3-6-2024

SÁNCHEZ FERNÁNDEZ, ÁLVARO

IES - MUTXAMEL

Proyecto de Desarrollo De Aplicaciones Web

2º DAW

ÍNDICE

[Título 2](#_Toc168573168)

[Introducción 2](#_Toc168573169)

[Objetivo 2](#_Toc168573170)

[Justificación 2](#_Toc168573171)

[Análisis de lo existente 2](#_Toc168573172)

[Requisitos funcionales de la aplicación 3](#_Toc168573173)

[Análisis de Diseño 4](#_Toc168573174)

[Diagrama de arquitectura 5](#_Toc168573175)

[Diagrama de casos de uso 6](#_Toc168573176)

[Diagrama de clases 7](#_Toc168573177)

[Diseño de datos 7](#_Toc168573178)

[Codificación 8](#_Toc168573179)

[Entorno de programación 8](#_Toc168573180)

[Lenguaje y herramientas 9](#_Toc168573181)

[Aspectos relevantes de la implementación 10](#_Toc168573182)

[Manual de usuario 14](#_Toc168573183)

[Requisitos de Instalación 14](#_Toc168573184)

[Software Necesario 15](#_Toc168573185)

[Visual Studio Code 15](#_Toc168573186)

[XAMPP 15](#_Toc168573187)

[Composer 16](#_Toc168573188)

[Laravel 16](#_Toc168573189)

[Node.js 16](#_Toc168573190)

[Conclusiones 16](#_Toc168573191)

[Conclusiones sobre el trabajo realizado 17](#_Toc168573192)

[Posibles ampliaciones y mejoras 17](#_Toc168573193)

[Bibliografía y webgrafía 17](#_Toc168573194)

[Libros, artículos y apuntes 17](#_Toc168573195)

[Direcciones Web 17](#_Toc168573196)

[Apéndice, si procede 17](#_Toc168573197)

# Título

El nombre que se ha elegido para este proyecto es el de “PeakPals”, que combina dos palabras en inglés:

* Peak: que significa “cima”, refiriéndose al punto mas alto de una montaña
* Pals: que significa “amigos” o “compañeros”

Por lo tanto, “PeakPals” se puede traducir como “Amigos de la cima” o “Compañeros de escalada”.

# Introducción

Objetivo  
El objetivo principal del desarrollo de esta aplicación web es proporcionar una plataforma integral y fácil de usar para escaladores que buscan compañeros de escalada. Esta aplicación pretende cubrir la necesidad de conectar a escaladores con intereses y niveles de habilidad similares, facilitando la organización y participación en eventos de escalada. Al implementar un sistema de registro e inicio de sesión, junto con funcionalidades avanzadas como la creación, edición y eliminación de eventos, y el uso de filtros en tiempo real, la aplicación aspira a crear una comunidad activa y comprometida de escaladores que pueden encontrar y unirse a eventos de manera eficiente y efectiva.

## Justificación

La elección de desarrollar este proyecto responde a una necesidad detectada en la comunidad de escaladores: la falta de una plataforma centralizada y específica que facilite la conexión entre individuos con el objetivo de practicar escalada juntos. Actualmente, la organización de eventos de escalada y la búsqueda de compañeros suelen depender de redes sociales genéricas o foros dispersos, lo cual puede ser ineficiente y limitante.

A nivel empresarial, este proyecto tiene el potencial de satisfacer varias necesidades clave:

* **Conexión y comunidad**: Facilitar la creación de una red de escaladores que puedan conectarse y compartir su pasión.
* **Organización y accesibilidad**: Proveer una herramienta eficiente para organizar y acceder a eventos de escalada, optimizando la planificación y participación.
* **Crecimiento y expansión**: Ofrecer una plataforma que pueda expandirse con nuevas funcionalidades y servicios, atrayendo a más usuarios y potencialmente generando ingresos a través de suscripciones premium, publicidad o colaboraciones con gimnasios y tiendas de escalada.

