Memoria Proyecto Integrado

FitVerum

Icono

Descripción generada automáticamente

Pedro Espigares Asenjo – 2º DAW Bilingüe

# Abstract

## Español

FitVerum es una plataforma de gestión de rutinas y citas que permite a los clientes llevar un seguimiento de las rutinas y ejercicios asignados por un entrenador personal mediante las entradas que este crea dentro de su perfil. Los entrenadores pueden llevar la gestión de citas / entrenamientos con sus clientes, junto a la gestión de las rutinas, ejercicios y dietas de estos últimos.

Todo esto será gestionado por un usuario administrador que será el encargado de asignar a los entrenadores de la página y de crear / modificar / borrar las máquinas que se pueden utilizar para los ejercicios puedan crear los entrenadores.

## Inglés

FitVerum is a routine and appointment management platform that allows clients to keep track of the routines and exercises assigned by a personal trainer through the entries they create within their profile. Trainers can manage appointments/workouts with their clients, along with the management of their clients' routines, exercises and diets.

All this will be managed by an administrator user who will oversee assigning trainers to the page and creating / modifying / deleting the machines that can be used for the exercises that the trainers can create.

# Índice

[Abstract 2](#_Toc135923289)

[Español 2](#_Toc135923290)

[Inglés 2](#_Toc135923291)

[Justificación 5](#_Toc135923293)

[Características generales 5](#_Toc135923294)

[Restricciones generales 6](#_Toc135923295)

[Prestaciones de FitVerum frente a otras del mercado 6](#_Toc135923296)

[Tecnología empleada 6](#_Toc135923297)

[Tecnologías base 6](#_Toc135923298)

[Librerías 7](#_Toc135923299)

[Requerimientos 7](#_Toc135923300)

[Hardware 7](#_Toc135923301)

[Software 8](#_Toc135923302)

[Análisis y diseño 8](#_Toc135923303)

[Diagrama de casos de uso 8](#_Toc135923304)

[Diagrama de clases 9](#_Toc135923305)

[Diagrama entidad relación base de datos 11](#_Toc135923306)

[Schema de base de datos 11](#_Toc135923307)

[Implementación 12](#_Toc135923308)

[Estilos 12](#_Toc135923309)

[Plantillas y formularios 13](#_Toc135923310)

[Envío de datos 14](#_Toc135923311)

[Conexión y fichero de configuración 15](#_Toc135923312)

[Consultas de base de datos 15](#_Toc135923313)

[Documentación de código 17](#_Toc135923314)

[Evaluación y prueba 18](#_Toc135923315)

[Testeo de base de datos 18](#_Toc135923316)

[Testeo de la aplicación web 18](#_Toc135923317)

[Manual de estilos 23](#_Toc135923318)

[Wireframes 23](#_Toc135923319)

[Criterios de accesibilidad 23](#_Toc135923320)

[Criterios de usabilidad 24](#_Toc135923321)

[Fuentes 25](#_Toc135923322)

[Tamaños principales 25](#_Toc135923323)

[Mapa de colores 26](#_Toc135923324)

[Puntos de ruptura 27](#_Toc135923325)

[Ejemplos de vistas 28](#_Toc135923326)

[Software utilizado 29](#_Toc135923327)

[Posibles mejoras 29](#_Toc135923328)

[Bibliografía 30](#_Toc135923329)

[Stack Overflow 30](#_Toc135923330)

[Usabilidad Web 30](#_Toc135923331)

[Principios de accesibilidad 30](#_Toc135923332)

[Adobe contrast color analyzer 30](#_Toc135923333)

# Justificación

## Características generales

Como ya se ha explicado previamente, FitVerum es un gestor de rutinas y citas. Esta página mezcla la del concepto de “horario” con el mundo fitness, lo cual permite a los usuarios de dicho mundo poder llevar cómodamente sus rutinas y observar la evolución que han hecho históricamente y marcarse nuevos retos además de tener un entrenador detrás supervisando dicha evolución.

## Restricciones generales

Algunas de estas restricciones están especificadas en el apartado de FAQs de la página en sí, pero estas son:

* Para poder ser entrenador, este tendrá que rellenar el formulario de contacto y ponerse en contacto con el administrador. El candidato a entrenador tendrá que enviarle la documentación necesaria para verificar su autenticidad como entrenador personal y darlo de alta como este. Toda esta comunicación se realizará de una manera externa a la aplicación.
* Para que un entrenador pueda poner a un usuario como cliente, debe haber un previo acuerdo. Esta comunicación será realizada también externamente a la aplicación. Si un entrenador pone a un usuario como cliente sin su consentimiento será revocado de su rol.
* Limitaciones propias de Firebase. Su plan gratuito cuenta con solo 5 GB de almacenamiento de imágenes y ficheros, por lo que la página no puede permitirse la subida de ficheros de mucho tamaño para no sobrepasar la limitación del plan gratuito.

## Prestaciones de FitVerum frente a otras del mercado

Tras realizar una búsqueda y consultar a personas que trabajan o pertenecen al mundo “fitness”, concluí que no existe una aplicación que combine un horario junto a la gestión de rutinas y ejercicios con posibilidad de ver su evolución. Tras dicha investigación, agrupé las peticiones de las personas en cuestión y las plasmé en la página. FitVerum es, en esencia, una combinación de funcionalidades ya existentes en otras páginas sueltas en una misma página.

# Tecnología empleada

## Tecnologías base

FitVerum está realizada con las siguientes tecnologías:

* Front End: Angular
* Back End: Firebase
* Estilos: SASS + Bootstrap

La elección de estas tecnologías es que Angular funciona muy bien y es fácil de implementar con Firebase, el cual te proporciona una base de datos en la nube, hosting gratuito y espacio de almacenamiento gratuito. Para los estilos he decidido usar SASS, ya que permite la reutilización de componentes y la creación de funciones y mixins, los cuales ahorran mucho tiempo a la hora de estilar una página. Además, sirve para poder modificar Bootstrap a tu gusto.

## Librerías

Además de estas tecnologías, se han utilizado distintas librerías, las cuales son:

* **Angular Calendar**: Componente de angular el cual muestra un calendario interactivo y con posibilidad de tener vista diaria, semanal y mensual.  
  <https://github.com/mattlewis92/angular-calendar>
* **Ngx-translate**: Librería i18n adaptada para Angular para internacionalización de la página.   
  <https://github.com/ngx-translate/core>
* **AOS**: Librería para animación al scroll.  
  <https://github.com/michalsnik/aos>
* **Ng2-charts**: Librería chartJS para gráficas adaptada para Angular.  
  <https://github.com/valor-software/ng2-charts>
* **EmailJS**: Librería para envíos de correos electrónicos,  
  <https://www.emailjs.com/>
* **SassDoc**: Sistema de documentación para SASS.  
  <http://sassdoc.com/>
* **Compodoc**: Sistema de documentación para Angular.  
  <https://compodoc.app/>
* **AngularFire**: Implementación de Firebase en Angular  
  <https://github.com/angular/angularfire>

# Requerimientos

## Hardware

Las tecnologías usadas en FitVerum no requieren un hardware realmente potente para funcionar (para desarrollo). Con un ordenador que soporte un sistema W10 es suficiente. En la página web pasa lo mismo, solo que se recomienda una buena conexión a Internet ya que la base de datos es en la nube.

