorkoutsComponent},
    {path: 'admin/video-sessions', component: VideoSessionsComponent},
    {path: 'admin/nutritional-plans', component: NutritionalPlansComponent},
    {path: 'admin/recipes', component: RecipesComponent},
    {path: 'admin/food', component: FoodComponent},
    {path: 'admin/groupings-plan', component: GroupingsPlanComponent},
    {path: 'admin/files', component: FilesComponent},
    {path: 'admin/forms', component: FormsComponent},
    {path: 'admin/programs', component: ProgramsComponent},

    {path: 'admin/team', component: TeamComponent},
    {path: 'admin/plans', component: PlansAndPaymentsComponent},

    {path: 'client/home/:clientId', component: HomeClientComponent},
    {path: 'client/info/:clientId', component: InfoClientComponent},
    {path: 'client/plan/:clientId', component: PlanClientComponent},
    {path: 'client/nutrition/:clientId', component: NutritionClientComponent},
    {path: 'client/files/:clientId', component: FilesClientComponent},
    {path: 'client/evolution/:clientId', component: EvolutionClientComponent},
```

Figura 6.2. Rutas del Frontend.

El archivo inicial es el *index.html*. Dentro de este archivo está el elemento *<app-root></app-root>*, que es definido dentro del componente principal en */src/app/app-component.ts.*

```
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'Frontend';
}
```

Figura 6.3. Componente principal.

En el archivo *.html* del componente principal se especifica el *router outlet* de Angular. Dicho marcador se utiliza para cargar los diferentes componentes de forma dinámica dependiendo el estado actual de la ruta.

6.1.2. Servidor

En la Figura 6.4 se muestra la estructura del proyecto que forma la API desarrollada mediante NodeJS y el framework Express.

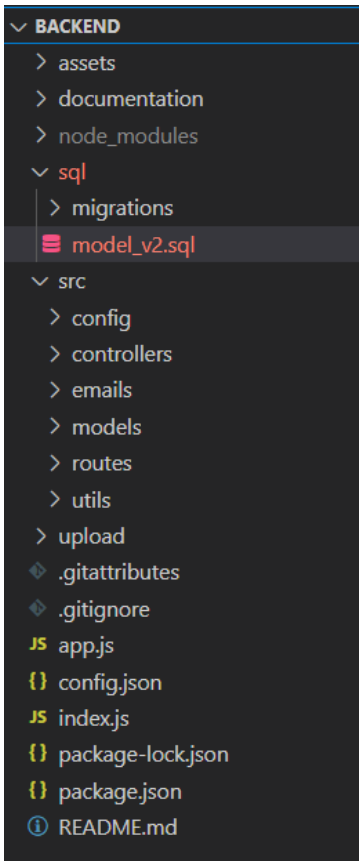

Figura 6.4. Estructura del servidor.

El proyecto se llama *BACKEND*, y cuenta con varios directorios organizados de la siguiente manera:

- ***node_modules:*** módulos adicionales utilizados en el desarrollo.
- ***assets:*** contiene ficheros estáticos como las imágenes o archivos que se devuelven en caso de error en alguna consulta.
- ***sql:*** contiene un archivo con la definición del modelo de las tablas en lenguaje SQL.
- ***/src/config:*** contiene los archivos de configuración para realizar la conexión con la base de datos MySQL y para enviar correos.
- ***/src/controllers:*** contiene diferentes controladores en los que se implementan o invocan a los métodos del modelo para realizar diferentes operaciones o devolver datos a partir de consultas a la API.

- ***/src/emails:*** contiene las plantillas de las vistas enviadas en los correos.
- ***/src/models:*** contiene los modelos de la aplicación. Cada modelo cuenta con métodos para crear, eliminar, buscar o actualizar el elemento.
- ***/src/routes:*** contiene los archivos de rutas que forman la API.
- ***/src/utils:*** contiene un archivo utilizado para obtener las rutas de los documentos o imágenes almacenados.
- ***upload:*** directorio para almacenar temporalmente imágenes o archivos.

El archivo *index.js* despliega un servidor que contiene la API en el puerto 3900 con la configuración establecida en el archivo *app.js* en el que se establece el *CORS*, el *middleware* y se especifican las rutas definidas en los archivos de rutas del directorio *src/routes.*

6.2. Chat y notificaciones

El chat y las notificaciones tienen un modo de funcionamiento muy similar. Ambas funcionalidades utilizan los sockets para el envío o recibimiento de mensajes y notificaciones de texto.

Los sockets utilizados se ejecutan en el servidor y en el cliente, creando un canal de comunicación fiable y eficiente que permiten el intercambio de la información.

En el *Backend* se ejecuta un socket en el archivo *app.js* para el chat de mensajes (Figura 6.5). Se reciben los mensajes emitidos desde un cliente y se reenvían para que les lleguen a los otros clientes. En el Frontend, en el componente que forma el chat de mensajes, se ejecuta el socket para recibir y mostrar los mensajes enviados en tiempo real (Figura 6.6), y se emiten por el socket los nuevos mensajes enviados por el cliente una vez validados.

Las notificaciones tienen un funcionamiento ligeramente distinto, en lugar de ejecutar un socket que este esperando los mensajes en el servidor para ser reenviados, se crean y emiten cada vez que se produce un evento que debe de ser notificado, y en el Frontend se escuchan estas notificaciones en todas las vistas del usuario para mostrarlas en tiempo real.

```
var server = require('http').createServer(app);
global.io = require('socket.io')(server, {
    cors: {
        origin: '*',
    }
});

io.on('connection', function(socket) {
    socket.on('save-message', function(newMessage) {
        console.log(newMessage);
        io.emit('new-message-' + newMessage.toId, { message: newMessage });
    });
});
```

Figura 6.5. Socket del chat de mensajes en el servidor.

```
this.socket.on('new-message-' + this.user.id, function(data) {
  var data_all = {
    fromId: data.message.fromId,
    toId: data.message.toId,
    msg: data.message.msg,
    created: data.message.created,
  };

  this.messages.push(data_all);
  console.log(data_all);


}.bind(this));
```

Figura 6.6. Socket del chat de mensajes en el Frontend.

6.3. Pagos con Stripe

Los pagos realizados en la plataforma por parte de los clientes que contratan un plan de entrenamiento personal de los disponibles se realizan a través de Stripe. Para realizar estos pagos se sigue un flujo de datos como el descrito en la Figura 6.7.

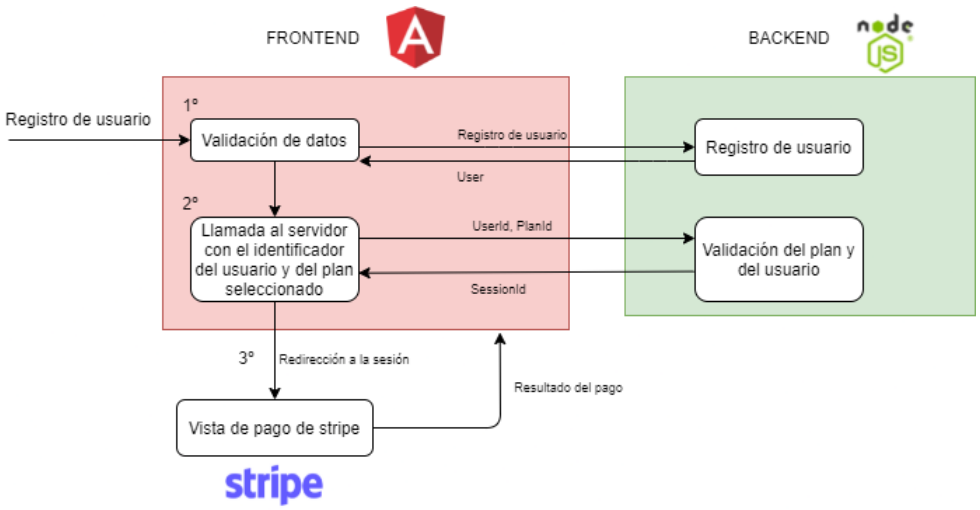

Figura 6.7. Flujo de datos de un pago.

Para poder realizar los pagos habrá que crear una cuenta en Stripe y crear un par de claves pública y privada en la parte de *Desarrolladores* para utilizarlas en los proyectos creados. La clave pública se utilizará en el Frontend y la clave secreta en el *Backend* (Figura 6.8).

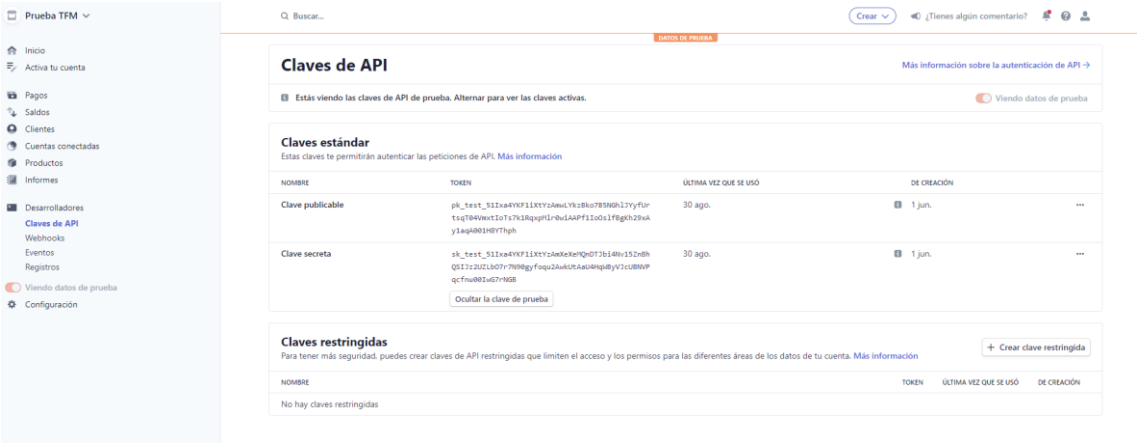

Figura 6.8. Claves de API de Stripe.

El flujo de datos comienza en la parte del *Frontend*, mediante la instalación del módulo ngx-stripe en el proyecto:

```
npm install ngx-stripe
```

Una vez instalado este módulo se importará en el archivo *app.module* del proyecto*:*

```
import { NgxStripeModule } from 'ngx-stripe';
```

, y se añadirá dentro de los *imports* indicando la clave pública obtenida en la página de Stripe.

```
    NgxStripeModule.forRoot('pk_test_51Ixa4YKF1iXtYzAmwLYkzBko7B5NGhlJYyfUrts
qT04VmxtIoTs7k1RqxpMlr0wiAAPf1IoOslfBgKh29xAy1aqA001H8YThph'),
```

Una vez hechos estos pasos, ya se podrá utilizar Stripe en cualquier componente. En el componente *SignupComponent* se importará el servicio de Stripe en el constructor.

Cuando se hayan validado los datos de registro, enviado al servidor y obtenido la respuesta de que el usuario se ha registrado correctamente, se hará un envió de datos al servidor con el identificador del usuario obtenido y el identificador del plan seleccionado para obtener el identificador de la sesión de Stripe. Con el identificador de sesión obtenido, si la anterior respuesta es correcta, se utilizará el servicio de Stripe para redirigir al usuario a la vista de pago.

```
this._stripeService.redirectToCheckout({ sessionId: response.id })
```

Si el pago se realiza correctamente se redirigirá al usuario a la vista de *Inicio de sesión.*

En la parte del *Backend* únicamente habrá que implementar la ruta y el método a ejecutar. Este método validará que el plan y el usuario existen en la aplicación y creará la sesión de Stripe cuyo identificador se devolverá al *Frontend* (Figura 6.9).

```
router.post('/stripe/checkout', StripeController.checkout);
```

```
const session = stripe.checkout.sessions.create({
    payment_method_types: ['card'],
    line_items: [{
        price_data: {
            currency: 'eur',
            product_data: {
                name: name,
            },
            unit_amount: price * 100,
        },
        quantity: 1,
    }, ],
    mode: 'payment',
    success_url: 'http://localhost:4200/login',
    cancel_url: 'http://localhost:4200',
});
```

Figura 6.9. Sesión de Stripe creada en el Backend.

La clave secreta se indicará en la declaración del módulo de Stripe utilizado.

6.4. Almacenamiento de la información

Para almacenar toda la información utilizada en la aplicación web se ha utilizado una base de datos MySQL. Se crea una base de datos con el mismo nombre especificado en el archivo de configuración de la base de datos del proyecto *Backend* y las tablas disponibles en el archivo */src/model.sql*.

Los imágenes y archivos manejados en la aplicación se almacenan en directorios en una carpeta oculta en el directorio personal del usuario del ordenador en el que se ejecuta la aplicación. De esta forma esa carpeta es menos probable que sea eliminada de forma accidental y no afecte a otras tareas de la aplicación.

6.5. Sistema de control de versiones

El proyecto se ha almacenado en un repositorio de Github. Github es una plataforma que utiliza el sistema de control de versiones de Git. En las Figuras 6.10 y 6.11 se pueden ver las contribuciones realizadas en ambos proyectos.

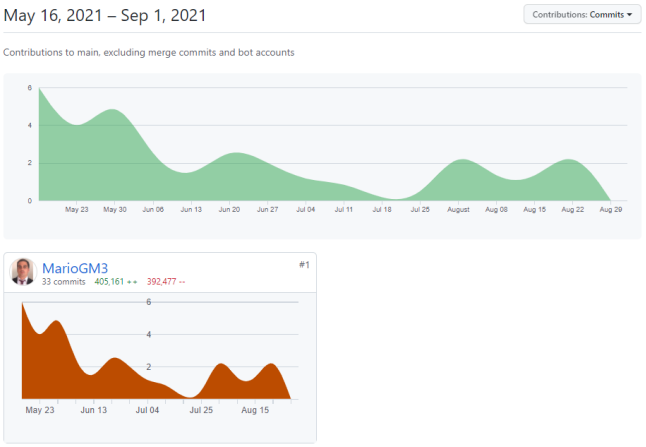

Figura 6.10. Proyecto TFM-BACKEND.