## Análisis de lo existente

En términos de innovación, este proyecto introduce varias características que no están ampliamente disponibles en las aplicaciones existentes en el mercado:

* **Filtros en tiempo real**: Aunque algunas aplicaciones ofrecen filtrado de eventos, pocos permiten una experiencia de búsqueda y filtrado tan dinámica y en tiempo real como la propuesta en este proyecto.
* **Integración de mapas**: La inclusión de mapas interactivos para la creación y visualización de eventos añade un nivel de interactividad y precisión que mejora significativamente la experiencia del usuario.
* **Sistema de roles y herramientas administrativas**: La implementación de un sistema de roles con funcionalidades administrativas avanzadas es una característica diferenciadora que permite una gestión más eficiente de la comunidad y los eventos.

Existen aplicaciones en el mercado que pueden equipararse parcialmente con este proyecto, como Meetup y algunas plataformas específicas de escalada como Mountain Project y TheCrag. Sin embargo, estas aplicaciones a menudo carecen de la integración profunda de funcionalidades específicas para escaladores que este proyecto ofrece, tales como los filtros en tiempo real y la integración avanzada de mapas para la creación y visualización de eventos. Además, el enfoque en una comunidad de escaladores con una herramienta específica para organizar y unirse a eventos, junto con la gestión centralizada de estos eventos, aporta un valor añadido que no se encuentra comúnmente en las plataformas existentes.

# Requisitos funcionales de la aplicación

Esta aplicación esta enfocada a escaladores que buscan a compañeros de escalada.

La aplicación dispone con un sistema de registro e inicio de sesión para poder identificarte como escalador, en este deberás poner nombre de usuario, correo, contraseña y nivel de escalada. Los usuarios no registrados podrán ver los eventos existentes y filtrarlos por todas sus propiedades, a diferencia de los usuarios registrados que además de poder visualizar los eventos y filtrarlos podrán unirse a ellos, crear sus propios eventos, editarlos y eliminarlos.

Todos los filtros disponibles para usuarios no registrados y registrados son los siguientes:

* Filtro por nombre del evento
* Filtro por fecha de inicio del vento
* Filtro por fecha de finalización del evento
* Filtro por nivel de escalada
* Filtro por estado del evento (finalizado o en curso)
* Filtro por tipo de evento (escalada deportiva, rocódromo o via ferrata)
* Filtro por distancia del evento

El filtro extra con el que contaran los usuarios registrados es:

* Filtro por eventos creados

Cabe mencionar que todos estos filtros son en tiempo real.

Además, la aplicación cuenta con un apartado en la página principal en la cual podrás ver todos los eventos a los cuales te has unid, esto facilita la navegación del usuario por la aplicación ya que permite visualizar y entrar en los eventos en los que ya este unido ya sea para visualizarlos como para salir de este.

La aplicación cuenta con mapas, los cuales se sitúan en:

* El formulario de creación de un evento, para que el usuario clicando en el mapa pueda indicar donde quiere se sitúe su evento.
* En los propios eventos, para indicar la ubicación que se ha especificado en el formulario de creación de evento.
* En la página principal, donde se muestran todas las ubicaciones de todos los eventos. Cada marcador de cada evento se puede clicar, para que muestre una breve descripción del evento junto a un botón el cual te lleva al evento en cuestión

Por otra parte, la aplicación cuenta con un sistema de roles, que separa a los usuarios en dos tipos, “user” o “admin”, los “users” que se refiere a los usuarios registrados y ‘logados’, podrán realizar todas las acciones mencionadas anteriormente mientas que los “admin” además de las funcionalidades que tienen los “users” cuentas con una página extra llamada “AdminTool” en la cual podrán ver, editar o eliminar todas las propiedades de todos los usuarios y eventos existentes.

Por ultimo la aplicación cuenta con un evento en base de datos que se ejecuta cada una vez al día. Este evento comprueba la fecha de finalización de cada evento comparando con la fecha actual, en caso de la fecha de finalización del evento sea igual o menor a la fecha actual este me marcará como finalizado.