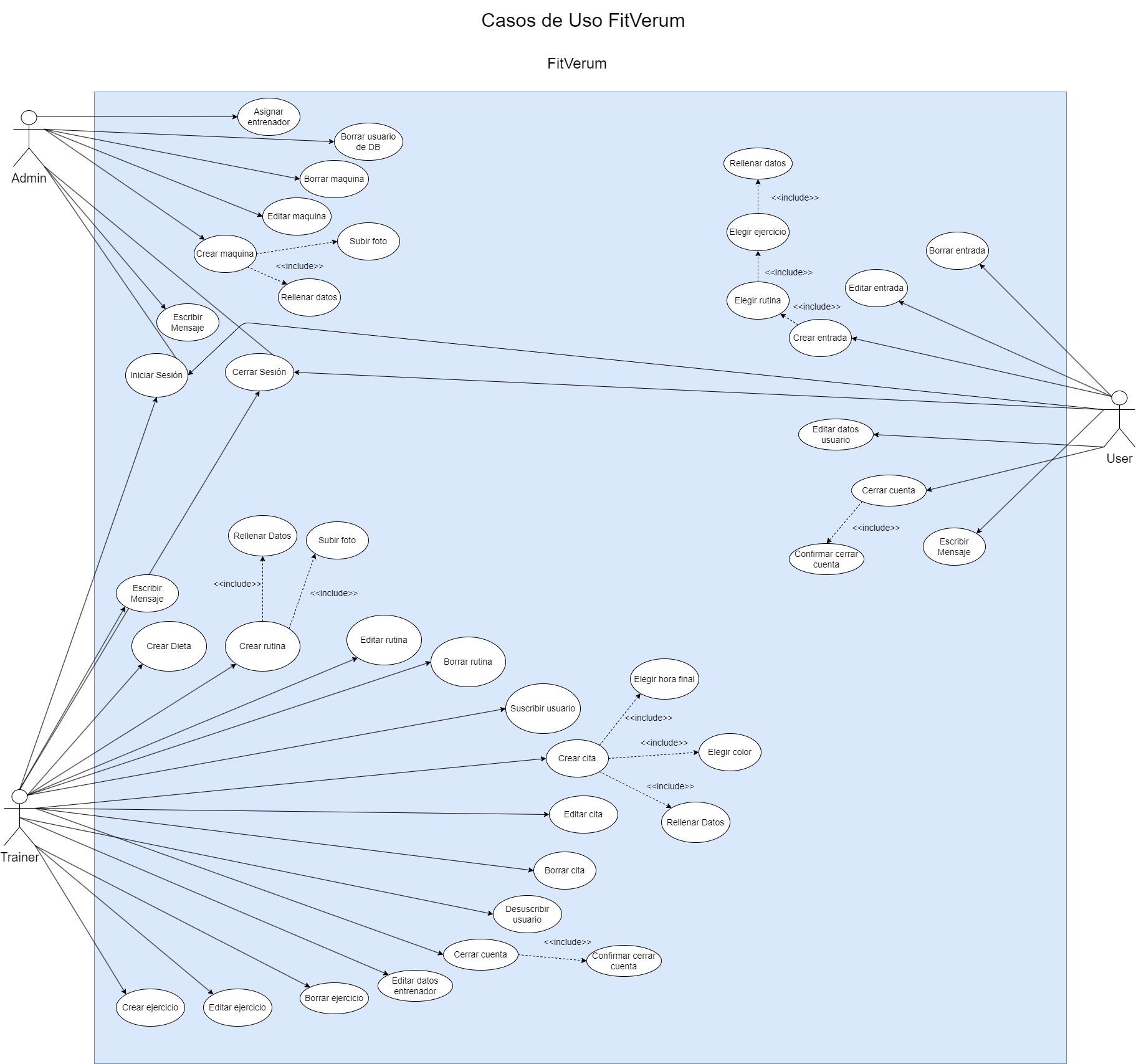
## Software

El único software que se necesita para el modo desarrollo de FitVerum es Node JS para poder utilizar su gestor de paquetes NPM e instalar las dependencias necesarias para su funcionamiento. En la página web lo único que se necesita es un navegador para poder visitarla y tener activada la ejecución de JavaScript.

# Análisis y diseño

FitVerum, al ser una página dedicada a la relación entre un entrenador personal y su cliente, las relaciones tanto de base de datos como de clases son mayoritariamente entre ambos. Aunque la base de datos de firebase es NoSQL, la he tratado como una base de datos relacional con sus primary keys y foreign keys correspondientes. Al ser NoSQL, no tiene las restricciones propias para las foreign keys, not null, autoincrement, etc. Pero la base de datos está montada para que “simule” el comportamiento de una base de datos relacional.