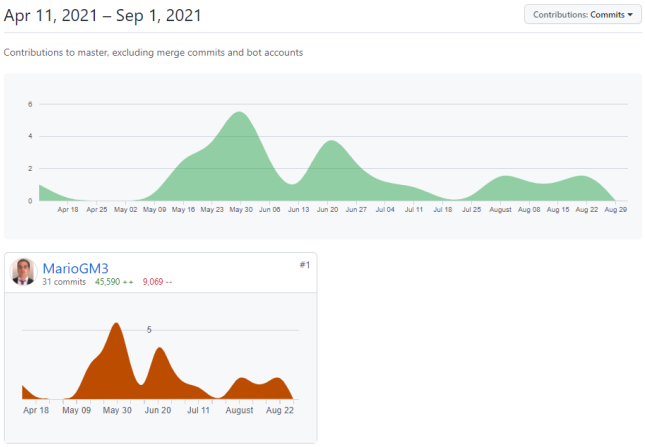

Figura 6.11. Proyecto TFM-FRONTEND.

7. Resultados

En esta sección se muestran las diferentes funcionalidades con las que cuenta la aplicación final mediante capturas de las vistas que nos podemos encontrar. En cada apartado se realizará una pequeña descripción de las funciones más relevantes.

7.1. Página web, inicio y registro

En la Figura 7.1 se puede ver la pantalla principal de la aplicación web. En ella se muestra información relativa a los métodos empleados y los servicios ofrecidos de forma general. En la parte superior se pueden observar dos botones: uno para iniciar sesión y otro para realizar el registro.

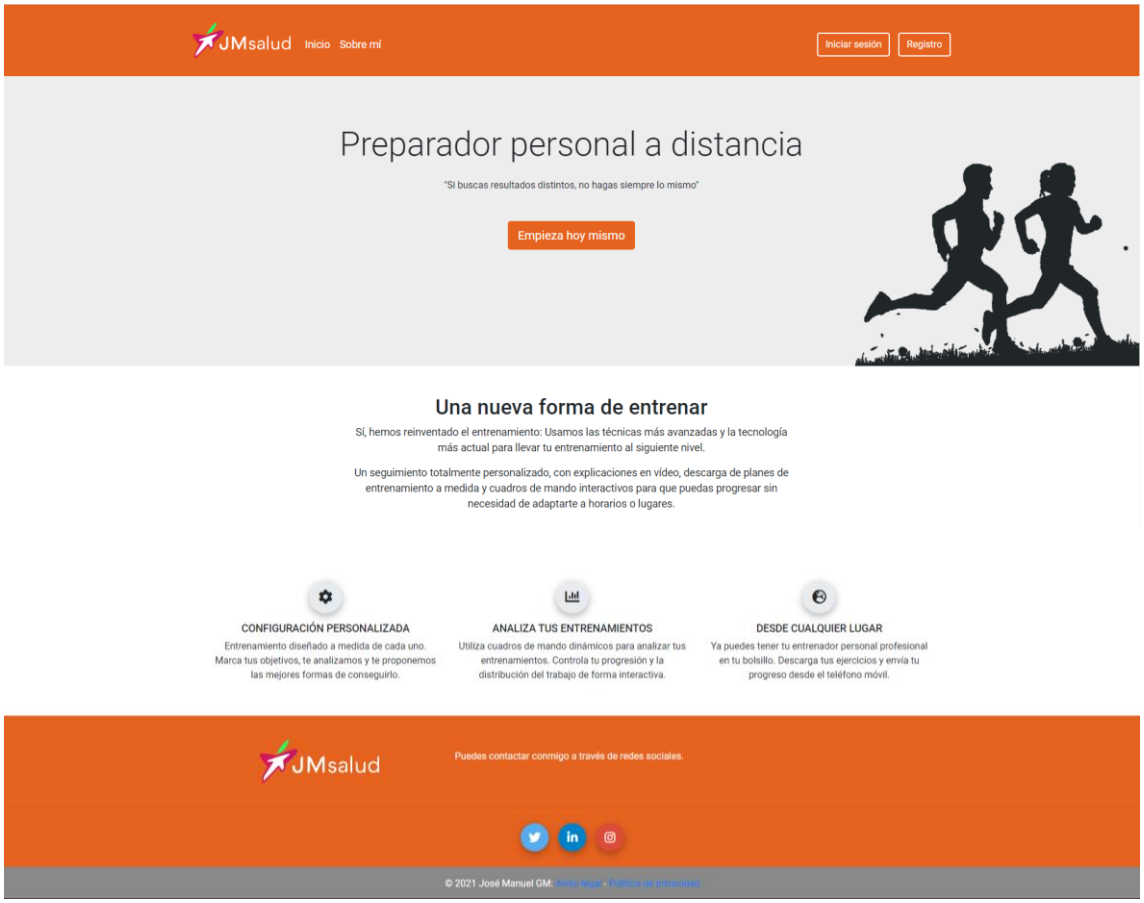

Figura 7.1. Pantalla principal.

Si hacemos clic en el botón de registro se mostrará la vista de la Figura 7.2.

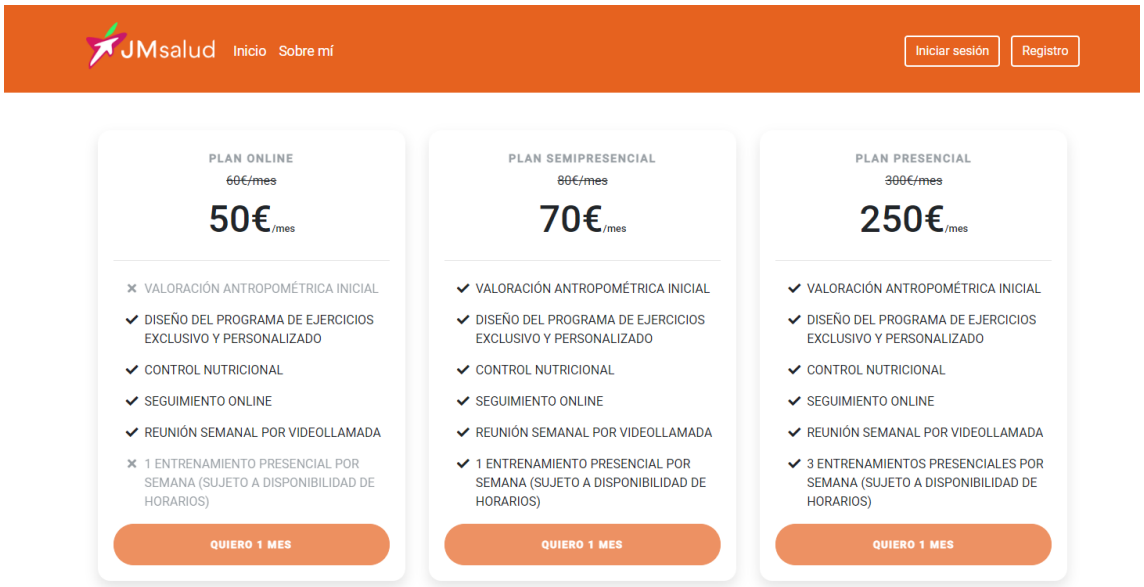

Figura 7.2. Pantalla *Seleccionar plan.*

Para realizar el registro habrá que seleccionar uno de los planes disponibles haciendo clic en la parte inferior de ellos, en el botón *Quiero 1 mes.* Cada plan disponible muestra las diferentes características que se ofrecen, el nombre, el precio, y la oferta, en caso de que tuviera.

En algunos casos no es necesario realizar este registro, el administrador de la aplicación podrá añadir clientes manualmente y proporcionar las credenciales de acceso a los mismos vía email u otro medio. Se explicará con más detalle estos casos en el apartado de *Administración.*

Para iniciar sesión se utilizará el correo y contraseña del registro, y dependiendo del rol del usuario (administrador, colaborador o cliente) se mostrará una pantalla principal diferente.

7.2. Configuración del perfil, chat y notificaciones

Las vistas o funcionalidades mostradas en este apartado son comunes, independientemente del rol que tenga el usuario. Una vez que el usuario inicia sesión, en la parte superior derecha, podemos ver diferentes botones como los que se muestran en la Figura 7.3.

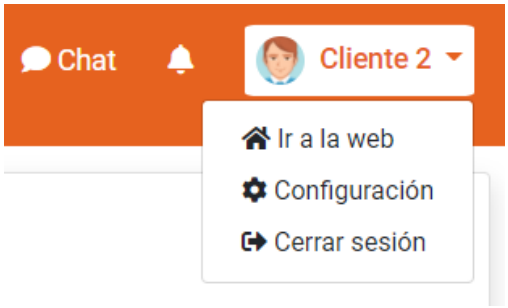

Figura 7.3. Chat, Notificaciones y Configuración.

Cada usuario, dependiendo del rol, tendrá en la parte izquierda de la vista del chat los usuarios con los que podrá enviarse mensajes:

- **Administrador:** podrá enviar mensajes a todos los colaboradores y clientes.
- **Colaborador:** podrá enviar mensajes al administrador y a los clientes a su cargo.
- **Cliente:** podrá enviar mensajes al administrador y a los colaboradores que tenga asignados.

La pantalla de *Configuración* se muestra en le Figura 7.4.

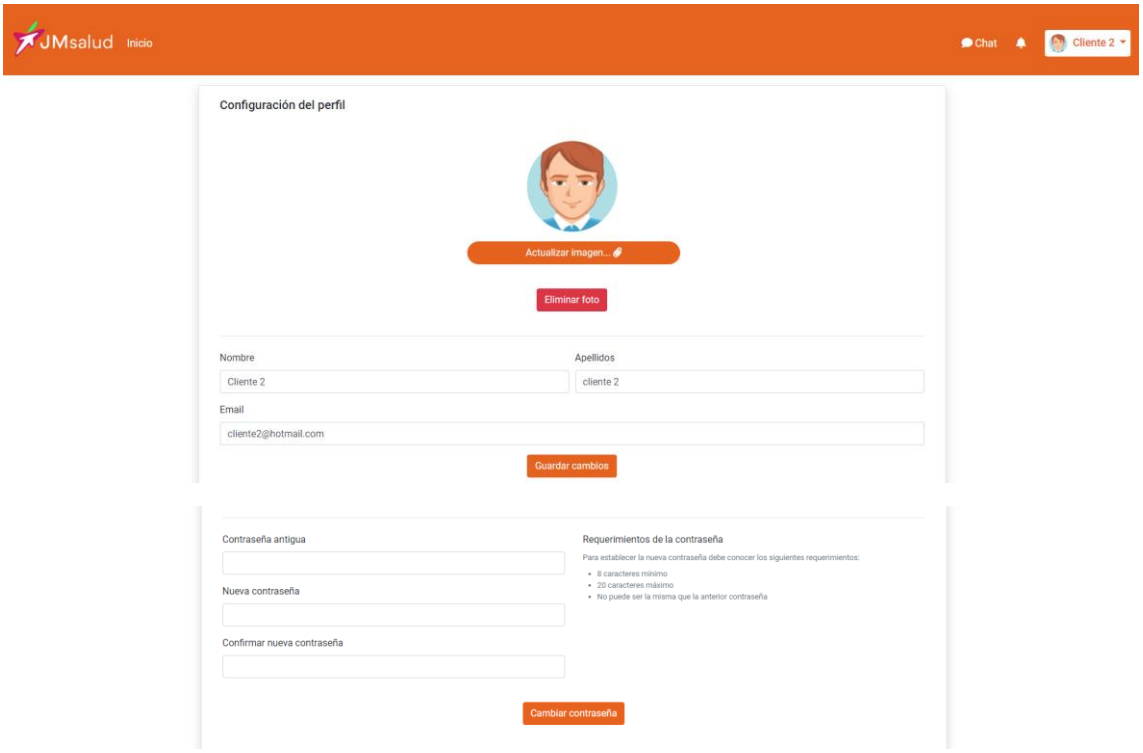

Figura 7.4. Pantalla de *Configuración*.

En la parte superior se puede ver la foto de perfil que se mostrará en la aplicación y se podrá actualizar subiendo otra foto o eliminar, y se mostrará la imagen por defecto.

Debajo de la opción anterior se puede modificar el nombre, los apellidos y el email. Para actualizar los cambios habrá que hacer clic en *Guardar cambios.*

En la parte inferior se podrá actualizar la contraseña de inicio de sesión, para ello habrá que introducir la contraseña actual y la nueva contraseña que cumpla con los requisitos mostrados.

En la parte superior, al lado del botón de acceso al chat se pueden ver las notificaciones recibidas. Si se hace clic sobre el botón se mostrará una lista de las notificaciones no leídas, y en la parte inferior un botón para ver todas las notificaciones disponibles leídas y no leídas.

7.3. Administración

En la Figura 7.5 se puede ver la pantalla inicial que un tiene usuario administrador al iniciar sesión.

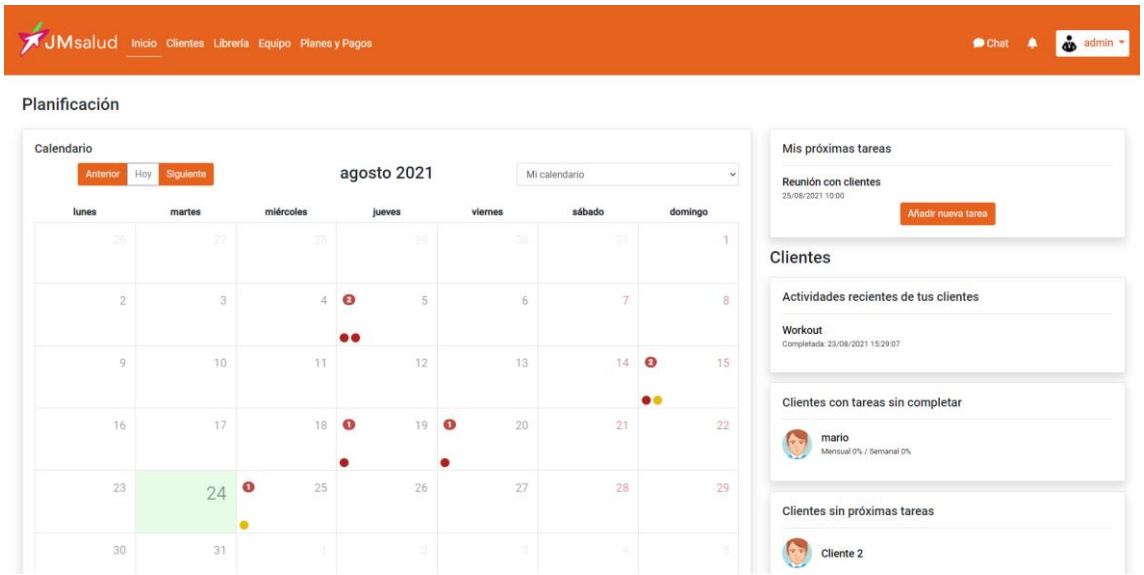

Figura 7.5. Pantalla principal del administrador.

