UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

REESTRUCTURACIÓN DE LA PLATAFORMA WEB DE TMR PARA ADAPTARSE A LOS DESARROLLOS FUTUROS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

FRANCO ANDRÉ MIRANDA OYARZÚN

PROFESORA GUÍA:
MARÍA CECILIA BASTARRICA PIÑEYRO

MIEMBROS DE LA COMISIÓN:
JOSÉ PINO URTUBIA
MARCEL AUGSBURGER BECERRA

SANTIAGO DE CHILE
2023

Resumen

La empresa TMR se dedica a prestar servicios de cronometraje de distintas disciplinas deportivas como descenso, running o motocross a organizaciones que realizan estos eventos.

TMR cuenta con una aplicación de escritorio y una aplicación web, que es en donde se centra este trabajo de memoria, conformada por el *editor* y los *Tiempos En Vivo*. El editor permite a los organizadores tener un control sobre sus eventos, y en los Tiempos En Vivo se permite a los espectadores visualizar los resultados parciales y finales de las competencias.

Con el fin de poder captar más clientes en el mercado, TMR desea mejorar su producto. La empresa busca tener una aplicación con una interfaz más moderna, similar a aplicaciones de la competencia, pero principalmente busca diferenciarse por funcionalidades específicas que otros no proporcionan.

La aplicación ha ido creciendo a lo largo de los años sin una estructura planificada y con una implementación que manejaba de forma conjunta la interfaz, la lógica de negocio y el almacenamiento de los datos. Es por ello que añadir una nueva funcionalidad o modificar alguna ya existente implicaba tanto un enorme trabajo como agregar complejidad, lo que hacía aún menos extensible la aplicación.

Este trabajo de título aborda la reestructuración completa de la plataforma web de TMR. Por un lado, se actualiza el backend para que siga una combinación de patrones arquitectónicos de microservicios y en capas. De esta forma, se desacopla el manejo de la interfaz del procesamiento de los servicios, y permite a su vez incorporar nuevas funcionalidades de forma sencilla y clara. Se ilustra esta característica con la extensión de la herramienta para tener la capacidad de manejar inscripciones, lo que hasta antes de la realización de este trabajo no existía. Por otro lado, se rediseñaron las interfaces de modo de ofrecer un aspecto más moderno e intuitivo para el usuario final.

La herramienta se encuentra actualmente en producción y ya se ha utilizado la funcionalidad del manejo de inscripciones en una competencia real. Se espera que dentro de las próximas semanas existan varios clientes que contraten los servicios de TMR, o pasen de la versión original a la versión desarrollada en este trabajo.

Dedicado a mi familia, amigos, y mi yo del pasado. Se logró.

Agradecimientos

Muchas gracias papá, mamá, Romina, Nona y Tata, por siempre ayudarme, darme el espacio para desenvolverme tranquilo en los estudios, aguantar mis manías y hacer los tremendos sacrificios por mi. También, gracias Bruno y Pelusa, por significar una enorme compañía y apoyo en mis años de adolescencia. Les mando saludos al cielo de los perritos.

Muchas gracias a mi colegio, y en particular a los profesores Rodrigo Ríos y Hugo Carrasco. Ustedes me inculcaron el amor por las matemáticas y por las ciencias, me hicieron notar que soy capaz de hacer lo que me proponga y me demostraron que los valores humanos son más importantes que una nota.

Muchas gracias a los chiquillos del IV°B, mis amigos del colegio y amigos de toda la vida. Sin ustedes definitivamente no sería quién soy, mi casa es su casa y celebro sus triunfos como si fuesen los míos. En particular, quiero mencionar al Jose, Nacho, Lucho y Kony. Tenemos mil historias y anécdotas juntos, y espero que sean mil historias más y que hasta el final estemos unidos.

Muchas gracias a los chiquillos de la play. Simplemente por haber entrado en un directo de la PS4 en 2015 empezó a formarse un gran grupo que se mantiene hasta el día de hoy. Gracias por ser como son, por estar siempre ahí y por aguantar mis gritos, enojos y cahuines interminables.

Muchas gracias a toda la gente que conocí durante estos años universitarios. Desde aquellos amigos de plan común, con los que siempre estudiábamos y jugábamos en la biblioteca, hasta los amigos del DCC, durante y después de la pandemia, con los que estuve innumerables horas por Discord haciendo trabajos o jugando. Aquí quiero hacer mención especial a la Monse, la Julia, el Cris, el Rodrigo, los chiquillos de Anakena FC y los de Proyecto de Software.

Muchas gracias a mi profesora guía Cecilia, por siempre darse el tiempo para ayudarme revisando mis avances y proponiendo mejoras.

Muchas gracias a todas aquellas personas con las que he conseguido entablar confianza para salir, jugar, conversar sobre la vida o los problemas que tenemos. También desde aquí agradecerle a todas esas personas que fueron parte de mi vida y que por distintas razones ya no lo son, quiero que sepan que me quedo con lo mejor de ustedes y que de todos aprendí algo nuevo. También son parte de este proceso.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Índice de Códigos

Capítulo 1

Introducción

1.1. Contexto

La empresa TMR [26] ofrece hace más de 10 años soluciones de cronometraje deportivo, enfocadas a eventos y entrenamientos en diversas disciplinas. La aplicación a través de TimerCrono, su línea de prestación de servicios, se ha consagrado a nivel sudamericano, pudiendo estar presente en importantes competencias del cono sur, como: Columbia Snow Challenge [25] en Valle Nevado, Epica Parque Nacional Torres del Paine XCM [29], Valparaíso Cerro Abajo [11] y, desde 2019, en su versión colombiana Monserrate Cerro Abajo [10].

Entre los productos ofrecidos, se encuentra un sistema de chips y transponders orientados a pruebas masivas como running o motocross. También, a través de fotoceldas de precisión se pueden cronometrar pruebas individuales como descenso o slalom. Todas las tecnologías antes mencionadas son completamente desarrolladas por TMR, y es por ello que es la empresa de soluciones tecnológicas más flexible y completa de Sudamérica.

Cuando un organizador de algún evento deportivo requiera los servicios y paga por ello, se le entregan las credenciales de la plataforma web conocida como editor o administrador web. Esta componente está enfocada en el organizador del evento, y entre sus características más importantes se encuentra un listado de eventos, con información como el nombre del evento, la disciplina a la que corresponde, la fecha de realización, entre otros, así como información sobre los corredores de cada evento, como nombre y apellido, dorsal, categoría, tiempo, etc. Toda la información antes nombrada se puede editar según las necesidades del organizador, de ahí el nombre de editor o administrador web.

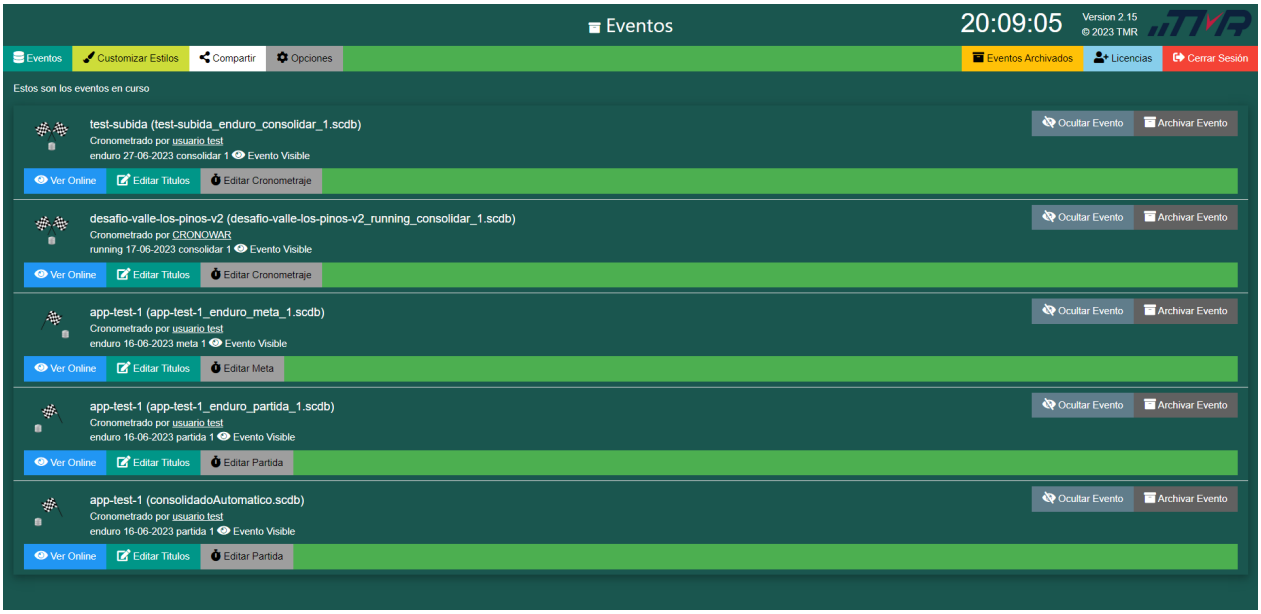


Figura 1.1: Interfaz principal del editor web

La figura 1.1 muestra la interfaz principal del editor web antes de realizar este trabajo de memoria. Aquí, un organizador puede ver todos sus eventos y cambiar su información, tanto la propia del evento como la de cada uno de los competidores.

Existe otra componente importante llamada Tiempos En Vivo, la cual está enfocada en el corredor y público general, en donde a medida los corredores van finalizando se van actualizando sus tiempos y están a disposición del público separados por categorías. De esta forma, solo con la URL se pueden consultar los tiempos de una manera sencilla e inmediata, lo que es muy valorado en el rubro, sobre todo por los organizadores de los eventos.

Posicion	Dor	Competidor	Categoria	Team	Pais	LAPS	Total
1	234	JHON_GHILLMAR ROJAS	CICLOTURISTA_30K	TODO_TERRENO	BOL	1	01:13:40.614
2	222	MARCO_ANTONIO ALVARADO_LA_FUENTE	CICLOTURISTA_30K	TEAM_TODO_TERRENO	BOL	1	01:14:11.974
3	218	ABSALON_JORDAN GARCIA	CICLOTURISTA_30K	MYALOE_BOLIVIA	BOL	1	01:17:54.079
4	213	JAVIER MONTECINOS_DELGADILLO	CICLOTURISTA_30K	TEAM_CER_2	BOL	1	01:18:40.442
5	215	PABLO MALDONADO_MARANON	CICLOTURISTA_30K	INDEPENDIENTE	BOL	1	01:18:41.861
6	232	LUIS_ENRIQUE NORIEGA_LENS	CICLOTURISTA_30K	AGUA_RICA	BOL	1	01:22:17.953
7	216	FERNANDO VARGAS	CICLOTURISTA_30K	TEAM_TODO_TERRENO	BOL	1	01:22:57.040
8	224	JORGE_ALEJANDRO SEJAS_NUMBELA	CICLOTURISTA_30K	TEAM_AVENTURA	BOL	1	01:23:32.626
9	210	JIMMY VISCARRA_ATTALA	CICLOTURISTA_30K	T4_TEAM	BOL	1	01:25:17.221
10	227	FREDDY_HERLAN ROJAS_ARAMAYO	CICLOTURISTA_30K	TEAM_CERO_EMISIONES	BOL	1	01:25:21.706
11	228	SANTOS_HUMBERTO LIMA_CONDE	CICLOTURISTA_30K	TEAM_CERO_EMISIONES	BOL	1	01:25:30.242
12	220	MAURICIO VASQUEZ	CICLOTURISTA_30K	COPELME_TEAM	BOL	1	01:26:09.680

Figura 1.2: Interfaz de los Tiempos En Vivo

En la figura 1.2 se muestra la interfaz de los Tiempos En Vivo. Los asistentes a un evento pueden acceder a esta vista mediante un link y ver en tiempo real los resultados de cualquier corredor en cualquier categoría. La información se divide en dos tablas: la tabla de posicionamiento y la tabla de rezagados. La tabla de posicionamiento se utiliza para conocer los lugares en que finalizó cada corredor en la competición, mientras que la tabla de rezagados muestra la información de todos los corredores, incluso de aquellos que no están rankeados debido a descalificación u otros motivos.

1.2. Problema

TMR ha funcionado por más de 10 años utilizando la herramienta con la arquitectura actual que posee. Sin embargo, el modelo de datos se está implementando junto al procesamiento de los datos y las interfaces de usuario, por lo que no permite que los desarrollos se realicen de manera eficiente ni que el código se pueda mantener, ya que cada servicio se implementa de diferente manera y no se mantiene un orden fijo.

Lo nombrado anteriormente implica riesgos para el negocio. Por ejemplo, no se pueden agregar nuevas funcionalidades de manera correcta ni corregir fallos críticos en un corto tiempo, que a su vez puede provocar errores al mostrar los resultados de una competición. Esto, en el peor caso, puede significar la pérdida de un cliente y que la reputación de la empresa empeore.

Debido a lo desarrollado anteriormente es que es necesario reestructurar la plataforma, adquiriendo una nueva arquitectura que permita el desarrollo de nuevas funcionalidades con un código que sea mantenible.

1.3. Objetivos

El objetivo general de este trabajo de memoria es realizar una reestructuración de la plataforma web de TMR, para que ésta deje de ser un cuello de botella para el crecimiento y diversificación de la empresa.

El trabajo a realizar se puede dividir en tres objetivos específicos:

- *Desacoplar el monolito actual.* Separar las peticiones del usuario, el procesamiento de los datos y las operaciones en base de datos. Con esto, se optimizarían los tiempos de desarrollo y la plataforma sería más mantenible y extensible. Este desarrollo se debe realizar manteniendo operativos los servicios actuales de TMR, ya que la empresa mantiene compromisos permanentes con las organizaciones de diversas competencias, por lo que el servicio entregado no puede interrumpirse.
- *Rediseñar completamente la plataforma web.* Es necesario reestructurar la plataforma actual tanto en el frontend como en el backend. En el frontend, adoptar un diseño

más moderno e intuitivo para los clientes. En el backend, sumado al objetivo anterior, adaptar las respuestas de los servicios a lo requerido en el nuevo frontend a desarrollar.

- *Crear una nueva funcionalidad que le entregue valor a la empresa: manejo de inscripciones.* Tomando en cuenta que para cada evento cronometrado por TMR participan decenas de corredores, que la inscripción de cada uno de ellos sea manejada por TMR puede generar enormes ganancias. Por ello, una vez reestructurada la plataforma, se implementará una funcionalidad completamente nueva que permita manejar las inscripciones.

Capítulo 2

Estado del Arte

2.1. Otras soluciones de cronometraje

Actualmente, existen otras empresas que ofrecen servicios de cronometraje deportivo en Latinoamérica, como lo son Cronosur [3], Race Timing [7] y CronoSystem [4]. Algunas diferencias a nivel general de las soluciones de estas empresas respecto a la solución ofrecida por TMR son:

- Sus soluciones están desarrolladas en el exterior, por lo que ninguna tiene implementada su propia tecnología.
- Es debido a lo anterior que sus modelos de negocios se basan en comprar tanto el hardware como el software y prestar los servicios, mientras que TMR desarrolla el hardware y el software.
- Son representadas por marcas de cronometraje extranjeras como MyLaps [6] o AlgeTiming [1], mientras que TMR es representada por empresas chilenas como CORFO [2] o FabLab [5].
- Se especializan en ciertas disciplinas mientras que TMR no, a cambio de tener un mayor espectro de disciplinas que son posibles de cronometrar.

La siguiente tabla resume lo recién mencionado:

	TMR	**Otras soluciones**
Desarrollo hecho en	Chile	Extranjero
Modelo de negocios	Desarrollar hardware y software y prestar servicios	Comprar hardware y software y prestar servicios
Representaciones	Empresas chilenas	Empresas extranjeras
Especialización	No	Sí

Tabla 2.1: Comparación entre TMR y otras soluciones de cronometraje latinoamericanas

Al entrar a los sitios web de las empresas antes mencionadas e ingresar a sus respectivas secciones de resultados, se puede notar que en prácticamente todas ellas se tiene una interfaz moderna y sencilla de usar para el usuario, sobre todo pensando en el público asistente a estos eventos. También, en cada una de ellas se puede filtrar por categoría o género y visualizar los datos de esos competidores. Por último, en algunos de los sitios es posible visualizar información histórica de cada corredor, ya sea en tablas o gráficos, mostrando, por ejemplo, sus posiciones en carreras anteriores, o el cambio en su posición en cada una de las etapas de una carrera. Algunas imágenes de los sitios mencionados se muestran en las figuras A.1 y A.2 del apéndice A.

2.2. Patrones de arquitectura de software

En este subcapítulo, se desarrollará lo que es un patrón de arquitectura de software, y se mostrarán algunos tipos que existen con sus respectivas ventajas y desventajas.

Los patrones de arquitectura de software son formas de capturar estructuras de diseño de probada eficacia, para que puedan ser reutilizadas. Cada patrón define los componentes que llevan a cabo alguna tarea de computación, sus interfaces y la comunicación entre ellos.

Las aplicaciones que no poseen una arquitectura formal generalmente son complicadas de cambiar. Como resultado, es muy difícil determinar las características arquitecturales de una aplicación sin entender completamente el trabajo de cada componente y módulo en el sistema, además de no tener claridad sobre el mantenimiento y el despliegue.

Existen 5 patrones principales de arquitectura de software [24], los cuales se desarrollarán en los siguientes subcapítulos.