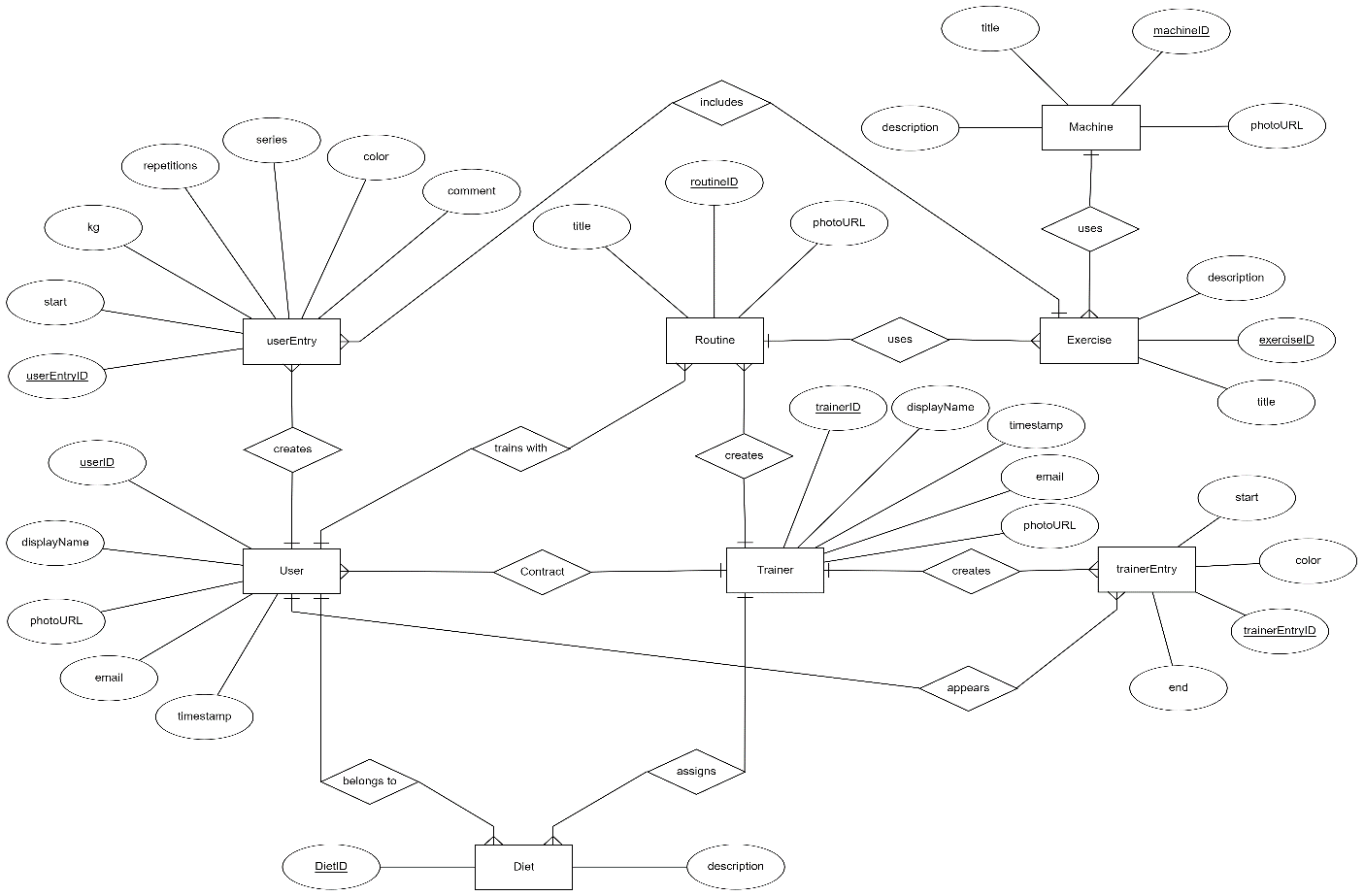
## Diagrama de casos de uso



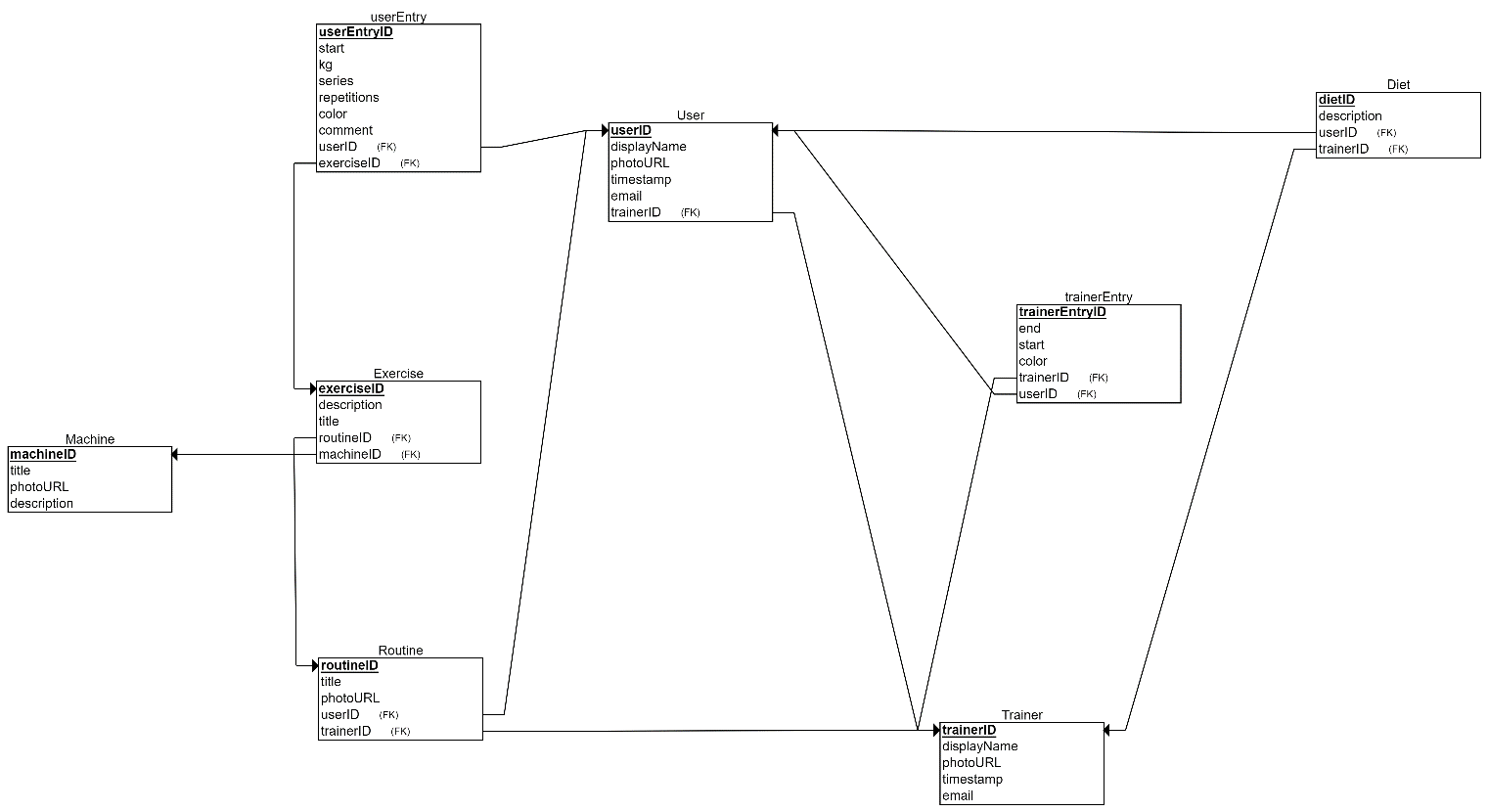
## Diagrama de clases



## Diagrama entidad relación base de datos



## Schema de base de datos



# Implementación

## Estilos

Con Angular, SASS funciona de una manera distinta a como funcionaría normalmente. Esto es porque los componentes de Angular son autocontenidos y contienen tanto su fichero Html, TypeScript y SCSS / CSS (según se configure). Por lo tanto, para poder utilizar las variables y los mixins dentro de los componentes hace falta importarlos dentro del fichero .scss del propio componente, al revés de cómo se haría normalmente.

Cada fichero de hojas de estilos contiene los estilos solo del componente al que pertenezca. Este tendrá el apartado de configuración “stylesheet” en su fichero .ts la ruta del fichero .scss de su componente. Un ejemplo de hoja de estilo con SASS es el siguiente, el cual pertenece al componente de dietas de un usuario:

Texto

Descripción generada automáticamente

## Plantillas y formularios

La página utiliza plantillas para los formularios creadas por mí. Son dos plantillas, una para los formularios de añadir rutinas, ejercicios, dietas, etc. Y otro para editarlos. La plantilla es a base de un mixin de SASS y el HTML correspondiente. En la documentación de estilos generada por SassDoc aparecen dichos mixins. Dejo el enlace a continuación:

<https://pedroespigares.github.io/FitVerum-SassDoc/>

Los formularios están validados por distintas expresiones regulares como son las siguientes:

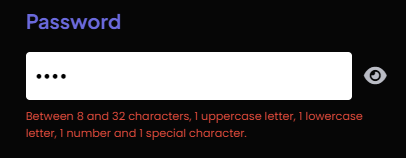
* Contraseña: Entre 8 y 32 caracteres, mínimo 1 mayúscula, 1 minúscula, 1 número y 1 carácter especial:

^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[0-9])(?=.\*[!@#$%^&\*\_=+-]).{8,32}$

* Email: Formato de correo electrónico validado por el tipo email de Html y esta expresión regular:

/^[a-zA-Z0-9.!#$%&'\*+/=?^\_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)\*$/

Además, el incumplimiento de estas expresiones se verá reflejado en la página con un mensaje de error. Ejemplo de incumplimiento de la expresión regular de la contraseña:



Los formularios que contengan subida de ficheros están limitados a que sean exclusivamente imágenes y como máximo 3 MB de tamaño para no colapsar el espacio de almacenamiento de Firebase con fotos que ocupen mucho espacio. Como pasa con las expresiones regulares, el incumplimiento de estas especificaciones se verá reflejado en la página con un mensaje de error (igual que en la anterior, en la cual se pondrá en un color rojo el mensaje con el formato de la imagen). La condición de esta restricción es el siguiente:

Texto

Descripción generada automáticamente

## Envío de datos

Texto

Descripción generada automáticamenteAl haber realizado la aplicación con Firebase como servicio de backend, los métodos de envío de datos ya vienen predefinidos con su servicio de autentificación. Un ejemplo es el siguiente, utilizado para el registro de usuarios con el método convencional de email y contraseña:

## Conexión y fichero de configuración

Al utilizar la librería de Angular Fire para la conexión y configuración entre mi aplicación y Firebase, dicha conexión se crea automáticamente en el fichero environments.ts del proyecto:

Texto

Descripción generada automáticamente

## Consultas de base de datos

Nuevamente, al estar hecho el proyecto con Firebase, he utilizado los métodos predefinidos de este para las consultas a mi base de datos, como pueden ser collecionData, getDoc y la cláusula where con el método query. Fundamentalmente he utilizado 4 tipos de consultas que me devuelven lo siguiente (con ejemplos de la propia aplicación):

1. Texto

   Descripción generada automáticamenteConsulta que devuelve una tabla / colección entera:
2. Consulta que devuelve un item de la colección:



1. Consulta que devuelve un atributo específico de un item de la colección:

Texto

Descripción generada automáticamente

1. Consulta con cláusula where

Texto

Descripción generada automáticamente

# Documentación de código

Como he especificado previamente, para la documentación del código he utilizado la librería [Compodoc](https://compodoc.app/) que permite la generación de documentos HTML que toman en cuenta todos los comentarios escritos a lo largo del código. Además, al estar adaptada y preparada para Angular, también toma en cuenta el fichero de rutas, guards, servicios, etc. Al ser ficheros HTML, he hecho como con la documentación generada por SassDoc y la he subido a producción en GitHub Pages para facilitar su consulta. Dejo el enlace a continuación:

<https://pedroespigares.github.io/FitVerum-CompoDoc/>

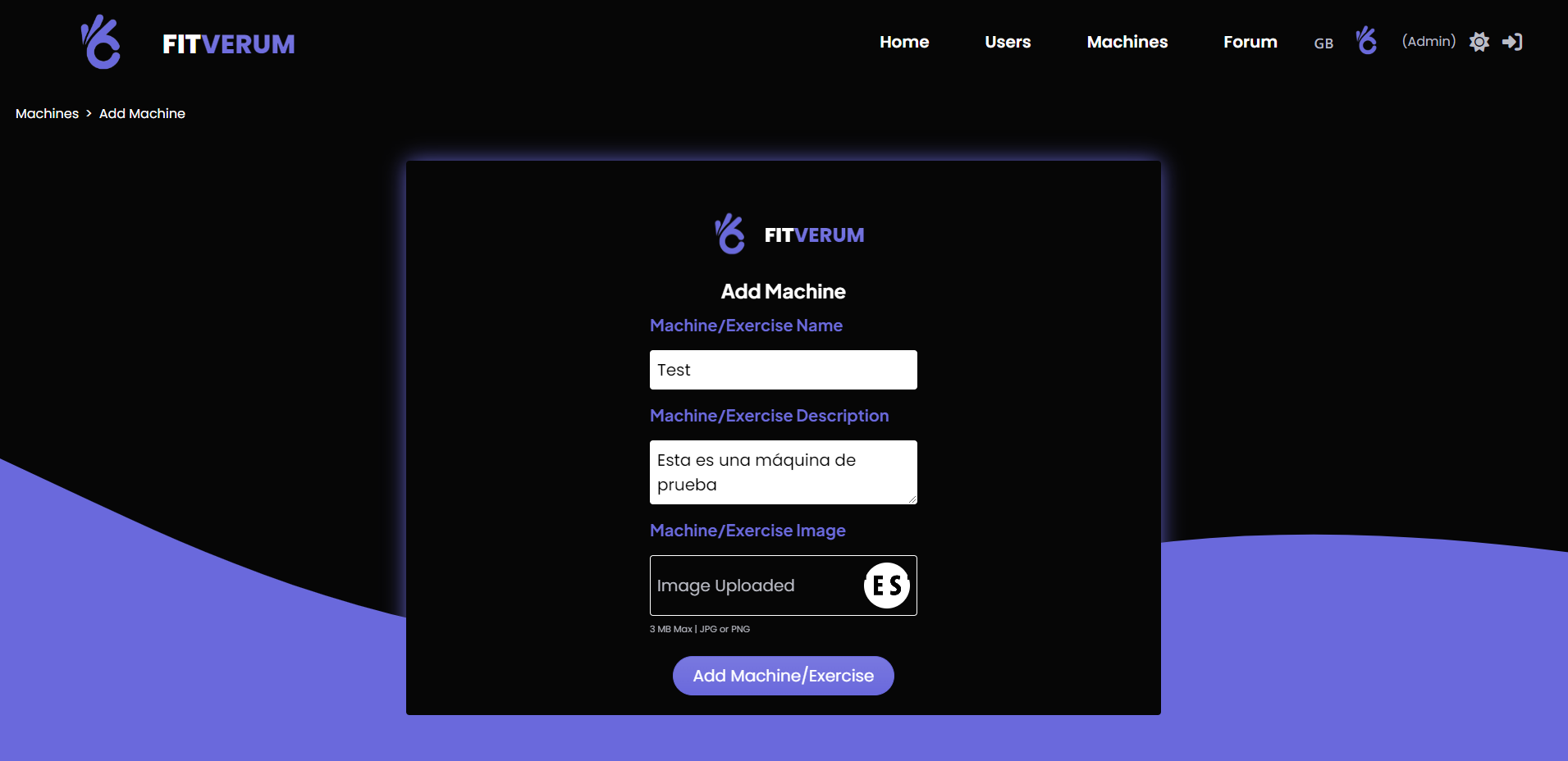
# Evaluación y prueba

## Testeo de base de datos

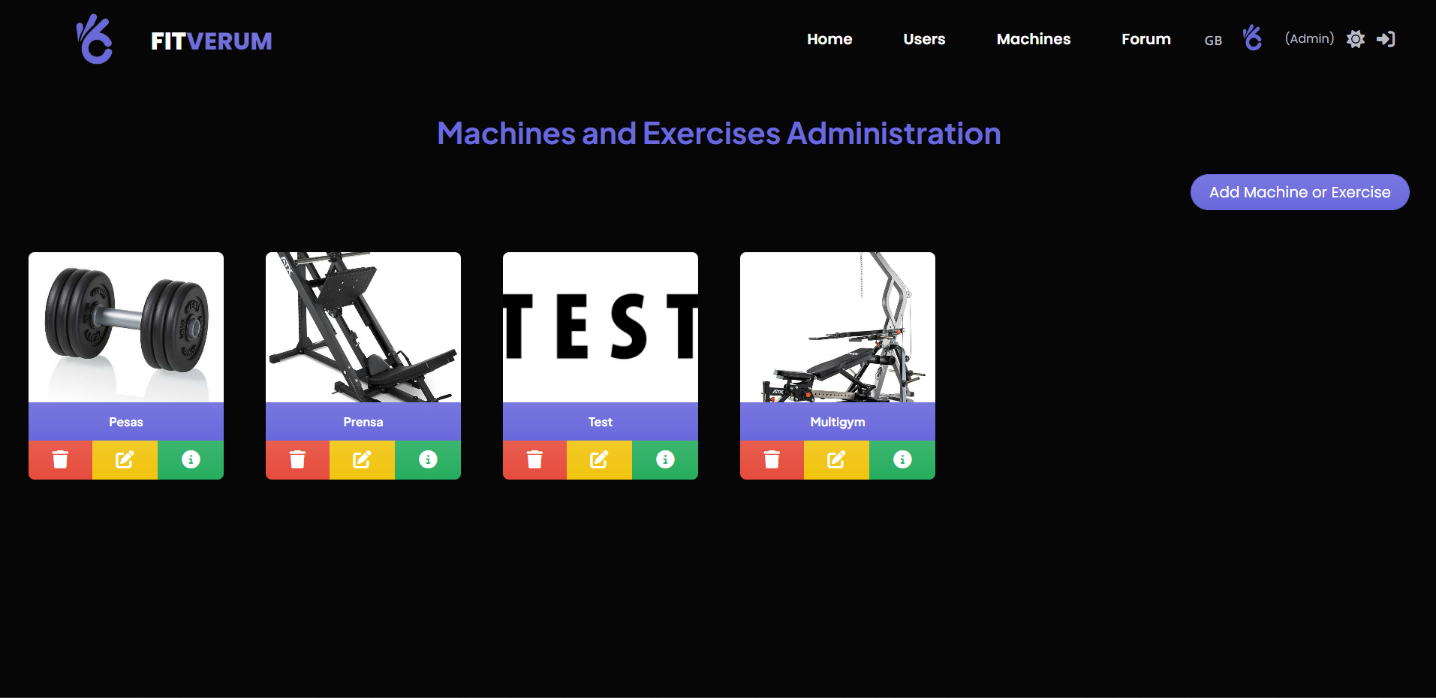
Unas de las desventajas que tiene la base de datos de Firebase es que, al ser NoSQL, tiene más permisividad a la hora de validar los datos introducidos. Como no tiene restricción de tipo de datos, ni de valores nulos y tampoco de claves foráneas, todo tiene que ser validado en el front antes de poder escribirse en la base de datos, por lo que su testeo ha sido un poco a base de prueba y error.

## Testeo de la aplicación web

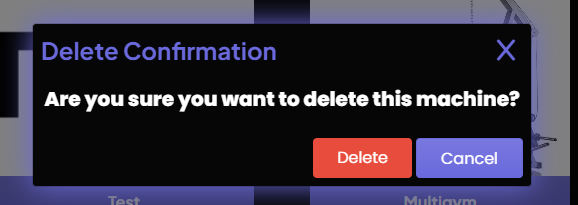
A la hora de testear la aplicación web, al ser un gestor de citas y de rutinas, su testeo obvio es la de crear rutinas y citas a mano. Con ayuda externa a la hora del desarrollo, he ido descubriendo casos de uso que he utilizado para mejorar / arreglar la aplicación. Los formularios de añadir rutinas, dietas y máquinas (y los de editar) funcionan exactamente de la misma manera, cada uno manejando sus datos. Por ejemplo, el añadir una rutina también estará relacionada con un cliente que se ha elegido previamente. Dejo aquí una prueba de añadir una máquina / ejercicio:

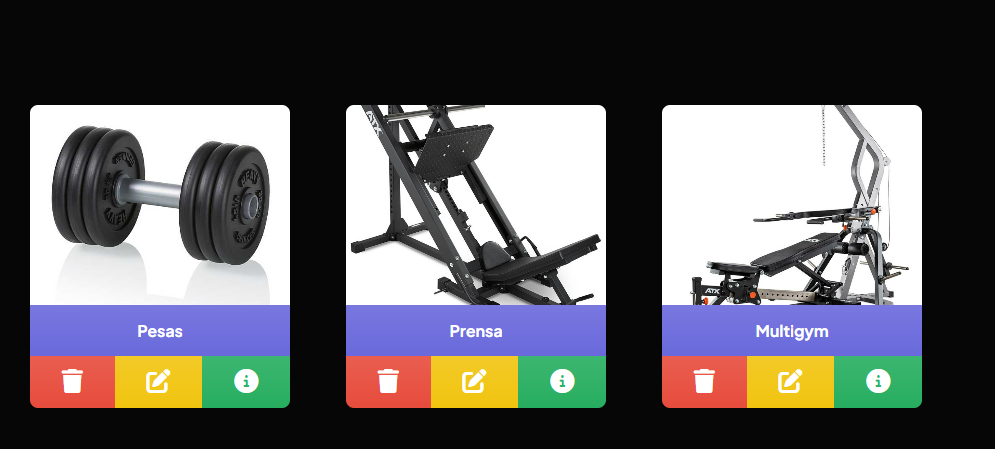


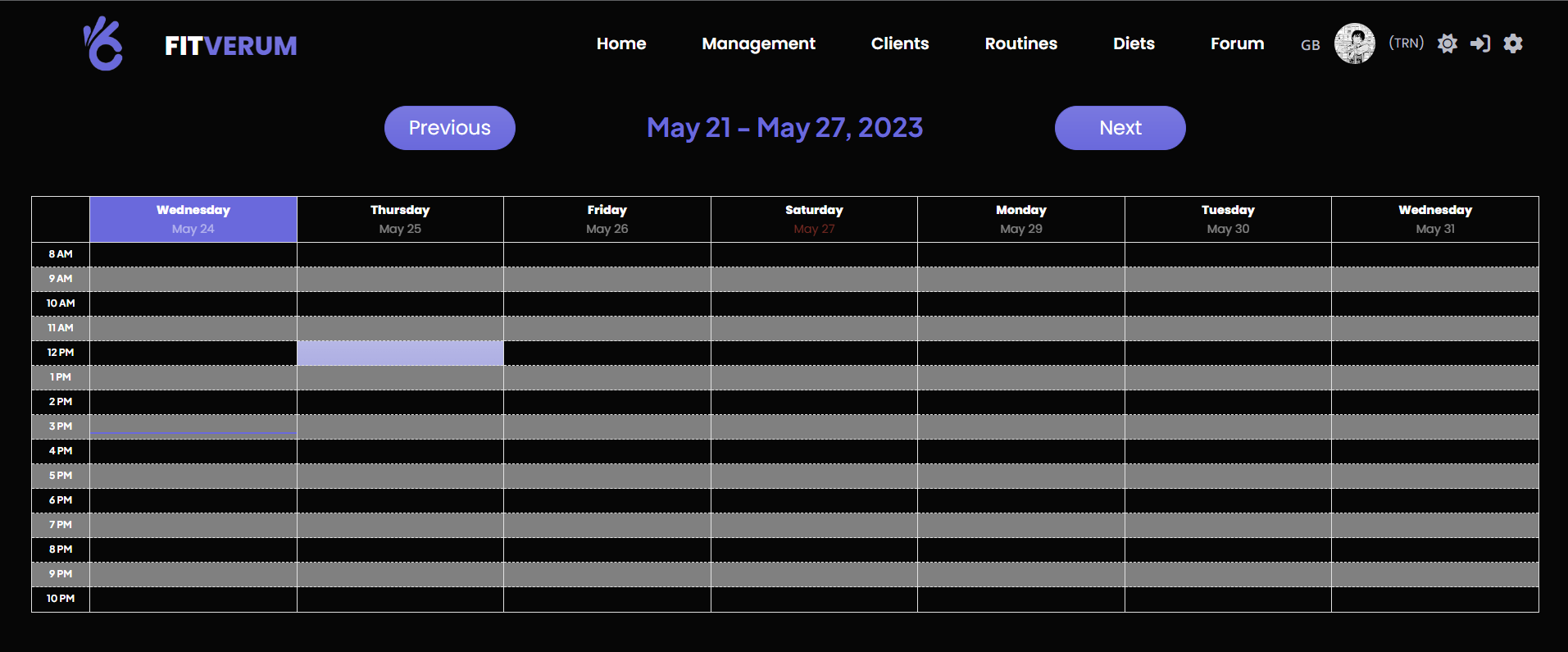
El botón de añadir máquina o ejercicio no será clickeable hasta que todos los campos hayan sido rellenados y no sean solo espacios en blanco. Una vez añadida, nos llevará a la página de consulta de las máquinas, donde ya aparecerá la máquina añadida.



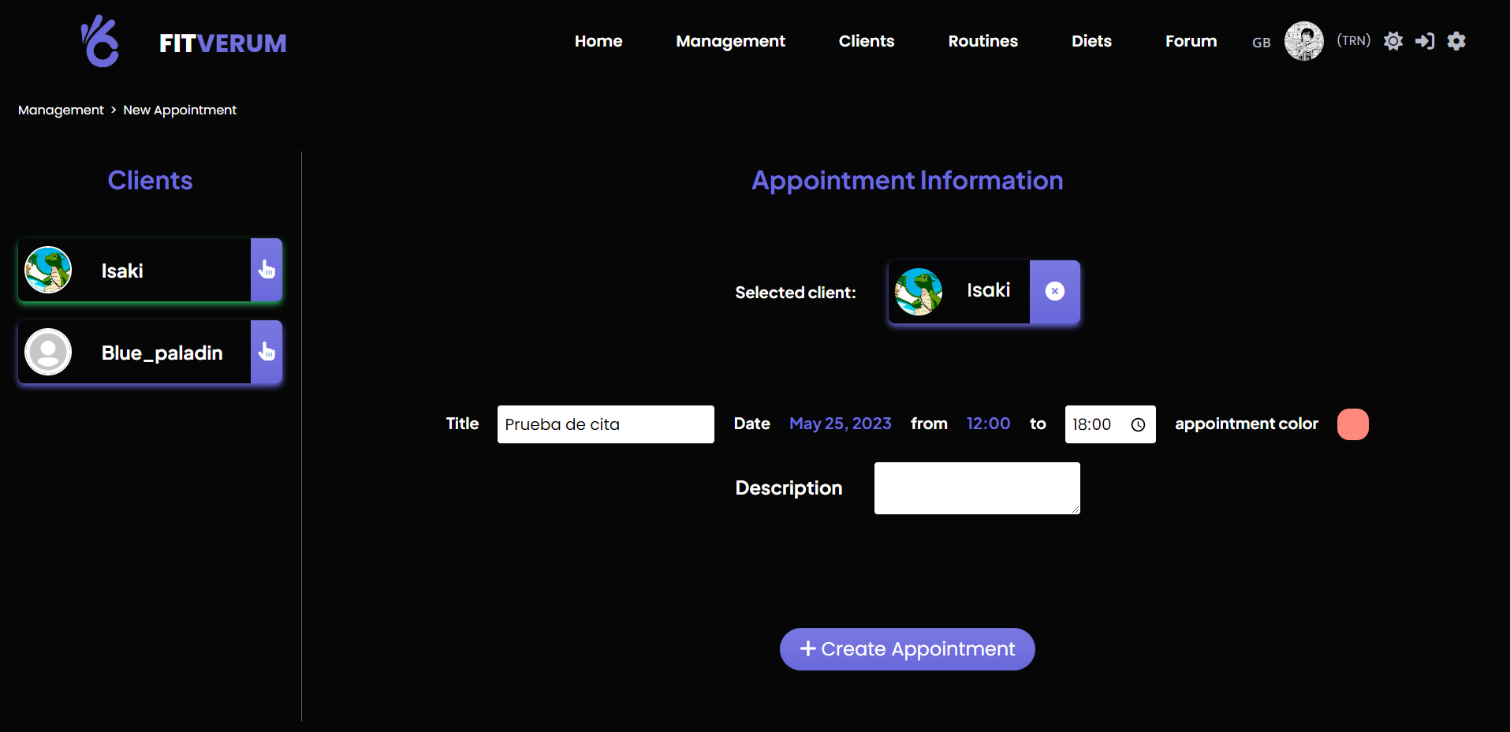
Dicha máquina podrá ser borrada con su botón correspondiente, donde aparecerá un modal para confirmar dicha decisión



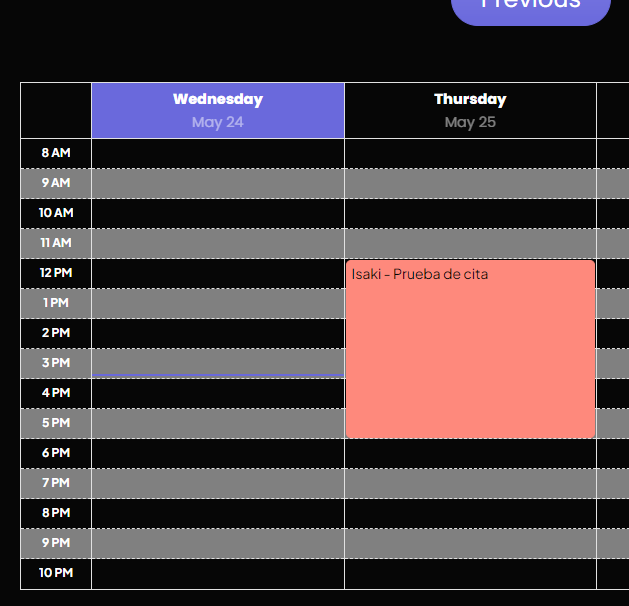
Una vez confirmado el borrado, podemos observar que dicha máquina ya no aparece en la lista. Dicho borrado también elimina la foto subida del almacenamiento.

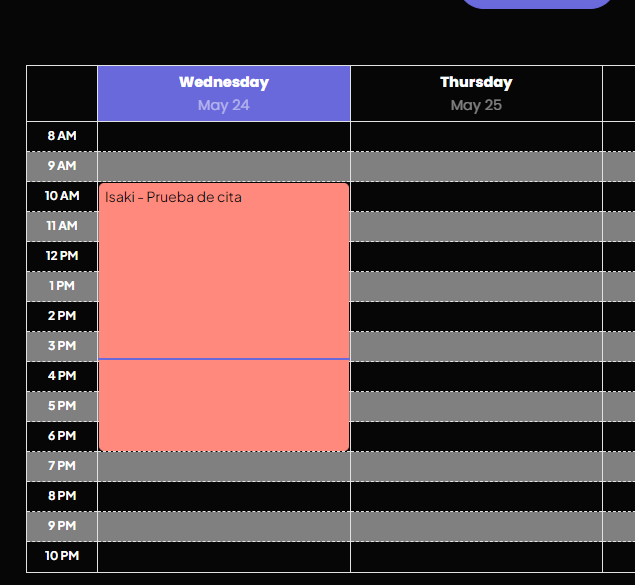
El entrenador tiene un calendario donde podrá elegir día y hora para asignar una cita:

Una vez elegido una hora, te saldrá la vista donde podrás elegir al cliente al que quieres reservar la cita junto a la información de esta. Hasta que no rellenes todos los campos menos el de descripción (que es opcional), no podrás guardar la cita. El cliente elegido aparecerá con un box shadow verde. La hora final de la cita tiene que ser mayor a la hora elegida y menor a las 22:00. Si no se cumplen estas condiciones, saltará un error en la página.