Se muestra un calendario con todos los eventos y sesiones creados por el administrador. Existe una opción (Figura 7.6) para que el administrador seleccione los calendarios de cualquier otro usuario de la plataforma y vea sus tareas.

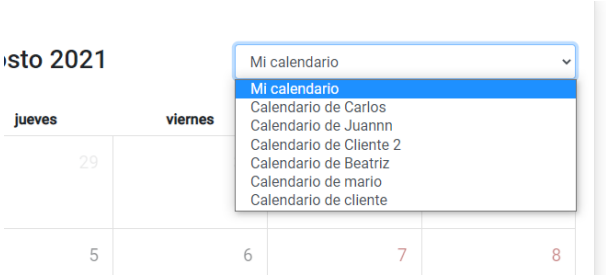

Figura 7.6. Selector de calendarios.

Los eventos, sesiones y tareas de los usuarios aparecerán representados en el calendario con un círculo de color. Si se quiere mostrar la información de un evento, sesión o tarea en concreto habrá que hacer clic sobre este círculo.

En la parte derecha se muestran las próximas tareas que tiene el administrador y un apartado de *Clientes*, en el que se muestran las actividades recientes realizadas por los clientes, las clientes con tareas sin completas, los clientes sin próximas tareas y los clientes cuyas subscripciones están a punto de finalizar.

7.3.1. Clientes

En la Figura 7.7 se puede ver la pantalla de administración de la vista de clientes.

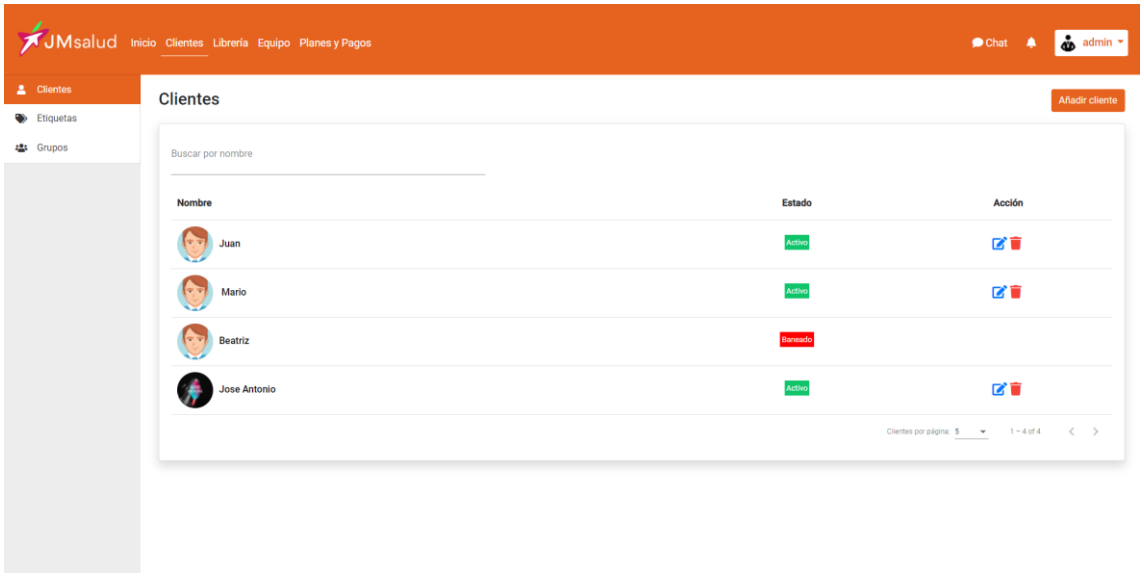

Figura 7.7. Pantalla de *Clientes.*

En esta lista aparece un listado con los clientes disponibles y su estado en la plataforma. A la derecha se tienen opciones de configuración y una opción para banearlos.

En la parte izquierda se dispone de un menú lateral en el que se podrá ir a la vista de etiquetas o de grupos.

En la vista de etiquetas que se puede ver en la Figura 7.8, se podrán añadir nuevas etiquetas para asignárselas a los clientes disponibles y poder filtrarlos a la hora de enviarles ciertas tareas o documentos.

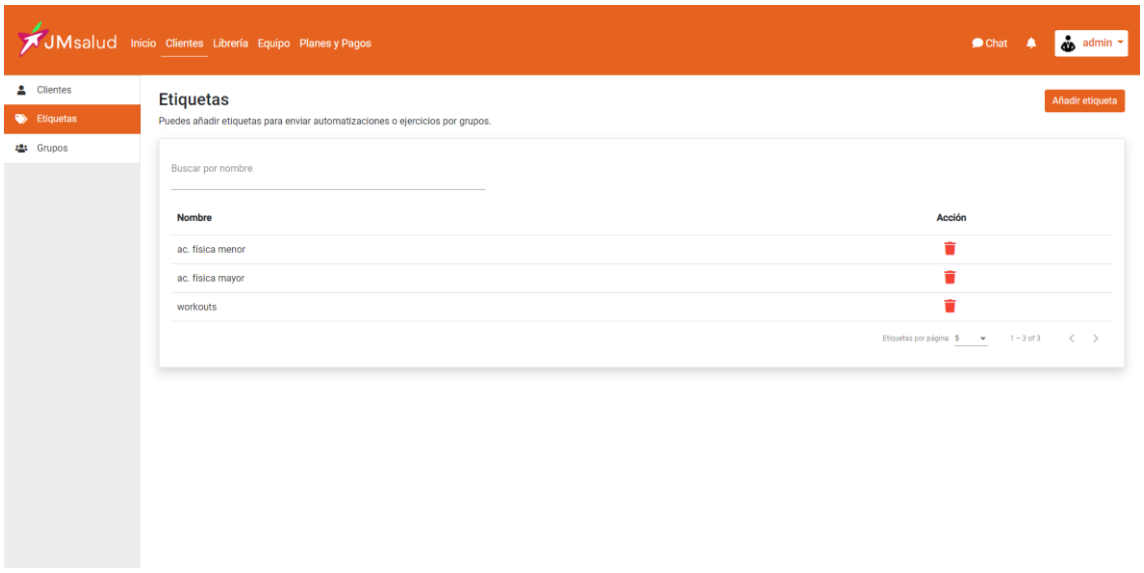

Figura 7.8. Pantalla de *Etiquetas de clientes.*

Para añadir una nueva etiqueta bastará con hacer clic en el botón de la derecha *Añadir etiqueta* e introducir un nombre (Figura 7.9). No se podrán añadir dos etiquetas iguales.

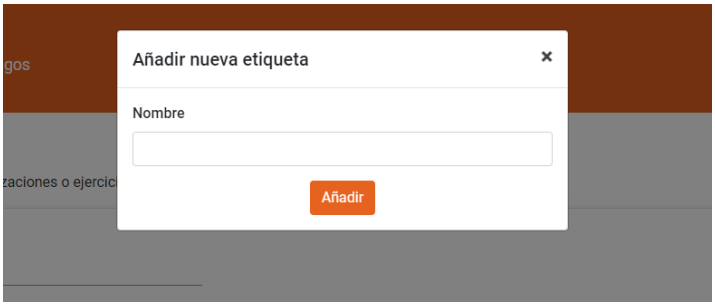

Figura 7.9. Añadir nueva etiqueta de cliente.

La función de grupos forma parte de las líneas de trabajo futuras. Dicha funcionalidad permitirá agrupar los clientes en grupos de usuarios para que la asignación de tareas sea más fácil y dinámica.

7.3.2. Librería

En la parte izquierda de cada vista disponible dentro de la librería de ejercicios, nutrición, … se dispone de un menú como el de la Figura 7.10 para desplazarse entre las diferentes vistas.

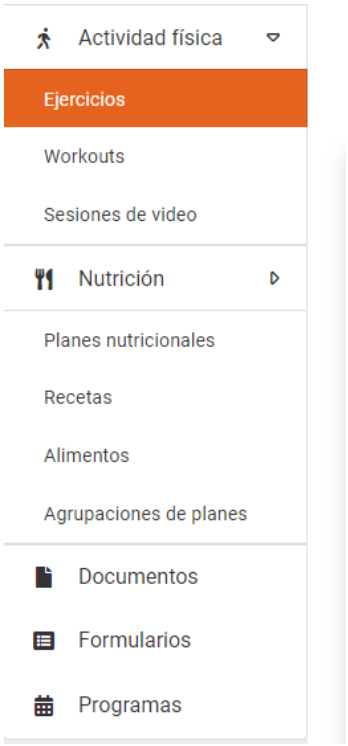

Figura 7.10. Menú de la *Librería*.

En la vista de *Ejercicios* que se puede ver en la Figura 7.11 se puede consultar la lista de ejercicios de la librería, modificarlos, eliminarlos y añadir nuevos ejercicios haciendo clic en el botón *Añadir ejercicio.*

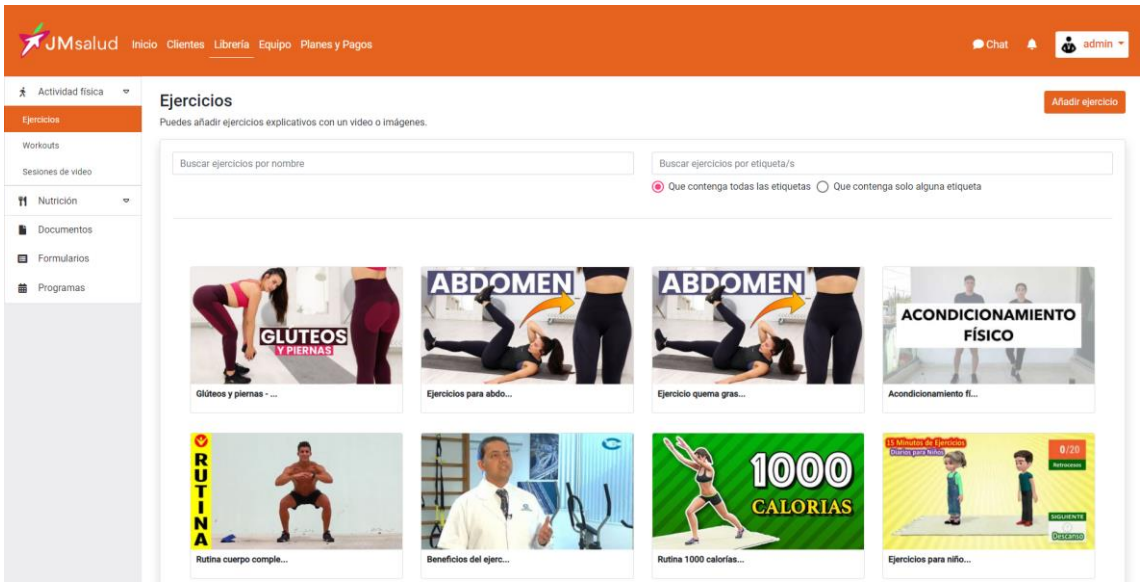

Figura 7.11. Vista de *Ejercicios.*

En la vista de *Workouts* que se puede ver en la Figura 7.12 se pueden ver todos los workouts disponibles en la librería y editar, eliminar o añadir nuevos.

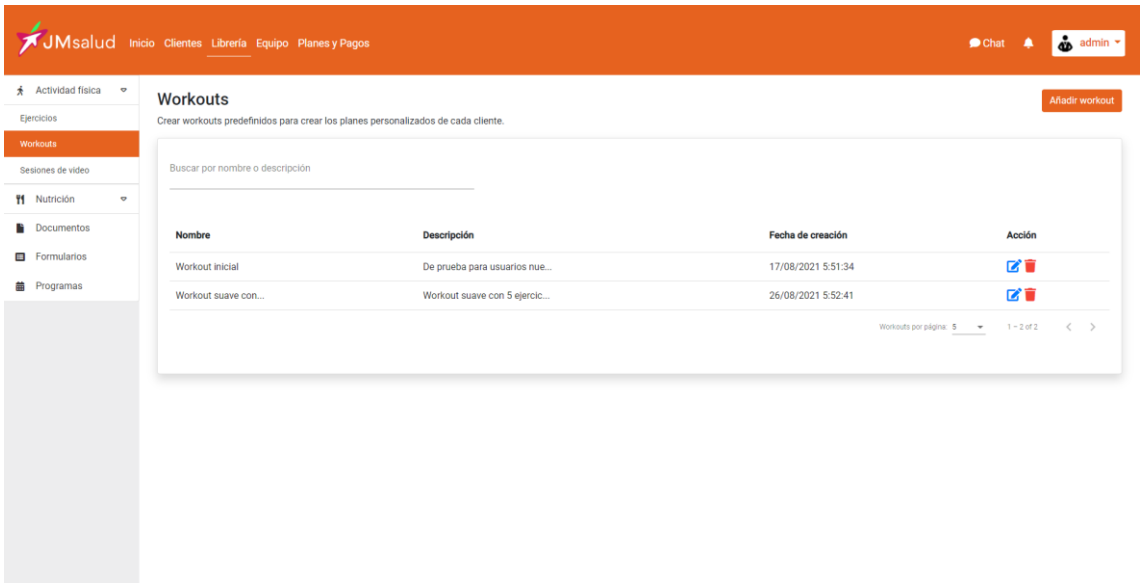

Figura 7.12. Vista de *Workouts.*

Para crear un nuevo *workout* se añadirá un nombre, una pequeña descripción, y diferentes ejercicios intercalados en bloques o con descansos de diferente duración.

En la vista de *Sesiones de video* que se puede ver en la Figura 7.13 se podrán consultar las sesiones de video añadidas y crear nuevas sesiones para utilizarlas en las sesiones programadas con los clientes. Estas sesiones se podrán modificar o eliminar.