2.2.1. Microkernel

Consiste en dos tipos de componentes: un sistema central y plugins. El sistema central contiene solo la mínima funcionalidad requerida para el funcionamiento del sistema. Los plugins son módulos independientes que se especializan en cierto procesamiento o en características adicionales, lo que entrega extensibilidad, flexibilidad y aislamiento de las características específicas de la aplicación. Un esquema representativo de este patrón se encuentra en la siguiente figura:

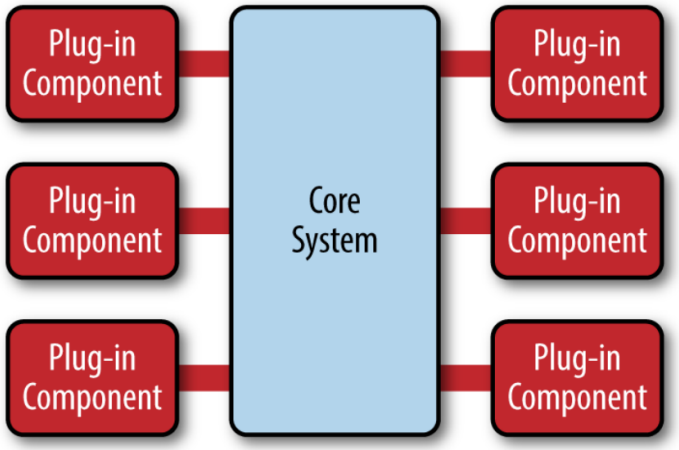


Figura 2.1: Esquema del patrón de arquitectura Microkernel. Fuente: Software Architecture Patterns (p. 22), por M. Richards

Se utiliza principalmente cuando se desarrollan sistemas con componentes intercambiables, como por ejemplo, aquellas aplicaciones basadas en productos, es decir, aquellas que están empaquetadas y disponibles para su posterior descarga.

Un ejemplo de uso de esta arquitectura son los editores de código. Las descargas de éstos proveen un software básico, pero a medida se les agregan distintos plugins, se convierten en un producto altamente personalizable y útil.

- **Ventajas**
  - *Facilidad de despliegue.* Dependiendo de cómo fue implementado el patrón, los plugins pueden ser agregados dinámicamente en tiempo de ejecución.
  - *Alta agilidad general.* Cada cambio puede ser desarrollado rápidamente de manera aislada a través de pequeños plugins.
  - *Facilidad de testeo.* Cada módulo puede ser testeado de manera independiente.
  - *Alto rendimiento.* Se pueden incluir solo las características que se necesitan en cada caso.
- **Desventajas**
  - *Poca escalabilidad.* Como las implementaciones de este patrón de arquitectura generalmente son pequeñas, son desarrolladas como una única unidad, y por lo tanto, no son muy escalables.
  - *Dificultad en el desarrollo.* Requiere un diseño precavido y gobernanza de contratos, haciendo que sea más complejo de implementar.

2.2.2. Microservicios

Este patrón de arquitectura consiste en distintos componentes, donde cada uno de ellos comprende servicios disponibles para comunicarse a pedido. Estos componentes son independientes entre sí, es decir, operan de manera independiente uno del otro, y por lo tanto, al actualizar uno de ellos no existe un impacto en otros componentes o servicios.

Existen muchas formas de implementar este patrón, siendo una de ellas la topología basada en API REST. Esta topología consiste en servicios pequeños que contienen pocos módulos, donde cada uno de ellos tiene su propia lógica de negocio, independiente del resto de servicios. Cada servicio es accedido mediante una interfaz REST. Es útil para sitios web que exponen pequeños servicios individuales en una API. Un esquema de esta topología se encuentra en la siguiente figura:

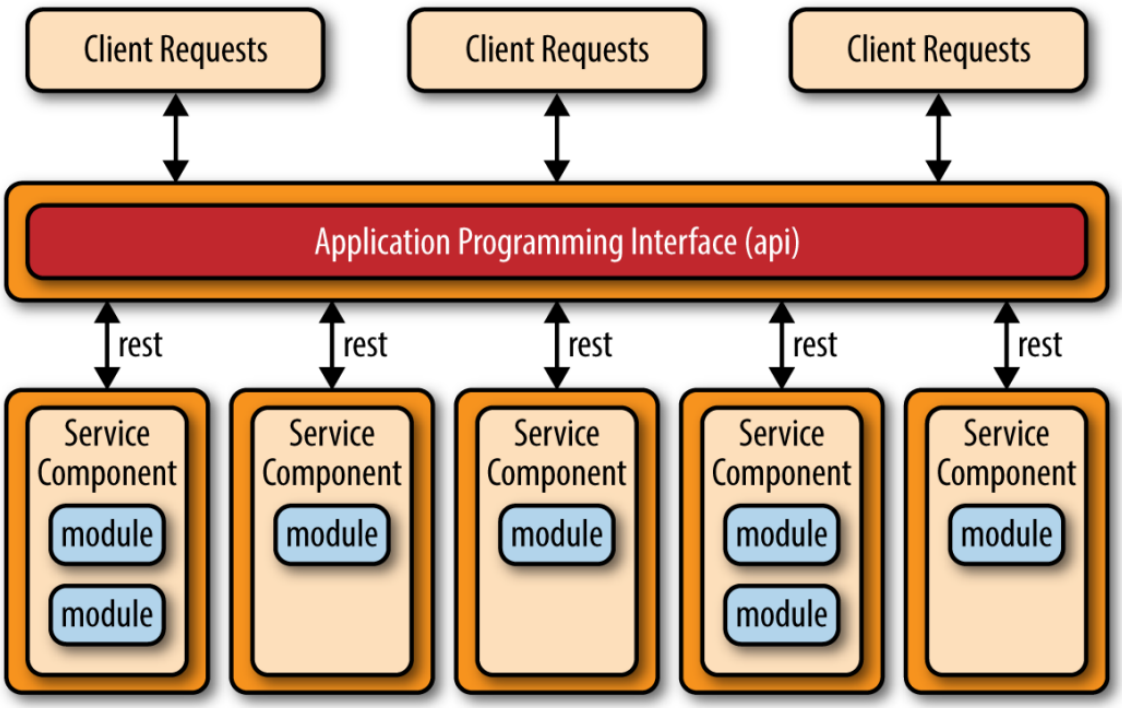


Figura 2.2: Topología basada en API REST del patrón de arquitectura Microservicios. Fuente: Software Architecture Patterns (p. 30), por M. Richards

Se suele utilizar en sitios web con pequeños componentes, en centros de datos corporativos, o para desarrollar rápidamente nuevos negocios y aplicaciones web.

- **Ventajas**
  - *Facilidad de despliegue.* Como los servicios son desplegados como unidades de software separadas, resulta sencillo realizar 'hot deployments'.
  - *Alta agilidad general.* Cada cambio a realizar en un servicio se encuentra completamente aislado del resto, por lo que es factible adaptarse a los cambios en el

entorno.

  - *Facilidad de testeo.* Como cada servicio está implementado de forma independiente, el testeo se puede realizar de manera exhaustiva para cada uno de ellos.
  - *Alta escalabilidad.* Cada servicio puede ser escalado de manera independiente según sea requerido.
  - *Facilidad en el desarrollo.* Al estar cada funcionalidad aislada en distintos servicios independientes, el desarrollo es más pequeño y con un enfoque aislado.

- **Desventajas**
  - *Bajo rendimiento.* En general, este patrón no suele utilizarse en aplicaciones de alto rendimiento debido a la naturaleza distribuida de los microservicios.

2.2.3. En capas

Es el patrón de arquitectura de software más común. Se caracteriza por tener sus componentes organizados en distintas capas horizontales, donde cada una de ellas realiza un trabajo en específico. En general, consiste en cuatro capas: presentación, negocio, persistencia y datos, aunque en algunos casos las capas de negocio y de persistencia se combinan en una sola capa de negocios.

La capa de presentación maneja lo relacionado con la interfaz de usuario y la comunicación con el navegador. La capa de negocios se encarga de ejecutar funciones específicas relacionadas con las peticiones hechas. La capa de datos solo almacena la información. Un esquema de este patrón se encuentra en la siguiente figura:

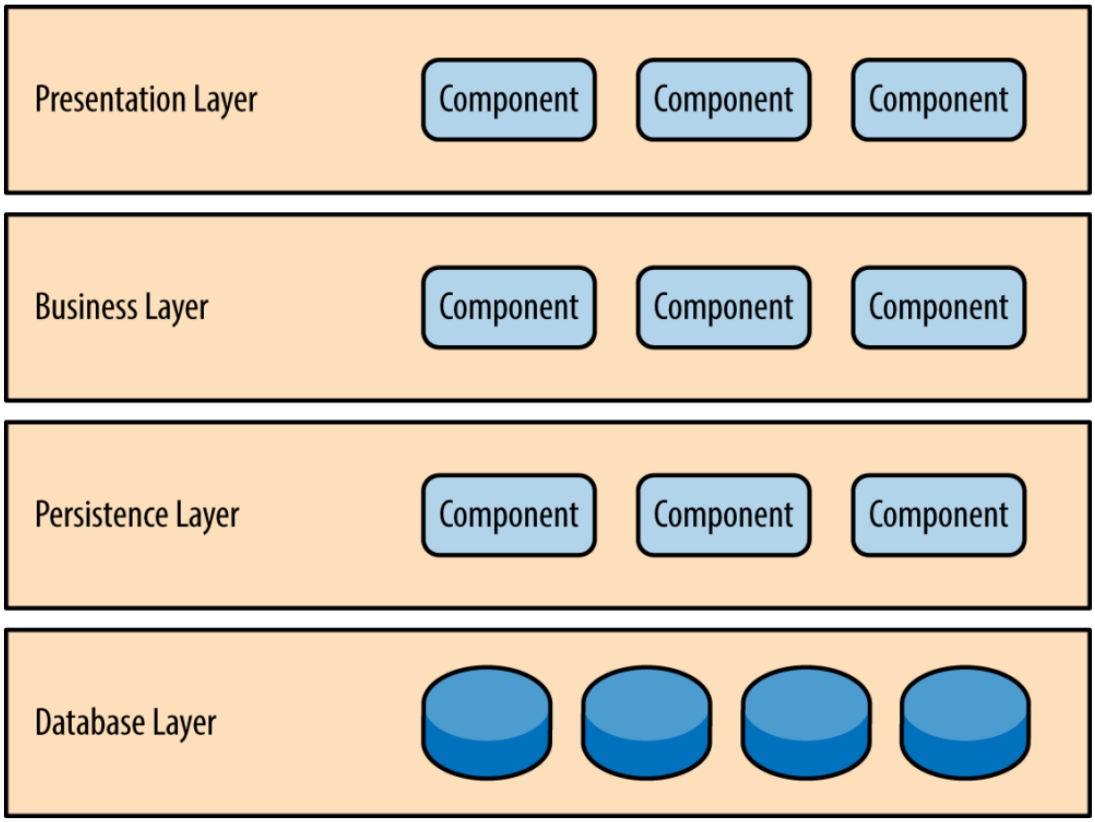


Figura 2.3: Esquema del patrón de arquitectura en capas. Fuente: Software Architecture Patterns (p. 2), por M. Richards

Se utiliza principalmente en aplicaciones que necesitan ser construidas rápidamente o en aquellas que requieran normas estrictas de mantenimiento. También, le puede ser útil a equipos inexpertos debido a la simplicidad del patrón.

- **Ventajas**
  - *Facilidad de testeo.* Como cada componente pertenece a capas específicas en la arquitectura, el resto de capas se puede ignorar, por lo que el testeo se vuelve sencillo.
  - *Facilidad en el desarrollo.* Muchas compañías desarrollan aplicaciones separando conjuntos de habilidades por capas, por lo que este patrón se vuelve una opción natural.
- **Desventajas**
  - *Dificultad en el despliegue.* Un pequeño cambio en un componente puede requerir volver a desplegar la aplicación completa, por lo que se requieren despliegues que se ejecuten fuera de hora o en fines de semana.
  - *Baja agilidad general.* Consume mucho tiempo hacer cambios con este patrón de arquitectura debido a la naturaleza monolítica de la mayoría de sus implementaciones.

- *Bajo rendimiento.* Se deben pasar por múltiples capas para cumplir con una solicitud, lo que disminuye el desempeño considerablemente.
- *Poca escalabilidad.* Esta arquitectura se puede escalar replicando la aplicación completa en múltiples nodos, lo que lo hace una operación cara.

2.2.4. Basado en eventos

Este patrón consiste en dos topologías principales: el mediador y el corredor.

La topología del mediador se utiliza cuando se requiere de muchos pasos dentro de un evento. Está compuesto por cuatro tipos de componentes: una cola de eventos, un mediador, canales de eventos y procesadores de eventos. En primer lugar, se envía un evento a la cola de eventos. Luego, el mediador recibe el evento y envía otros eventos asíncronos a los canales. Los canales son escuchados por los procesadores de eventos, y éstos reciben el evento y ejecutan la lógica de negocio. Un esquema de esta topología se muestra a continuación:

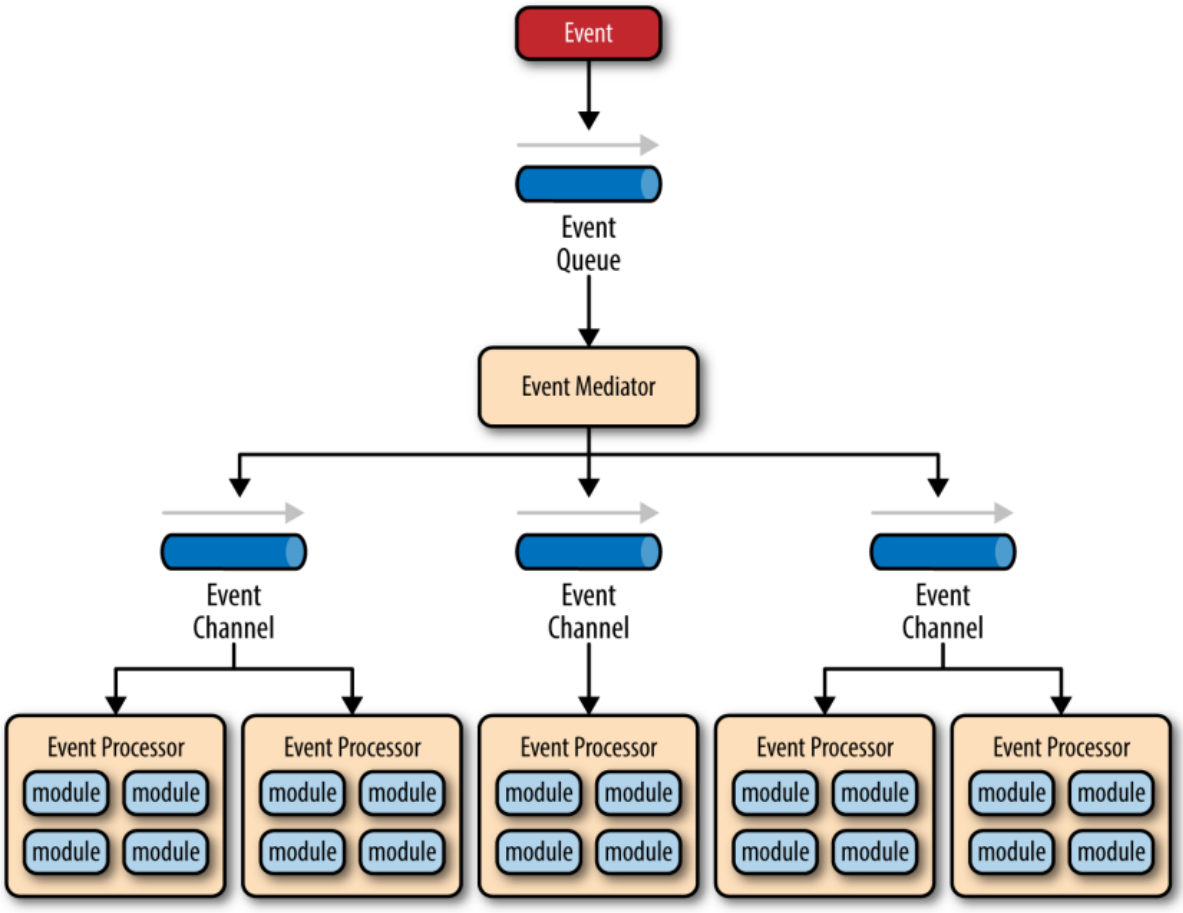


Figura 2.4: Topología del mediador del patrón de arquitectura basado en eventos. Fuente: Software Architecture Patterns (p. 12), por M. Richards

La topología del corredor se diferencia de la del mediador en que no existe un mediador de eventos. Está compuesto por dos tipos de componentes principales: el corredor y el procesador de eventos. El corredor contiene todos los canales de eventos que son usados en el flujo, y es en estos canales en donde el procesador de eventos procesa y publica un nuevo evento indicando que la acción se realizó. Un esquema de esta topología se encuentra en la siguiente figura:

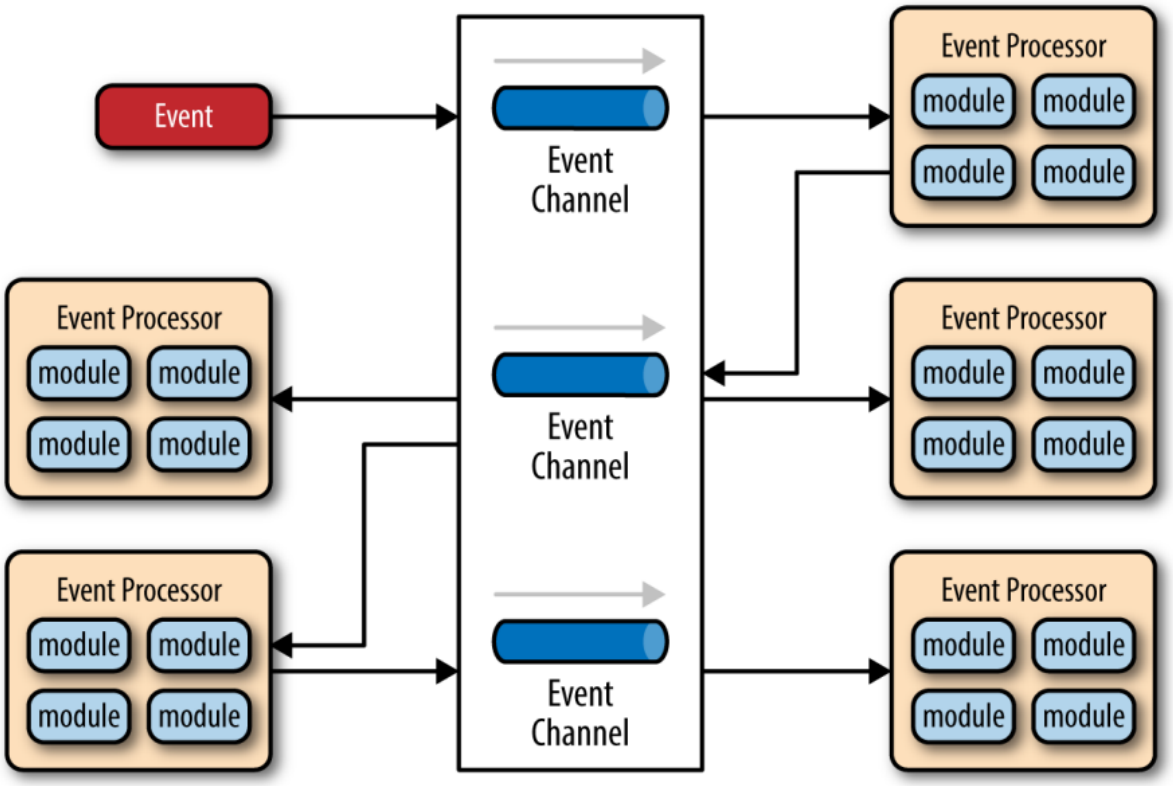


Figura 2.5: Topología del corredor del patrón de arquitectura basado en eventos. Fuente: Software Architecture Patterns (p. 16), por M. Richards

Este patrón de arquitectura de software se suele utilizar en sistemas con un flujo de datos asíncrono y en interfaces de usuario.