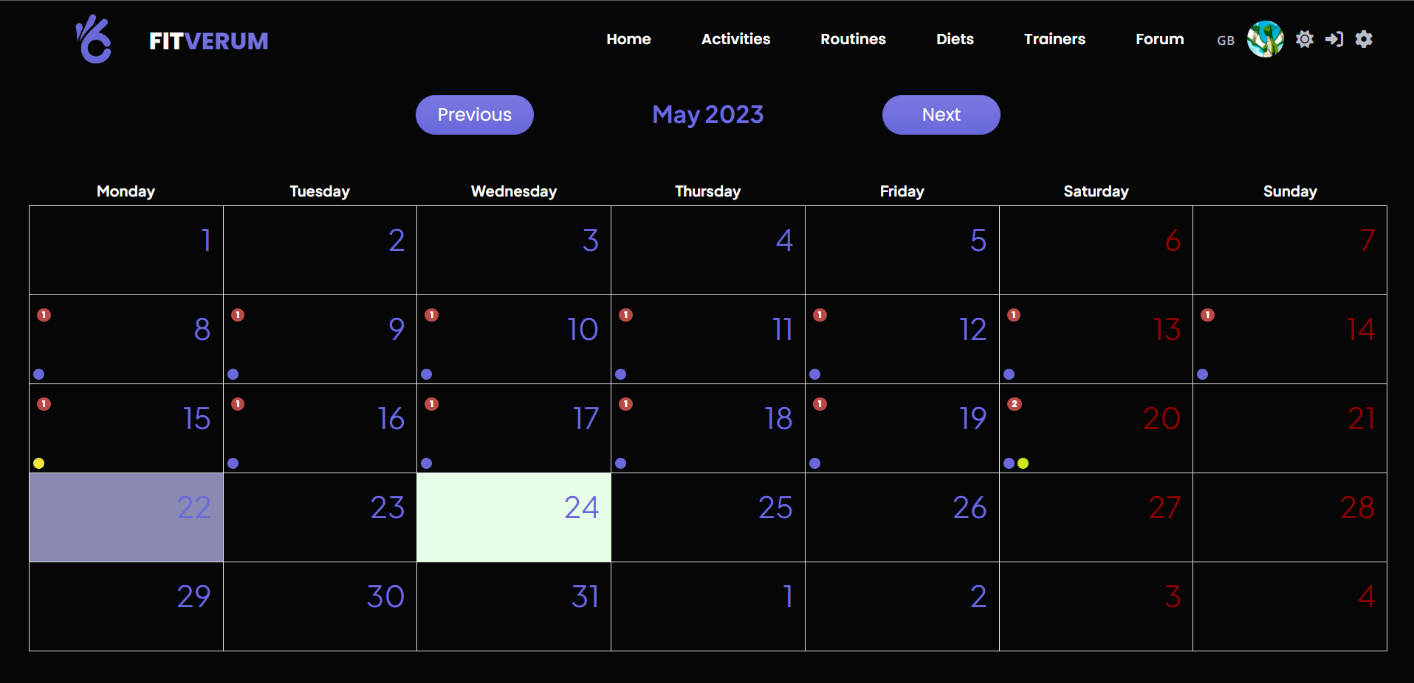
Una vez creada la cita, aparecerá la cita en el calendario desde la hora clickada hasta la hora elegida en el formulario. Cuando clickas en la cita, aparecerá un modal con la información de esta junto a la opción de borrarla y volver al calendario. Para editar la hora y día de la cita se puede arrastrar y soltar en la zona que quieras de calendario. También se puede ampliar las horas que ocupa agrandándola desde los extremos.



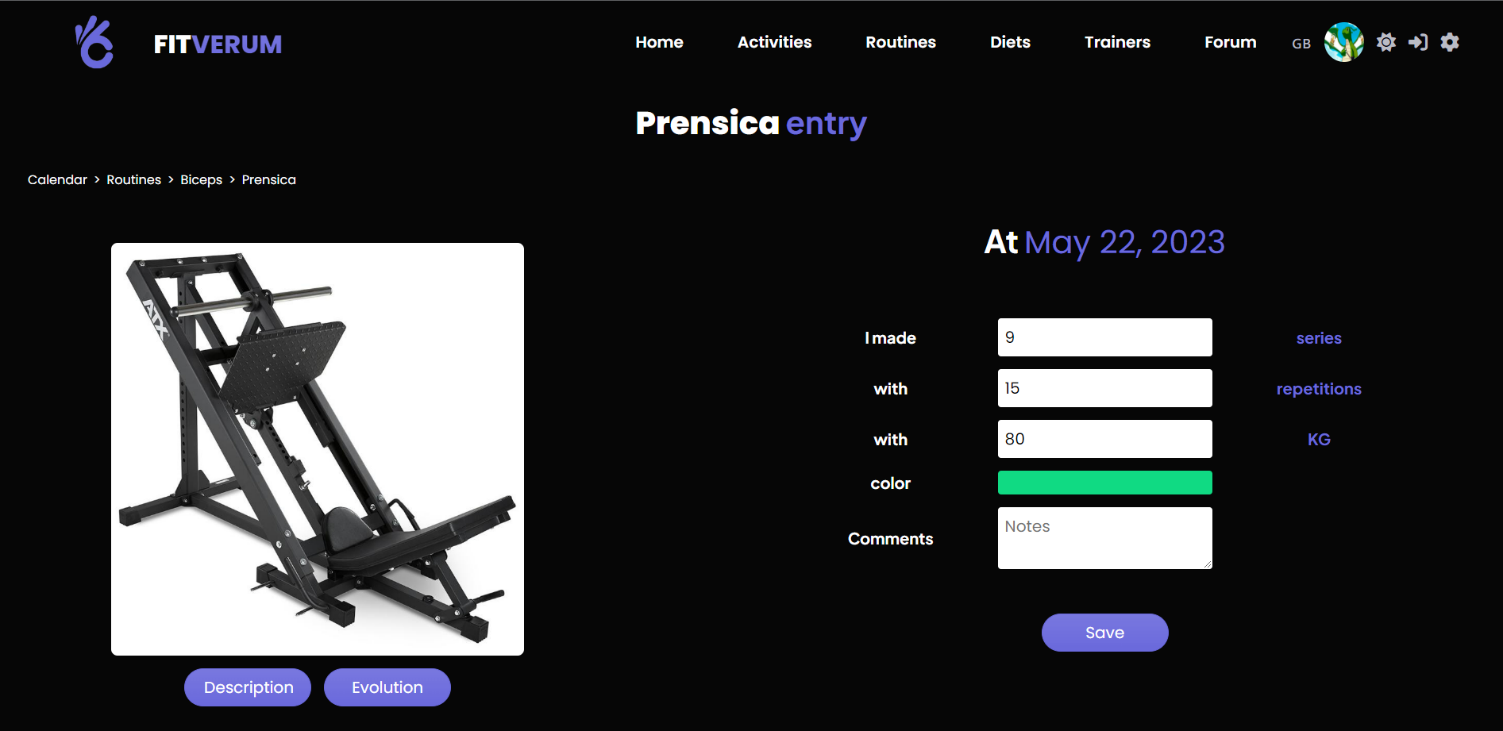


En la zona del usuario es parecida, ya que elegirá el día donde quiere introducir una entrada clickando al dicho día en el calendario. Los puntos en los días son entradas introducidas por el usuario, mientras que el número con fondo rojo en la esquina superior izquierda son el número de entradas que tienen en el mismo día.

