

**Máster Desarrollo Web Fullstack**

**Proyecto final**

**Alumno: Rodrigo García López NIF: 52901425E**



[1. Sinopsis de la aplicación 2](#_Toc1)

[2. Puesta en marcha de la aplicación 2](#_Toc2)

[3. Vistas de la aplicación 3](#_Toc3)

[3.1 Landing page 3](#_Toc4)

[3.2 Mapa 4](#_Toc5)

[3.3 Panel 6](#_Toc6)

[3.3 Perfil de usuario 7](#_Toc7)

[3.4 Ajustes del perfil 8](#_Toc8)

[4. Estructura de datos de la aplicación 9](#_Toc9)

[5. API 10](#_Toc10)

# Sinopsis de la aplicación

Nombre: Climber’s Best Friend

Caso de uso: Red social para escaladores.

Tecnologías usadas:

* Front-end: Vue.js 3
  + Gestión de estilos: Tailwind CSS
  + Cliente de llamadas HTTP: Axios
  + Mapa interactivo: Leaflet
  + Analizador de archivos KML: leaflet-omnivore
  + Carrusel de medios: vue3-carousel
* Back-end: Laravel
  + Gestión de usuarios: Jetstream
  + Gestión de rutas: Inertia.js
* Gestor de bases de datos: MySQL

# Puesta en marcha de la aplicación

Se necesita tener Laravel y XAMPP en nuestro equipo.

* Versión de PHP usada: 8.3.9
* Versión de Laravel usada: 11.7.0

Para instalar las librerías necesarias e inicializar la base de datos, después de modificar el archivo .env encontrado en la raíz del proyecto, deberemos ejecutar los siguientes comandos:

* composer install
* npm install
* php artisan migrate
* php db:seed —class=RouteSeeder

Para ejecutar la aplicación y verla en nuestro navegador, deberemos ejecutar los siguientes comandos (en diferentes ventanas de terminal):

* php artisan serve
* npm run dev

Podremos ver la aplicación en la dirección localhost:8000.

La aplicación junto con la memoria se encuentra en el siguiente repositorio:

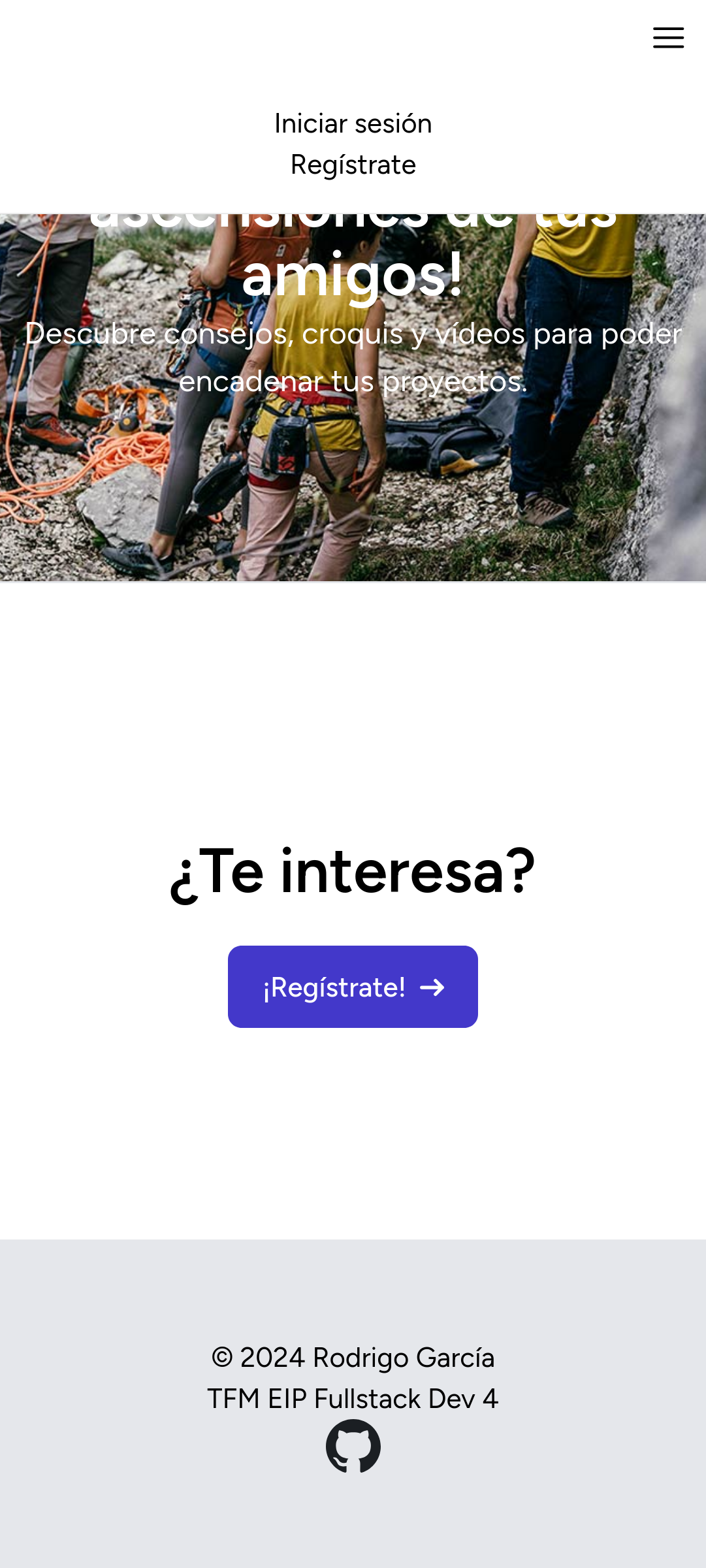
<https://github.com/litoralWasTaken/climbers-best-friend>