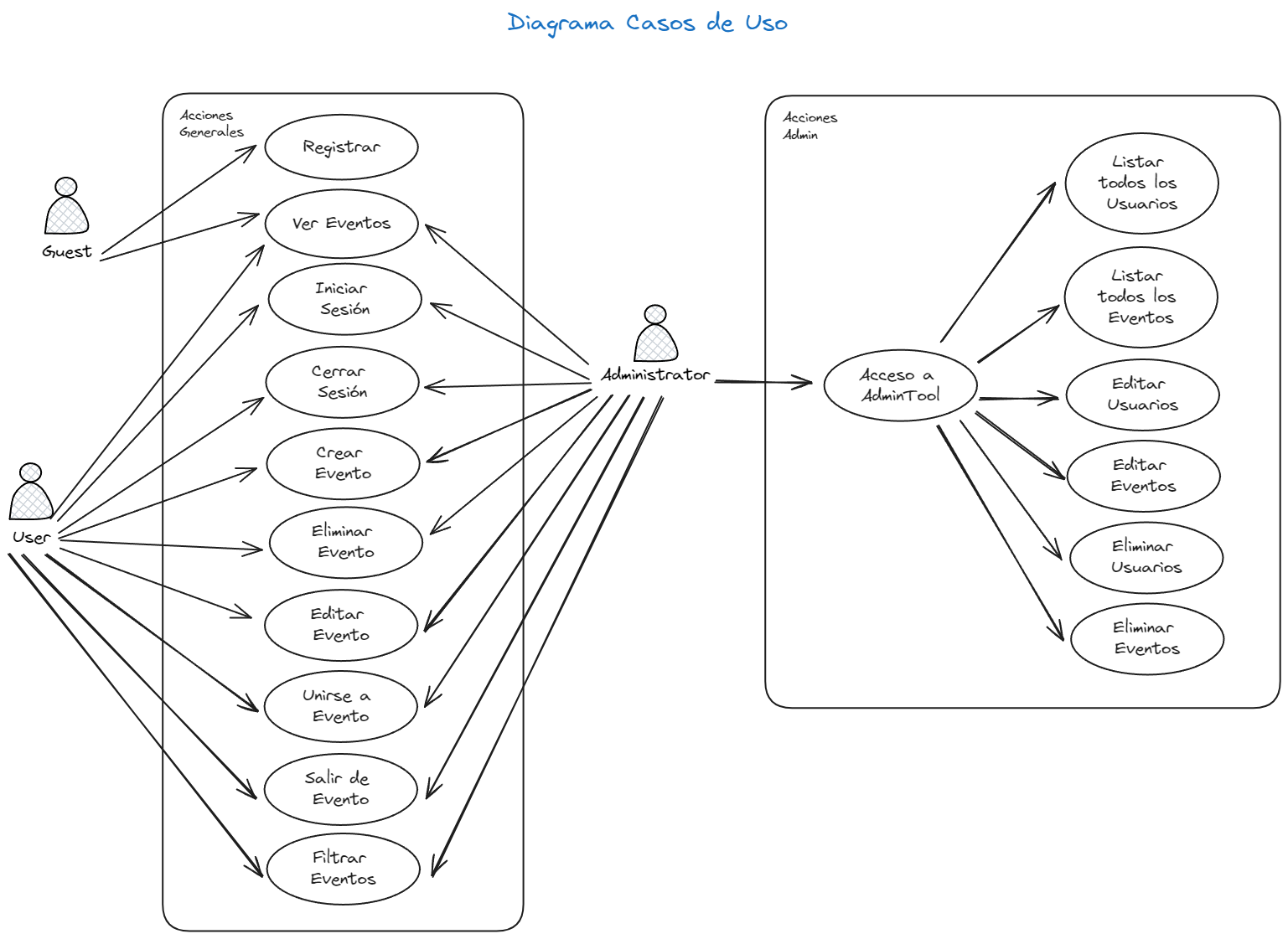
# Análisis de Diseño

## Diagrama de arquitectura

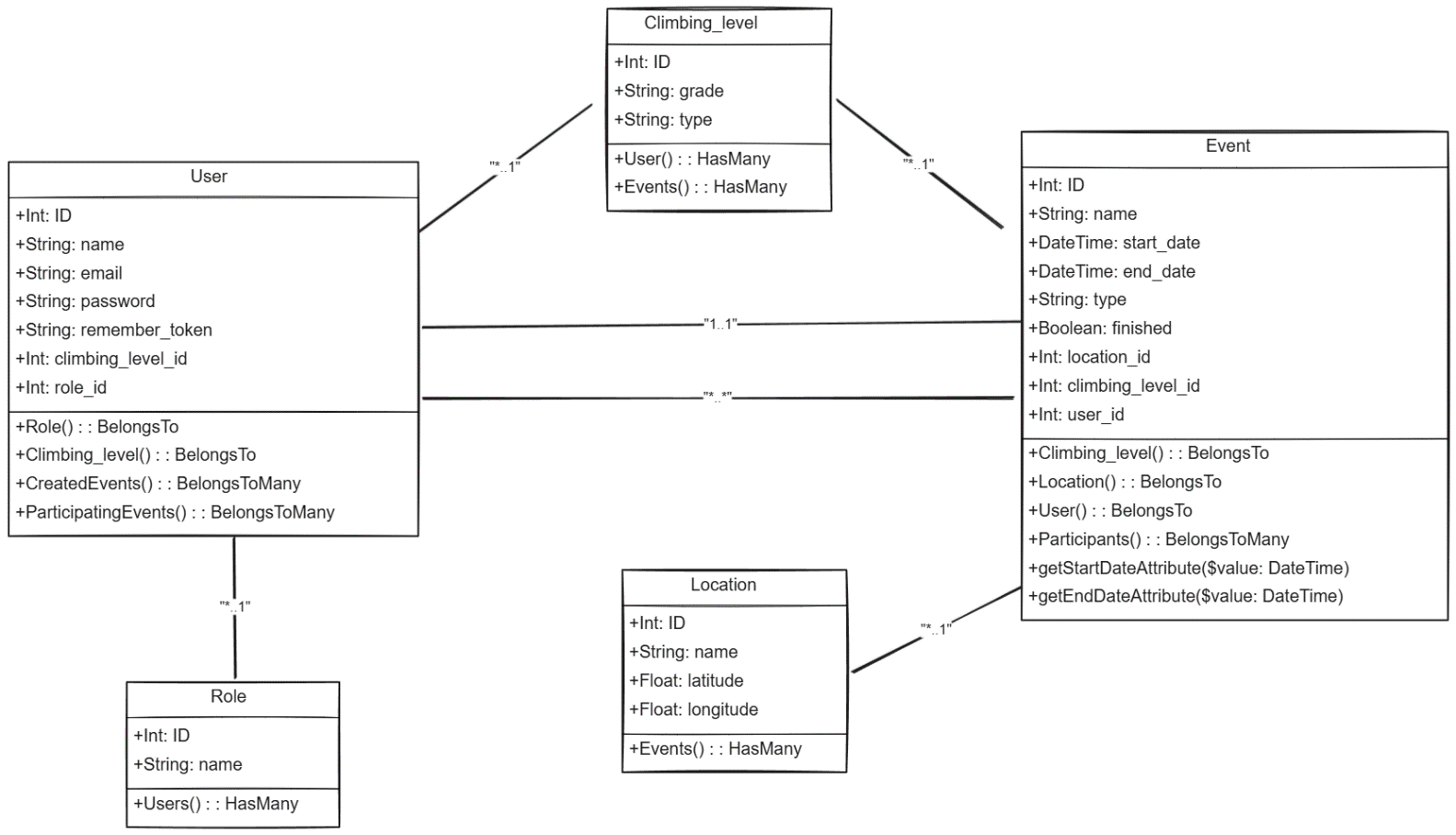
Diagrama

Descripción generada automáticamente

## Diagrama de casos de uso



## Diagrama de clases



## Diseño de datos

Diagrama

Descripción generada automáticamente

# Codificación

## Entorno de programación

Para el desarrollo de esta aplicación web se han utilizado Visual Studio Code y phpMyAdmin. A continuación se justifica la elección de estos entornos:

**Visual Studio Code:**

* **Versatilidad y Extensiones:** Visual Studio Code (VS Code) es conocido por su versatilidad y la amplia variedad de extensiones disponibles que pueden mejorar y personalizar el entorno de desarrollo. Esto incluye soporte para Laravel, React.js, y otras tecnologías relevantes para el proyecto.
* **Facilidad de Uso:** VS Code proporciona una interfaz intuitiva y fácil de usar, con características como la autocompletación de código, la integración de Git y un depurador integrado, que facilitan el proceso de desarrollo.
* **Popularidad y Comunidad:** VS Code es uno de los entornos de desarrollo más populares y cuenta con una gran comunidad de desarrolladores. Esto asegura una amplia disponibilidad de recursos, documentación y soporte.