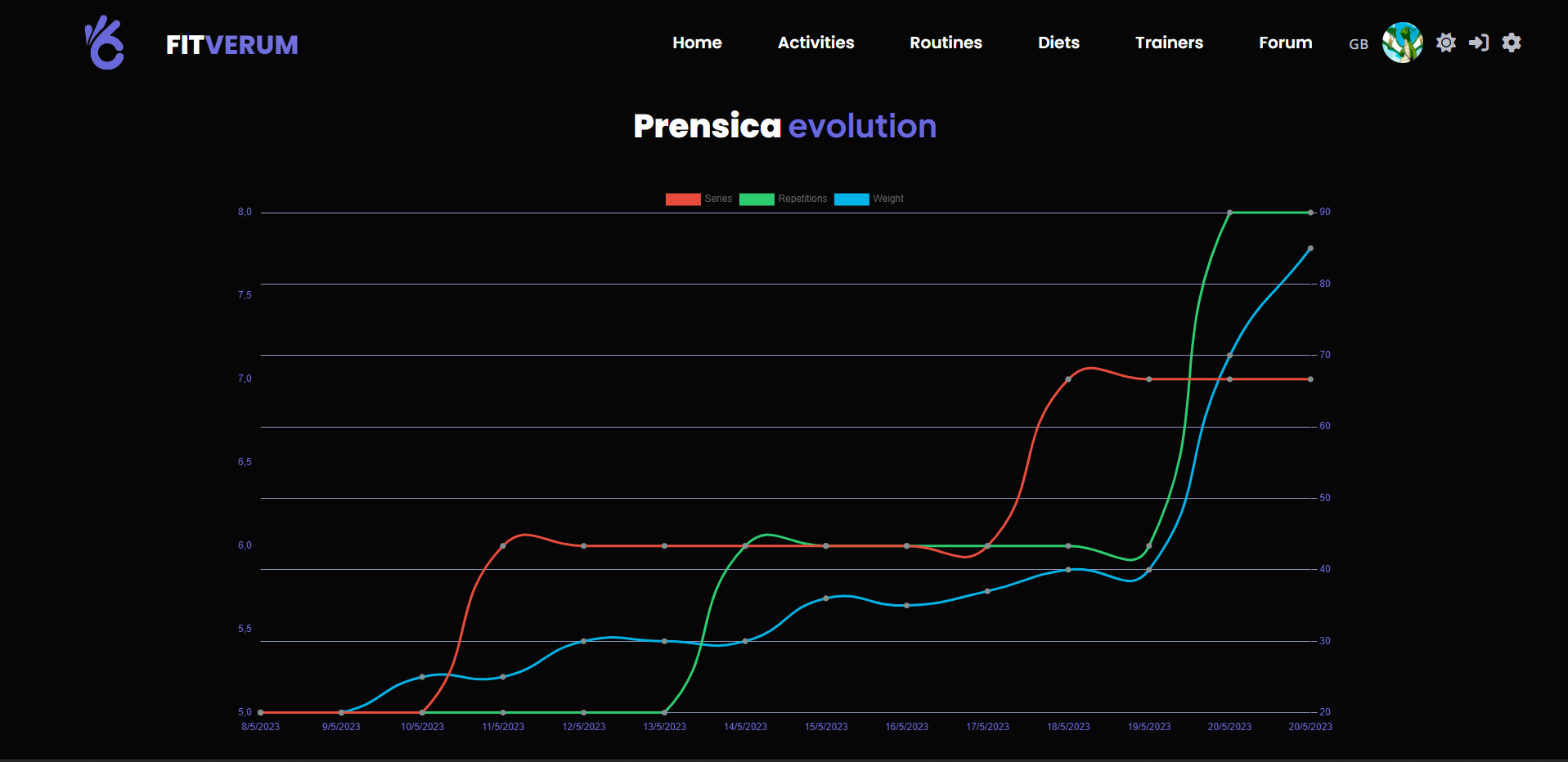




Una vez clickado, el usuario tendrá que elegir la rutina a la que pertenece el ejercicio que quiere apuntar y elegir dicho ejercicio, llevando a la página de instrucción de datos. Como pasaba previamente en la página análoga de la gestión del entrenador, hasta que todos los datos no estén completos, no te dejará crear dicha entrada. El campo de comentario es opcional.



Una vez creada aparecerá junto a las otras en el calendario. Los botones “Evolution” o “See evolution” abrirán una página que mostrará una gráfica con la evolución de dicho ejercicio y usuario.



# Manual de estilos

<https://pedroespigares.github.io/FitVerum-SassDoc/index.html>

## Wireframes

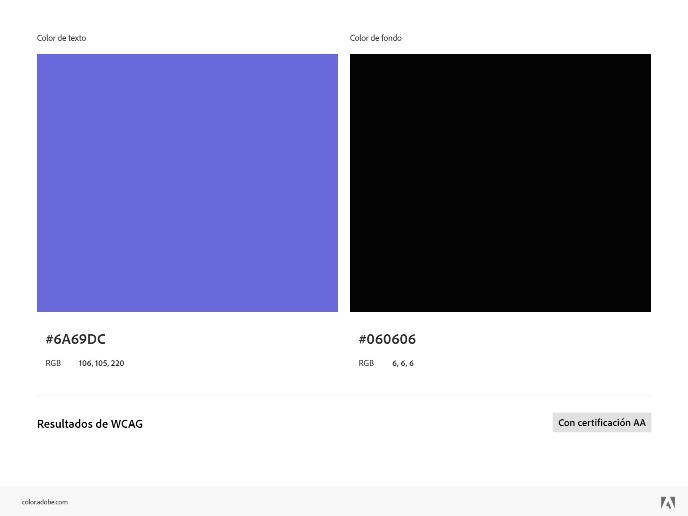
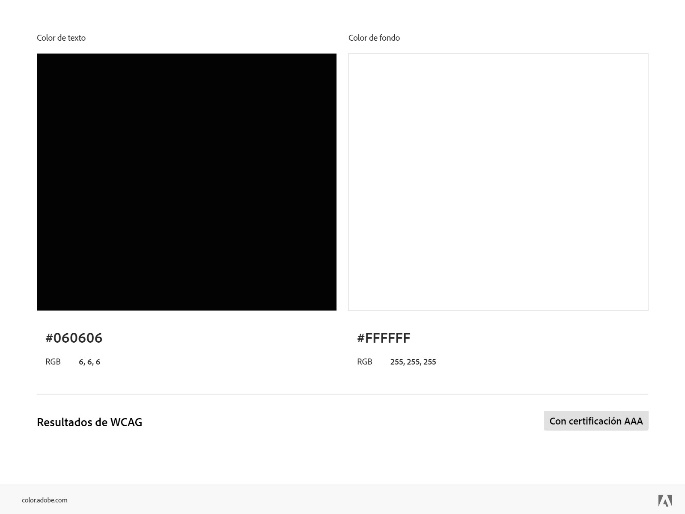
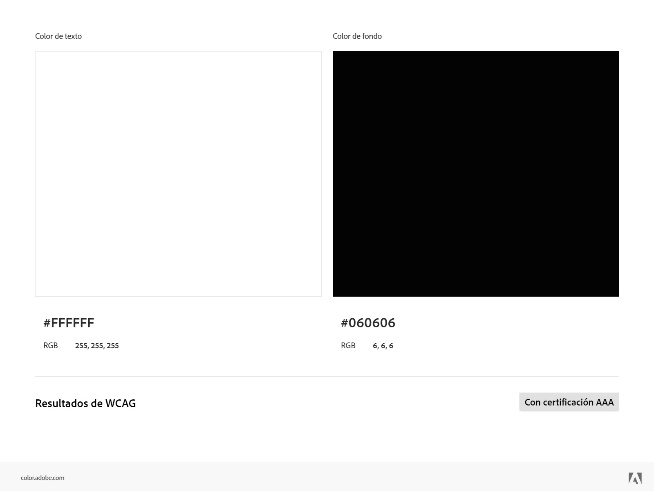
Los wireframes de la página están en el siguiente PDF:

<https://drive.google.com/file/d/1vp5giqh9aSp-zUk6QE5FWeVA-OOp1UN7/view?usp=sharing>

## Criterios de accesibilidad

Los colores elegidos para la página web cumplen con los criterios de conformidad nivel AA a nivel de contraste en títulos y elementos que tengan el color primario (#6A69DC)

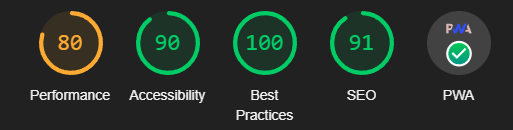
y el nivel AAA de contraste en textos.

Gráfico, Gráfico de rectángulos

Descripción generada automáticamente

## Criterios de usabilidad

* **Simplicidad**: La página sigue una línea estética minimalista, sencilla e intuitiva que permite a cualquier tipo de usuario utilizar la página eficientemente. Además, la página utiliza solo 3 colores para los elementos principales y otros 3 colores claramente diferenciados e intuitivos para las acciones que se puedan realizar sobre los elementos de la página
* **Rapidez y accesibilidad**: Según la herramienta Lighthouse de Google:



* **Interacción**: La página es totalmente interactiva y reactiva en tiempo real a los cambios que se realizan en la base de datos.
* **Universalidad**: En el apartado siguiente se explicará con más detalle, pero la página web está adaptada a todo tipo de soportes, es decir, es responsive.

