



TÉCNICO SUPERIOR EN  
DESARROLLO DE APLICACIONES WEB

DEPARTAMENTO DE INFORMÁTICA



MANUAL TÉCNICO

Autor: Adrián Gutiérrez de la Osa  
Curso académico: 2022/2023

# Índice

<b>1.- Introducción.....</b>	<b>3</b>
<b>2.- Arquitectura de la aplicación.....</b>	<b>3</b>
2.1.- Introducción.....	3
2.2.- Frontend.....	3
2.3.- Backend.....	4
<b>3.- Documentación técnica.....</b>	<b>4</b>
3.1.- Desarrollo.....	4
3.1.1.- Diseño de la base de datos.....	4
3.1.2.- Diagrama de casos de uso.....	6
3.2.- Pruebas realizadas.....	7
<b>4.- Despliegue de la aplicación.....</b>	<b>7</b>
<b>5.- Propuestas de mejora.....</b>	<b>10</b>

# 1.- Introducción

Workflow es una aplicación web destinada al registro y seguimiento de entrenamientos en el gimnasio.

Su principal objetivo es proporcionar a sus usuarios un mecanismo para llevar un control de todo lo que hacen y de su progresión, permitiéndoles tomar mejores decisiones a nivel de planificación, organización etc.

## 2.- Arquitectura de la aplicación

### 2.1.- Introducción

La aplicación se ha desarrollado utilizando el framework de PHP [Laravel](#) en su versión 10. Adicionalmente, se ha hecho uso de Laravel Breeze, un paquete de “scaffolding” que aporta más funcionalidades a Laravel y mejora la experiencia de desarrollo.

La aplicación se ha desarrollado en 2 dispositivos diferentes, un Windows 10 y un mac OS.

En Windows 10 he utilizado [Laragon](#) para crear el entorno de desarrollo con la versión de PHP y MySQL deseadas.

En el caso de mac OS, he utilizado [Laravel Herd](#), una herramienta que genera un entorno de desarrollo específico para trabajar con Laravel y PHP.

Adicionalmente, se han utilizado diversas herramientas en ambos equipos; Github Desktop, Visual Studio Code, Composer, NodeJS, npm...

### 2.2.- Frontend

En cuanto al desarrollo del lado del cliente, el stack tecnológico ha sido el siguiente:

- HTML
- CSS
- JavaScript

Más en concreto, y dado que el proyecto está desarrollado con Laravel, se ha hecho uso de su motor de plantillas, [Blade](#).

A nivel de estilos, se ha utilizado una combinación de estilos nativos de CSS con la librería [Tailwind CSS](#) en su versión 3.1.0.

También he obtenido algunos componentes y/o ideas de diseño de la plataforma [Merakiui](#), que utiliza Tailwind CSS para dar estilos a sus componentes..

Y a nivel de JavaScript, se han utilizado diversas librerías para aportar mayor funcionalidad y dinamismo a la aplicación:

- [jQuery](#): versión 3.7.1
- [Datatables](#): versión 1.13.6
- [Apache echarts](#): versión 5.4.3
- [SweetAlert2](#): versión 11.10.1
- [AlpineJs](#): versión 3.4.2

## 2.3.- Backend

En lo que al backend respecta, y puesto que hemos utilizado Laravel, el lenguaje de programación escogido es PHP.

A nivel de versiones, durante el desarrollo se han usado las versiones 8.1 y 8.2, y ahora en producción, la 8.2.

Aprovechando el ecosistema de Laravel, he utilizado [Breeze](#) como starter kit, y me he basado en [Sanctum](#) para la autenticación.

Centrándonos en la base de datos, he optado por utilizar el gestor MySQL en su versión 8.0.3.

Y gracias a que estamos usando Laravel, podemos hacer uso de Eloquent, el ORM de Laravel que nos permite establecer relaciones e interactuar con los datos de una forma mucho más sencilla y potente.