**phpMyAdmin:**

* **Gestión de Bases de Datos:** phpMyAdmin es una herramienta web libre que permite la administración de bases de datos MySQL a través de un navegador web. Es ampliamente utilizado para realizar tareas como la creación, modificación y eliminación de bases de datos, tablas y registros.
* **Interfaz Gráfica:** Proporciona una interfaz gráfica que facilita la gestión de bases de datos sin necesidad de escribir comandos SQL manualmente, lo cual es especialmente útil para tareas de administración y mantenimiento.
* **Acceso Remoto:** Permite la gestión remota de bases de datos, lo cual es beneficioso para los desarrolladores que necesitan acceder y administrar la base de datos desde diferentes ubicaciones.

## Lenguaje y herramientas

Para este proyecto se han utilizado los siguientes lenguajes de programación:

* PHP
* JavaScript
* MySql
* HTML
* CSS

Los framewors son:

* Laravel
* React.js
* Tailwind

Herramientas utilizadas:

* Draw.io para diagramas y esquemas
* Excalidraw para diagramas y esquemas
* Google Maps Platform (Google Maps API)

Justificación de las Tecnologías Utilizadas

Laravel 10:

* Framework Robust: Laravel es un framework PHP conocido por su robustez y su amplia gama de características integradas que facilitan el desarrollo de aplicaciones web complejas. Laravel 10, siendo la última versión, proporciona mejoras en rendimiento, seguridad y nuevas funcionalidades.
* Ecosistema Rico: Laravel tiene un ecosistema rico que incluye herramientas como Eloquent ORM para la gestión de bases de datos, un sistema de enrutamiento elegante, y soporte para autenticación y autorización, lo cual simplifica enormemente el desarrollo de aplicaciones web.
* Comunidad y Soporte: Laravel cuenta con una gran comunidad de desarrolladores y una extensa documentación, lo cual facilita la resolución de problemas y la implementación de buenas prácticas.

Inertia.js y React.js:

* Aplicaciones Monolíticas Modernas: Inertia.js permite construir aplicaciones monolíticas modernas sin necesidad de separar completamente el frontend y el backend. Esto simplifica el desarrollo y despliegue, manteniendo la experiencia de una SPA (Single Page Application) sin la complejidad de gestionar una API separada.
* React.js para el Frontend: React.js es una biblioteca de JavaScript ampliamente utilizada para construir interfaces de usuario interactivas y dinámicas. Su enfoque basado en componentes facilita la creación y mantenimiento de interfaces complejas y reutilizables.
* Desempeño y Experiencia de Usuario: La combinación de Inertia.js y React.js permite crear una experiencia de usuario fluida y rápida, mejorando significativamente el desempeño de la aplicación al reducir la necesidad de recargas completas de página y proporcionando actualizaciones instantáneas de contenido.

## Aspectos relevantes de la implementación

Uno de los aspectos más relevantes de este proyecto es el uso de la API de Google Maps, que permite mostrar mapas para ubicar los eventos.