## Fuentes

**Fuente para títulos**: Plus Jakarta Sans

<https://fonts.google.com/specimen/Plus+Jakarta+Sans>

**Fuente para texto**: Poppins

<https://fonts.google.com/specimen/Poppins>

## Tamaños principales 🡪

|  |  |
| --- | --- |
| Tamaño | Uso |
| Landing-text-title | Exclusivo para el título del hero de la landing page. |
| Xxxl | Títulos de las secciones de la landing page |
| Xx | Título de las vistas (h1) |
| Xl | Botones en vista completa |
| Lg / md-lg / md | Textos y botones en vista completa |
| Sm / xs / xxs | Textos y botones en vista Reducida |

## Mapa de colores



## Puntos de ruptura

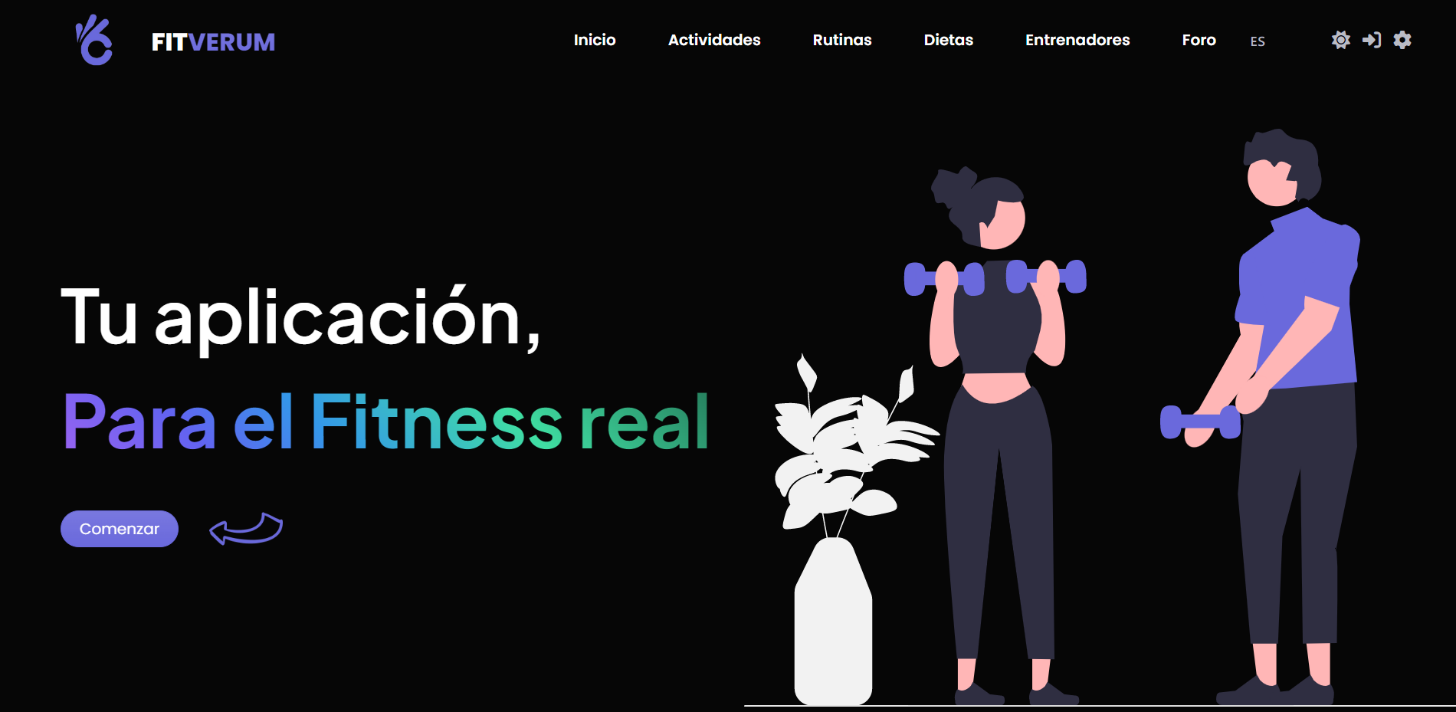
Texto

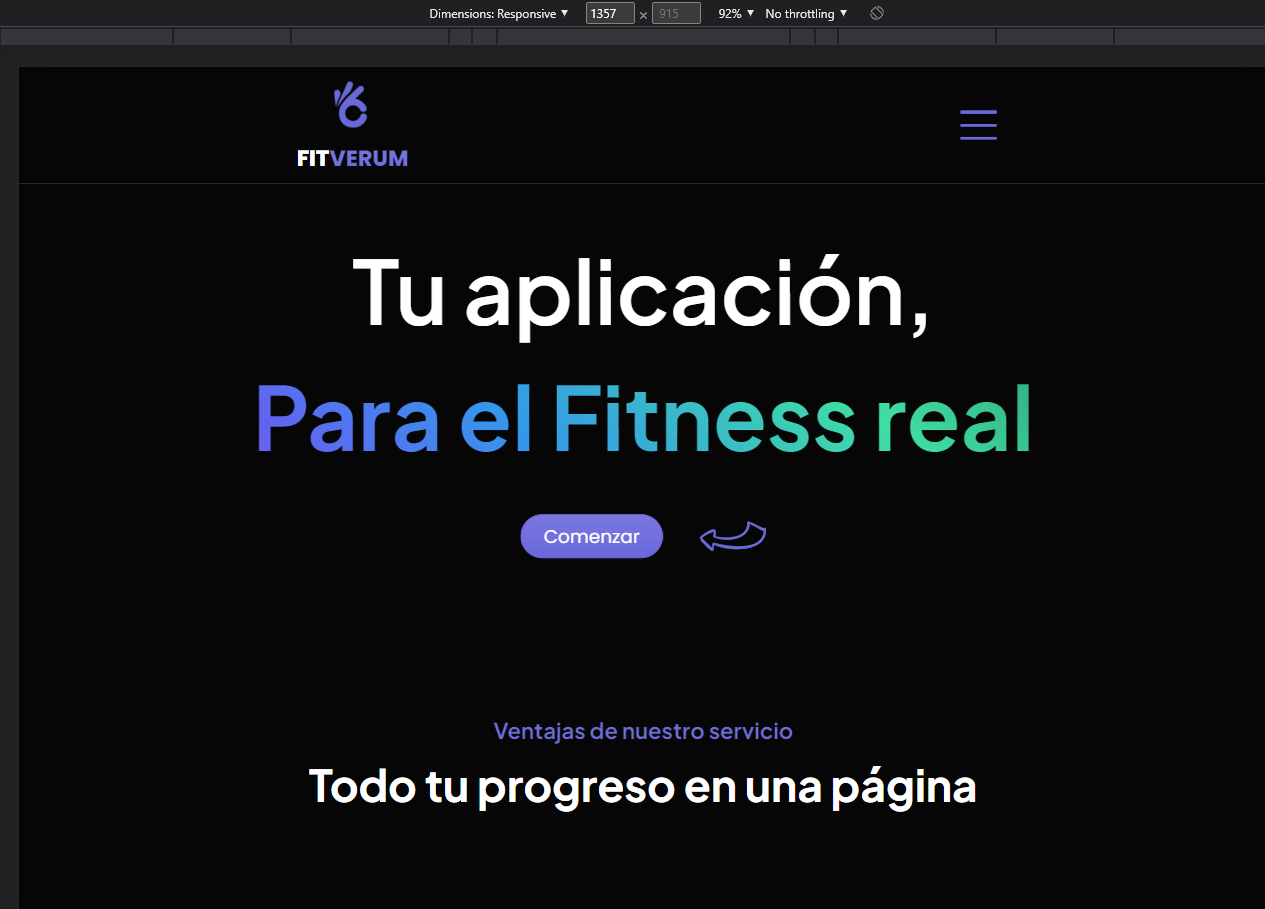
Descripción generada automáticamente

|  |  |
| --- | --- |
| Tamaño | Dispositivo |
| Desktop - min (1750px) | Vista pantalla desktop |
| Xl - min (1500px) | Vista pantalla portátil |
| Lg - min (992px) | Vista Tablet grande |
| Md - min (768px) | Vista Tablet pequeña |
| Sm - min(576px) | Vista móvil grande |
| Xs - min(0px) | Vista móvil pequeño |

## Ejemplos de vistas

Desktop:



Tablet:

Móvil:

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

# Software utilizado

Para este proyecto he utilizado Visual Studio Code como entorno de desarrollo y Firebase como aplicación web para el servicio backend. Para la creación de diagramas he utilizado Draw.io. Para la gestión de versiones he utilizado Git.

# Posibles mejoras

Mejoras en el foro, como podría haber sido enviar imágenes o el poder responder a los posts. El código podría haber sido mejor organizado creando un servicio específico para cada uno de los apartados que engloban varios componentes, como pueden ser trainers, administration, etc. En vez de utilizar el mismo servicio de base de datos para todo, es decir, subdividir dicho servicio. No utilización de try & catch en el código, aunque para las promesas y realizo manejo de excepciones. Mejoras en la optimización del código, ya que Angular tiene algunos métodos como los “lazy components” que permitirían una mayor rapidez en la carga de datos.

# Bibliografía

## Stack Overflow

<https://stackoverflow.com/>

## Usabilidad Web

<https://blog.hubspot.es/website/que-es-usabilidad-web#:~:text=La%20usabilidad%20web%20se%20refiere,un%20esfuerzo%20y%20tiempo%20m%C3%ADnimos>.

## Principios de accesibilidad

<https://www.w3.org/WAI/fundamentals/accessibility-principles/es>

## Adobe contrast color analyzer

<https://color.adobe.com/es/create/color-contrast-analyzer>