- **Ventajas**
  - *Facilidad de despliegue.* En general, con este patrón los despliegues resultan sencillos, debido a la naturaleza desacoplada de cada procesador de eventos.
  - *Alta agilidad general.* Como cada componente tiene un solo objetivo y está completamente desacoplado del resto, los cambios a realizar se encuentran aislados a uno o pocos procesadores de eventos.
  - *Alto rendimiento.* En general, se consigue alto rendimiento debido a la capacidad de realizar operaciones asíncronas paralelamente.

  - *Alta escalabilidad.* Cada procesador de eventos puede ser escalado de manera separada.

- **Desventajas**
  - *Dificultad de testeo.* Debido a la naturaleza asíncrona de este patrón, se requiere de testeo especializado.
  - *Dificultad en el desarrollo.* El desarrollo se complica debido tanto a la naturaleza asíncrona como a que se requiere un avanzado manejo de errores.

2.2.5. Basado en el espacio o en la nube

Obtiene su nombre a partir de la idea de la memoria compartida distribuida. Este patrón minimiza los factores que limitan el escalado de aplicaciones, lo que se consigue eliminando la restricción de tener una base de datos central y utilizando cuadrículas de datos replicados en memoria. Está diseñada para evitar el colapso funcional bajo una gran carga al dividir tanto el procesamiento como el almacenamiento entre varios servidores.

Este patrón está compuesto por dos componentes: una unidad de procesamiento y un middleware virtualizado.

La unidad de procesamiento contiene los componentes de la aplicación y su contenido depende del tipo de aplicación, ya que aplicaciones pequeñas pueden estar desplegadas en una sola unidad de procesamiento, mientras que en aplicaciones más complejas sus funcionalidades pueden estar divididas en varias unidades. El middleware virtualizado se encarga de las comunicaciones y la limpieza. Contiene componentes que controlan aspectos como la sincronización de la información y el manejo de peticiones. Un esquema de este patrón de arquitectura se muestra a continuación:

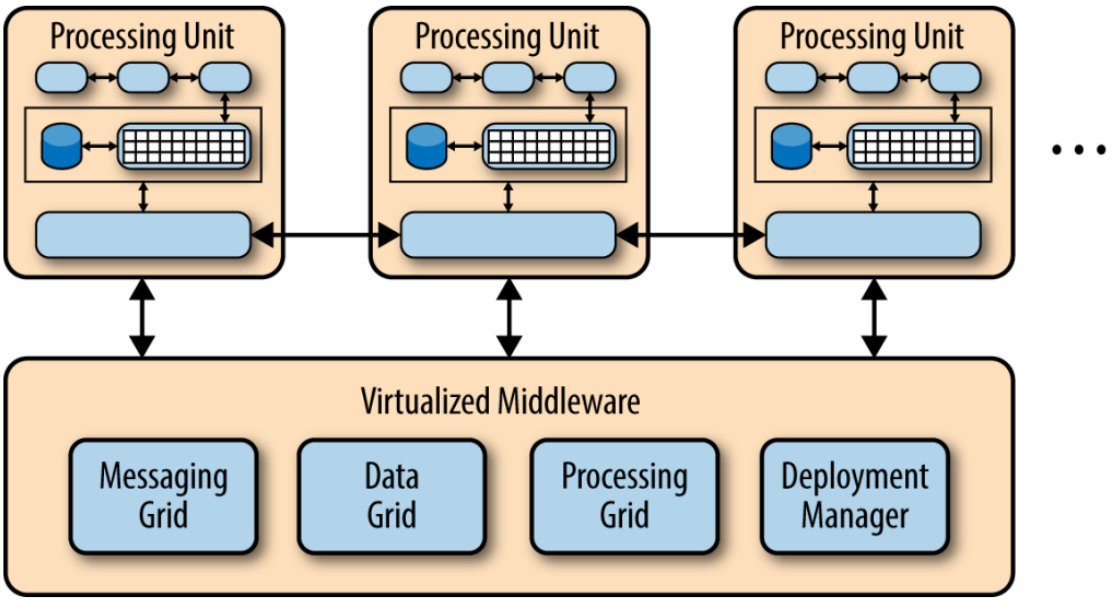


Figura 2.6: Esquema del patrón de arquitectura basado en el espacio o en la nube. Fuente: Software Architecture Patterns (p. 39), por M. Richards

La mayoría de aplicaciones en las que se utiliza este patrón son sitios web que reciben una petición desde un navegador y ejecutan alguna acción, como por ejemplo un sitio de subastas, en donde continuamente se reciben pujas por parte de los usuarios.

- **Ventajas**
  - *Facilidad de despliegue.* Las herramientas basadas en la nube permiten a las aplicaciones ser fácilmente subidas a los servidores, simplificando el despliegue.
  - *Alta agilidad general.* Como las unidades de procesamiento pueden subirse y bajarse de un servidor rápidamente, las aplicaciones responden bien a cambios relacionados con la carga de usuarios.
  - *Alto rendimiento.* Este rendimiento se alcanza mediante el acceso a los datos en memoria y al almacenamiento de datos en caché.
  - *Alta escalabilidad.* Se depende muy poco de una base de datos centralizada.
- **Desventajas**
  - *Dificultad de testeo.* Conseguir una alta carga de usuarios en ambiente de testeo es caro y consume mucho tiempo.
  - *Dificultad en el desarrollo.* No existe mucha familiarización con las herramientas y productos usados para crear este tipo de arquitectura.

2.3. Tecnologías

En este subcapítulo, se mostrarán algunas de las tecnologías más utilizadas para el desarrollo de aplicaciones web. Éstas varían según las necesidades de cada proyecto, sin embargo, existen tecnologías ampliamente adoptadas y populares. A continuación, se muestran algunas de ellas:

- *Frontend.* Para el desarrollo del lado del cliente se pueden utilizar tecnologías como:
  - *HTML* [9]. Es el lenguaje de marcado para crear la estructura y contenido de las páginas web.
  - *CSS* [21]. Se utiliza para dar estilo y diseñar las páginas web.
  - *JavaScript* [15]. Lenguaje de programación que permite agregar interactividad y funcionalidad a las páginas web.
- *Backend.* Para el desarrollo del lado del servidor se pueden utilizar tecnologías como:
  - *Node.js* [13]. Entorno de tiempo de ejecución de JavaScript que permite ejecutar JavaScript en el lado del servidor.
  - *Python* [28]. Lenguaje de programación interpretado, dinámico y multiplataforma, ampliamente utilizado en aplicaciones web y desarrollo de software.
  - *PHP* [20]. Lenguaje de programación interpretado del lado del servidor y adaptado especialmente al desarrollo web.
- *Frameworks.* Proporcionan una estructura y conjunto de herramientas para facilitar el desarrollo web. Algunos ejemplos son:
  - *React* [22]. Framework de JavaScript para construir interfaces de usuario interactivas. Se enfoca en la creación de componentes reutilizables.
  - *Laravel* [23]. Framework de PHP que sigue el patrón Modelo-Vista-Controlador y tiene características que permiten agilizar el desarrollo web.
  - *Django* [32]. Framework de Python que sigue el patrón Modelo-Vista-Controlador. Enfocado en la eficiencia y seguridad.
  - *Bootstrap* [27]. Framework de CSS utilizado para facilitar y acelerar el proceso de diseño y desarrollo de aplicaciones web, proporcionando herramientas que permiten construir interfaces responsivas y atractivas.
- *Bases de datos.* Es necesario tenerlas para almacenar información. Algunos sistemas de gestión de bases de datos son:
  - *PostgreSQL* [30]. Sistema de gestión de bases de datos relacional de alto rendimiento y robusto.
  - *MariaDB* [12]. Sistema de gestión de bases de datos relacional de código abierto. Ofrece seguridad y soporte para múltiples plataformas.
  - *SQLite* [19]. Sistema de gestión de bases de datos relacional de código abierto y ligero. Destaca por su simplicidad, portabilidad y eficiencia.
  - *MongoDB* [16]. Sistema de gestión de base de datos no relacional orientada a documentos que proporciona flexibilidad y escalabilidad.

Capítulo 3

Solución

3.1. Situación inicial

La aplicación web, como ya fue explicado anteriormente, está compuesta por el editor y los Tiempos En Vivo. El editor web consta de distintas interfaces y modales, los cuales se nombran a continuación junto a su funcionalidad:

- **Interfaces**
  - *Eventos.* Contiene todos los eventos actuales de un usuario.
  - *Eventos Archivados.* Contiene todos los eventos archivados por un usuario y que no se muestran al público.
  - *Personalización de Estilos.* Permite editar la apariencia de los Tiempos En Vivo.
  - *Licencias.* Acá un usuario puede adquirir alguna de las licencias ofrecidas por TMR.
  - *Editar Cronometraje.* En esta interfaz se muestra una tabla con la información de todos los corredores, como sus datos personales y los tiempos realizados en la competición.
- **Modales**
  - *Compartir.* Aquí se muestra el código QR del usuario y un embed, ambos correspondientes a los Tiempos En Vivo.
  - *Opciones.* Se pueden configurar las distintas opciones del usuario, como si se quiere mostrar la tabla de rezagados en los Tiempos En Vivo, seleccionar la zona horaria, o personalizar los nombres de algunos campos y ocultar otros.
  - *Editar Títulos.* Aquí se puede editar la información de un evento, como su nombre, fecha de realización o responsable, además de poder escribir un mensaje, el cual se mostrará al público.
  - *Editar Cronometraje.* A este modal se accede al hacer clic sobre alguna fila de la tabla de la interfaz de edición de cronometraje, y se puede editar la información del corredor en el evento.

Todo el frontend antes mencionado está hecho a partir de imprimir strings PHP. Desde aquí se hacen peticiones a los servicios del backend. Estos hacen una operación en base de datos y entregan una respuesta, para luego ser procesada en el frontend.

Algunos de los servicios del backend se nombran a continuación. Todos ellos retornan la información en texto plano:

- *Autenticación.* Iniciar y cerrar sesión.
- *Personalización del usuario.* Personalización de estilos para los Tiempos En Vivo.
- *Opciones del usuario.* Zona horaria, nombres personalizados de campos y campos a ocultar en los Tiempos En Vivo, seleccionar si se muestra la tabla de rezagados en los Tiempos En Vivo, entre otros.
- *Eventos.* Crear, actualizar y eliminar un evento, obtener y editar la información de todos los eventos de un usuario, obtener y editar la información de los corredores de un evento, entre otros.

Internamente, la aplicación está construida de forma monolítica, ya que las capas de interfaz de usuario, lógica de negocios y acceso a datos están sobre la misma plataforma. Es por esto que es necesario separar cada capa, ya que el frontend y el backend al estar combinados en un mismo programa se vuelve complicado incorporar nuevas funcionalidades y que el código sea mantenible en el tiempo.

El siguiente diagrama muestra la estructura inicial de la aplicación:

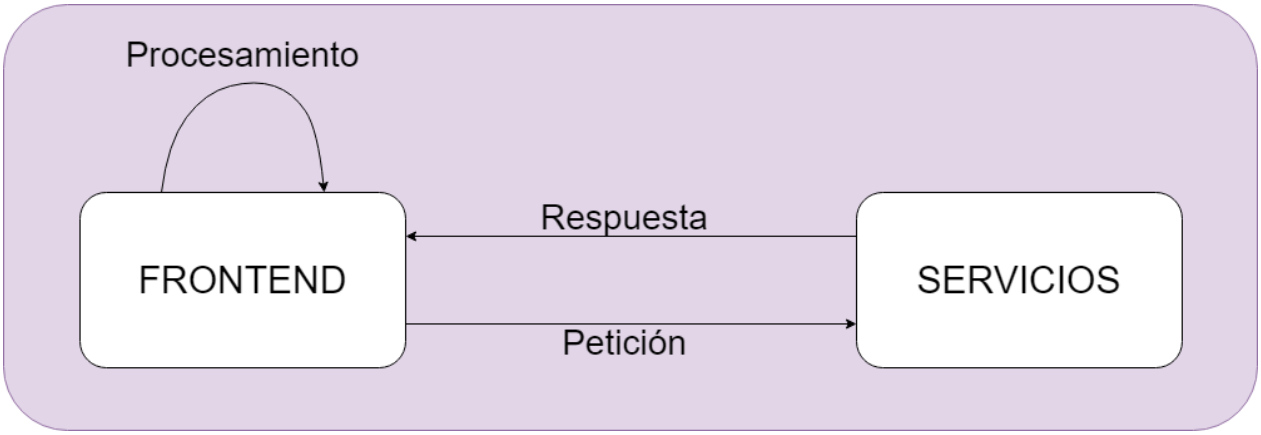


Figura 3.1: Estructura inicial de la aplicación

El modelo de datos se divide en dos partes: archivos SQLite y una base de datos MariaDB gestionada desde PHPMyAdmin [17]. En los archivos SQLite se encuentra información sobre los competidores, los eventos, disciplinas, tiempos, entre otros. En la base de datos MariaDB se encuentra información sobre los usuarios que tienen acceso al editor web, sus opciones de personalización y parámetros tales como su zona horaria. No existe ninguna relación entre entidades.

Una vez explicada cómo se encuentra la aplicación actualmente, ahora se desarrollará en detalle el trabajo realizado.

3.2. Desacoplar el monolito actual

Para cumplir con este objetivo, en primer lugar se debe definir qué patrón de arquitectura de software utilizar. Para esta aplicación, lo más requerido es que exista:

- *Facilidad en el desarrollo.* Se debe tener un desarrollo pequeño y aislado para identificar rápidamente los errores de código en caso de algún problema. Además, para TMR es muy importante la personalización de características, por lo tanto, se debe tener una arquitectura con la cual no sea costoso agregar nuevas funcionalidades.
- *Facilidad en el testeo.* Se debe comprobar la robustez de la plataforma, para asegurar su funcionamiento durante las competencias.
- *Facilidad en el despliegue.* Es muy importante realizar 'hot deployments', ya que pueden identificarse problemas críticos antes o durante una competición.
- *Alta agilidad para realizar cambios.* Relacionado con el primer punto, al identificar rápidamente los errores, no consume tanto tiempo realizar cambios.

Debido a lo recién nombrado, se opta por una combinación entre microservicios y en capas, donde todos los servicios serán independientes entre sí y cada uno estará formado por tres capas.

Con esta arquitectura, por un lado, se lograría que cada servicio esté aislado del resto de servicios, y que cada capa esté aislada de las otras capas, por lo que el backend sería sencillo de desarrollar y mantener, además de tener facilidad para implementar nuevas características, lo que aporta extensibilidad. Por otro lado, resultaría sencillo realizar 'hot deployments', ya que los servicios son desplegados como unidades de software separadas, y no sería necesario volver a desplegar la aplicación completa. Por último, el testeo se vería facilitado, ya que cada servicio se puede testear de forma exhaustiva, y con ello asegurar el funcionamiento correcto de la aplicación. De esta forma, de los objetivos específicos mencionados en la sección 1.3, se cumpliría el primer objetivo, y se vería facilitado el desarrollo del tercero.