Al estar utilizando React.js se utiliza la dependencia "@vis.gl/react-google-maps" que no permite utilizar componentes predefinidos de React.js para así utilizar esta API. Los componentes que se utilizan son los siguientes:

* **APIProvider**: este junto con su prop “apiKey” nos permite utilizar la api key previamente solicitada en Google Maps Platform para poder hacer peticiones a la API de Google Maps
* **Map**: Este componente renderiza el propio mapay se utiliza dentro de APIProvider. Los **Props** que utilizan son los siguientes:
  + **mapId**: este se solicita en el Google Maps Platform para definir el estilo del mapa
  + defaultCenter: este admite un objeto literal el cual tiene como propiedad ‘lat’ y ‘lng’ para definir la latitud y longitud. Esta propiedad a diferencia de center permite mover el mapa.
  + **defaultZoom**: admite como parámetros números enteros, es decir un int para definir el zoom por defecto del mapa. Este a diferencia de zoom permite cambiar el zoom una vez renderizado el mapa.
  + **zoomControl**: admite booleano como parámetro y define si se puede hacer zoom o no en el mapa
  + **streetViewControl**: admite booleano como parámetro y habilita o deshabilita el Street view del mapa
  + **mapTypeControl**: admite booleano como parámetro y habilita o deshabilita cambiar entre el modo satélite del mapa
* **AdvancedMarker**: se trata de un marcadores que se pueden poner dentro del mapa. El prop que se utiliza es:
  + **Position**: admite como parámetro un objeto literal con las propiedades ‘lat’ y ‘lng’ para definir la latitud y longitud.
* **InfoWindow**: este componente es un popUp que se suele utilizar junto a un estado para definir si está abierto o no y la función onClick dentro AdvancedMarker para abrir este popUp
* **Pin**: no se utiliza en el proyecto, pero también es interesante ya que permite cambiar los estilos de los AdvancedMarker con las siguientes propiedades:
  + **Background**: define el color del fondo
  + **BorderColor**: define el color de borde
  + **glyphColor**: define el color del punto central
  + **Scale**: define el tamaño de del marcador

A continuación, un ejemplo de un componente que contiene todo lo visto anteriormente:

export default function MapMultipleMarkerComponent({

*events*,

*envApiKey*,

*envMapID*,

*userCoords*,

}) {

    const defaultLocation = {

        lat: 40.42579089487254,

        lng: -3.713155412568466,

    };

    const [open, setOpen] = useState({

        state: false,

        event: undefined,

    });

    const handleGoEvent = (*id*) => {

        Inertia.get(`/event/${parseInt(*id*)}`);

    }

    return (

        <APIProvider *apiKey*={*envApiKey*}>

            <div *className*="h-full w-full">

                <Map

*defaultZoom*={*userCoords* ? 9 : 5}

*defaultCenter*={*userCoords* ?? defaultLocation}

*mapId*={*envMapID*}

*zoomControl*={false}

*streetViewControl*={false}

*mapTypeControl*={false}

                >

                    {*events* &&

*events*.map((*event*) => (

                            <>

                                <AdvancedMarker

*position*={{

                                        lat: parseFloat(

*event*.location.latitude

                                        ),

                                        lng: parseFloat(

*event*.location.longitude

                                        ),

                                    }}

*onClick*={() => setOpen({

                                        state: true,

                                        event: *event*

                                    })}

                                />

                            </>

                        ))}

                    {open.state && (

                        <InfoWindow

*position*={{

                                lat: parseFloat(open.event.location.latitude),

                                lng: parseFloat(open.event.location.longitude),

                            }}

*onClose*={() => setOpen(false)}

                        >

                            <div *className*="text-gray-700 text-lg flex flex-col">

                                <p> {open.event.name} </p>

                                <p> {open.event.location.name} </p>

                                <p> {open.event.climbing\_level.grade} </p>

                                <button

*className*="p-3 bg-blue-700 rounded text-white underline"

*onClick*={() => handleGoEvent(open.event.id)}

                                >

                                    Go

                                </button>

                            </div>

                        </InfoWindow>

                    )}

                </Map>

            </div>

        </APIProvider>

    );

}

Otro aspecto relevante del proyecto es el uso de Accessors en el modelo de evento, estos permiten poner condiciones a la hora de solicitar datos en la base de datos, en este caso se han utilizado para modificar las fechas al formato deseado o así no tener que hacerlo de forma manual cada vez que se solicitan las fecha de inicio y de fin de los eventos.