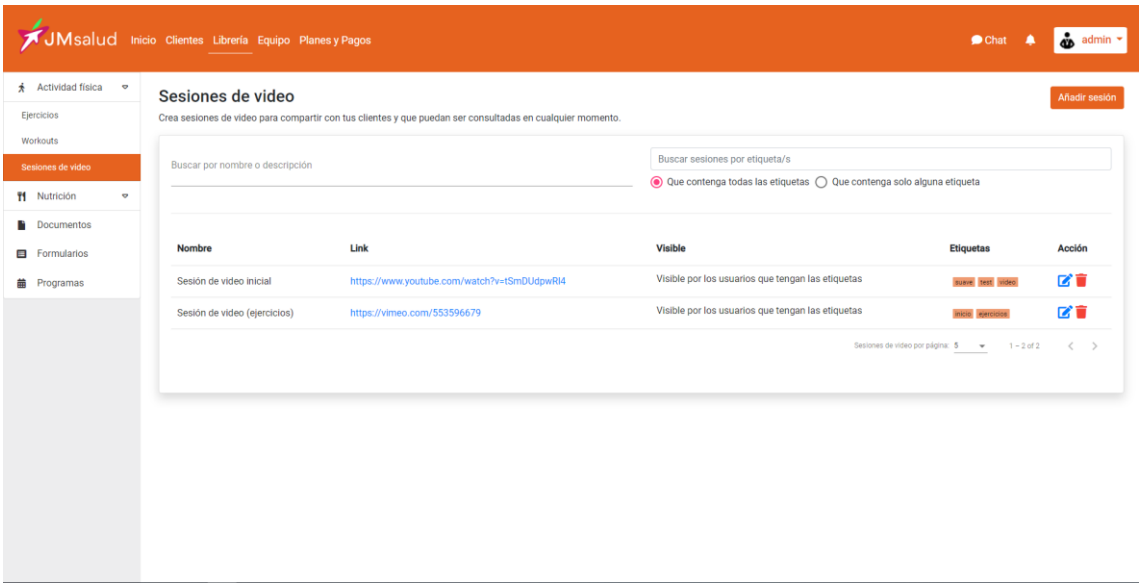

Figura 7.13. Vista de *Sesiones de video.*

Para añadir una nueva sesión de video habrá que añadir un nombre, unas instrucciones y un *link* a un vídeo de una plataforma externa. Se podrán añadir etiquetas para filtrar la sesión y seleccionar la visibilidad.

En la vista de *Planes nutricionales* que se puede ver en la Figura 7.14 se puede ver el listado de planes nutricionales disponibles en la plataforma y se podrán añadir, editar o eliminar los planes existentes.

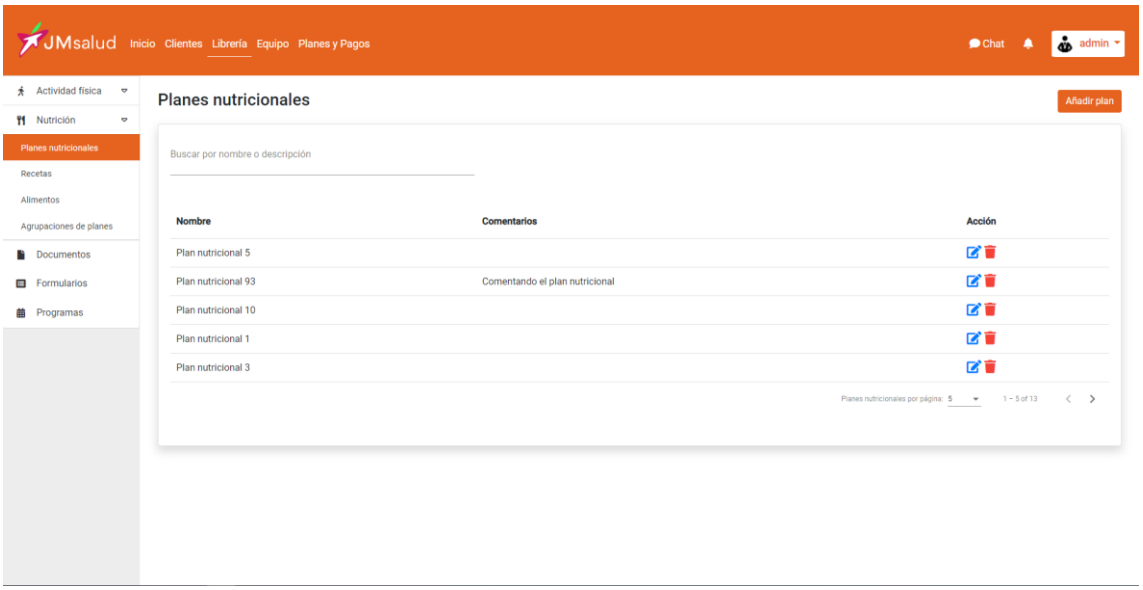

Figura 7.14. Vista de *Planes nutricionales.*

Para añadir un nuevo plan, habrá que hacer clic en el botón superior derecho *Añadir plan*. Se añadirá un nombre, unos comentarios y una o más comidas completadas con alimentos disponibles.

En la vista de recetas que se puede ver en la Figura 7.15 se podrá ver la lista de recetas añadidas en la plataforma y añadir, editar o eliminar nuevas recetas.

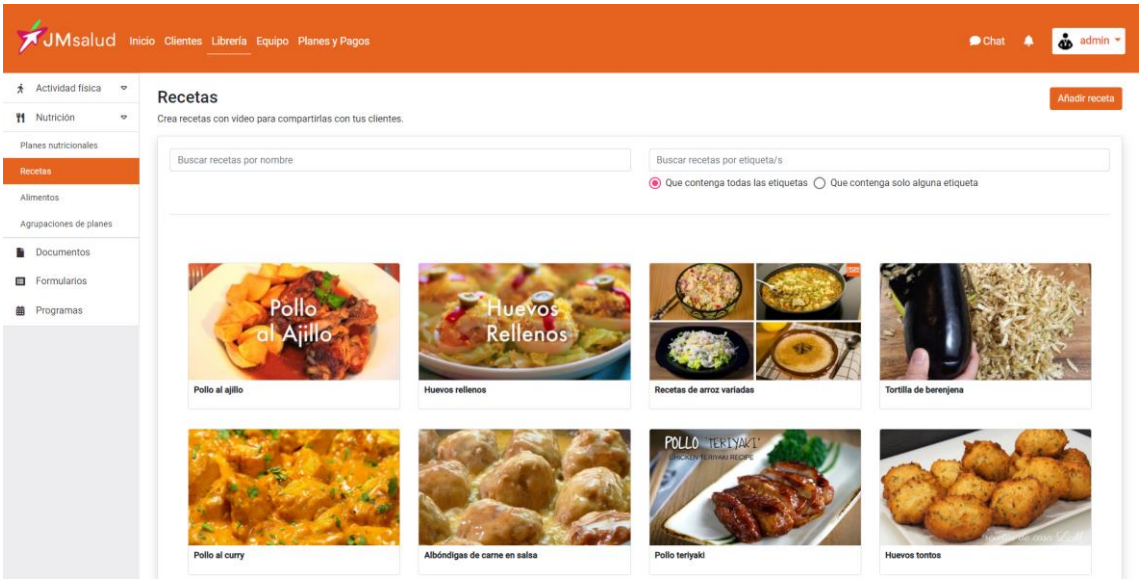

Figura 7.15. Vista de *Recetas.*

Para añadir una nueva receta se dispone de un modal similar al de añadir nuevos ejercicios. Se añadirá un nombre, unas instrucciones y el enlace a un vídeo o hasta tres imágenes si se desea. Además, se podrá añadir etiquetas para poder filtrar las recetas.

En la vista de alimentos de la Figura 7.16 se puede ver la lista de alimentos añadida en la plataforma, y añadir, editar o eliminar nuevos alimentos.

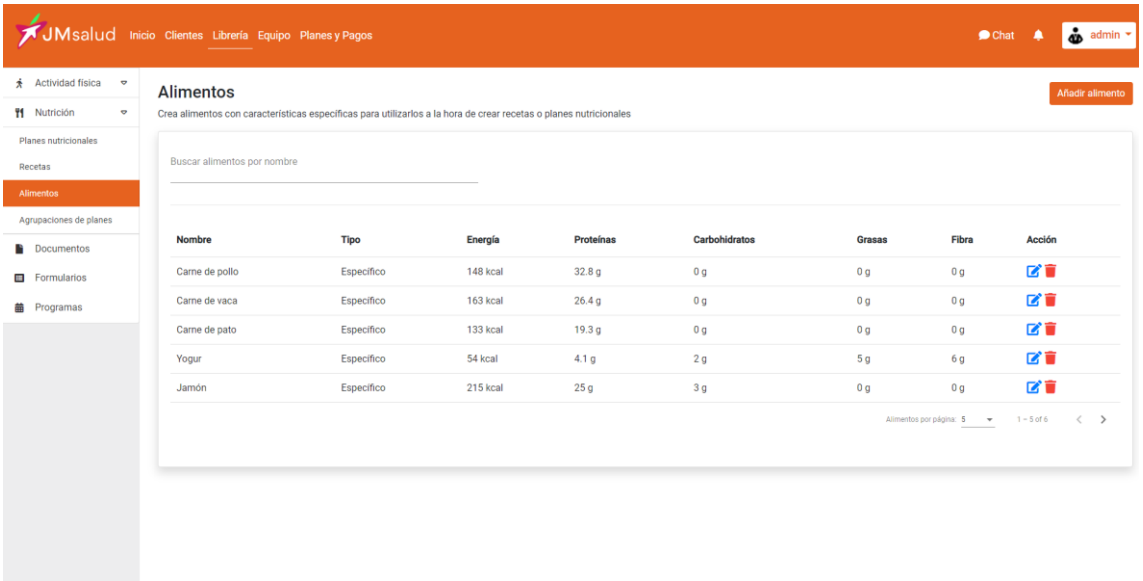

Figura 7.16. Vista de *Alimentos.*

Para añadir nuevos alimentos habrá que hacer clic en el botón superior derecho *Añadir alimento*. El nuevo elemento dispondrá de un nombre, un tipo, un enlace opcional y el valor nutricional para 100 gramos de ese alimento.

En algunos casos se podrá compartir un agrupamiento de varios planes nutricionales con un cliente. En la Figura 7.17 se puede ver la vista en la que se muestra una lista con las agrupaciones ya existentes, y se podrá añadir, editar o eliminar otras agrupaciones.

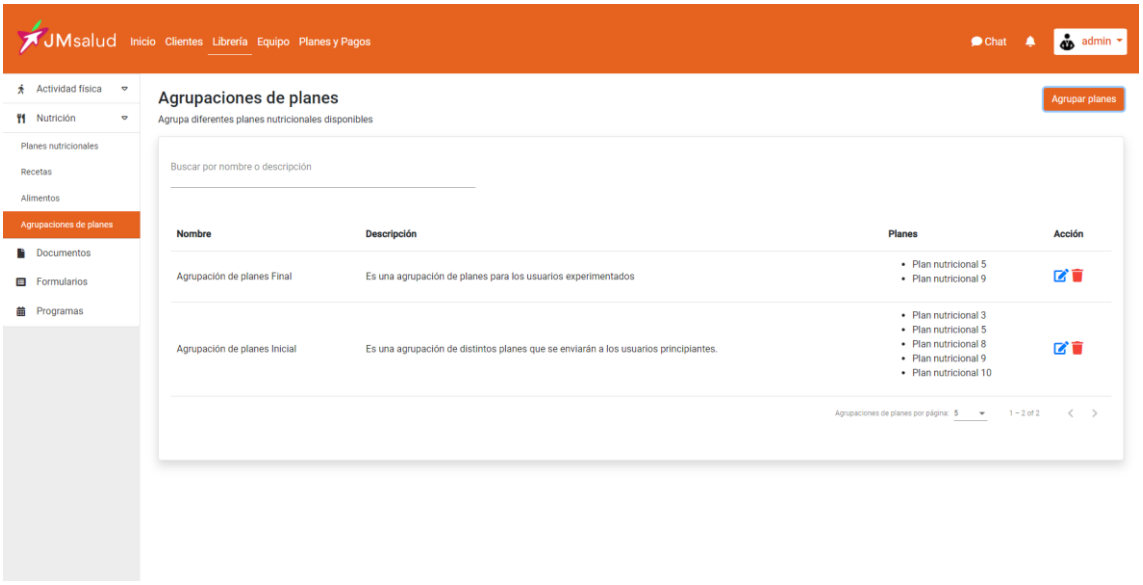

Figura 7.17. Vista de *Agrupaciones de planes.*

Para añadir una nueva agrupación de planes se hará clic en la parte superior derecha, en el botón *Agrupar planes*. Se abrirá un nuevo modal y se indicará un nombre, una descripción y se seleccionarán los planes nutricionales que se quieran agrupar.

En la vista de *Documentos* de la Figura 7.18 se puede ver la lista de documentos en formato PDF disponibles y su visibilidad dentro de la plataforma. Además, se podrán añadir, editar o eliminar otros documentos. Para ver un documento existente se puede hacer clic en el nombre del documento y se abrirá una nueva ventana del navegador mostrando el documento.

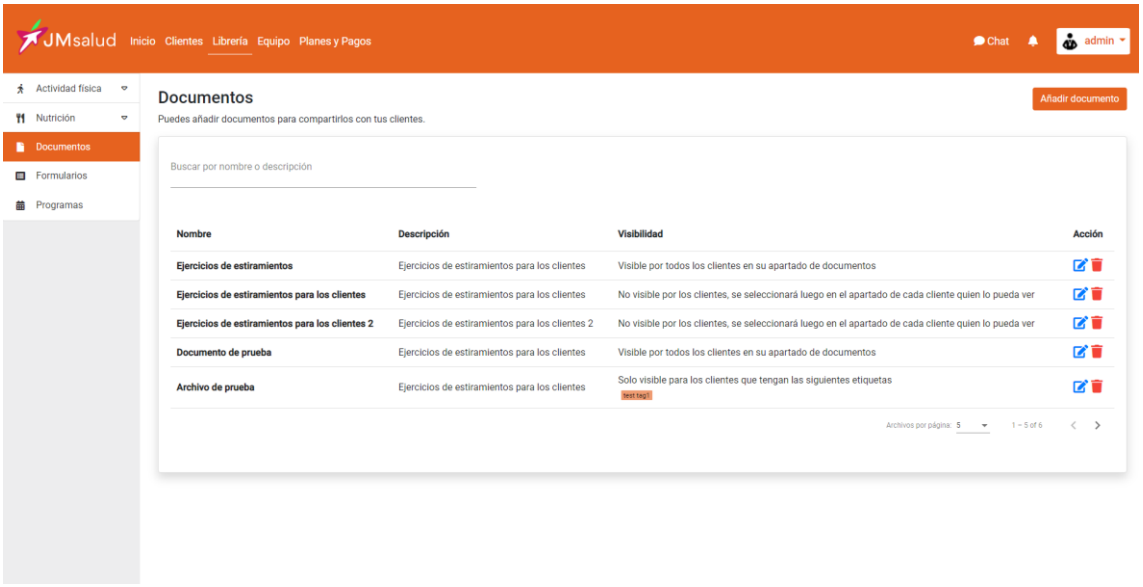

Figura 7.18. Vista de *Documentos.*

Para añadir un nuevo documento se hará clic en el botón superior derecho *Añadir documento*. Se añadirá un nombre, una descripción, se elegirá la visibilidad y se subirá un archivo haciendo clic en el botón *Subir pdf*….