# Vistas de la aplicación

## 3.1 Landing page

Ruta: /

Landing page de la aplicación.



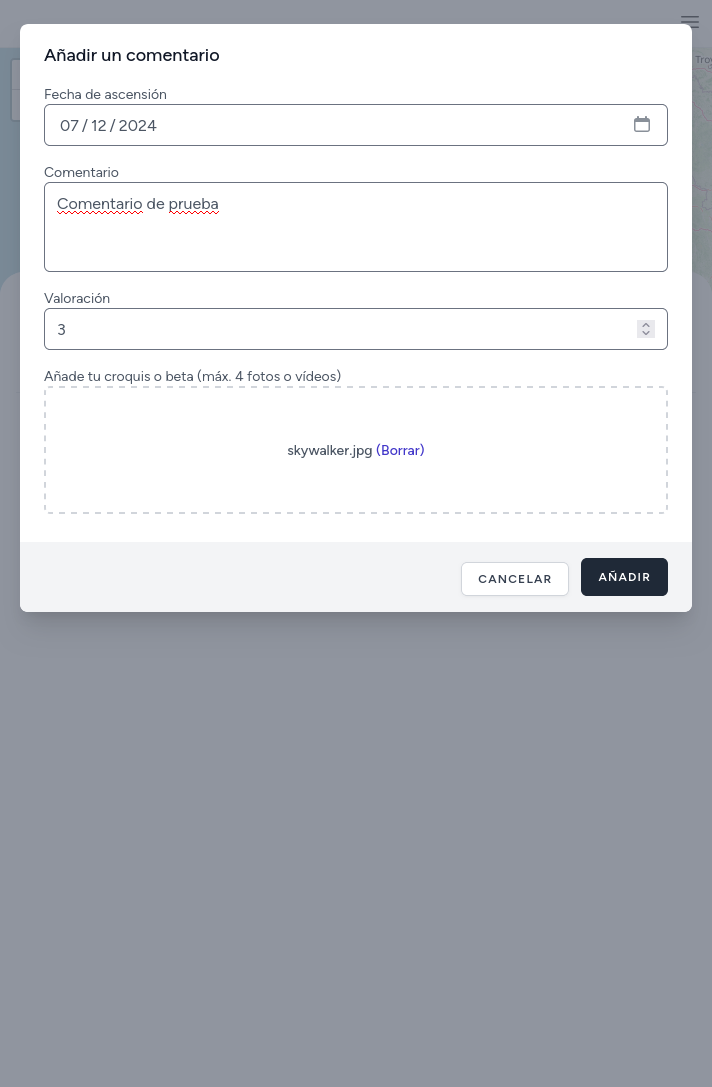
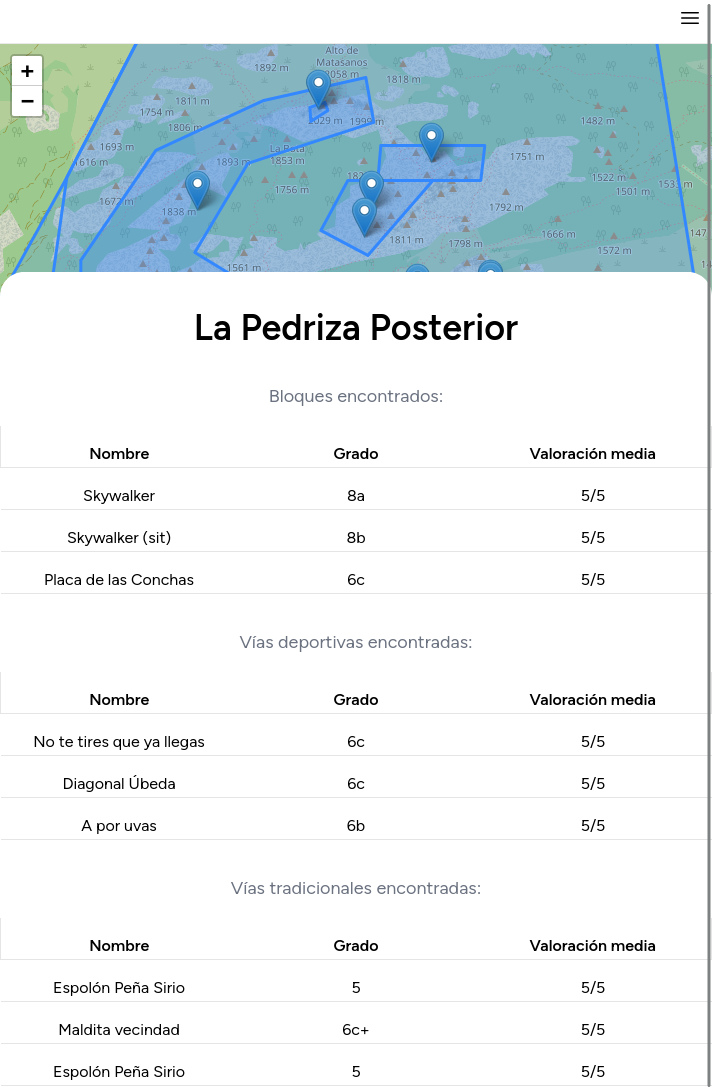
## 3.2 Mapa

Ruta: /map

Vista principal de la aplicación. Podremos ver diferentes sectores de escalada, marcados con puntos azules. Al hacer clic en ellos, podremos ver las diferentes rutas de ese sector categorizadas por estilo (bloque, escalada deportiva o escalada tradicional) junto con su grado.

Si además hacemos clic en las filas, podremos ver los comentarios de los usuarios sobre la ruta de escalada elegida. Los usuarios también podrán incluir material audiovisual referido al bloque.