El ejemplo de código es el siguiente:

*// Accessor para start\_date*

    public function getStartDateAttribute($value)

    {

        return Carbon::createFromFormat('Y-m-d H:i:s', $value)->format('d-m-Y H:i');

    }

*// Accessor para end\_date*

    public function getEndDateAttribute($value)

    {

        return Carbon::createFromFormat('Y-m-d H:i:s', $value)->format('d-m-Y H:i');

    }

Por ultimo tenemos un evento en la base de datos, este se ejecuta una vez al día y recorre todos los eventos en busca de eventos ya finalizados comparando la fecha de finalización de los eventos con la fecha actual, en caso de la fecha de finalización se igual a menos a la fecha actual cambia el atributo ‘finished’ de la tabla evento a ‘true’.

Código del evento:

CREATE DEFINER=`root`@`localhost`

EVENT `update\_finished\_events`

ON SCHEDULE EVERY 1 DAY

STARTS '2024-05-17 00:00:00'

ON COMPLETION PRESERVE

ENABLE DO

UPDATE event SET finished = TRUE

WHERE end\_date <= CURDATE() AND finished = FALSE

* CREATE DEFINER=root@localhost: Define el creador del evento, en este caso, el usuario root desde localhost.
* EVENT update\_finished\_events: Nombra el evento update\_finished\_events.
* ON SCHEDULE EVERY 1 DAY: Establece que el evento se ejecute cada día.
* STARTS '2024-05-17 00:00:00': Especifica la fecha y hora en que el evento comenzará a ejecutarse por primera vez.
* ON COMPLETION PRESERVE: Indica que el evento se conservará después de su ejecución. Esto significa que si el evento se completa, se mantendrá en la lista de eventos programados.
* ENABLE: Habilita el evento para que se ejecute según lo programado.
* DO UPDATE event SET finished = TRUE WHERE end\_date <= CURDATE() AND finished = FALSE: La acción que se ejecutará cuando se dispare el evento, que en este caso es actualizar la tabla event, estableciendo finished en TRUE para aquellos eventos cuyo end\_date sea menor o igual a la fecha actual y que aún no estén marcados como finished.

# Manual de usuario

## Registrarse

Captura de pantalla de computadora

Descripción generada automáticamente

Interfaz de usuario gráfica

Descripción generada automáticamente

## Logarse

Una captura de pantalla de un celular

Descripción generada automáticamente

## Modificar perfil

Captura de pantalla de un celular

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

## Eliminar perfil

Interfaz de usuario gráfica

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

## Des logarse

Captura de pantalla de un celular

Descripción generada automáticamente

## Ver Evento

Captura de pantalla de un celular

Descripción generada automáticamente

## Unirse a evento

## Salir de evento

## Listado de “joined events”

## Filtrado de eventos

## Mapa de Evento

# Requisitos de Instalación

## Software Necesario

### Visual Studio Code

Para instalar Visual Studio Code se ira a la la [página oficial](https://code.visualstudio.com/), para luego clicar en el botón que dice Download For Windos. Luego de descargarse el instalador se ejecutará y aparecerá un ventana donde se tendrá que marcar las dos primeras casillas en el aparado ‘Otros:’

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

### XAMPP

Para instalar XAMPP se ira a la [página oficial](https://www.apachefriends.org/es/index.html) y se clicara en el botón que dice descargar. Una vez descargado el instalador se clicará en el para iniciarlo, tras esto se abrirá una ventana de configuración donde tendremos que seleccionar los siguientes componentes.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

### Composer

Para instalar Composer se dirigirá a la [página oficial](https://getcomposer.org/) en la cual se hará clic en el enlace que dice Composer-Setup.exe ubicado en el primer apartado llamado Windows Installer. Una vez descargado y ejecutado seguiremos los pasos del instalador e indicaremos la ruta donde está instalado php que es la siguiente: C:\xampp\php\php.exe

### Laravel

Para instalar Laravel se ejecutará el siguiente comando en la consola de comandos: composer global require laravel/installer

### Node.js

Para instalar Node.js se ira a la [página oficial](https://nodejs.org/en), una vez ahí se descargará la última versión y ejecutará el instalador.

# Conclusiones

## Conclusiones sobre el trabajo realizado

## Posibles ampliaciones y mejoras

# Bibliografía y webgrafía

## Libros, artículos y apuntes

## Direcciones Web

# Apéndice, si procede