En la vista de *Formularios* que podemos ver en la Figura 7.19 podemos ver los formularios disponibles en la plataforma para compartir con los clientes y ayudar a crear su planificación personal. Podremos añadir, editar o eliminar formularios.

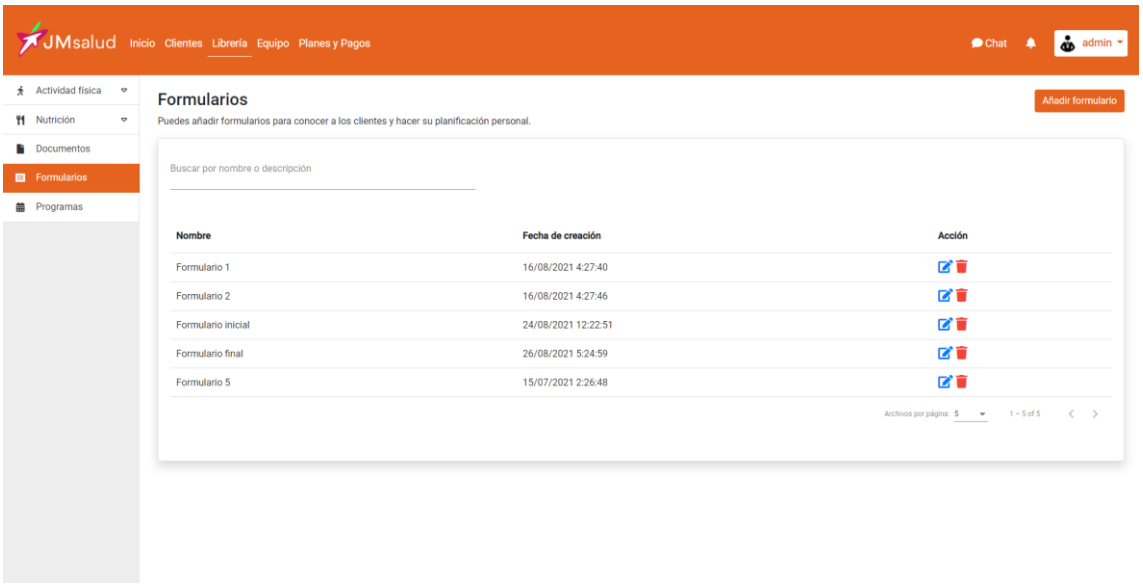

Figura 7.19. Vista de *Formularios.*

Para añadir un nuevo formulario habrá que hacer clic en el botón superior derecho *Añadir formulario*. En este modal se añadirá un nombre y las preguntas del formulario.

Los programas se utilizarán para tener programadas ciertas tareas durante varios días en varias semanas y asignárselas a los clientes que se desee. En la Figura 7.20 se puede ver una lista con los programas disponibles. Se podrá añadir, editar o eliminar los programas.

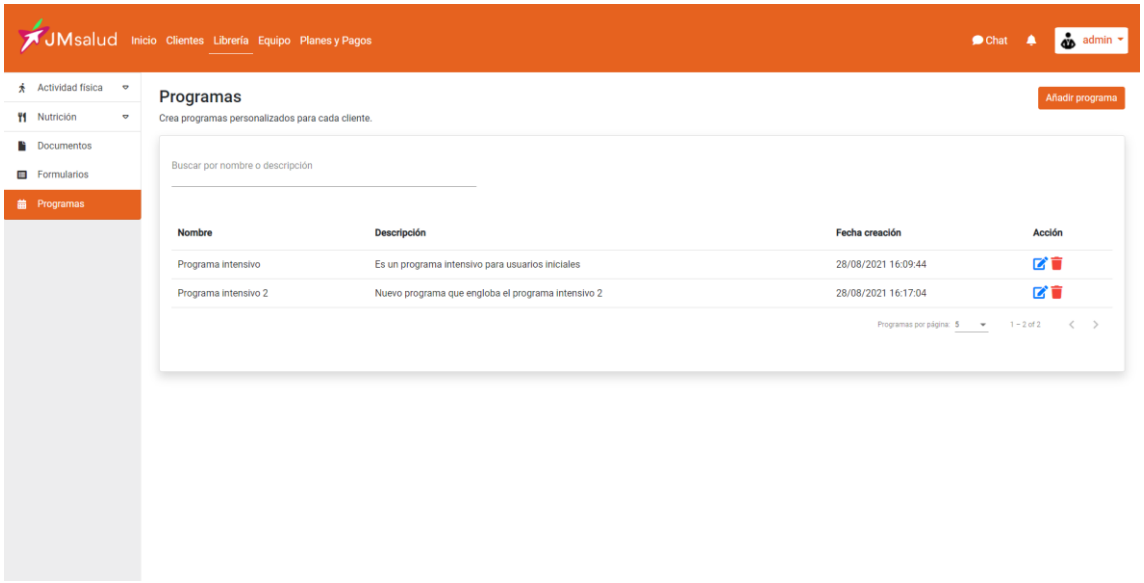

Figura 7.20. Vista de *Programas.*

Para añadir nuevos programas se hará clic en el botón superior derecho *Añadir programa* y se mostrará un modal. En este modal se añadirá un nombre, una descripción, se podrá enlazar otro programa, y se añadirán varias semanas con varias tareas en cada día como se desee.

7.3.3. Equipo

En esta vista el administrador podrá añadir nuevos miembros a su equipo de entrenadores personales. Introduciendo su nombre, apellidos y correo electrónico (Figura 7.21), el nuevo entrenador personal recibirá un correo en el que se le proporcione una contraseña de acceso a la plataforma.

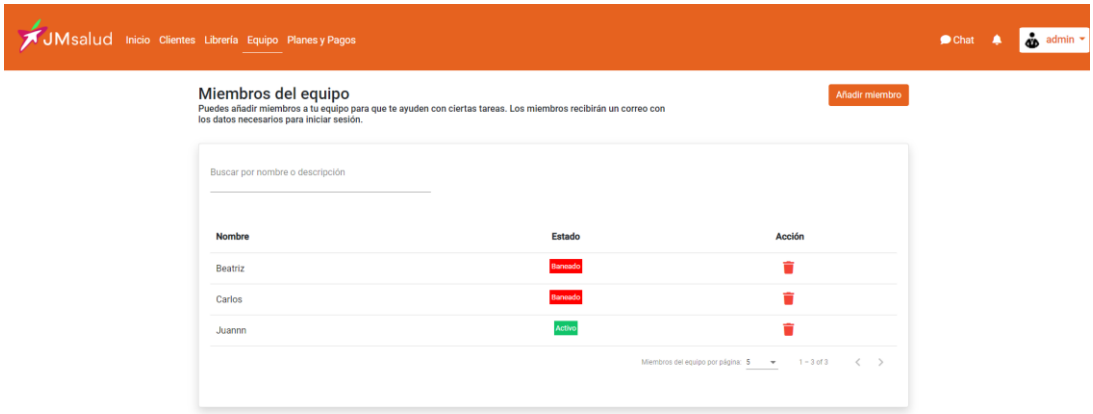

Figura 7.21. Pantalla de *Equipo*

El administrador podrá eliminar a miembros de su equipo en la lista de miembros, haciendo clic en el botón de la papelera.

7.3.4. Planes y pagos

En esta vista (Figura 7.22) se puede realizar la gestión de los planes disponibles y activos en cada momento en la aplicación.

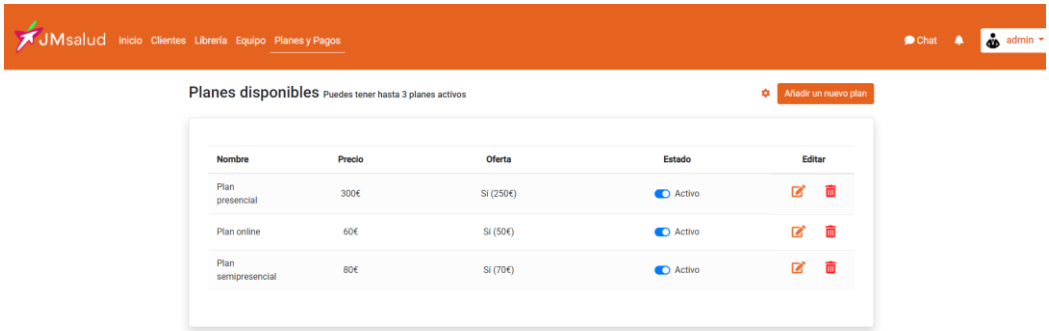

Figura 7.22. Pantalla de administración de *Planes y pagos.*

Los planes se podrán añadir haciendo clic en el botón superior *Añadir un nuevo plan*. Se podrá añadir un nombre, un precio, activar una oferta con un precio menor y añadir diferentes opciones descriptivas pudiéndolas activar o desactivar.

En la lista inferior se podrá eliminar un plan existente, editarlo (Figura 7.23) o activarlo. Se pueden tener activos un máximo de tres planes a la vez.

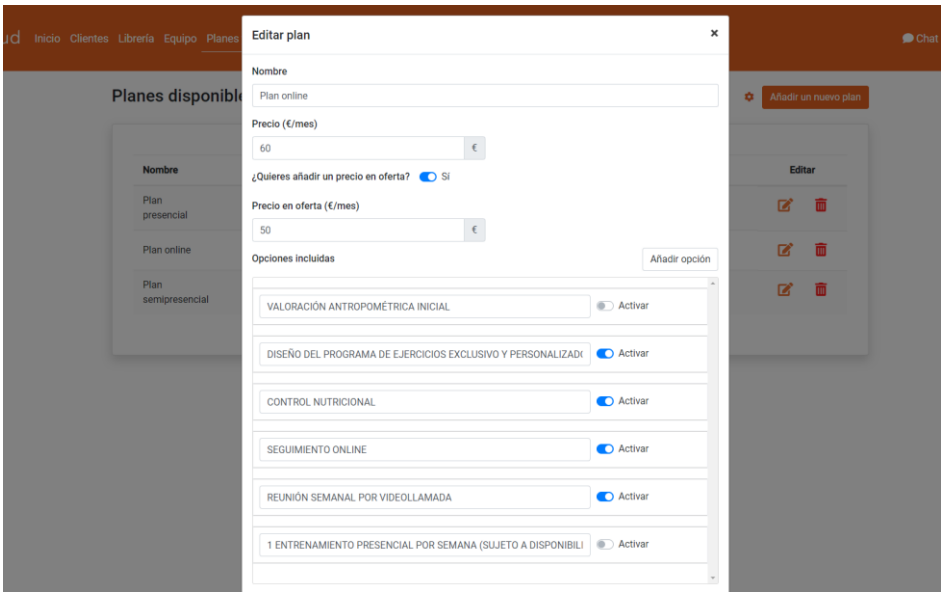

Figura 7.23. Editar un plan.

En la parte superior, al lado del botón para añadir un nuevo plan, se podrá abrir un nuevo modal para ordenar los planes activos disponibles y que se muestren en la pantalla de *Registro* en ese orden (Figura 7.24).

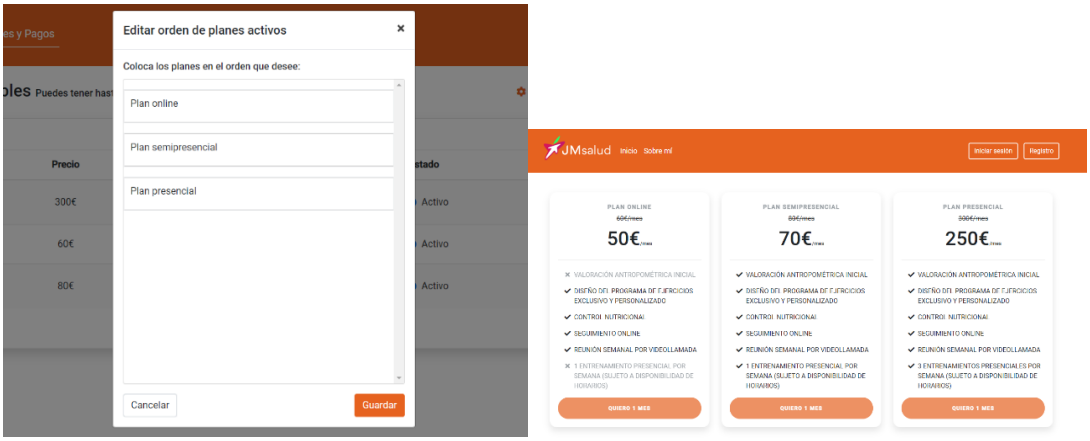

Figura 7.24. Ordenar planes activos.

7.3.5. Vista de cliente específico

Los administradores podrán acceder a la vista de un cliente específico desde la vista de *Clientes* o desde su página principal, desde la lista de clientes con tareas sin completar o asignadas. Dentro de esta vista específica se mostrarán las vistas mostradas en el siguiente apartado de clientes, pero con opciones para administrar al cliente.

En la vista de *Planificación* se mostrará un botón para añadir nuevas tareas en el calendario en la parte superior derecha, y se podrán añadir tareas directamente haciendo clic en un día del calendario (Figura 7.25).