La principal debilidad que tendría el uso de esta arquitectura es que la aplicación no tendría un gran rendimiento. Esto es debido a la creación de múltiples tablas al momento de subir un evento, sumado al aislamiento de cada microservicio, y al hecho de que se debe pasar por varias capas en cada uno de ellos. Aun así, este apartado no es tan relevante en esta aplicación, ya que es más importante la integridad de los datos que la velocidad de respuesta de los servicios.

Luego, se deben definir las tecnologías a utilizar. Conversando con el equipo de TMR, se decide usar PHP 7.3 sin ningún framework, MariaDB y SQLite como base de datos, y solo agregar Bootstrap como nueva herramienta para el desarrollo del frontend. Esta decisión se

toma debido a que si se optaba por otras tecnologías, se hubiera tenido que utilizar otro servidor, lo que implica un costo económico adicional para TMR, ya que es indispensable mantener activos los servicios actuales mientras se desarrolla la nueva plataforma. Además, el servidor actual, usado hace años por la empresa, usa como gestor de bases de datos a MariaDB, funciona muy bien con los archivos SQLite, es sencillo subir actualizaciones a los archivos PHP, y la inclusión de Bootstrap no implica costos adicionales, ya que solo corresponde a clases CSS predefinidas, ofreciendo una mayor agilidad en el desarrollo y un diseño responsivo para que la interfaz sea compatible con muchas pantallas y navegadores.

Pasando al desarrollo de la solución, como fue mencionado anteriormente, cada servicio del backend se dividió en tres capas. La primera capa, o capa de presentación, corresponde a los endpoints para hacer peticiones de tipo GET y POST luego de haber realizado verificaciones de seguridad. La segunda capa corresponde a la lógica de negocios o controlador, en donde se recibe la información desde el endpoint, se procesa y se manda una petición a la próxima capa, para luego recibir una respuesta, procesarla, y retornarla al usuario a través de la capa mencionada en primer lugar. La tercera capa, o capa de datos, se encarga de realizar las operaciones CRUD. En caso de leer información, ésta es entregada a la segunda capa para su procesamiento.

El siguiente diagrama muestra la estructura final de la aplicación, en donde cada servicio posee sus tres capas:

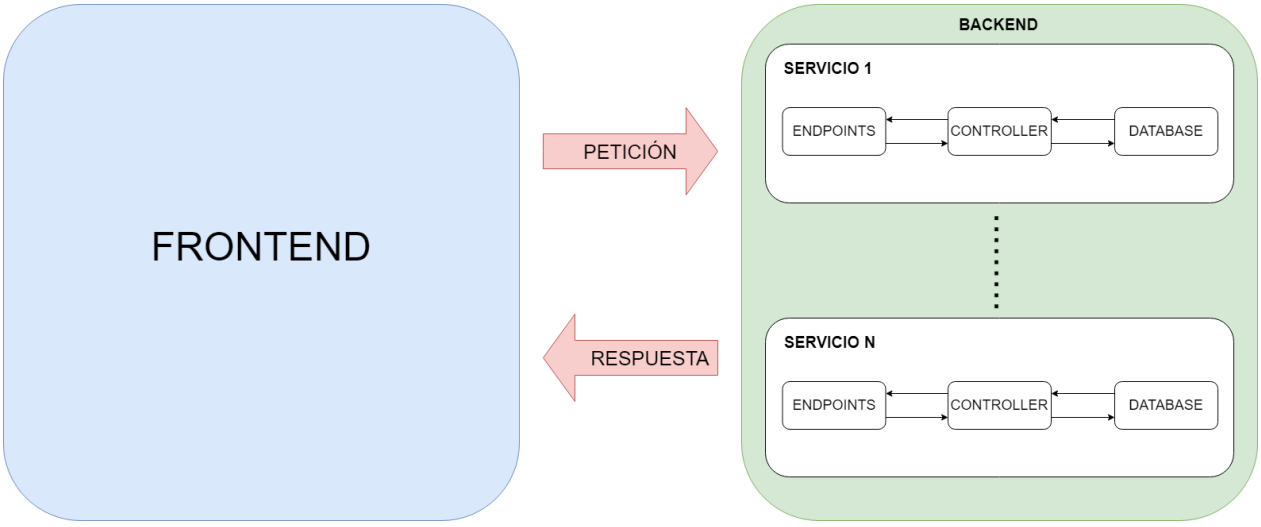


Figura 3.2: Estructura final de la aplicación

De esta forma, el trabajo del backend y del frontend queda completamente definido. El backend se encarga de procesar la información tal que el frontend solo se encargue de mostrarla. Es por ello que se espera que los tiempos de desarrollo de las funcionalidades, tanto de las ya existentes en el editor web original como aquellas completamente nuevas, se optimice, ya que para implementar una nueva funcionalidad basta con agregar sus tres capas, las cuales no dependen de las otras funcionalidades ya existentes.

Anteriormente, ya se explicó la estructura de cada servicio, cómo interactúan sus capas entre sí y de qué se encarga cada una. Por lo tanto, a continuación se mostrará el trabajo

realizado para todos los servicios y se especificará la implementación en concreto para cada uno de ellos, con tal de no repetir lo desarrollado previamente.

- **Autenticación**
  - **Iniciar Sesión**
    - Inicia la sesión de un usuario.
    - Si el usuario es válido y tiene una licencia activa, se crea un token de inicio de sesión.
  - **Cerrar Sesión**
    - Cierra la sesión de un usuario.
    - En caso de que el correo del usuario sea válido y éste tenga su sesión activa, se elimina el token.
- **Personalización**
  - **Obtener personalización de estilos**
    - Obtener los colores y banners personalizados de un usuario para ser mostrados en los Tiempos En Vivo.
    - En caso de que no exista información para los estilos del usuario, se configura una respuesta por defecto.
  - **Configurar personalización de estilos**
    - Configurar los colores y banners personalizados de un usuario para ser mostrados en los Tiempos En Vivo.
    - En el controlador se hace una limpieza de caracteres y se estandarizan los códigos hexadecimales de los colores, para que todos sean de 6 caracteres en lugar de 3.
- **Opciones**
  - **Obtener consolidación automática**
    - Obtener si el usuario tiene activada la opción para consolidar eventos automáticamente. Consolidar se refiere a la acción de unir en un solo evento dos o más eventos. Por ejemplo, se cronometran por separado 3 eventos que en realidad son parte de uno solo, por lo tanto, se consolidan y queda un evento resultante con la información de los 3 eventos.
    - Si no existe esta información para el usuario dado, se configura por defecto que esté desactivada la consolidación automática.
  - **Obtener campos ocultos**
    - Obtener los campos que el usuario desea ocultar al público en los Tiempos En Vivo. Por ejemplo, si el usuario decide ocultar 'Tiempo Total', en ninguna tabla de los Tiempos En Vivo se verá dicha información.
    - En caso de ocurrir algún error con la respuesta de la base de datos o la información no existe, no habrá ningún campo que se deba ocultar.

  - **Obtener mostrar tabla de rezagados**
    * Obtener si el usuario desea mostrar las tablas de rezagados.
    * Si no existe esta información para el usuario, por defecto la respuesta es que sí se muestre la tabla de rezagados.
  - **Obtener nombres personalizados**
    * Obtener los nombres de los campos personalizados por el usuario. Similar al servicio para ocultar campos, al usuario se le da la opción de personalizar los nombres de las columnas en los Tiempos En Vivo. Por ejemplo, puede decidir que la columna 'Tiempo Total' pase a llamarse 'Total' o 'Time'.
    * Si existe algún error con la respuesta de la base de datos o esta información no existe, todos los campos mantienen sus nombres originales.
  - **Obtener zona horaria**
    * Obtener la zona horaria configurada por el usuario. Actualmente, existen las opciones UTC-3, UTC-4 y UTC-5.
    * Si no existe información al respecto para el usuario, la respuesta por defecto es la zona horaria de Chile que, a julio de 2023, corresponde a UTC-4.
  - **Configurar opciones**
    * Configurar las opciones nombradas: consolidación automática, campos ocultos, mostrar tabla de rezagados, nombres personalizados y zona horaria.
    * En caso de que para algunas de las opciones anteriores se envíe un dato inválido, se configura la opción por defecto respectiva. También, en caso de recibir algún nombre personalizado para un campo, éste será limitado a 32 caracteres.

- **Eventos**
  - **Mostrar evento**
    * Mostrar u ocultar un evento de los Tiempos En Vivo.
    * En la capa de datos se alterna el valor del estado del evento, es decir, si ya se mostraba, se oculta, y viceversa.
  - **Archivar evento**
    * Ocultar un evento de la interfaz principal del editor web y los Tiempos En Vivo.
    * Análogo al servicio para mostrar un evento. Además, al archivar un evento, automáticamente éste se oculta.
  - **Eliminar evento**
    * Eliminar un evento definitivamente. Toda su información se pierde.
    * En la capa de datos simplemente se elimina el archivo SQLite, por lo tanto, la información del evento se pierde definitivamente.
  - **Obtener información de eventos**
    * Obtener los nombres, fechas, lugares, responsables, categorías, disciplina, tipo de portal y estados de todos los eventos subidos por un usuario.

    * En el endpoint se recibe un parámetro eventos, cuyo valor define si se retornan todos los eventos, solo los actuales (no archivados) o solo los archivados. En el controlador, se ordenan los eventos por fecha y por nombre.

- **Configurar información de un evento**
    * Configurar nombre, fecha, lugar, responsable y estado de un evento.
    * En el controlador, se verifica que el estado enviado sea válido, que se haya enviado una fecha válida y que se haya enviado un responsable del evento. En caso de que no, se configura como un evento nuevo, con la fecha actual y el responsable siendo el usuario.
- **Subir resultados (creación de evento)**
    * Crear un evento subiendo tanto su propia información como los datos de los corredores.
    * El controlador traduce la información enviada de los tiempos y estados de los corredores. Por ejemplo, cada tiempo pasa de formato HH:MM:SS.mmm a milisegundos, y cada estado pasa de un valor de texto a un valor numérico. Una vez hecho esto, se crean las tablas correspondientes dependiendo principalmente de la disciplina del evento. Estas tablas se pueden dividir en tres: parámetros propios del evento, parámetros propios de los corredores, y datos de los tiempos de los corredores. Una vez creadas mediante la capa de base de datos, se verifica si el usuario tiene la consolidación automática activada. En caso de que sí, se busca realizar una consolidación mediante el servicio de consolidación de eventos, que se desarrollará más adelante.
- **Actualizar resultados**
    * Actualizar los resultados de un corredor, como la información de sus tiempos.
    * En el controlador, se traduce la información de la forma descrita en el servicio anterior. Luego, se hacen verificaciones como que el dorsal del corredor a actualizar sea válido y que los tiempos cronometrados sean mayores o iguales a 0. En caso de que corresponda a una disciplina lineal, se envía la información a la base de datos para que sea actualizada. En caso de que sea una disciplina cíclica, se calcula inmediatamente el tiempo total y la mejor vuelta para que esta información también sea actualizada como es debido.
- **Obtener resultados de un evento**
    * Obtener los datos de cada corredor de un evento.
    * En el controlador, una vez recibidos los datos de cada corredor, algunos de éstos se deben traducir de la manera inversa a la explicada en el servicio para subir resultados. Por ejemplo, los tiempos están guardados en milisegundos y se deben traducir a formato HH:MM:SS:mmm.
- **Consolidación de eventos**
    * Consolidar dos o más eventos.
    * En el controlador se chequea que se hayan recibido al menos dos eventos a consolidar. Luego, si lo que se va a realizar es una consolidación automática, proveniente del servicio para subir eventos, se buscan todos los eventos

que sean compatibles con el último evento creado y se consolidan entre ellos. Si es una consolidación manual, se revisa que los eventos a consolidar sean compatibles. En caso de serlo, se crea una nueva base de datos y se copia la información de cada evento a la nueva base.

Además de la reimplementación de los servicios nombrados con la nueva arquitectura, cada uno de ellos lleva un log para guardar registro de las acciones que realizan los usuarios. En un futuro, esto puede ser útil para hacer análisis de qué servicios son los más utilizados y poder tomar decisiones estratégicas en base a ello. También, se ha tenido especial cuidado en documentar el código y seguir las convenciones de PHP.

3.3. Rediseño completo del editor web

Para cumplir con este objetivo, se dividió el trabajo en dos: el trabajo en backend y el trabajo en frontend.

Desde TMR se limitaba a los clientes a usar ciertos caracteres en los nombres de sus eventos, corredores, u otra información. Esto es debido a que, originalmente, los servicios del backend retornaban la información en texto plano, y se utilizaban algunos caracteres para separar la información. Por lo tanto, el uso de ciertos caracteres por parte del usuario puede provocar errores, como 'pipe' o 'virgulilla'. Es por esto que el trabajo principal en el backend consistió en que todos aquellos servicios que entregan información lo hagan en formato JSON, porque de esta forma se evita el problema de los caracteres, además de ser un formato común y ampliamente utilizado en el mundo de la computación.

Los servicios del backend que se actualizaron fueron:

- Obtener personalización de estilos
- Obtener opciones (consolidación automática, campos ocultos, etc.)
- Obtener información de eventos
- Obtener resultados de un evento

El hecho de retornar la información como JSON, sobre todo para el último servicio nombrado, facilitó mucho el trabajo a realizar en frontend.

En cuanto al trabajo hecho en frontend, el objetivo era desligarse del diseño original del editor, ya que se quería adoptar una interfaz moderna e intuitiva para los clientes de TMR. Para esto, se utilizó una plantilla de Bootstrap como base para comenzar a construir el editor.

El desarrollo del trabajo se llevó a cabo mediante la siguiente metodología. En primer lugar, se desarrolló la interfaz de manera estática mediante HTML, CSS y Javascript con Bootstrap. Luego, se transformó a dinámica a través de las herramientas que ofrece PHP, como las variables $_SESSION, $_POST y $_GET (parámetros de la sesión, parámetros enviados por POST y parámetros enviados por GET). Luego, para todas las interfaces, a excepción

del inicio de sesión, se verificó que el usuario está validado. De estarlo, se muestra la interfaz correctamente, en caso contrario, se redirige automáticamente a una vista de error 401, ya que es un recurso al que no se tiene acceso.

A continuación, se nombrará el trabajo específico hecho para cada interfaz y se comparará con la versión anterior del editor web, omitiendo los cambios hechos en el diseño:

3.3.1. Inicio de sesión

La interfaz de inicio de sesión de la versión final contempla un formulario típico que pide usuario y contraseña, además de ofrecer las opciones para restablecer contraseña y crear una cuenta. Una imagen de la nueva interfaz de inicio de sesión se muestra en la figura B.1 del anexo B.

Al ingresar las credenciales, se hace una petición POST al servicio de inicio de sesión del backend. De ser correctas, se redirige a la vista de eventos actuales, y en caso contrario, se muestra una alerta indicando que las credenciales son incorrectas.

Se agregan los botones *Olvidaste tu contraseña?* y *Quiero tener una cuenta TMR!*. Al hacer clic en el primer botón, se da la opción para enviar un correo al equipo TMR, con tal de recuperar el acceso al editor web. Al pulsar el segundo botón, el usuario será redirigido a la vista de compra de licencias.

El principal cambio hecho en esta vista respecto a la versión anterior del editor web, además de las diferencias estéticas, es que se ofrece la posibilidad de contactar con el equipo de TMR directamente.

3.3.2. Eventos actuales y archivados

Las interfaces de eventos actuales y archivados constan de un listado de eventos, en donde para cada uno se puede ver su información y editarla, además de la posibilidad de decidir entre ocultar o mostrar el evento, o archivar, desarchivar o eliminar el evento, dependiendo si corresponde a la vista de eventos actuales o eventos archivados. En las figuras C.1 y C.2 del anexo C se muestran unas imágenes de la versión inicial y final de la interfaz de eventos.

Los eventos se obtienen a partir del servicio creado en el backend. Para cada evento se puede ver información como fecha de realización, cronometrador, tipo de portal, número de especial o etapa y disciplina. A la derecha de cada tarjeta de eventos se muestran dos botones. Si es un evento actual, los botones serán para ocultar o mostrar el evento y para archivarlo. Si es un evento archivado, los botones serán para desarchivarlo o para eliminar el evento definitivamente. Para todas estas acciones se hace uso de los servicios creados en el backend.

Finalmente, en la parte inferior de la tarjeta de eventos hay tres botones. El botón *Ver Online* es para visualizar el evento en los Tiempos En Vivo. El botón *Editar Títulos* es un modal para editar la información del evento, como su nombre, fecha, entre otros. Por último, el botón *Editar Cronometraje* redirige a una interfaz para editar la información de

los competidores.

La principal diferencia entre la vista de eventos actuales y eventos archivados es que los eventos actuales se visualizan en gris mientras que los archivados en amarillo, además de que cada evento actual puede ocultarse o archivarse, mientras que cada evento archivado puede desarchivarse o eliminarse definitivamente.

Los cambios entre la versión inicial y final es que, inicialmente, ambas interfaces eran archivos completamente diferentes, mientras que en la versión final simplemente se diferencian mediante un parámetro enviado por GET, lo cual reduce la complejidad del proyecto, y hace que la solución sea más eficiente y mantenible.

3.3.3. Personalización

En esta interfaz, se pueden personalizar los colores e imágenes a mostrar en los Tiempos En Vivo. Unas imágenes de las versiones inicial y final de esta interfaz se muestran en las figuras D.1 y D.2 del anexo D.