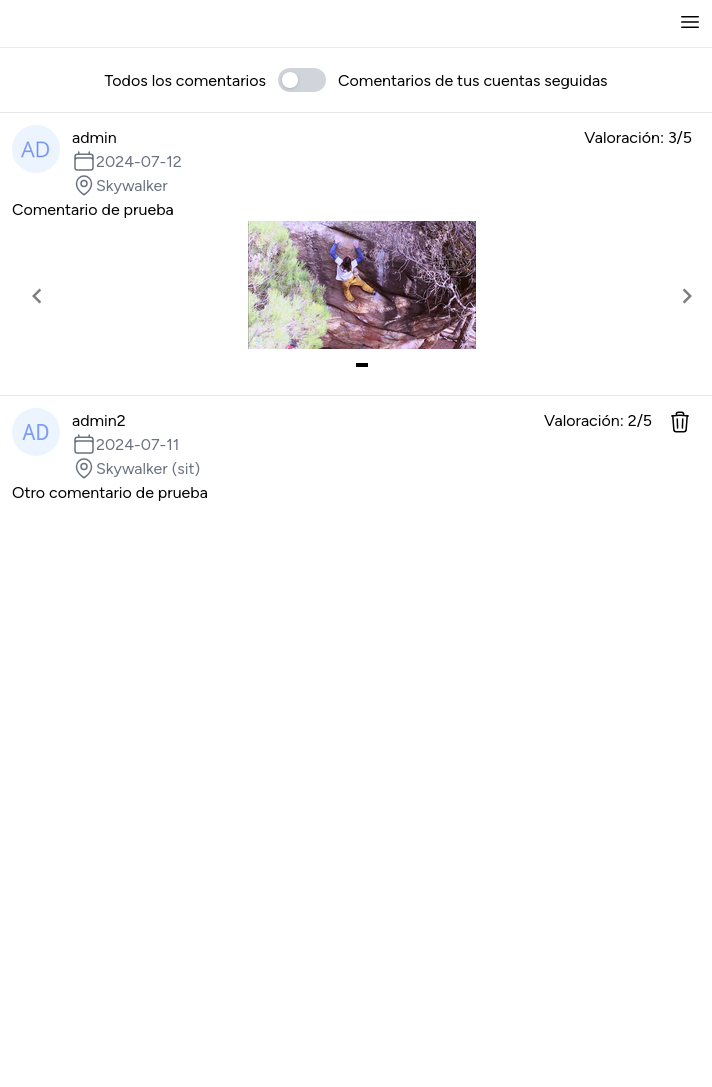
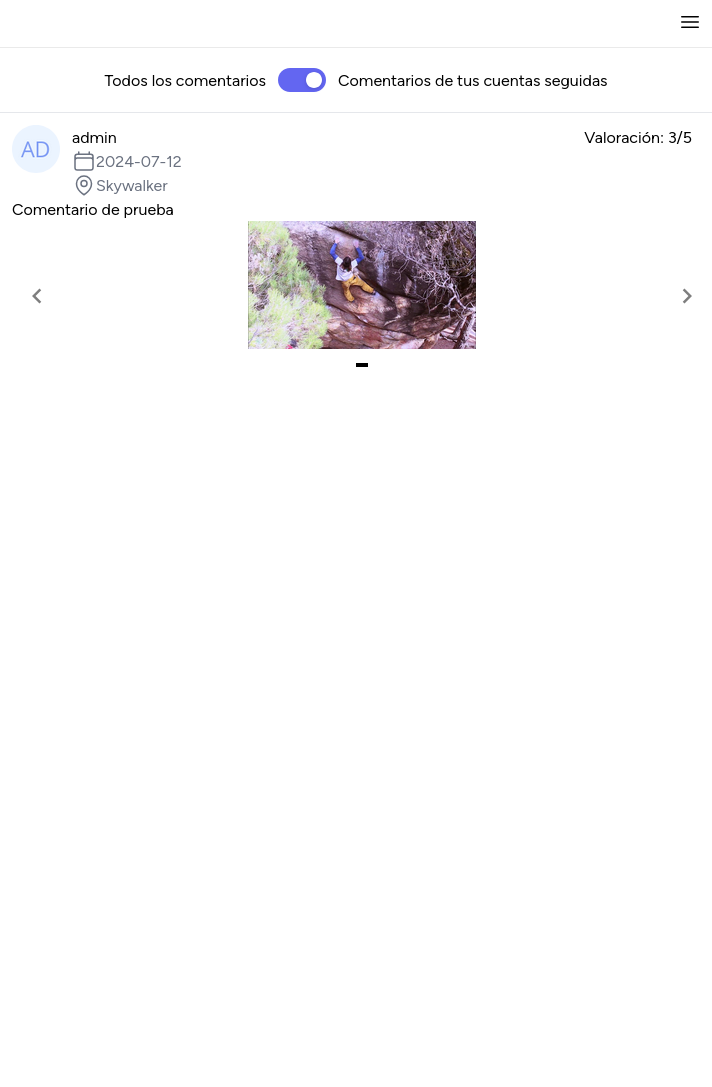
(Sólo disponible mapa de La Pedriza. Más mapas de sectores pueden ser añadidos mediante archivos KML.)