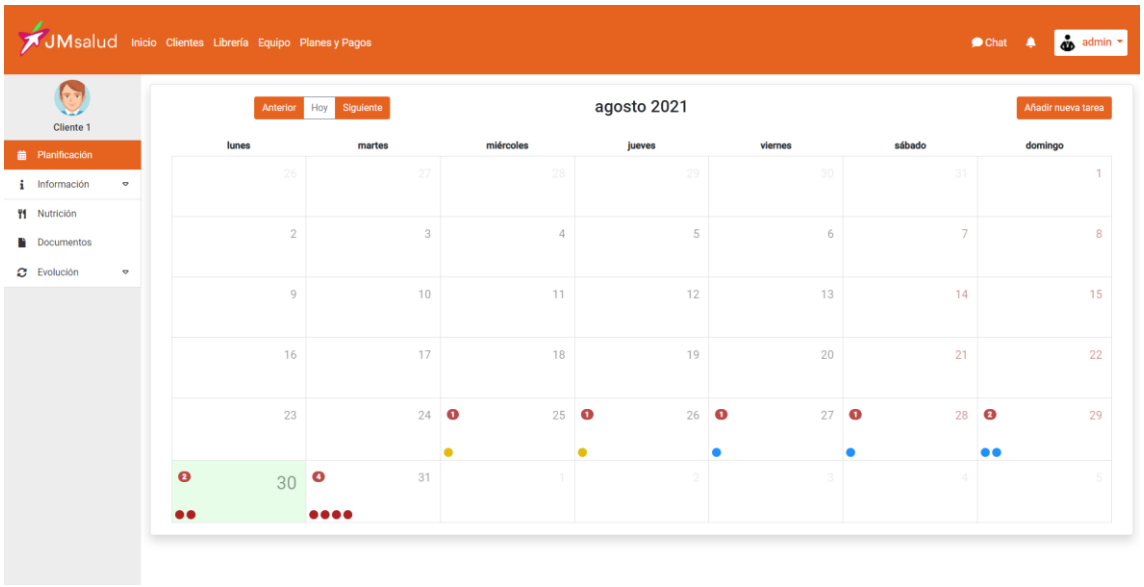

Figura 7.25. Vista específica del cliente desde un administrador.

Para añadir una nueva tarea se mostrará un modal como el de la Figura 7.26.

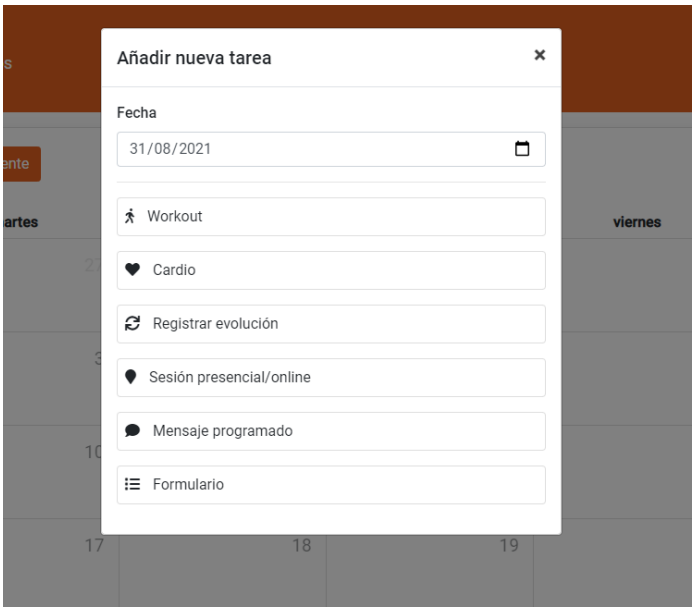

Figura 7.26. Modal para añadir una nueva tarea al cliente.

En la vista de *Información general* el administrador dispondrá de botones y opciones para asignar o eliminar entrenadores personales, enviar, ver respuestas o eliminar formularios asignados y modificar datos personales del cliente como los que se ven en la Figura 7.27.

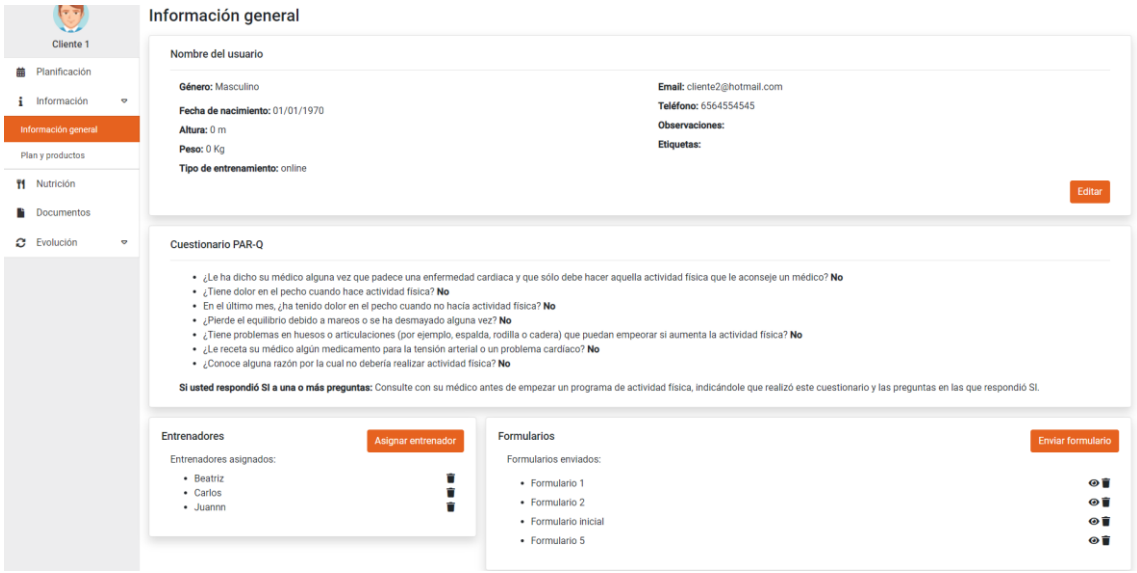

Figura 7.27. Vista de *Información general* del cliente desde un administrador.

En la vista de *Nutrición* que se puede ver en la Figura 7.28 el administrador podrá agregar o compartir planes y recetas, modificar la visibilidad de los planes o eliminar los planes y las recetas.

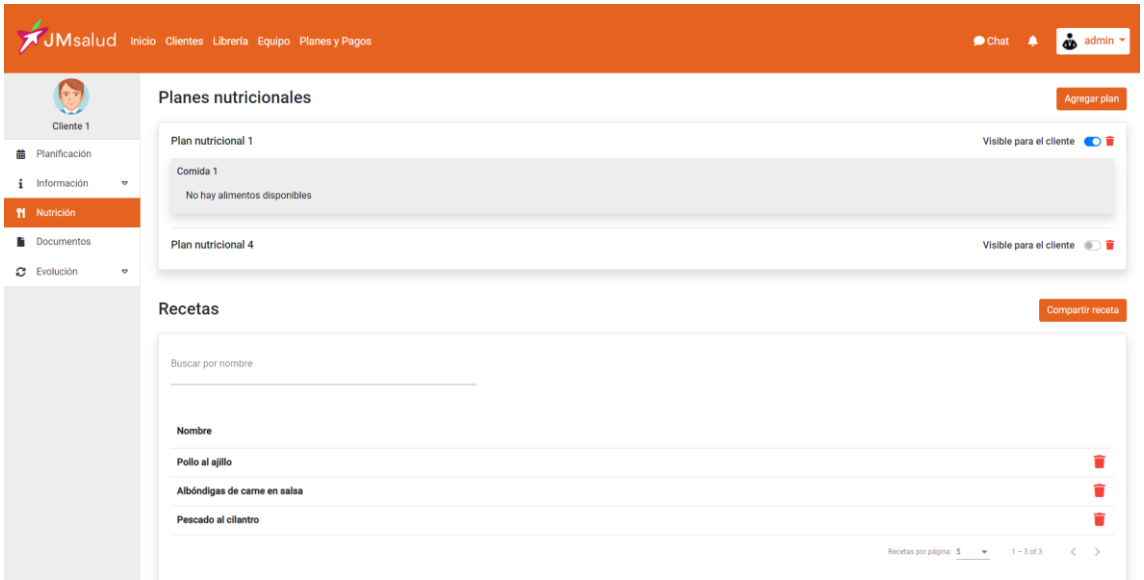

Figura 7.28. Vista de *Nutrición* del cliente desde un administrador.

En la vista de *Documentos* que se puede ver en la Figura 7.29, el administrador podrá enviar nuevos documentos al cliente o eliminar documentos ya compartidos.

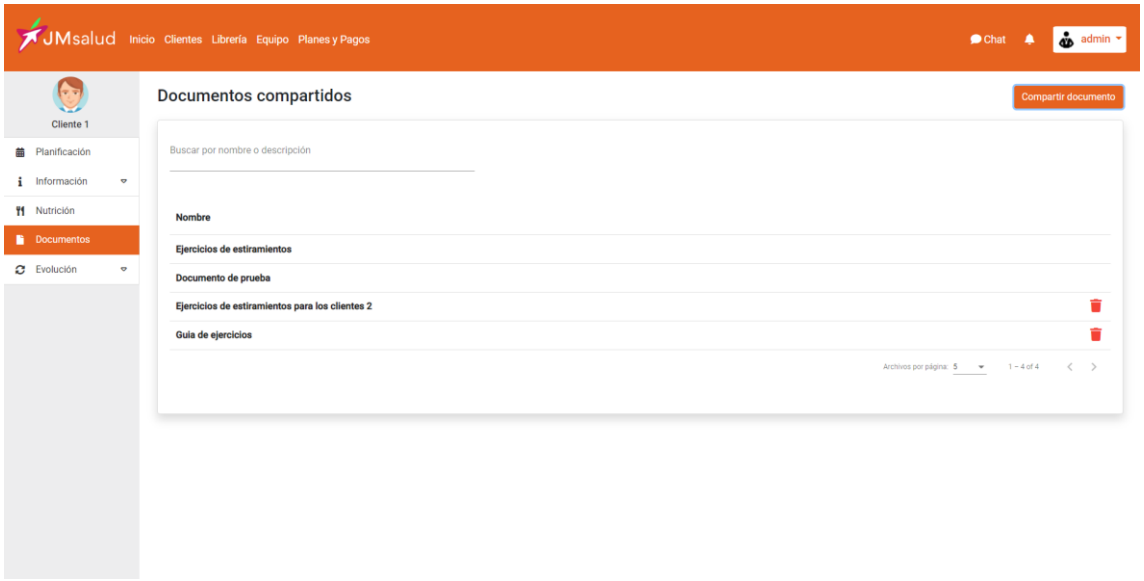

Figura 7.29. Vista de *Documentos* del cliente desde un administrador.

Las vistas de *Evolución* son exactamente iguales a las mostradas en el apartado de clientes.

Los usuarios con el rol de *Colaborador* dispondrán de las mismas vistas que un administrador quitando las vistas de *Librería*, *Equipo* y *Planes y Pagos*. En la vista de clientes se mostrarán los clientes asignados a ese entrenador personal.

7.4. Cliente

En la Figura 7.30 se muestra la pantalla principal de un usuario con el rol de cliente al iniciar sesión. En ella se puede ver un calendario en el que se muestran las tareas asignadas y no finalizadas que hay que completar.

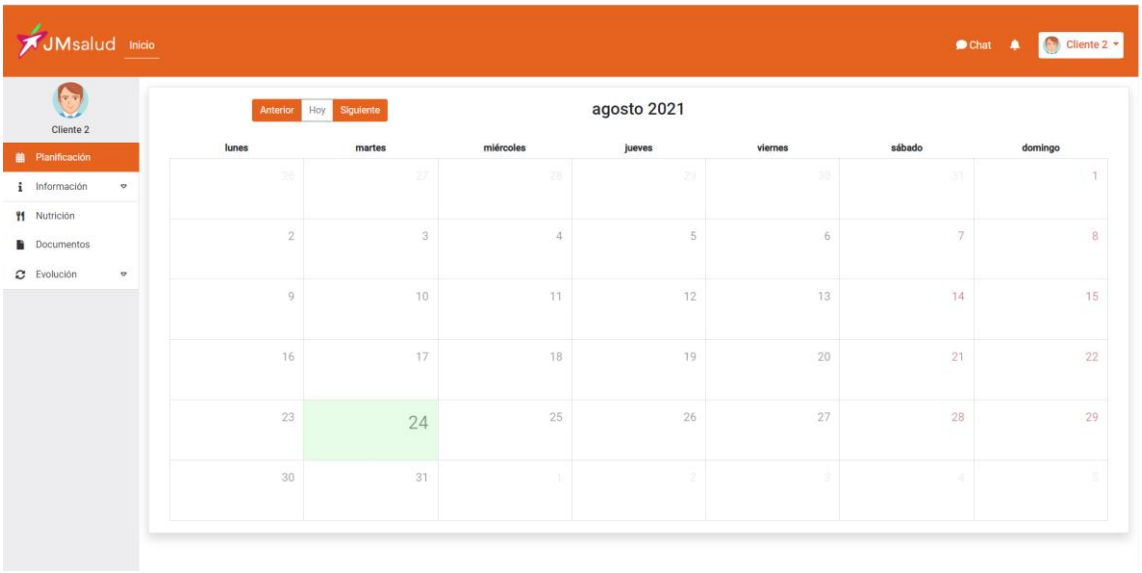

Figura 7.30. Pantalla principal de un cliente.

En la parte izquierda se muestra un menú de navegación para que el cliente se desplace entre las diferentes vistas a las que tiene acceso. En la Figura 7.31 lo podemos ver.

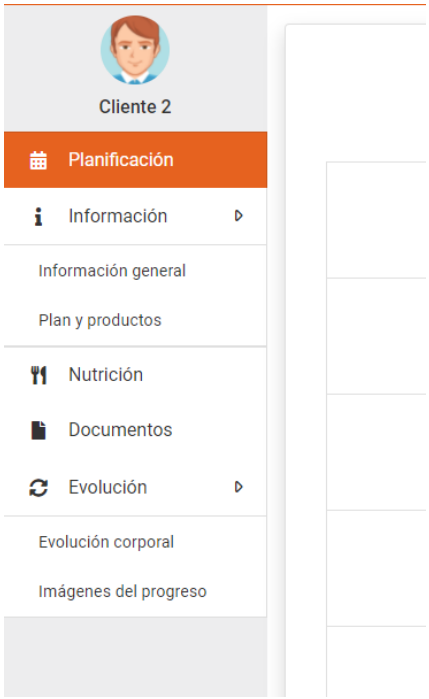

Figura 7.31. Menú de navegación del cliente.

7.4.1. Información general

En esta vista se puede ver la información general relacionada con el usuario, los entrenadores personales asignados, los formularios recibidos y el cuestionario PAR-Q como podemos ver en la Figura 7.32.