## 3.- Documentación técnica

### 3.1.- Desarrollo

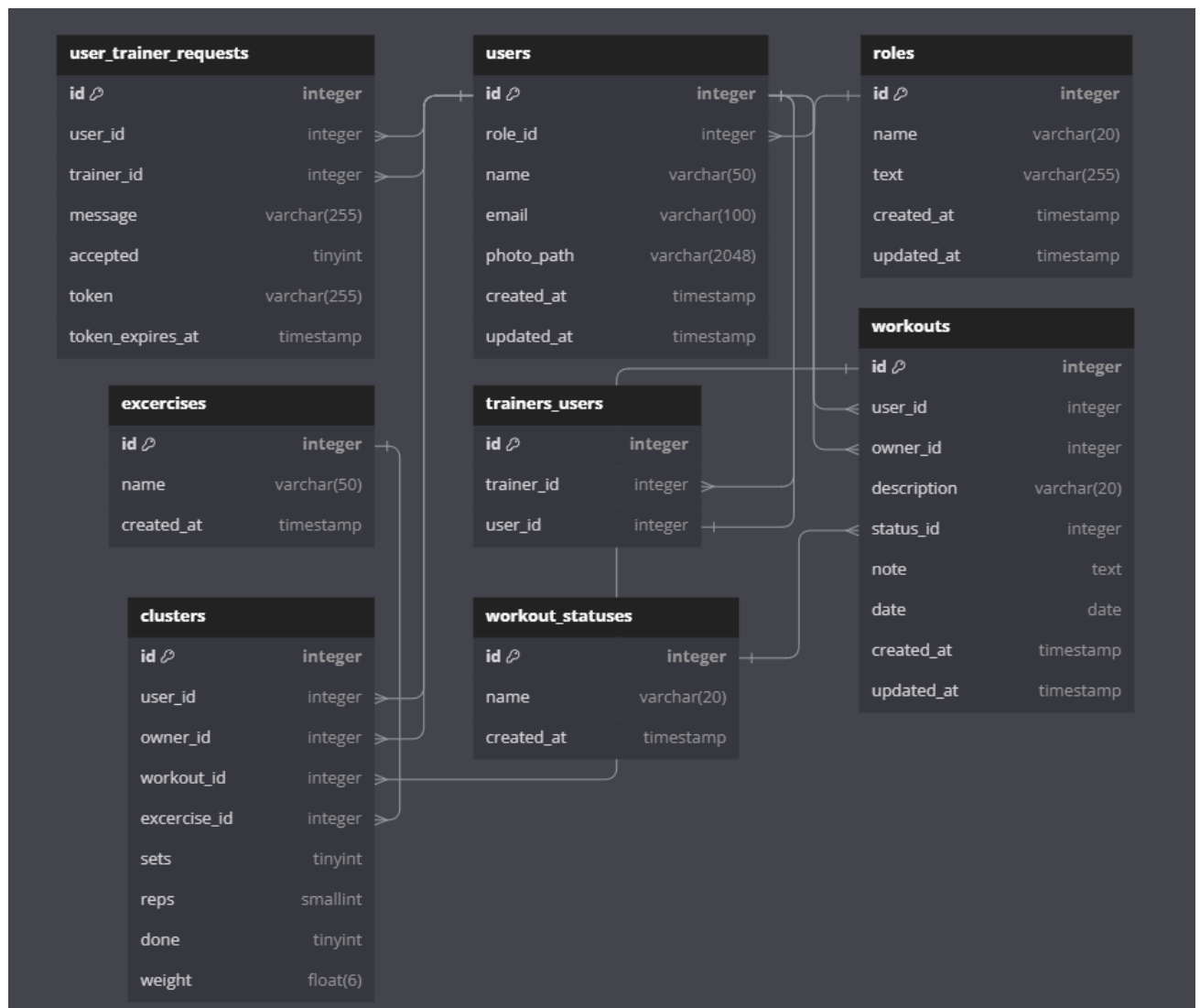
#### 3.1.1.- Diseño de la base de datos

La base de datos se ha ido creando y desarrollando poco a poco a medida que avanzaba el proyecto con las migraciones de Laravel.