En la parte superior se muestran dos enlaces, los cuales corresponden, respectivamente, a la imagen que se mostrará en la parte superior de los Tiempos En Vivo y dónde redirigirá esta imagen, que usualmente corresponde al sitio web del cliente de TMR. Los siguientes campos corresponden a los colores que se mostrarán en las diferentes secciones de los Tiempos En Vivo, y normalmente, al igual que con la imagen, suele corresponder a los colores de la empresa cliente. Por último, en la parte inferior hay un botón que permite volver a la configuración por defecto, tanto de las imágenes como de los colores.

Los cambios hechos en esta vista son, en primer lugar, que en los campos correspondientes a colores ahora se puede elegir el color mediante un selector en lugar de escribirlo en hexadecimal, lo que es mucho más simple para el usuario. También, los colores se dividieron en 3 grupos, para que el usuario sepa las zonas que se verán afectadas por el cambio.

3.3.4. Compartir

En esta interfaz, se muestran el código QR del usuario y un embed correspondiente a los Tiempos En Vivo. Se muestra la interfaz final de la sección de compartir en la figura E.1 del anexo E.

El código QR se obtiene mediante la API QR Code [18] y redirige a los Tiempos En Vivo del usuario. Existe un botón *Compartir por Whatsapp*, utilizado para enviar la URL de los Tiempos En Vivo a cualquier contacto a través de este medio.

Por otra parte, el embed se genera solo conociendo el nombre de usuario. Puede ser copiado ya sea seleccionando el texto completo o haciendo clic en el botón *Copiar Embed*.

Puede existir un caso borde en donde el código QR no se pueda obtener. En ese caso, esa sección se eliminará y solo se visualizará el embed personalizado.

La mayor diferencia con respecto a la versión inicial, es que actualmente corresponde a una interfaz nueva, mientras que inicialmente era un modal en la vista de eventos. Otra diferencia, es que en la versión inicial no se hace el manejo de errores, por lo tanto, puede no existir un QR y esa sección mostrarse vacía.

3.3.5. Opciones

En esta interfaz, se pueden configurar las distintas opciones del usuario. En la figura F.1 del anexo F, se muestra la interfaz final de las opciones.

Los dos primeros campos corresponden a switches para seleccionar si se consolidan automáticamente los eventos al subirse uno nuevo, y si se muestra la tabla de rezagados en los Tiempos En Vivo. Luego, existe un selector de zona horaria, en donde, a la fecha de escritura de este informe, están disponibles las zonas horarias UTC-3, UTC-4 y UTC-5. Dependiendo de la zona horaria seleccionada se muestra la hora actual correspondiente.

En la parte inferior, se pueden personalizar los nombres de los campos en los Tiempos En Vivo. Se muestra el nombre original, un campo de texto con el nombre personalizado y un switch para mostrar el campo o no. En la parte final existe un botón para que los campos vuelvan a tener sus valores por defecto.

La principal diferencia con respecto a la versión inicial, es que actualmente corresponde a una interfaz nueva, mientras que inicialmente era un modal en la vista de eventos.

3.3.6. Licencias

En esta interfaz, se muestran las tres diferentes licencias disponibles que ofrece TMR a sus clientes. En la figura G.1 del anexo G, se muestra la vista final de la compra de licencias.

Cada una de las licencias ofrecidas tiene un botón, el cual redirige al sitio Flow [8], una plataforma de pagos en línea ampliamente utilizada en Latinoamérica, que es donde se realiza el pago correspondiente.

En esta interfaz es importante el uso de un parámetro GET. Si no existe, se muestra la vista con una descripción de los beneficios de obtener una licencia TMR y las distintas licencias disponibles. En caso de existir, se mostrará una alerta en verde o en rojo, dependiendo de si el pago de licencia se completó o hubo un error. Esto último se hace ya que Flow requiere una URL a la que se redirigirá si el pago fue exitoso, y otra URL si no lo fue.

Las principales diferencias con respecto a la versión inicial, es que actualmente se muestra el mensaje con los beneficios de adquirir una licencia TMR, además de el uso del parámetro enviado por GET.

3.3.7. Editar Títulos

En este modal se puede editar la información de un evento, como el nombre, fecha, estado de la competición, lugar de realización y cronometrador, así como mostrar un mensaje al público en los Tiempos En Vivo. Una imagen del modal final de la edición de títulos se encuentra en la figura H.1 del anexo H.

Este modal se puede visualizar al hacer clic en el botón *Editar Títulos* de la tarjeta de un evento en la interfaz de eventos (actuales o archivados). Aquí, se rellena un formulario con la información mencionada y se edita a gusto del usuario. Si se pulsa el botón *Cerrar*, se cierra el modal y la información no se modifica.

Los principales cambios respecto a la versión inicial son, en primer lugar, que en la nueva versión se puede editar el estado de la carrera, y en segundo lugar, que tanto el estado como la fecha corresponden, respectivamente, a un selector y a un input de tipo fecha, lo que permite evitar errores de formato.

3.3.8. Editar Cronometraje

En esta interfaz se puede editar la información de los corredores de un evento, como su nombre, categoría, tiempos, entre otros. En el anexo I, en las figuras I.1, I.2, I.3 y I.4, se muestran imágenes de la interfaz de edición de cronometraje y del modal que permite la edición, tanto de la versión inicial del editor web como de la final.

En esta vista, se puede ver una grilla con la información de todos los corredores de un evento, separados por etapas. Cada etapa tiene su propia información, como por ejemplo los tiempos de cada corredor.

Al hacer clic en la primera celda de un corredor, se abre el modal de edición de cronometraje, en donde se puede editar la información del corredor. No todos los campos son editables, como por ejemplo el dorsal del corredor, el tiempo total o la mejor vuelta. Esto ya que el dorsal corresponde al identificador único de un corredor, y tanto la mejor vuelta como el tiempo total corresponden a un cálculo automatizado y no debe ser ingresado por el usuario. Si se pulsa el botón *Cerrar*, se cierra el modal y la información no se modifica.

Los cambios hechos respecto a la primera versión son, en primer lugar, que inicialmente no se soportaban aquellos eventos con portal partida o meta, ni los que constaban de más de una etapa, lo que ahora sí es soportado correctamente gracias al uso de JSON, que facilitó bastante separar la información de cada corredor y cada etapa. También, en la versión inicial todos los campos en el modal de edición eran de texto, en cambio, actualmente se tienen 3 tipos de campos: de texto, de tiempo y selectores, estos últimos para la selección de estado.

3.4. Desarrollo de una nueva funcionalidad: manejo de inscripciones

TMR se dedica exclusivamente al cronometraje de eventos utilizando la aplicación de escritorio TMR7. Este software requiere de una planilla Excel con la información de cada corredor, como sus dorsales, nombres y apellidos, equipo al que representa, entre otros. Son los propios organizadores o clientes de TMR los que se encargan, manualmente, de rellenar esta planilla con la información de inscripciones que ellos manejan, para luego ingresarla al software.

Sin embargo, debido a que TMR cronometra eventos deportivos prácticamente cada semana, el hecho de manejar las inscripciones de cada uno de ellos le puede generar enormes ganancias a la empresa, y es debido a esto que se decide implementar la funcionalidad del manejo de inscripciones, la cual no estaba presente hasta antes del desarrollo de esta memoria.

En cuanto al trabajo en frontend, un miembro de TMR es encargado de crear formularios de inscripción a eventos. Cada evento posee su propio formulario y cualquier persona que desee participar de un evento debe rellenar el correspondiente. Una imagen de uno de los formularios se muestra a continuación:

COPA CERRO 26: TOUR NACIONAL DH 2023 24-25 junio

$34.240

Nombres *	
Apellido Paterno *	
Apellido Materno *	
Rut * ejemplo: 16.954.377-K	
Género *	Seleccione Gen
País *	
Código UCI	
Categorías Open * Categorías por defecto a menos que ingreses un Código UCI	Infantil_Open (S
Email *	
Fecha de nacimiento * ejemplo: 04-09-1988	

Figura 3.3: Formulario de inscripción para un evento

En el formulario se puede ver el nombre del evento junto al precio de inscripción, y distintos campos que el usuario debe rellenar, como nombre, apellidos, RUT, correo electrónico o fecha de nacimiento en el formato DD-MM-AAAA. Una vez rellenos todos los campos, el usuario debe realizar el pago correspondiente para completar la inscripción. Es necesario que cada formulario tenga como etiqueta el usuario al que pertenece el evento, para luego en el editor web poder filtrarlos. Esto se explicará posteriormente.

Luego, se creó la nueva sección *Descargas* del editor web, en donde el cliente de TMR puede descargar tanto las inscripciones del evento que desee, como cualquier versión del software TMR7. Una imagen de la nueva interfaz de Descargas se muestra a continuación:

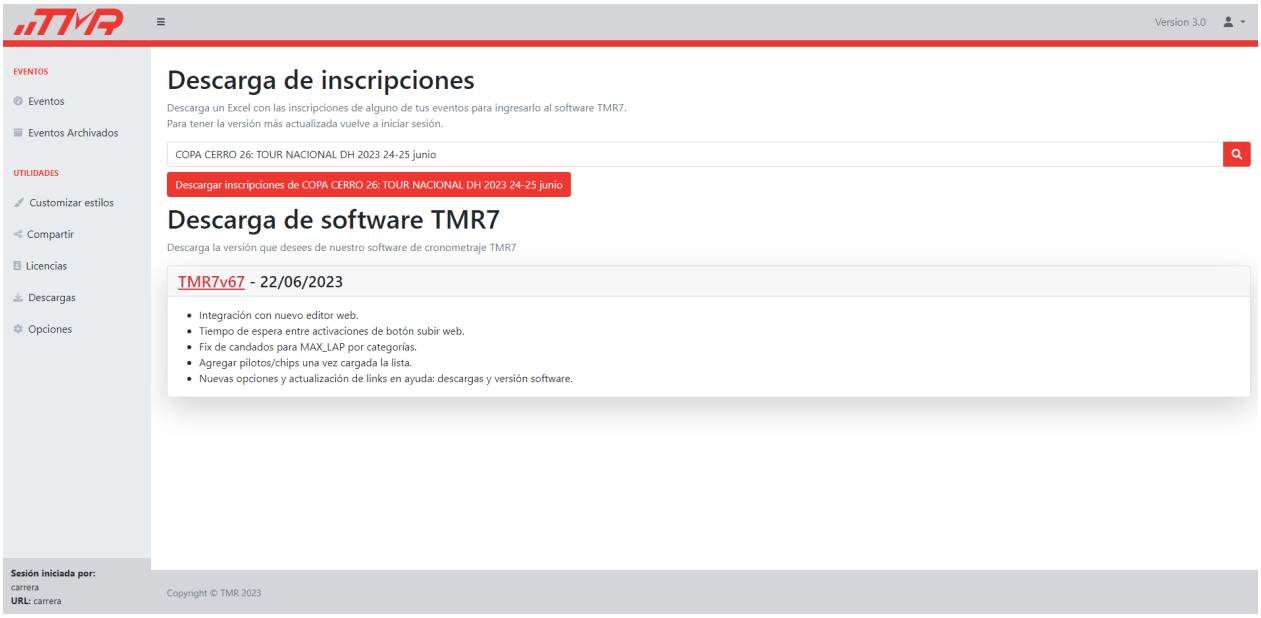


Figura 3.4: Interfaz de la nueva sección de descargas

En la parte superior, se encuentra la sección para descargar las inscripciones a eventos organizados por el usuario. Al buscar y seleccionar alguno, aparecerá un botón para descargar las inscripciones de ese evento en el formato de planilla Excel requerido por el software TMR7. Si un usuario no tiene eventos organizados con inscripciones manejadas por TMR, esta sección no se mostrará.

En la parte inferior, se puede ver un listado de las distintas versiones del software TMR7, junto a su fecha de lanzamiento y una descripción de los cambios realizados en dicha versión. Al hacer clic en el enlace en rojo, el usuario descargará esa versión.

En cuanto al trabajo a realizar en el backend, se deben obtener los datos de las inscripciones. La información de todas las inscripciones de todos los eventos está en un archivo CSV, el cual se debe procesar y, dependiendo del evento requerido, se obtiene su información y se convierte al formato Excel necesario.

El archivo CSV con los datos de las inscripciones se encuentra en WordPress [31], sin embargo, para obtenerlos es necesario tener la sesión iniciada en el administrador. Para ello, se crea una función que inicia sesión con las credenciales válidas y se obtienen las cookies del usuario, y con ello, hacer una petición GET a la URL desde la cual se obtienen los datos y se guarda el archivo CSV en el servidor. Esta acción se realiza cada vez que un usuario inicia sesión, para tener el archivo lo más actualizado posible. Como las credenciales de inicio de sesión son información sensible, se guardan en un archivo de variables de ambiente, tal que no se encuentren embebidas en el código.

Antes de procesar la información del archivo CSV, se debe notar que en cada línea, además de encontrar la información enviada en el formulario, se encuentra un identificador del evento y un identificador de la inscripción. Para procesar, en primer lugar, se filtran los datos correspondientes al evento requerido. Luego, teniendo todos los identificadores de inscripciones y el identificador del evento, se hace una consulta a la base de datos de WordPress para obtener

todas las inscripciones completadas, es decir, válidas. Luego, solo entre aquellos corredores con inscripción válida se crea la planilla Excel con el formato necesario.

Por último, se crea un servicio en el backend encargado de obtener los eventos con inscripciones de un usuario. Para ello, se lee el archivo CSV y se obtienen todos los identificadores de eventos. Luego, se hace una consulta a la base de datos de WordPress con estos identificadores y el nombre de usuario, con tal de obtener los nombres de todos los eventos que tengan como etiqueta el nombre del usuario, lo que fue explicado anteriormente. Es con este servicio que en la parte superior de la sección *Descargas* el usuario puede encontrar todos sus eventos con inscripciones manejadas por TMR.

Capítulo 4

Evaluación

4.1. Tests unitarios para cada servicio

Para testear el primer objetivo, se siguieron dos metodologías. Para los servicios que obtienen información, se hicieron tests unitarios en donde se compararon las respuestas obtenidas con las respuestas entregadas originalmente. Para los servicios que modifican información, se hicieron tests en donde se realizaba alguna operación de modificación, y luego se obtenía la información desde la base de datos, ya sea MariaDB o SQLite. Para todas las pruebas se utilizó el usuario de testing de TMR.

A continuación, se muestra un pseudocódigo de los tests para los servicios que obtienen información:

```
1    function testGET($servicioOriginal, $servicioFinal, $parametros) {
2        // Se obtiene la respuesta del servicio original
3        $respuestaOriginal = get($servicioOriginal, $parametros);
4        // Se obtiene la respuesta del nuevo servicio
5        $respuestaFinal = get($servicioFinal, $parametros);
6        // Ambas respuestas deben ser iguales
7        return $respuestaOriginal === $respuestaFinal;
8    }
```

Listing 4.1: Pseudocódigo de tests para servicios que obtienen información

A continuación, se muestra un pseudocódigo de los tests para los servicios que modifican información:

```
1    function testModificacion($servicioModificacion, $servicioGET,
     $parametrosModificacion, $parametrosGET, $respuestaEsperada) {
2        // Se modifica la data
3        post($servicioModificacion, $parametrosModificacion);
4        // Se obtiene la nueva data desde la base de datos
5        $respuestaGET = get($servicioGET, $parametrosGET);
6        // La respuesta obtenida debe ser igual a la respuesta esperada
7        return $respuestaGET === $respuestaEsperada;
8    }
```

Listing 4.2: Pseudocódigo de tests para servicios que modifican información

4.2. Reuniones y simulación de uso de la plataforma

Durante el desarrollo del nuevo frontend para el editor web, semana por medio se llevaban a cabo reuniones junto al CTO de TMR y tres desarrolladores. Aquí, se mostraban las interfaces implementadas durante los días anteriores, se recibía feedback y se planificaba la próxima reunión, a la cual se llegaba con las correcciones hechas y más interfaces desarrolladas. En total, hubo cuatro reuniones completamente enfocadas en la revisión del frontend.

Una vez hecho lo anterior, se esperaba validar la plataforma con los clientes de TMR. Sin embargo, las reuniones con ellos estaban fijadas fuera del plazo de entrega de esta memoria. Por lo tanto, se optó por simular casos de uso reales, los cuales se presentan a continuación:

4.2.1. Subida de un evento de partida, de meta y consolidación entre ellos

Antes de la realización de una competición, el organizador puede subir un evento de partida, en donde simplemente se ven los competidores y sus horas de partida previstas. Luego, durante la carrera, se puede subir un evento de meta, en donde se muestran los corredores con sus horas de llegada a la meta.

Teniendo estos dos eventos, el organizador puede consolidarlos entre ellos para tener un evento partida-meta en donde, sumado a la información individual de ellos, se ven los tiempos, estados, y posición de cada corredor en la competición.

Por lo tanto, para testear este caso de uso, se activa la consolidación automática en las opciones del editor. Luego, utilizando el software TMR7, ya integrado con los nuevos servicios implementados, se sube un evento de partida. Luego, se sube un evento compatible con el anterior pero de meta. Al momento de subirlo, en el editor existen los tres eventos previstos: el evento de partida, el evento de meta, y el evento partida-meta, consolidado automáticamente.

4.2.2. Subida y consolidación de eventos cíclicos de distintas etapas

Los eventos cíclicos son aquellos en donde se hace el mismo recorrido varias veces seguidas. Un organizador puede subir un evento cíclico de la etapa X, luego subir otro con el mismo nombre de la etapa Y, para luego consolidarlos, obteniendo un solo evento cíclico de etapa X-Y.