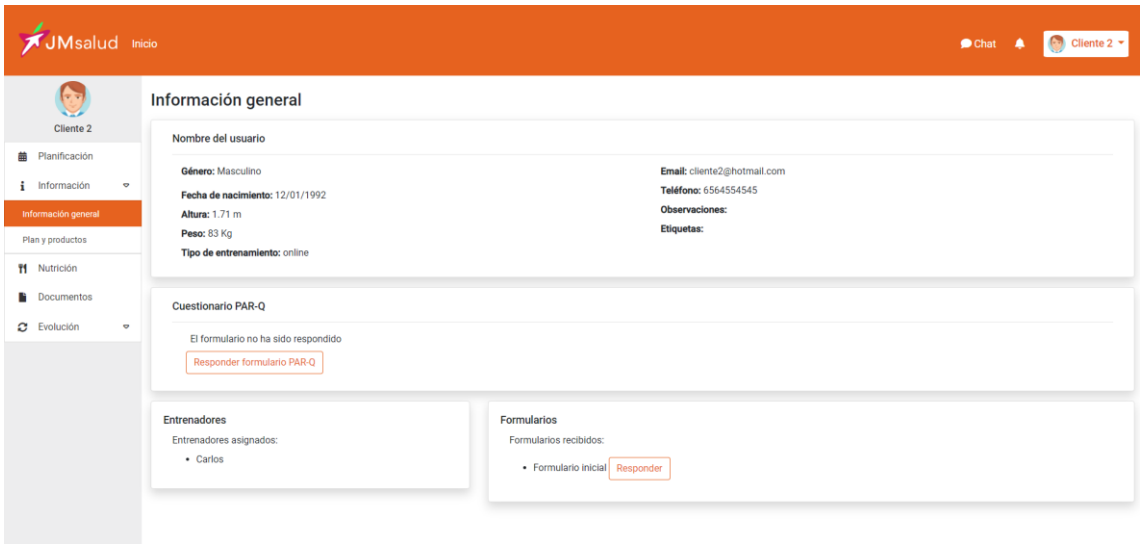

Figura 7.32. Pantalla de *Información general*.

Si el Cuestionario PAR-Q no ha sido respondido aún se mostrará un botón *Responder formulario PAR-Q* en esa sección. Si hacemos clic en el botón se mostrará el modal de la Figura 7.33.

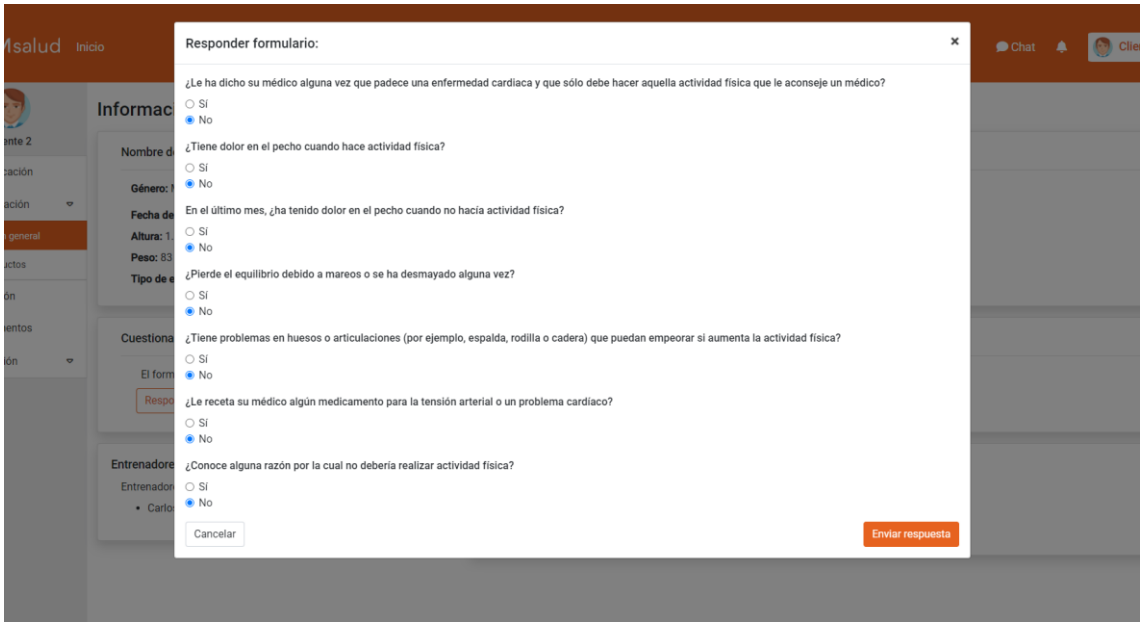

Figura 7.33. Modal responder Cuestionario PAR-Q.

Una vez seleccionadas las respuestas se hará clic en *Enviar respuesta* para guardarlas y en la sección correspondiente ya aparecerán las respuestas guardadas como se puede ver en la Figura 7.34.

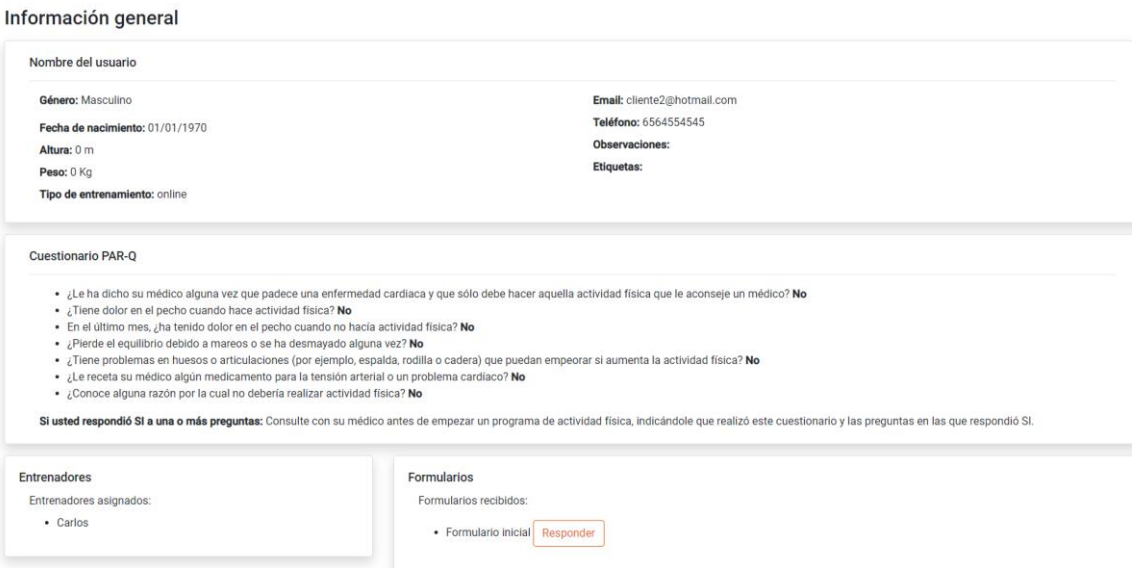

Figura 7.34. Cuestionario PAR-Q.

Para responder un formulario recibido se hará clic en el botón *Responder* del formulario de la lista mostrada y se mostrará el modal de la Figura 7.35.

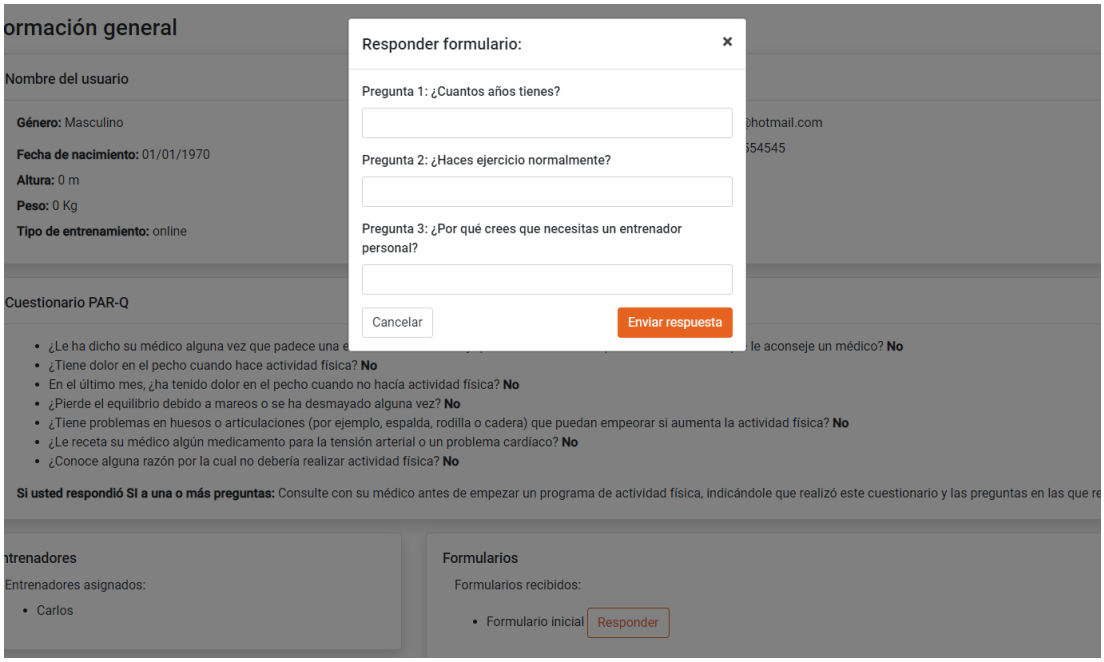

Figura 7.35. Modal para responder un formulario.

Una vez respondido se hará clic en *Enviar respuesta* para guardar los cambios.

7.4.2. Plan y productos

En esta vista se muestra el plan actual y los productos disponibles. Se encuentra dentro de las líneas de trabajo futuras.

Se podrán modificar los planes cuando se desee o configurar pagos automáticos a la aplicación para la renovación mensual de los mismos.

7.4.3. Nutrición

En esta vista se muestran los planes nutricionales asignados a cada cliente. Si no se tienen asignados planes nutricionales se mostrará la vista de la Figura 7.36.

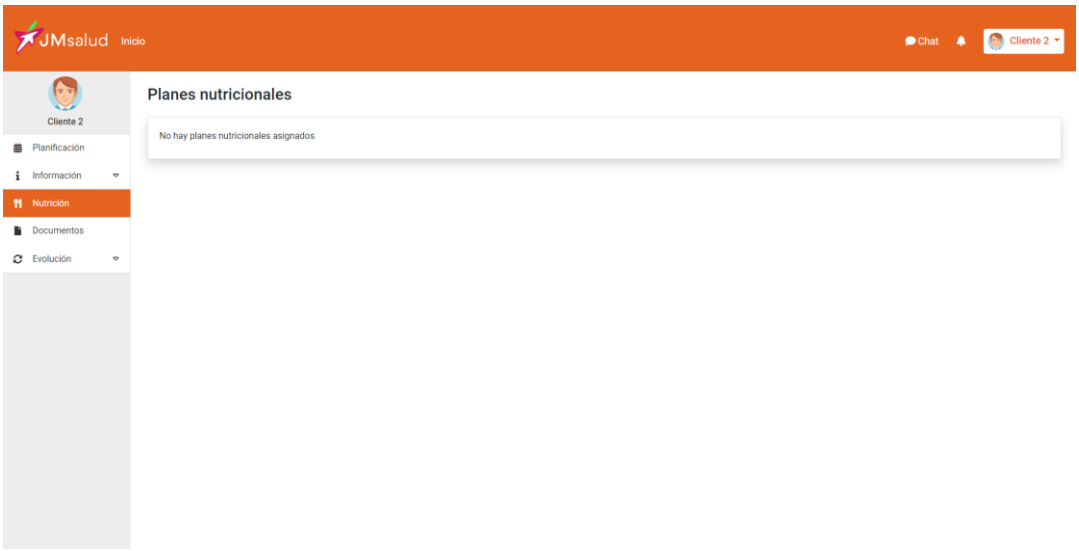

Figura 7.36. Pantalla *Planes nutricionales* sin asignar.

Si el cliente tiene un plan nutricional asignado tendrá el aspecto de la Figura 7.37.

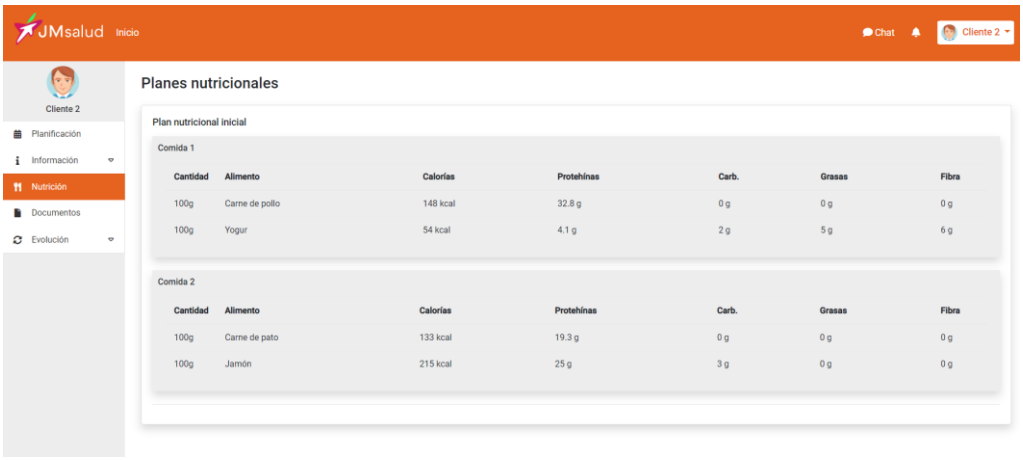

Figura 7.37. Pantalla *Planes nutricionales* asignados.

7.4.4. Documentos

En el apartado de *Documentos* se pueden ver y descargar los documentos compartidos en formato *.pdf*. En la Figura 7.38 Podemos ver la vista.

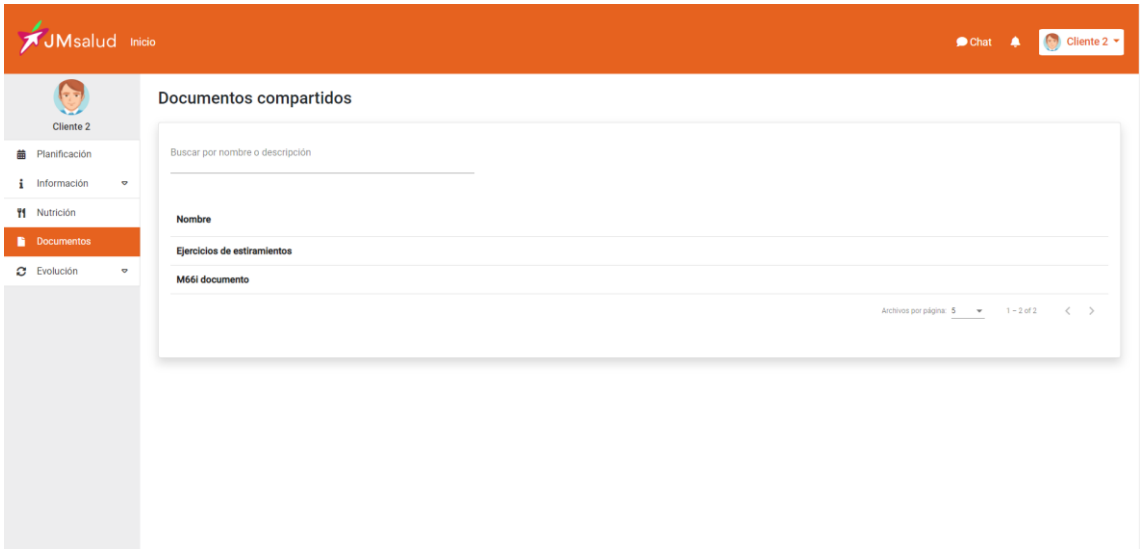

Figura 7.38. Pantalla *Documentos* del cliente.