## 3.3 Panel

Ruta: /dashboard

Vista de todos los comentarios de la aplicación. Con el interruptor de la sección superior se pueden ver los comentarios de las cuentas que sigue el usuario. (Conectado como usuario admin2, siguiendo a admin)



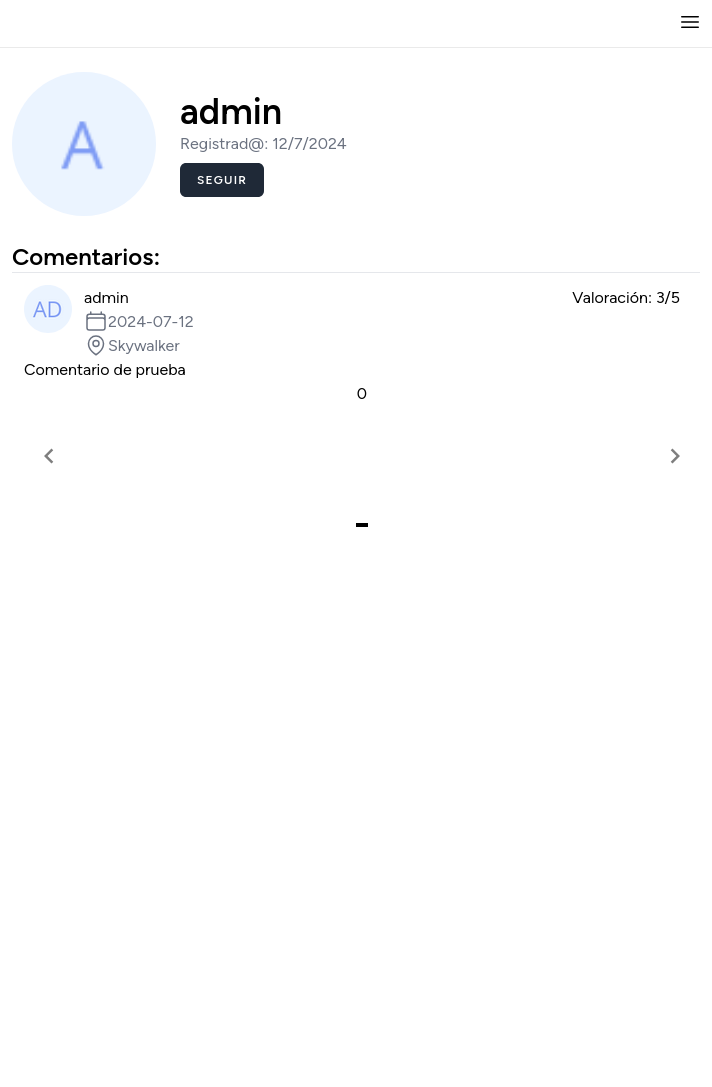
## 3.3 Perfil de usuario

Ruta: /user/(id del usuario)

Si se quiere ver el perfil del usuario conectado, no se indicará el ID en la ruta.

Se podrá ver la información y comentarios de un usuario específico. Aquí también se podrá seguir a un usuario y ver si el susodicho le sigue.