Para probar esto, se activa la consolidación automática en las opciones del editor web. Luego, mediante el software TMR7, se sube un evento cíclico con etapa 1 y otro con etapa 2, ambos de la misma disciplina. Al momento de subir el segundo evento, automáticamente se consolidan ambos, y en la interfaz de Eventos del editor se muestran los tres eventos previstos: el evento con etapa 1, el evento con etapa 2, y el evento con etapa 1-2.

4.2.3. Mostrar, archivar y eliminar un evento

Todos los eventos inicialmente se muestran en los Tiempos En Vivo. Un organizador tiene la capacidad de:

- Ocultar el evento al público.
- Archivar el evento y ocultarlo de los Tiempos En Vivo y de la vista principal del editor.
- Eliminar el evento y perder su información definitivamente.

Para testear este caso de uso, se utilizarán los eventos antes mencionados. Todos ellos se muestran en los Tiempos En Vivo.

En el editor, se hace clic en la opción para ocultar uno de los eventos. Éste dejó de mostrarse al público en los Tiempos En Vivo y sigue mostrándose en la interfaz principal del editor.

Luego, en otro evento se pulsa la opción para archivarlo. Éste ya no se muestra en los Tiempos En Vivo y tampoco en la interfaz principal del editor, sino que en la vista de Eventos Archivados.

Por último, se utiliza el mismo evento archivado previamente para probar que se elimina de manera correcta. Estando en la interfaz de Eventos Archivados, se pulsa el botón para eliminar el evento, y éste deja de aparecer, por lo que fue eliminado exitosamente.

4.2.4. Editar información de un evento

Para todos sus eventos, y como fue dicho en la sección 3.3.7, el organizador puede editar su información o mostrar un mensaje en los Tiempos En Vivo.

Para probar este caso, se utilizará uno de los eventos subidos previamente. Estando en la interfaz principal del editor, se hace clic en el botón *Editar Títulos*. Aquí, se abre un modal, se edita toda la información y se guarda. Al volver a cargar la vista de Eventos, se puede ver que el nombre del evento, la fecha y el cronometrador cambiaron, y al entrar en los Tiempos En Vivo, estos cambios se ven reflejados y se muestra un mensaje bajo el nombre del evento.

4.2.5. Editar la información de los corredores un evento

Una vez subidos los resultados de los competidores, el organizador puede editar la información de éstos. La información más relevante que se puede cambiar de un corredor es:

- Nombre
- Categoría

- Equipo
- País
- Tiempo
- Estado
  - OK
  - No comienza
  - No termina
  - Descalificación
- Penalización
- Hora de partida
- Hora de meta

El caso de uso más general con el que se puede testear es con el evento cíclico de varias etapas subido anteriormente, ya que sumado a los campos anteriores, posee varios tiempos según la cantidad de vueltas realizadas, además del campo de Mejor Vuelta, que se calcula automáticamente al cambiar otro. También, los tiempos de varias etapas son independientes entre sí, por lo que no deberían afectarse entre ellas.

Para el testeo, se hace clic sobre la opción *Editar Cronometraje* del evento mencionado. Aquí, el primer corredor en la primera etapa tiene 2 vueltas, siendo la segunda vuelta su mejor tiempo con 15 segundos menos que en la primera.

Luego, en la primera etapa se cambia el nombre del corredor y se guarda. Al volver a cargar la vista de edición de cronometraje, se puede ver que en ambas etapas el nombre del competidor cambió, al igual que en los Tiempos En Vivo.

Luego, al mismo corredor en la primera etapa se le suman 10 segundos en la segunda vuelta y 20 segundos de penalización. Al guardar, su tiempo total en esa etapa aumentó en 30 segundos y la mejor vuelta pasó a ser la primera, además de que en los Tiempos En Vivo este cambio se ve correctamente. La segunda etapa no se vio afectada.

Por último, al mismo competidor se le asignará un estado no válido, como el de descalificación, en la segunda etapa. Al hacer esto, en los Tiempos En Vivo el corredor deja de mostrarse en la tabla de posicionamiento, solo mostrándose en la tabla de rezagados. Se ve que tiene estado OK en la primera etapa, pero estado de descalificación en la segunda.

4.2.6. Actualización de los estilos

Un organizador puede editar los estilos y la imagen del logo que se ve en los Tiempos En Vivo.

Para testear este caso de uso, en la vista de personalización de estilos se cambian los colores y la imagen del logo y se guarda. Al hacer esto, en la vista de personalización quedan guardados los nuevos estilos, y éstos se ven reflejados en los Tiempos En Vivo.

Un ejemplo de esto se puede ver entre la figura D.2 del anexo D y la figura 1.2, en donde se ven los colores de la interfaz de personalización y cómo se ven los Tiempos En Vivo con estos estilos.

4.2.7. Actualización de las opciones

Un organizador puede cambiar las opciones a su gusto, con tal de que en los Tiempos En Vivo se muestre una información más clara según se estime conveniente.

Para probar este caso de uso, se actualizan distintas opciones y se ven los cambios reflejados en los Tiempos En Vivo. La única opción no testeada en este punto es la consolidación automática, ya probada anteriormente al subir los eventos.

A continuación, se muestran las distintas opciones que fueron testeadas:

- *Mostrar tabla de rezagados*. Al desactivar esta opción ninguna tabla de rezagados de ningún evento se ve en los Tiempos En Vivo.
- *Zona horaria*. Inicialmente, se tiene seleccionada la zona horaria de Chile. Luego, se selecciona la zona horaria de Argentina. En los Tiempos En Vivo, se puede ver que la hora de última actualización de los eventos cambió en una hora.
- *Campos personalizados*. Se cambia el nombre del campo 'LAPS' por 'Vueltas'. En los Tiempos En Vivo este cambio se ve reflejado.
- *Ocultar campos*. Se oculta el campo 'Team'. En los Tiempos En Vivo no se muestra este campo en ninguna tabla de ningún evento.

4.3. Validación manual de la planilla Excel generada por la nueva funcionalidad implementada

Para el desarrollo de la nueva funcionalidad, se automatizó el proceso de generación de la planilla Excel que va al software TMR7 para cronometrar un evento. Sin embargo, el archivo CSV desde el cual se genera contiene todas las inscripciones, y en la planilla solo deben estar aquellos corredores con inscripciones completadas válidas.

Desde el administrador de WordPress se puede revisar el estado de todos los pedidos, entre ellos, las inscripciones a eventos. Por lo tanto, el CTO de TMR descargó por separado la planilla Excel ocupando la nueva funcionalidad, y el archivo CSV con toda la información sobre las inscripciones. Luego, por cada línea del archivo CSV, donde cada una representa una inscripción, se encargó manualmente de verificar que coincidan los estados con el del

administrador de WordPress, para luego, validar que en la planilla Excel solo aparezcan los corredores con inscripciones completadas.

Además, los días 1 y 2 de julio de 2023, se llevó a cabo, en la comuna de Puente Alto, el primer evento con inscripciones manejadas por TMR, llamado Copa Cerro 26 [14], en donde se inscribieron 281 corredores. Este evento corresponde a la primera ocasión en que se pone a prueba esta funcionalidad en un ambiente real de competencia, resultando un éxito. La plataforma funcionó correctamente, no hubo ningún problema, y el hecho de que TMR haya manejado las inscripciones permitió que se generaran mayores ganancias que sin esta funcionalidad.

Capítulo 5

Conclusiones

5.1. Trabajo realizado

Para la construcción de la nueva plataforma, fueron necesarias reuniones en distintas instancias con el equipo de TMR para identificar el problema a resolver, acordar los pasos a seguir, herramientas a utilizar y plazos a cumplir.

Se considera que el objetivo general se ha cumplido satisfactoriamente, ya que se logró una reestructuración completa de la plataforma web de TMR, con lo que se espera que ésta deje de ser un cuello de botella para el crecimiento y diversificación de la empresa. Sobre el cumplimiento de los objetivos específicos:

- *Desacoplar el monolito actual.* Con la reestructuración del backend de la aplicación, se logra una plataforma mantenible y extensible. Además, este desarrollo fue realizado mientras se mantenían operativos los servicios originales de TMR. Por lo tanto, se considera que este objetivo se cumplió exitosamente.
- *Rediseñar completamente la plataforma web.* Con la creación del nuevo frontend desde cero, se convierte la antigua interfaz en una moderna e intuitiva, ya que se permite que el usuario pueda navegar con facilidad en la plataforma, entendiendo cómo funciona. Además, se adaptaron correctamente algunos servicios del backend para que retornen la información de la manera requerida por el frontend. Por lo tanto, se considera que este objetivo se cumplió satisfactoriamente.
- *Crear una nueva funcionalidad que le entregue valor a la empresa: manejo de inscripciones.* Se desarrolla completamente la nueva funcionalidad que maneja las inscripciones. Con esta implementación, se generaron ganancias adicionales para TMR en el primer evento realizado con esta característica. En el futuro, se espera que continúe la tendencia y que TMR obtenga mayores ganancias por evento cronometrado. Se considera que este objetivo se cumplió exitosamente.

La nueva plataforma se encuentra activa en `https://timer-crono.com/TMRCloud/`. Para iniciar sesión, se pueden utilizar las credenciales gratuitas que ofrece TMR:

- Usuario: free@user.com
- Contraseña: 123
- Enlace de Tiempos En Vivo: `https://tmr.cl/free/tiempos.php`

Debido a la mantenibilidad y extensibilidad logradas, se espera que los desarrolladores futuros que integren la empresa sean capaces de implementar funcionalidades de una manera sencilla y eficiente. A su vez, debido a la transformación de la interfaz, se espera que la cantidad de clientes interesados en los servicios de TMR aumente, ya que la experiencia de usuario ha mejorado considerablemente. Por lo tanto, se concluye que la nueva plataforma aporta valor a TMR.

5.2. Trabajo futuro

Se esperaba que el desarrollo hecho en este trabajo de memoria se haya comenzado a utilizar desde el mes de junio de 2023 por algunos clientes de TMR. Sin embargo, debido a la situación climática, los eventos se vieron obligados a posponerse hasta el mes de agosto. Por lo tanto, es durante ese mes que se espera que se utilice la plataforma por primera vez en un ambiente real, por alguien externo a TMR.

Adicionalmente, una vez finalizada la reestructuración del backend de la aplicación, con cada uno de sus servicios funcionando correctamente, se espera retomar el desarrollo de la aplicación móvil de TMR, el cual se encontraba suspendido, integrando los nuevos servicios a ella.

Por último, ya que la plataforma es extensible, se han conversado y negociado varias nuevas características que se pueden agregar. Estas características han surgido de reuniones con el equipo de TMR, de tomar como referencia otros sitios web de cronometraje, o de peticiones de los clientes. Algunas de estas características corresponden a:

- Una página de perfil, en donde el usuario pueda personalizar su nombre de usuario y su enlace a los Tiempos En Vivo, tener una imagen de perfil, entre otros.
- Agregar temas a la interfaz. Por ejemplo, tema oscuro o un tema personalizado por cada usuario.
- Agregar traducciones, con tal de tener más de un idioma.
- A los corredores que tengan disponible su correo electrónico, enviarles las fotos oficiales de la competencia en que hayan participado.
- En los Tiempos En Vivo, mostrar una foto del corredor y su historial de eventos pasados, para lo cual sería necesario tener una base de datos de corredores en donde se almacene la información de éstos.
- Agregar estadísticas de competición. Por ejemplo, el tiempo promedio en completar la carrera, el mejor novato, el mejor equipo, diferenciar por género o país, etcétera.

- En los Tiempos En Vivo, mostrar los tweets relacionados con el evento y agregar un enlace al streaming de la competición, si es que está disponible.
- Mostrar un mapa en 3D del recorrido de la carrera, en donde se vea información como la distancia total, o la altura mínima y máxima sobre el nivel del mar.

Bibliografía

[1] Alge-Timing. Disponible en https://alge-timing.com/. Visitado el 23 de julio de 2023.

[2] CORFO. Disponible en https://www.corfo.cl/sites/cpp/homecorfo. Visitado el 23 de julio de 2023.

[3] Cronosur. Disponible en https://cronosur.cl/. Visitado el 23 de julio de 2023.

[4] CronoSystem. Disponible en http://cronosystem.cl/. Visitado el 23 de julio de 2023.

[5] FabLab. Disponible en http://www.fablab.uchile.cl/. Visitado el 23 de julio de 2023.

[6] MyLaps. Disponible en https://www.mylaps.com/. Visitado el 23 de julio de 2023.

[7] Race Timing. Disponible en https://racetiming.cl/httpdocs/. Visitado el 23 de julio de 2023.

[8] Flow. Disponible en https://www.flow.cl/, 2013. Visitado el 23 de julio de 2023.

[9] T. Berners-Lee. HTML. Disponible en https://html.com/document/, 1993. Visitado el 23 de julio de 2023.

[10] Red Bull. Monserrate Cerro Abajo. Disponible en https://www.redbull.com/cl-es/events/red-bull-monserrate-cerro-abajo-event, 2022. Visitado el 23 de julio de 2023.

[11] Red Bull. Valparaíso Cerro Abajo. Disponible en https://www.redbull.com/cl-es/events/cerroabajo, 2023. Visitado el 23 de julio de 2023.

[12] MariaDB Corporation. MariaDB. Disponible en https://mariadb.org/, 2012. Visitado el 23 de julio de 2023.

[13] R. Dahl. Node.js. Disponible en https://nodejs.org/en/about, 2009. Visitado el 23 de julio de 2023.

[14] Federación Deportiva Nacional de Ciclismo de Chile. Copa Cerro 26. Disponible en https://www.fdnciclismochile.cl/event/copa-cerro-26-nueva-fecha/, 2023. Visitado el 23 de julio de 2023.

[15] B. Eich. JavaScript. Disponible en https://developer.mozilla.org/en-US/docs/Web/JavaScript, 1995. Visitado el 23 de julio de 2023.

[16] D. Merriman et al. MongoDB. Disponible en https://www.mongodb.com/es, 2007. Visitado el 23 de julio de 2023.

[17] O. Müller et al. PHPMyAdmin. Disponible en https://www.phpmyadmin.net/, 1998. Visitado el 23 de julio de 2023.

[18] Foundata GmbH. QR Code. Disponible en https://goqr.me/api/doc/. Visitado el 23 de julio de 2023.

[19] R. Hipp. SQLite. Disponible en https://www.sqlite.org/index.html, 2000. Visitado el 23 de julio de 2023.

[20] R. Lerdorf. PHP. Disponible en https://www.php.net/manual/es/intro-whatis.php, 1994. Visitado el 23 de julio de 2023.

[21] H. Wium Lie. CSS. Disponible en https://www.css3.com/, 1994. Visitado el 23 de julio de 2023.

[22] Meta. React. Disponible en https://react.dev/, 2013. Visitado el 23 de julio de 2023.

[23] T. Otwell. Laravel. Disponible en https://laravel.com/, 2011. Visitado el 23 de julio de 2023.

[24] M. Richards. *Software Architecture Patterns: Understanding Common Architecture Patterns and when to Use Them*. O'Reilly Media, 2015.

[25] Columbia Sportswear. Columbia Snow Challenge. Disponible en https://corre.cl/evento/10039, 2022. Visitado el 23 de julio de 2023.

[26] TMR. TMR Solutions. Disponible en https://www.tmr.solutions/, 2023. Visitado el 23 de julio de 2023.

[27] Twitter. Bootstrap. Disponible en https://getbootstrap.com/, 2010. Visitado el 23 de julio de 2023.

[28] G. van Rossum. Python. Disponible en https://www.python.org/, 1991. Visitado el 23 de julio de 2023.

[29] Epica XCM. Epica Parque Nacional Torres del Paine. Disponible en https://tusdesafios.com/events/epica-parque-nacional-torres-del-paine-2018, 2018. Visitado el 23 de julio de 2023.

[30] A. Yu y J. Chen. PostgreSQL. Disponible en https://www.postgresql.org/, 1986. Visitado el 23 de julio de 2023.

[31] M. Mullenweg y M. Little. WordPress. Disponible en https://wordpress.com/es/, 2003. Visitado el 23 de julio de 2023.