Para ver/descargar un documento se hará clic sobre el nombre y se abrirá el documento en una pantalla nueva.

7.4.5. Evolución corporal

En esta vista se muestran los datos almacenados relacionados con la evolución corporal del cliente y se da la opción de almacenar nuevos datos. En la Figura 7.39 se muestra esta vista sin ningún dato disponible.

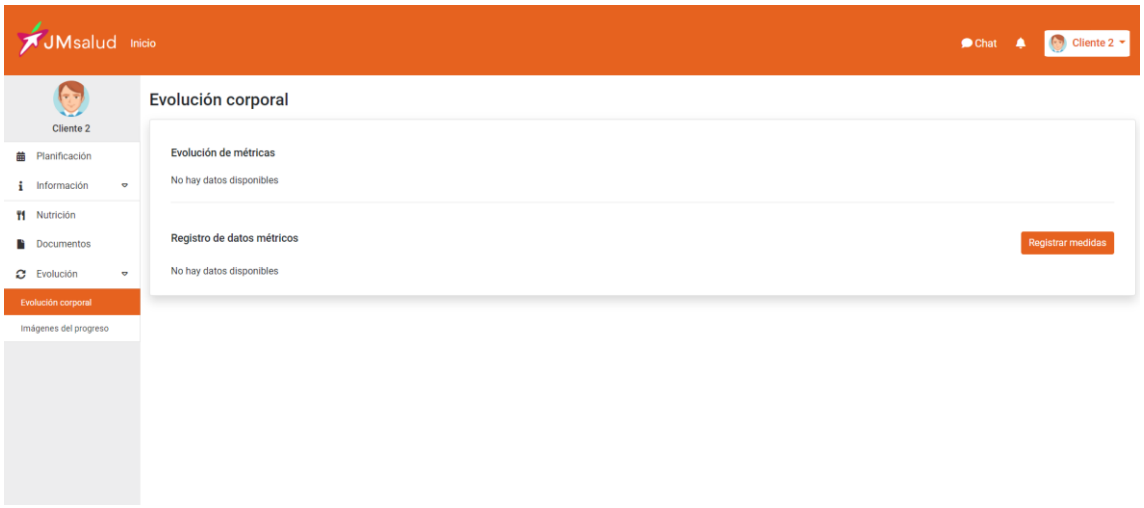

Figura 7.39. Pantalla *Evolución corporal* sin datos disponibles.

Para introducir nuevos datos el cliente hará clic en el botón *Registrar medidas*. Una vez hecho esto se abrirá el modal de la Figura 7.40.

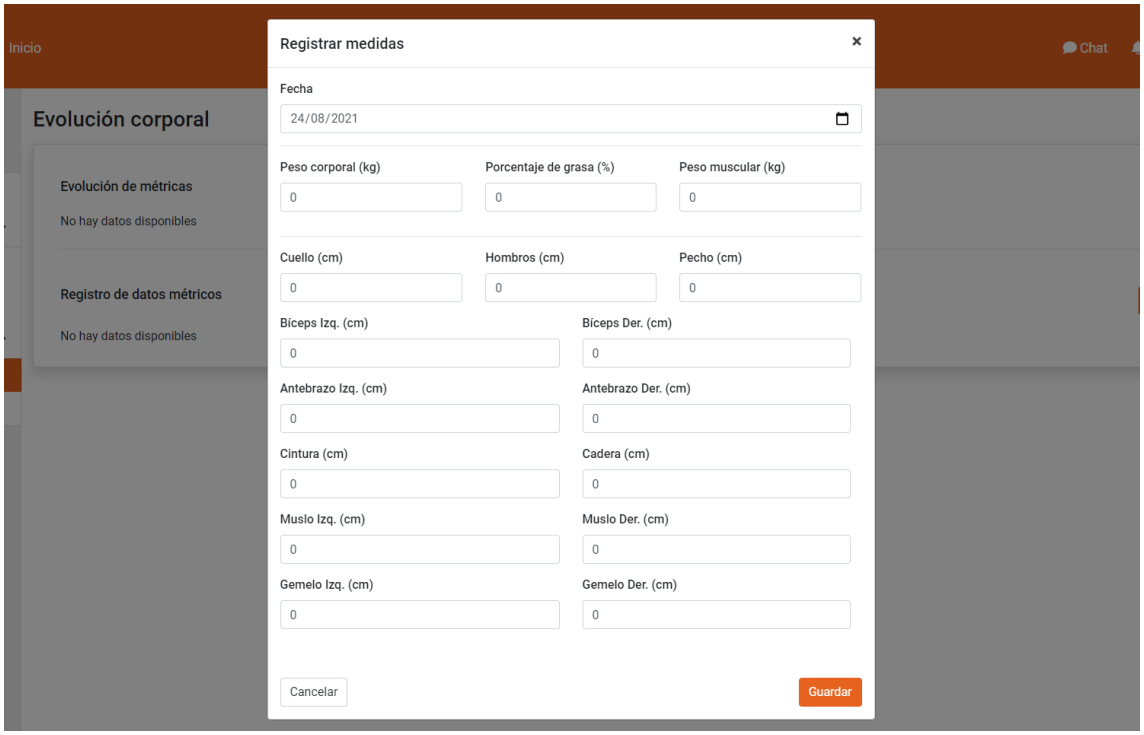

Figura 7.40. Modal *Registrar medidas.*

En este modal el cliente introducirá la fecha en la que se realizó las medidas y los valores de las medidas que desee (no es obligatorio introducir todas).

Una vez se hayan registrado varias medidas, en la parte superior se podrán ver tres tipos de gráficos: uno que muestra la evolución del peso corporal, otro la evolución del porcentaje de grasa y otro la evolución del peso muscular como se puede ver en la Figura 7.41.

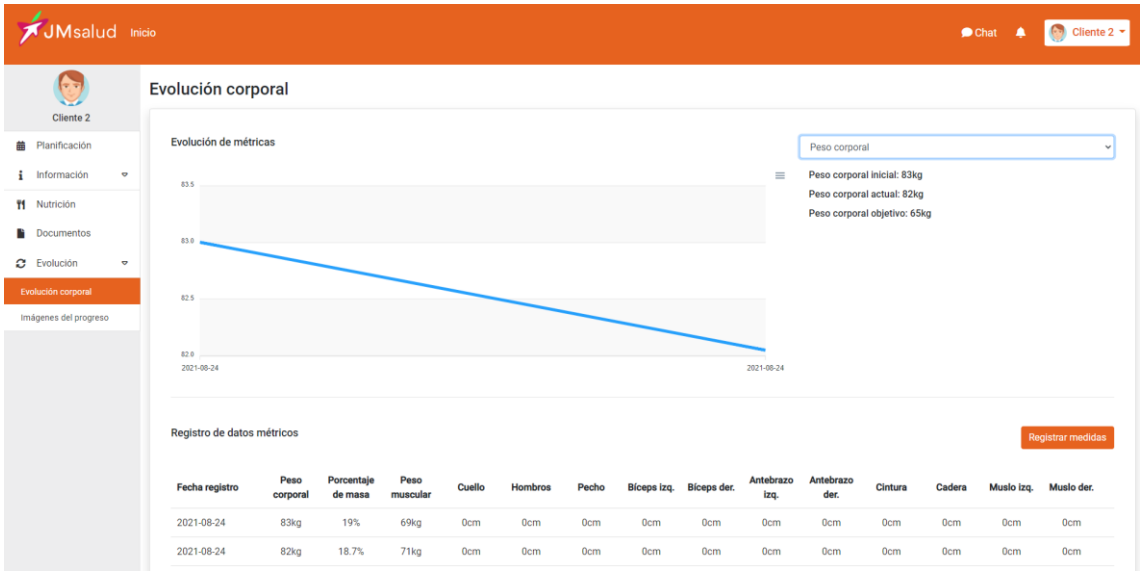

Figura 7.41. Gráficos de evolución corporal.

7.4.6. Imágenes del progreso

En este apartado el cliente podrá consultar en una tabla la evolución mediante imágenes corporales de su cuerpo. Además, podrá añadir nuevas imágenes en la fecha especificada haciendo clic en el botón *Añadir imágenes*, que abrirá un modal.

El cliente podrá subir una foto de frente, otra de espaldas y otra de perfil de su cuerpo. No será necesario subir las tres, si no la que se desee en ese momento.

En la lista mostrada el cliente podrá ver sus imágenes haciendo clic sobre el nombre.

8. Conclusiones y líneas de trabajo futuras

La aplicación web final desarrollada ha resultado ser una buena herramienta para la gestión online de los clientes de un entrenador personal. La presentación de todas las funcionalidades esenciales que pueden existir en una relación cliente-entrenador personal y la cercanía que proporciona a través de un chat personalizado hacen que la aplicación sea muy completa y ofrezca un buen servicio.

Los objetivos iniciales propuestos han sido completos, aunque la aplicación se encuentra actualmente en desarrollo para hacerla lo más completa posible. Para la realización del proyecto se han utilizado muchas tecnologías ya conocidas y otras muchas novedosas, que, gracias a un estudio profundo, se han aplicado de la forma correcta para la consecución de los objetivos.

A pesar de todas las funcionalidades ya existentes y opciones de configuración, la aplicación podría contar con muchas otras. Todas estas posibles líneas de trabajo futuras o propuestas de mejora de las funcionalidades ya existentes se resumen en la siguiente lista:

- Grupos de clientes. Los clientes se podrán agrupar en grupos para compartir contenido en común.
- Opción para descargar planes nutricionales u otra información en formato PDF.
- Compartir documentos o imágenes directamente por el chat.
- Notificaciones. Completar el sistema de notificaciones dentro de la aplicación.
- Notificaciones vía email. Desarrollar el sistema de notificaciones vía email.
- Consultar plan y productos de los clientes. Renovación con cobros automáticos o modificación del plan actual.
- Trabajo SEO y publicidad. Realizar las acciones necesarias en cuanto a posicionamiento en buscadores para conseguir que la aplicación pueda llegar a captar más clientes.
- Redimensión de imágenes subidas a la plataforma.
- Optimización de la plataforma. Reutilización de componentes, ordenación de ficheros CSS o mejora en el tratamiento de los datos.
- Otras opciones de registro. Registro mediante Facebook o cuenta de Google.
- Más opciones de pago. Pagos mediante PayPal.
- Configuración de permisos para colaboradores. Pantalla con configuración personalizada para cada colaborador.
- Análisis de cada plan nutricional. Mostrar estadísticas totales de los valores nutricionales de todos los alimentos que forman parte del plan.

9. Referencias

[1] elEconomista.es. (2021, 23 marzo). *El peligro de no hacer deporte en la pandemia: consejos de Salud Pública*. https://www.eleconomista.es/nacional/noticias/11119628/03/21/El-peligro-de-no-hacer-deporte-en-la-pandemia-consejos-de-Salud-Publica.html

[2] *Por qué elegir el gestor de base de datos MySQL*. (2018, 4 octubre). FP Online. https://fp.uoc.fje.edu/blog/por-que-elegir-el-gestor-de-base-de-datos-mysql/

[3] N. (2021, julio 12). *¿Qué es una SPA? (Angular))*. Programa en Línea. https://www.programaenlinea.net/una-spa-angular/#:%7E:text=%C2%BFTe%20gustar%C3%ADa%20aprender%20Angular%3F,entrada%2C%20generalmente%20el%20archivo%20index.

[4] *API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos*. BBVA API_Market. https://www.bbvaapimarket.com/es/mundo-api/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos/

[5] Beckwith. Justin. (2021). *Acerca de NodeJS*. Node.js. https://nodejs.org/es/about/

[6] *¿Qué es AngularJS y por qué deberías usarlo?* (2021, 15 abril). IfgeekthenEveris. https://ifgeekthen.everis.com/es/que-es-angularjs-y-por-que-deberias-usarlo

[7] *HTML: Lenguaje de etiquetas de hipertexto | MDN*. (2021, 8 agosto). https://developer.mozilla.org/es/docs/Web/HTML

[8] B., G. (2021, 9 agosto). *¿Qué es HTML? Explicación de los fundamentos del Lenguaje de marcado de hipertexto*. Tutoriales Hostinger. https://www.hostinger.mx/tutoriales/que-es-html

[9] Manz. *¿Qué es CSS? - CSS en español*. https://lenguajecss.com/css/introduccion/que-es-css/

[10] Microsoft. (2016, 14 abril). *Visual Studio Code Frequently Asked Questions*. https://code.visualstudio.com/docs/supporting/FAQ

[11] *Las 15 mejores extensiones de Visual Studio Code (VSCode 2021) para el desarrollo web*. (2021, 12 junio). Rafael Neto. https://rneto.es/blog/15-mejores-extensiones-visual-studio-code-desarrollo-web/

[12] *Getting started with GitHub Desktop - GitHub Docs*. (2021). GitHub Desktop. https://docs.github.com/en/desktop/installing-and-configuring-github-desktop/overview/getting-started-with-github-desktop

[13] *npm | npm Docs*. (2021). NPM. https://docs.npmjs.com/cli/v6/commands/npm/

[14] Robledano, A. (2020, 3 julio). *Qué es MySQL: Características y ventajas*. OpenWebinars.net. https://openwebinars.net/blog/que-es-mysql/

[15] Jonathan Herrera, info@easyappcode.com. (2021). *¿Qué es Stripe? Como funciona, ventajas e inconvenientes*. Easy App CODE. https://www.easyappcode.com/que-es-stripe-como-funciona-ventajas-e-inconvenientes