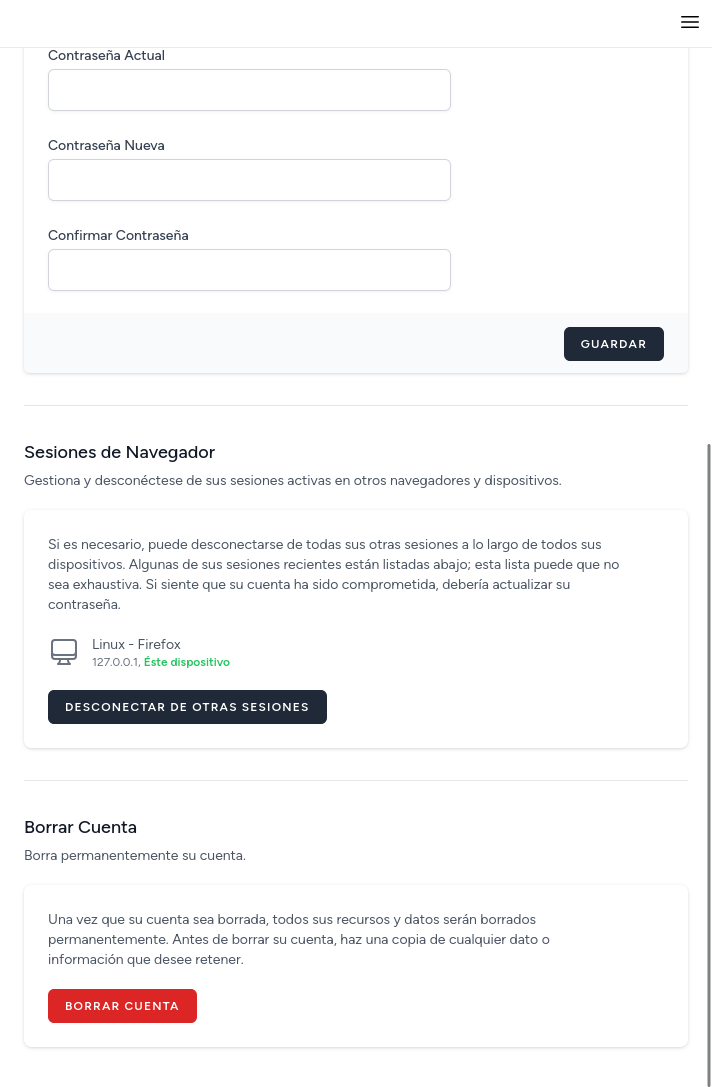
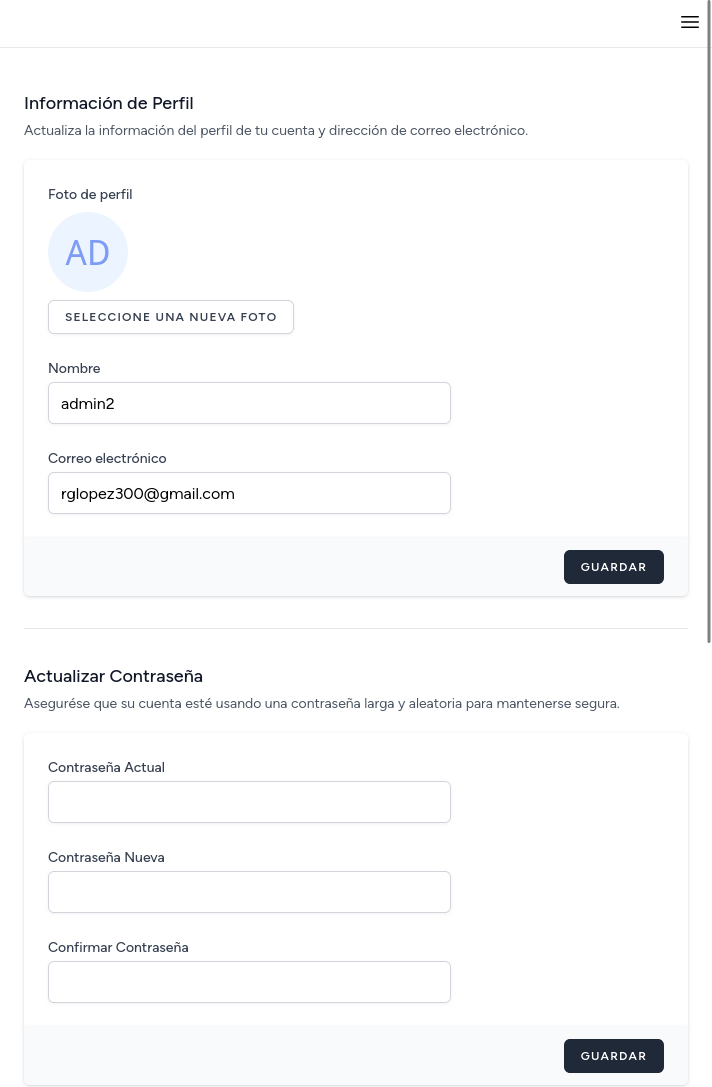
Para acceder a esta página se puede dar clic al nombre de usuario de un comentario en cualquiera de las secciones de la aplicación.