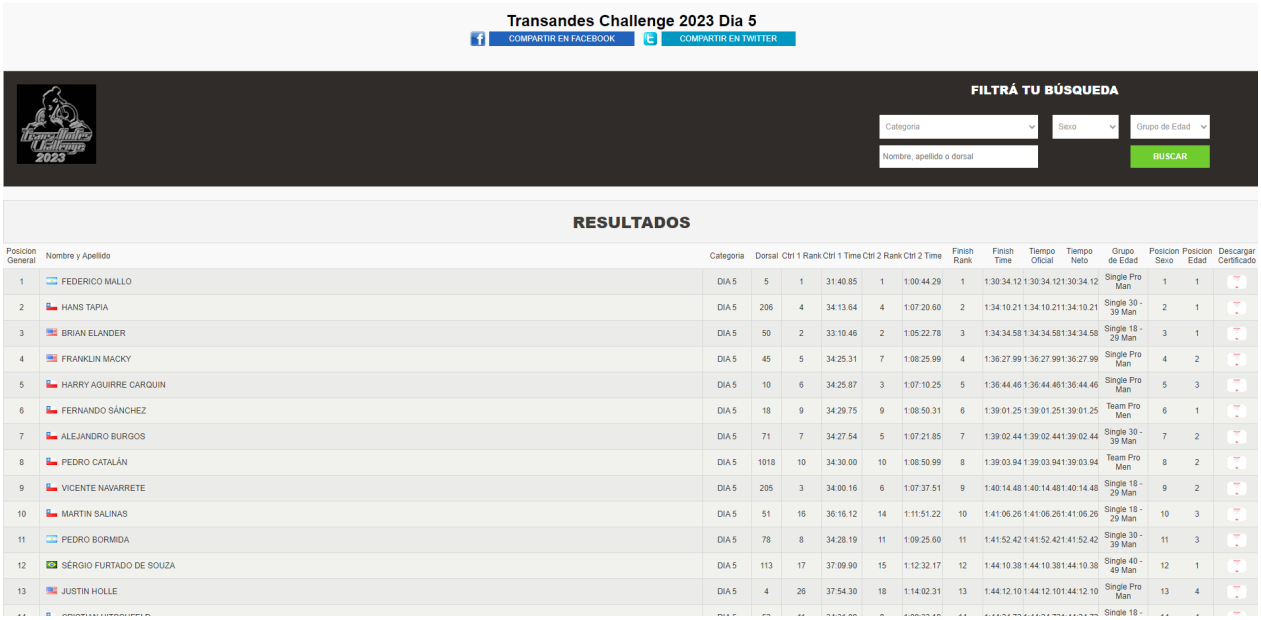
[32] A. Holovaty y S. Willison. Django. Disponible en https://www.djangoproject.com/, 2003. Visitado el 23 de julio de 2023.

ANEXOS

Anexo A

Sitios web de otras empresas de cronometraje

En el sitio web de Race Timing, se puede ver la tabla de resultados junto con la posibilidad de filtrar la búsqueda.



Posicion General	Nombre y Apellido	Categoria	Dorsal	Ctrl 1 Rank	Ctrl 1 Time	Ctrl 2 Rank	Ctrl 2 Time	Finish Rank	Finish Time	Tiempo Oficial	Tiempo Neto	Grupo de Edad	Posicion Sexo	Posicion Edad	Descargar Certificado
1	FEDERICO MALLO	DIA 5	5	1	31:40.85	1	1:00:44.29	1	1:30:34.12	1:30:34.12	1:30:34.12	Single Pro Man	1	1	
2	HANS TAPIA	DIA 5	206	4	34:13.64	4	1:07:20.60	2	1:34:10.21	1:34:10.21	1:34:10.21	Single 30 - 39 Man	2	1	
3	BRIAN ELANDER	DIA 5	50	2	33:10.46	2	1:05:22.78	3	1:34:34.58	1:34:34.58	1:34:34.58	Single 18 - 29 Man	3	1	
4	FRANKLIN MACKY	DIA 5	45	5	34:25.31	7	1:08:25.99	4	1:36:27.99	1:36:27.99	1:36:27.99	Single Pro Man	4	2	
5	HARRY AGUIRRE CARQUIN	DIA 5	10	6	34:25.87	3	1:07:10.25	5	1:36:44.46	1:36:44.46	1:36:44.46	Single Pro Man	5	3	
6	FERNANDO SÁNCHEZ	DIA 5	18	9	34:29.75	9	1:08:50.31	6	1:39:01.25	1:39:01.25	1:39:01.25	Team Pro Men	6	1	
7	ALEJANDRO BURGOS	DIA 5	71	7	34:27.54	5	1:07:21.85	7	1:39:02.44	1:39:02.44	1:39:02.44	Single 30 - 39 Man	7	2	
8	PEDRO CATALÁN	DIA 5	1018	10	34:30.00	10	1:08:50.99	8	1:39:03.94	1:39:03.94	1:39:03.94	Team Pro Men	8	2	
9	VICENTE NAVARRETE	DIA 5	205	3	34:00.16	6	1:07:37.51	9	1:40:14.48	1:40:14.48	1:40:14.48	Single 18 - 29 Man	9	2	
10	MARTIN SALINAS	DIA 5	51	16	36:16.12	14	1:11:51.22	10	1:41:06.26	1:41:06.26	1:41:06.26	Single 18 - 29 Man	10	3	
11	PEDRO BORMIDA	DIA 5	78	8	34:28.19	11	1:09:25.60	11	1:41:52.42	1:41:52.42	1:41:52.42	Single 30 - 39 Man	11	3	
12	SÉRGIO FURTADO DE SOUZA	DIA 5	113	17	37:09.90	15	1:12:32.17	12	1:44:10.38	1:44:10.38	1:44:10.38	Single 40 - 49 Man	12	1	
13	JUSTIN HOLLE	DIA 5	4	26	37:54.30	18	1:14:02.31	13	1:44:12.10	1:44:12.10	1:44:12.10	Single Pro Man	13	4	

Figura A.1: Sitio web de Race Timing. Fuente: https://racetiming.cl/httpdocs/

En el sitio web de Cronosur, se puede ver la información de un corredor junto a un gráfico comparativo con los tiempos de los demás competidores.