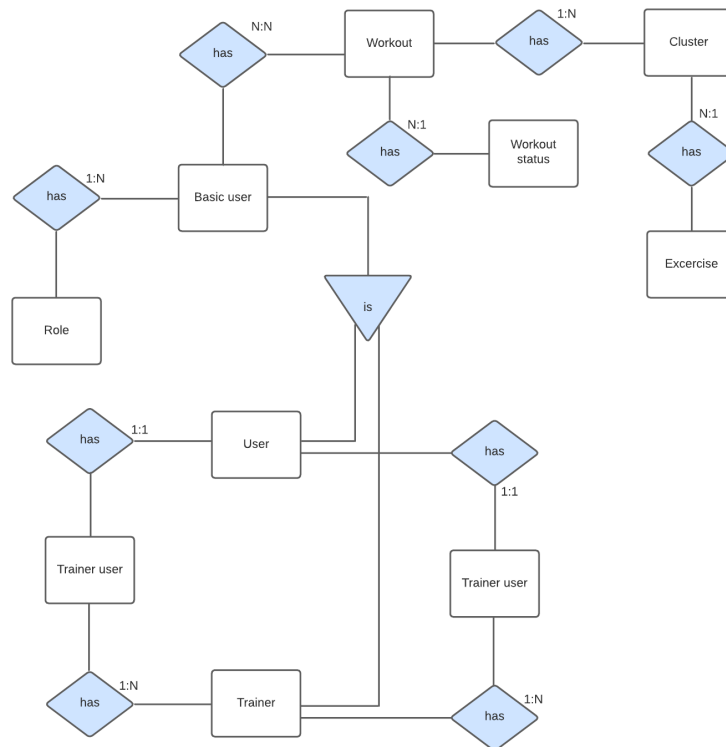
De este modo, en el directorio database/migrations se puede ver la creación de las tablas y en caso de haberlas, las modificaciones recibidas conforme el proyecto iba cogiendo forma.

No obstante, el diseño actual es el siguiente:

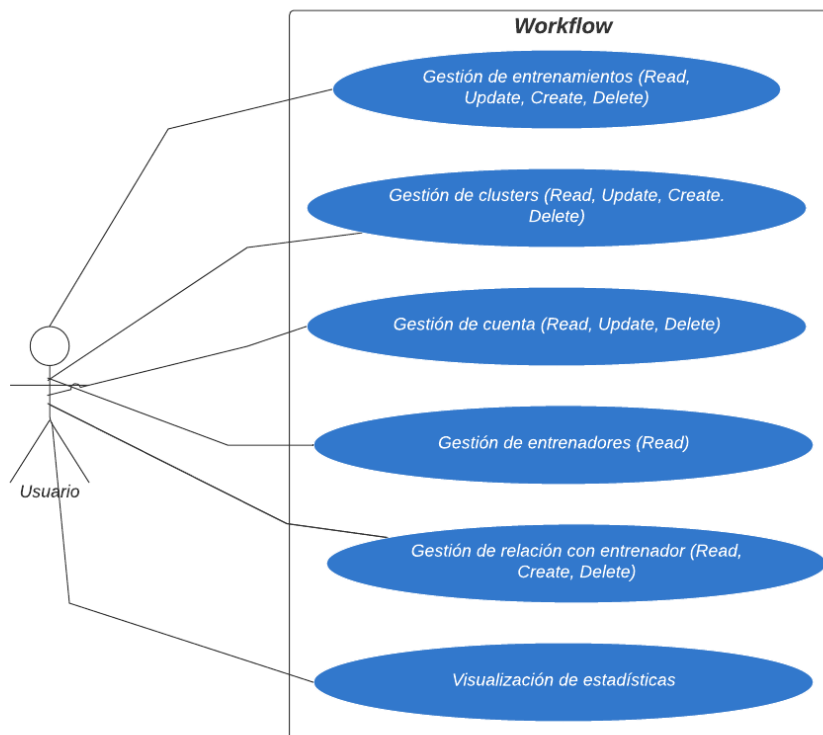
## Modelo relacional de la BBDD:

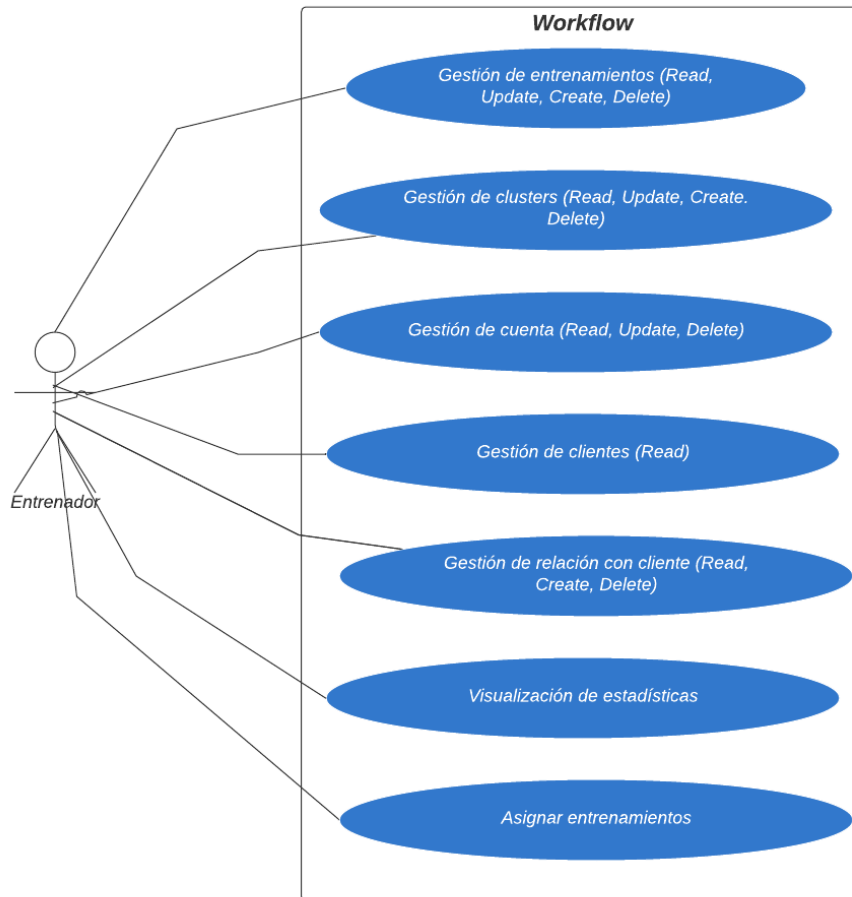


## Modelo entidad relación:



### 3.1.2.- Diagrama de casos de uso





### 3.2.- Pruebas realizadas

Para comprobar el funcionamiento de la aplicación, se han realizado pruebas de forma manual utilizando la aplicación como un usuario normal: crear entrenamientos, modificarlos, borrarlos...

Además, también se ha hecho probar la aplicación a varias personas en busca de errores o posibles mejoras.

## 4.- Despliegue de la aplicación

Dado que las tecnologías del lado del backend de la aplicación están muy extendidas, una gran cantidad de proveedores de alojamiento tienen soporte para estas.

En mi caso, tenía contratado un hosting web con soporte para PHP y MySQL, por lo que esta ha sido la opción escogida para el despliegue.

La empresa en cuestión es Raiola Networks, que utiliza cPanel para la gestión de sus sitios.

Partimos de la base de que tenemos contratado el servicio y un dominio bajo el que hacer el despliegue. En mi caso: <https://workflow.adriangutierrezd.com/>

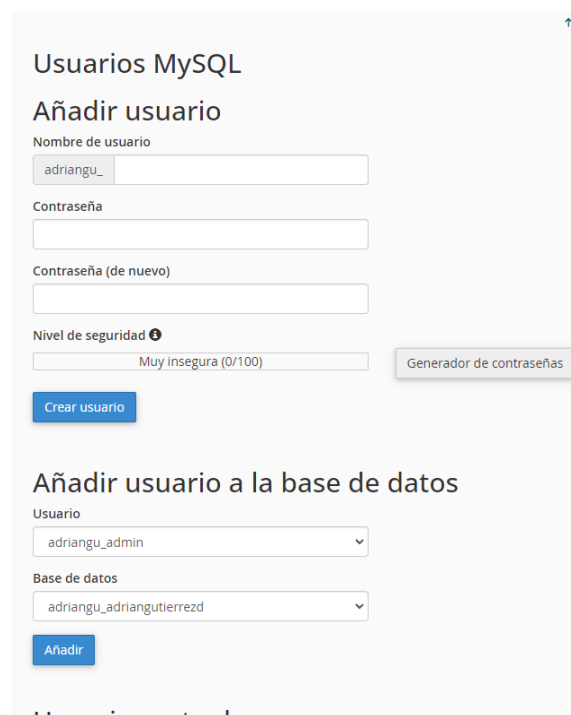
Una vez dentro del cPanel, tenemos que hacer 2 cosas:

- Gestión de base de datos
- Gestión de ficheros

Empezando por la base de datos, debemos crear una nueva base de datos con un usuario existente en el sistema o uno nuevo. Sea como sea, es importante tener acceso a las credenciales que necesitaremos después para establecer la conexión a la base de datos.