## 3.4 Ajustes del perfil

Ruta: /settings

Aquí se puede cambiar los parámetros del usuario conectado (foto de perfil, nombre, correo electrónico, contraseña.) También se pueden borrar las sesiones de otros navegadores y borrar la cuenta del usuario.



# Estructura de datos de la aplicación

Los datos contenidos en la base de datos de la aplicación están explicados en este diagrama de flujo:



Hay rutas de escalada (routes), las cuales se identifican por su latitud y longitud en el mapa. Esas rutas pueden contener muchos comentarios (route\_posts) (1-N), los cuales pueden contener muchos archivos multimedia (route\_post\_media) (1-N).

Cada usuario (users) puede tener muchos comentarios (1-N). Además de tener comentarios, muchos usuarios se pueden seguir entre ellos (user\_follows) (N-N).

# API

Para pasar datos del back-end al front-end, se puede hacer de dos formas: pasando propiedades (props) a las páginas mediante las rutas de Inertia.js declaradas en el archivo de páginas (web.php) o mediante API REST provista por Laravel.

Endpoints de API no protegidos:

* /api/routes/
  + Parámetros: N/A
  + Método de petición: GET
  + Descripción: Devuelve todas las rutas de la aplicación.
* /api/routes/latlng
  + Parámetros de URL: lat (latitud), lng (longitud)
  + Método de petición: GET
  + Descripción: Devuelve todas las rutas de un sector en específico identificado por latitud y longitud
* /api/posts
  + Parámetros: N/A
  + Método de petición: GET
  + Descripción: Devuelve todos los comentarios de todas las rutas de la aplicación, junto con sus archivos multimedia y datos del usuario
* /api/posts/followed/
  + Parámetros de URL: id (ID del usuario que hace la petición)
  + Método de petición: GET
  + Descripción: Devuelve todos los comentarios de los usuarios seguidos por el usuario que hace la petición (identificado por parámetro id)
* /api/posts/
  + Parámetros de URL: id (ID de la ruta)
  + Método de petición: GET
  + Descripción: Devuelve todos los comentarios de la ruta identificada por su ID numérica en la base de datos

Endpoints de API protegidos:

* /api/posts/upload
* Parámetros de cuerpo:
  + Comentario (object)
    - ascentDate: String
    - rating: Number (0-5)
    - comment: String
    - route\_id: Number
    - Media: Array [File | null, File | null, File | null, File | null]
* Método de petición: POST
* Descripción: Añade un comentario (fecha de ascensión, valoración, comentario, archivos multimedia) a una ruta de escalada identificada por su ID numérica.
* /api/posts/delete
  + Parámetros de cuerpo:
    - index: Number
  + Método de petición: POST
  + Descripción: Borra un comentario identificado por su ID numérica.
* /api/posts/user/follow
  + Parámetros de cuerpo:
    - id: Number
  + Método de petición: POST
  + Descripción: Crea seguimiento entre el usuario que hace la petición (identificado por Laravel) y el usuario indicado por su ID numérica.
* /api/posts/user/unfollow
  + Parámetros de cuerpo:
    - id: Number
  + Método de petición: POST
  + Descripción: Elimina el seguimiento entre el usuario que hace la petición (identificado por Laravel) y el usuario indicado por su ID numérica.