Una vez tienes una base de datos, creas un usuario y lo añades:



Finalmente debemos acceder a phpMyAdmin e importar la información de la base de datos. En mi caso al estar usando Laravel y su sistema de migraciones, puedo tener un “estado inicial” en local con el comando:

```
php artisan migrate:fresh --seed
```

Posteriormente, solo tengo que obtener un *dump* de la base de datos e importarla.



Por el lado de los ficheros, hay que recalcar que antes de comenzar cualquier proceso de despliegue debemos ejecutar un comando en nuestro sitio web:

*npm run build*

Este comando en realidad acaba ejecutando “vite build”. Pero ¿qué es Vite?

[Vite](#) es la herramienta que ha entrado a sustituir a Webpack en las nuevas versiones de Laravel para el desarrollo del frontend con JavaScript y Tailwind CSS.

De este modo, el comando “vite build” se encarga de empaquetar los archivos CSS y JavaScript para prepararlos para el despliegue en producción.

Sin este paso nuestra aplicación no funcionará.

Una vez los ficheros están listos para el despliegue, tenemos 2 posibilidades:

- Subir los ficheros por FTP hasta el directorio de nuestro dominio.
- Subir los ficheros de forma “manual” en un fichero .zip y descomprimirlos en el directorio adecuado.

Si empleamos la primera debemos crear un usuario FTP con acceso al directorio de interés y usar un cliente de FTP como Filezilla para la operación:

**Añadir cuenta FTP**

**Iniciar sesión**

@

**Dominio**

**Contraseña**

**Contraseña (de nuevo)**

**Nivel de seguridad ⓘ**

**Generador de contraseñas**

**Directorio**

**Cuota**

☐ 2000 MB

☒ Ilimitado

**Crear cuenta FTP**

Una vez los ficheros están subidos al servidor, debemos modificar los archivos clave como el .htaccess para poder acceder al sitio, y el .env con credenciales de base de datos, correo electrónico, claves privadas...

## 5.- Propuestas de mejora

Algunas funcionalidades a incluir en futuras versiones de la aplicación:

- **Uso de ejercicios personalizados:** en este momento los ejercicios de la plataforma son estáticos. De este modo, podemos ofrecer a los usuarios la posibilidad de añadir sus propios ejercicios y complementar a los de la plataforma.
- **Soporte para ejercicios de cardio:** la aplicación y su estructura está diseñada para registrar entrenamientos en el gimnasio, pero no permite registrar carrera, salto a la comba, bicicleta...
- **Registro de tiempo de entrenamiento:** añadir un “cronómetro” que permite al usuario indicar cuánto tiempo ha tardado en realizar un entrenamiento determinado, ya que es una métrica interesante para medir la intensidad.

Además, hay ciertas funcionalidades que no ha dado tiempo a desarrollar, y que se incluirían en estas versiones posteriores:

- **Anotaciones de entrenamientos:** la tabla de entrenamientos (*workouts*) tiene un campo “note” para poder añadir un comentario sobre el entrenamiento que no estamos mostrando al usuario ni para completarlo ni para visualizarlo.
- **Soporte para varias unidades:** la tabla de clusters (*clusters*) tiene un campo “unit” que quiero utilizar para permitir que cada usuario pueda expresar sus pesos en la unidad que quiera; kilogramos, libras...
- **Soporte multilingüe:** la aplicación se encuentra traducida al completo en español e inglés. Todos los literales de texto en el código están escritos en inglés y tienen una clave con su traducción a español en un fichero .json. En futuras versiones quiero incluir un selector de idioma español / inglés.

```
<div class="mt-2 text-center">
  <h3 class="modal-title" id="modal-title">{{__('Delete workout')}}</h3>
  <p class="mt-2 text-sm text-gray-500 dark:text-gray-400">
    {{__('If you continue, the training will be permanently deleted.')}}
  </p>
</div>
```