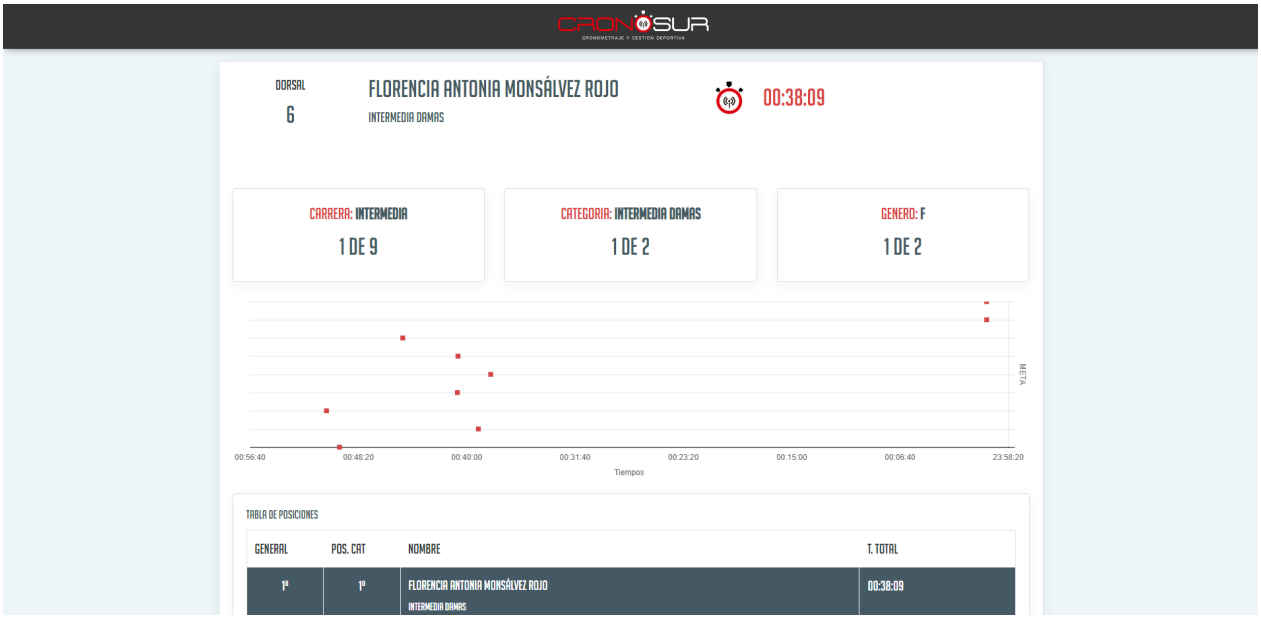


Figura A.2: Sitio web de Cronosur. Fuente: https://cronosur.cl/

Anexo B

Interfaz de inicio de sesión

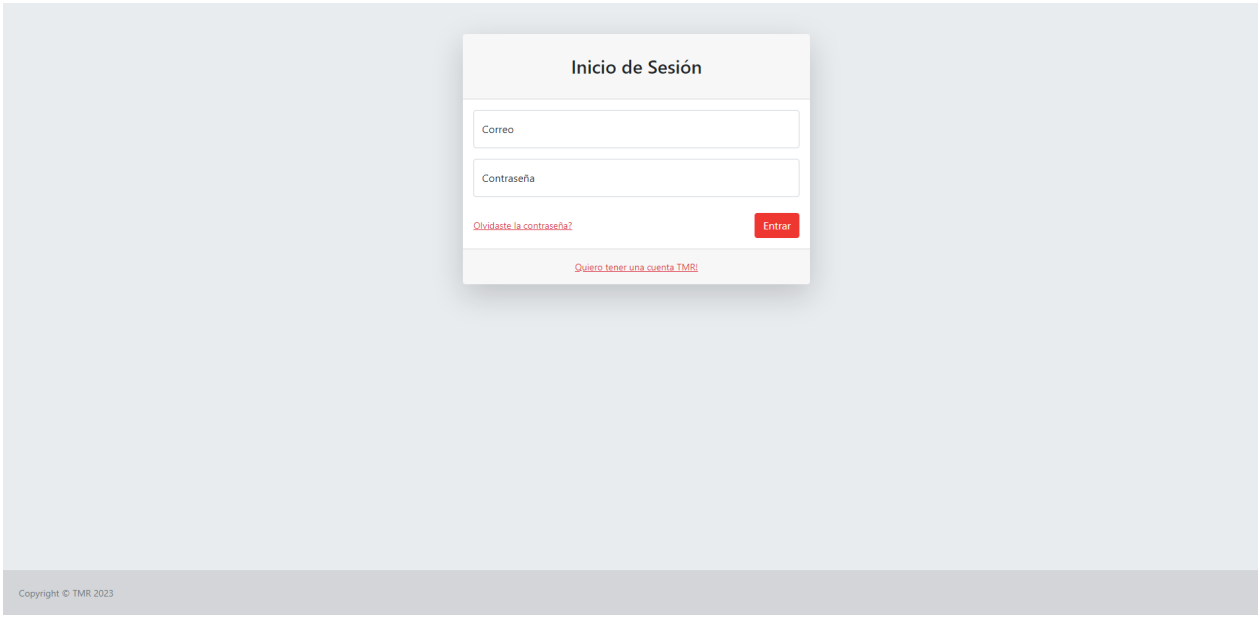


Figura B.1: Interfaz de inicio de sesión de la versión final

Anexo C

Interfaz de eventos

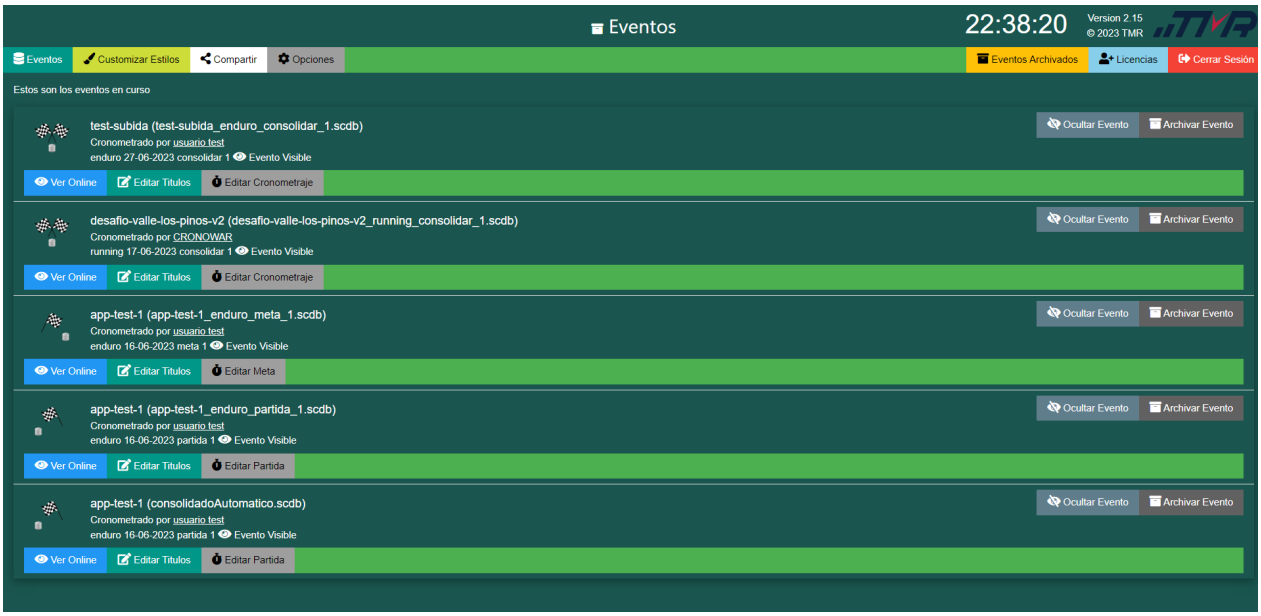


Figura C.1: Interfaz de los eventos actuales de la versión inicial

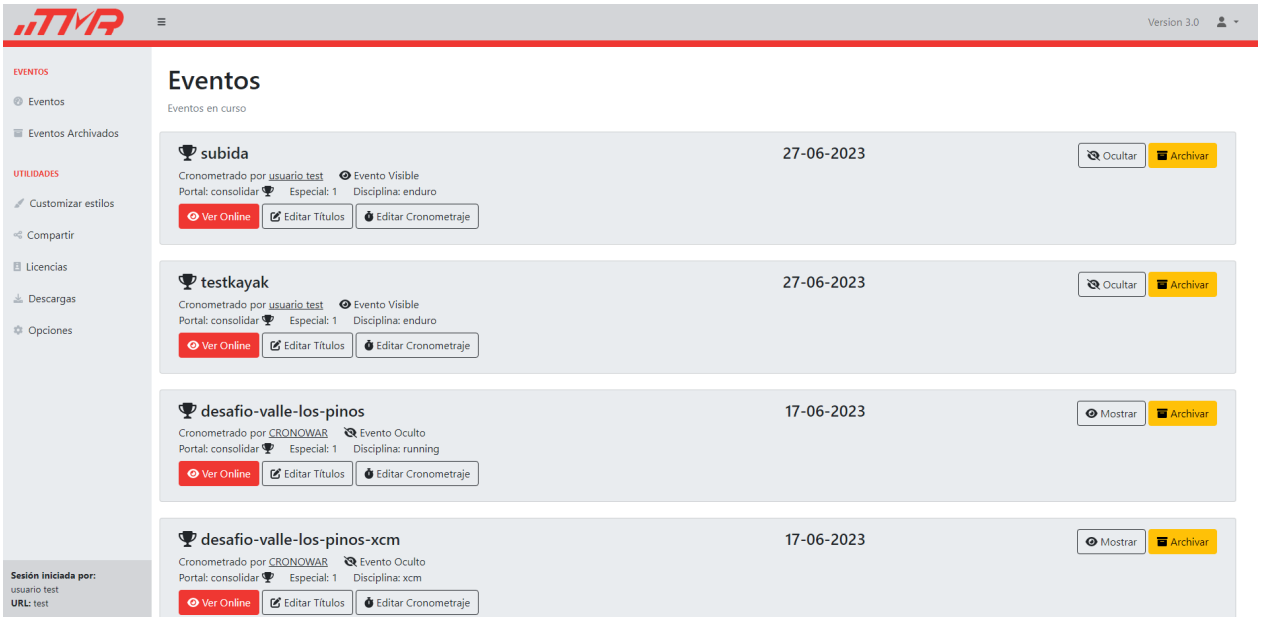


Figura C.2: Interfaz de los eventos actuales de la versión final

Anexo D

Interfaz de personalización

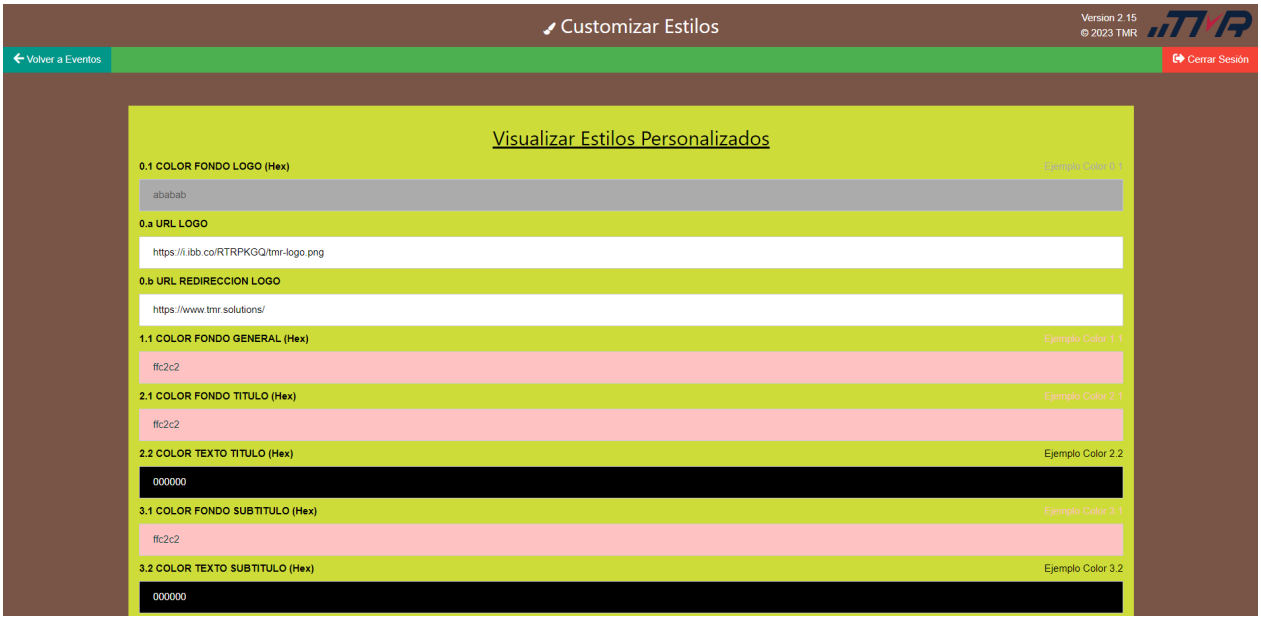


Figura D.1: Interfaz de personalización de estilos de la versión inicial

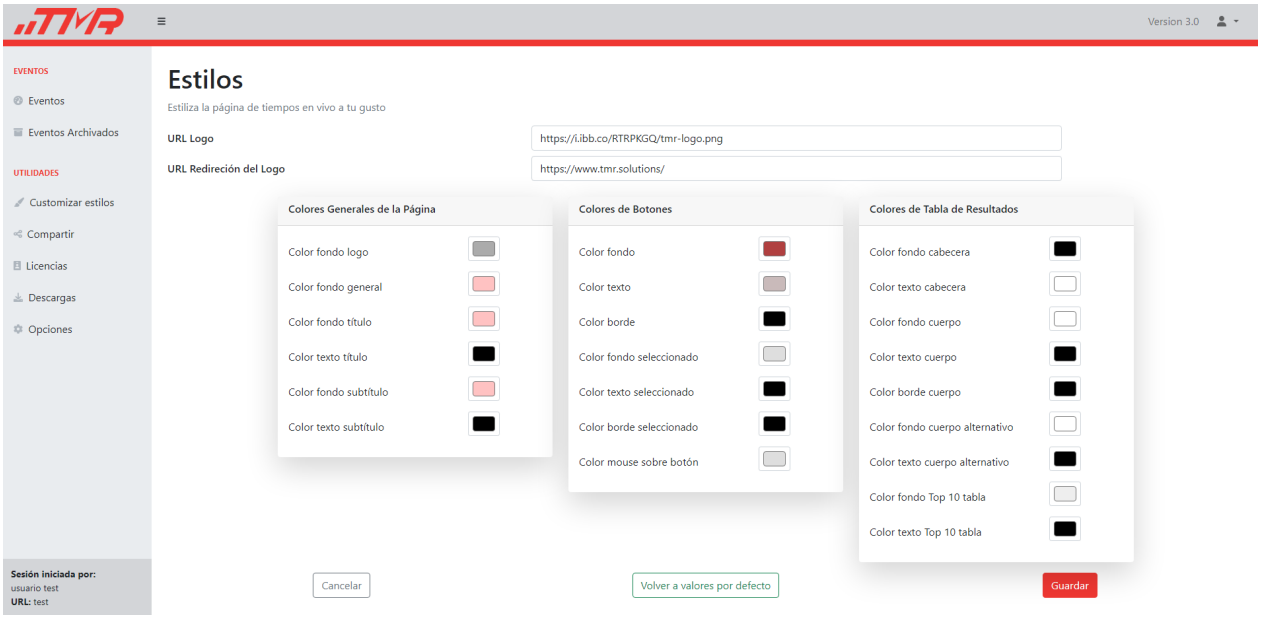


Figura D.2: Interfaz de personalización de estilos de la versión final

Anexo E

Interfaz de compartir

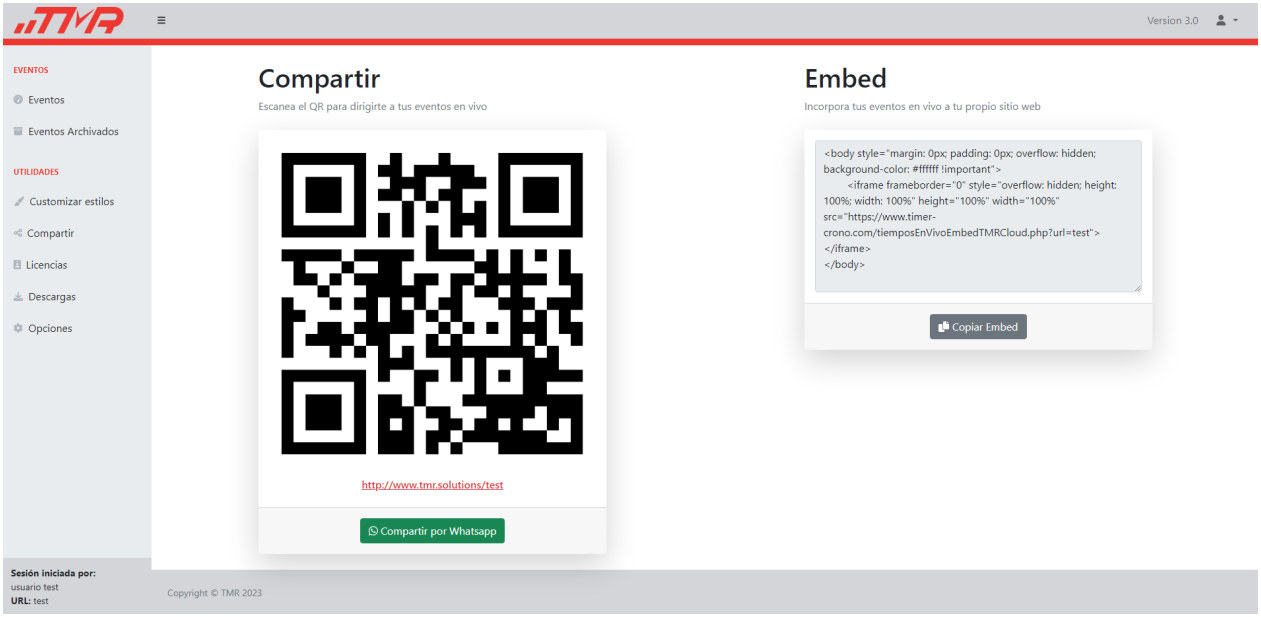


Figura E.1: Interfaz de la sección de compartir de la versión final

Anexo F

Interfaz de opciones

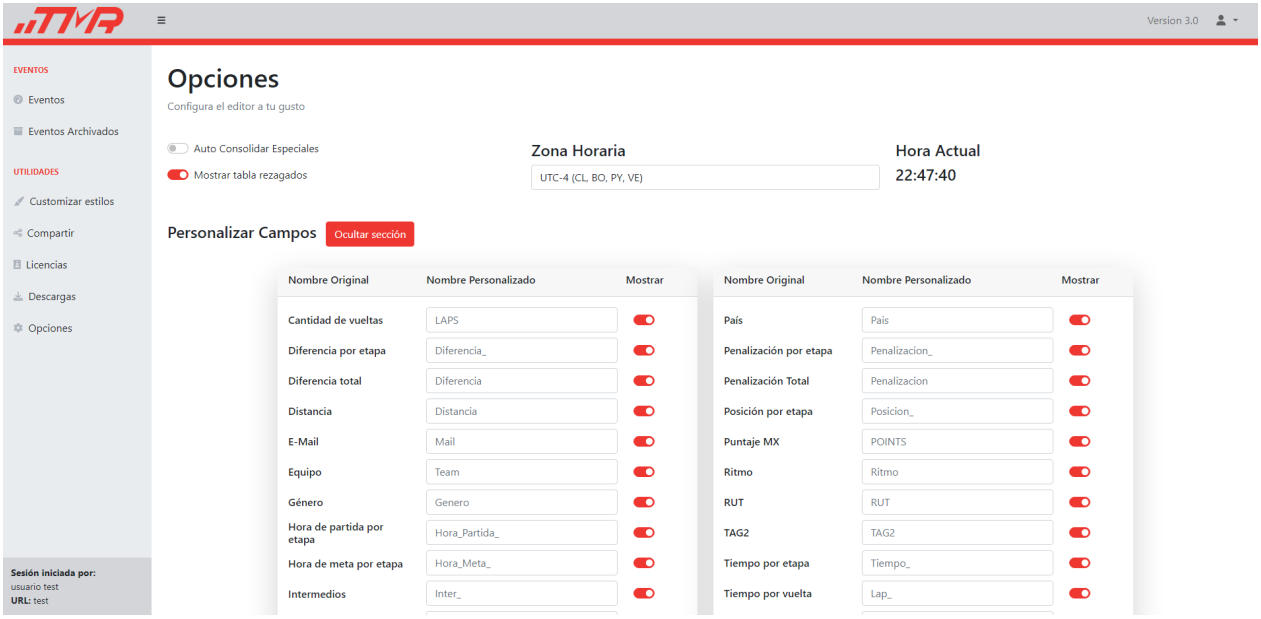


Figura F.1: Interfaz de las opciones de la versión final

Anexo G

Interfaz de licencias

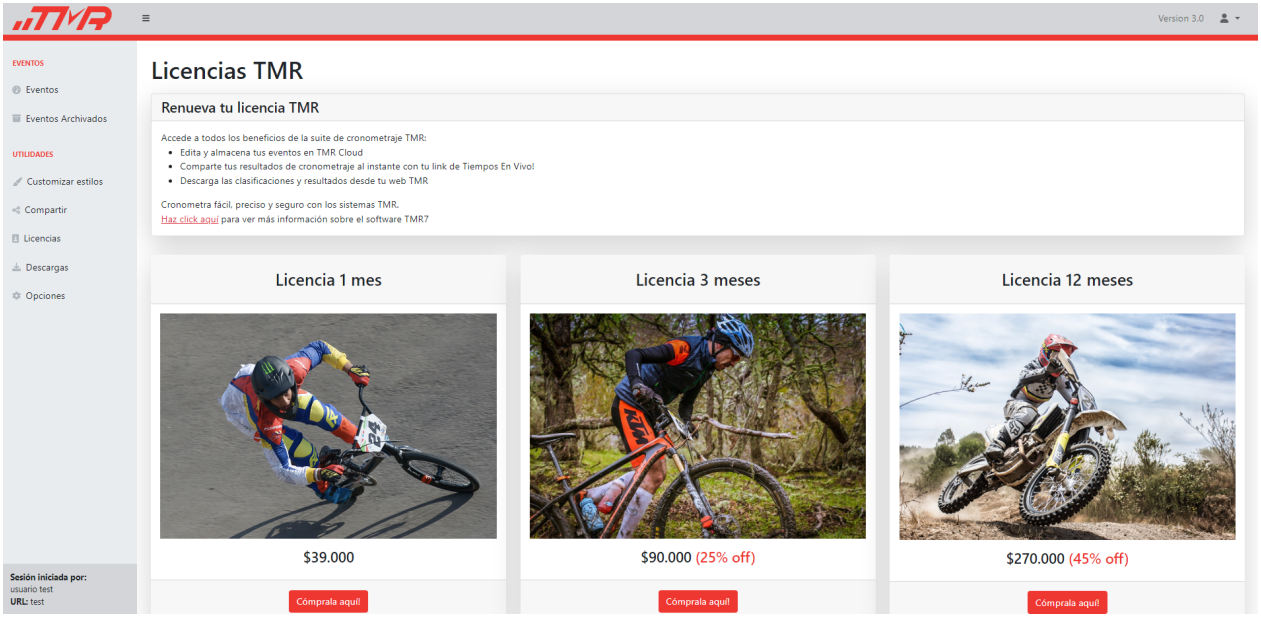


Figura G.1: Interfaz de compra de licencias de la versión final

Anexo H

Modal de edición de títulos

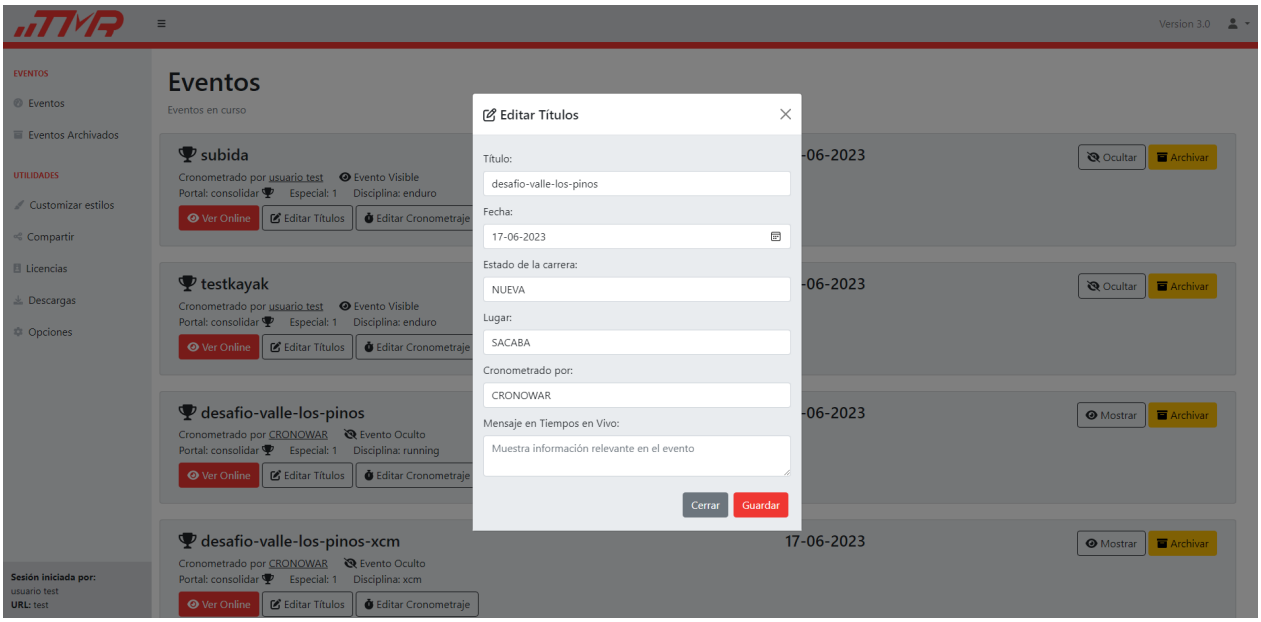
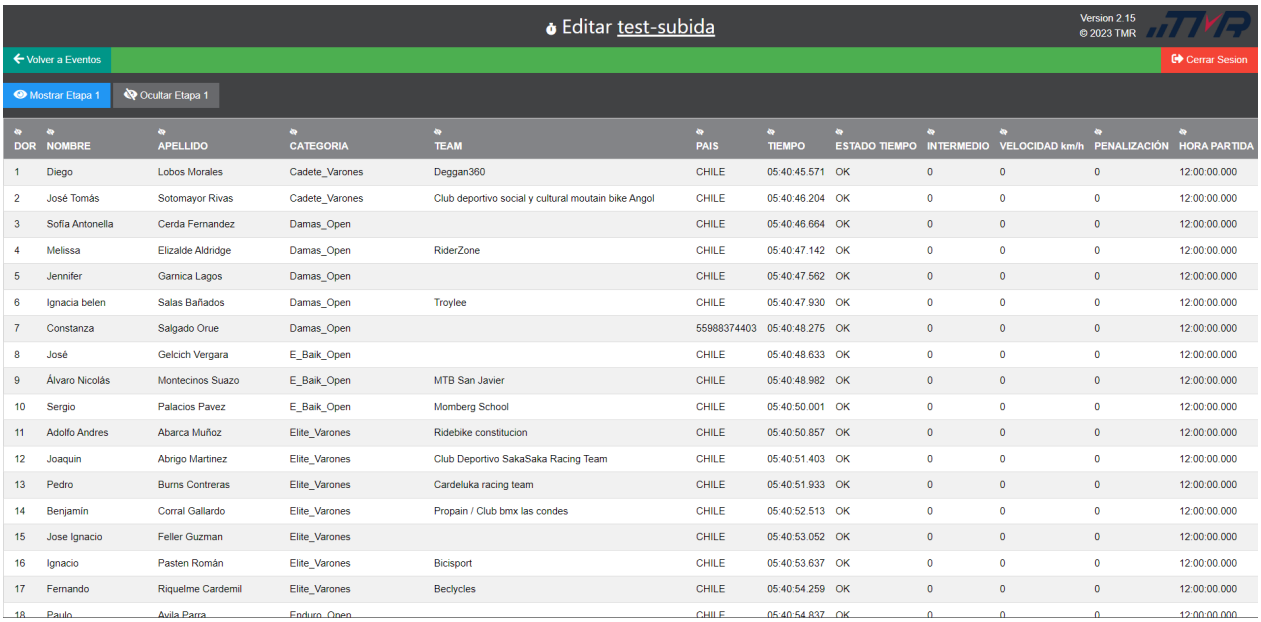


Figura H.1: Modal de edición de títulos de la versión final

Anexo I

Interfaz y modal de edición de cronometraje



DOR	NOMBRE	APELLIDO	CATEGORIA	TEAM	PAIS	TIEMPO	ESTADO TIEMPO	INTERMEDIO	VELOCIDAD km/h	PENALIZACIÓN	HORA PARTIDA
1	Diego	Lobos Morales	Cadete_Varones	Deggan360	CHILE	05:40:45.571	OK	0	0	0	12:00:00.000
2	José Tomás	Sotomayor Rivas	Cadete_Varones	Club deportivo social y cultural moutain bike Angol	CHILE	05:40:46.204	OK	0	0	0	12:00:00.000
3	Sofia Antonella	Cerda Fernandez	Damas_Open		CHILE	05:40:46.664	OK	0	0	0	12:00:00.000
4	Melissa	Elizalde Aldridge	Damas_Open	RiderZone	CHILE	05:40:47.142	OK	0	0	0	12:00:00.000
5	Jennifer	Garnica Lagos	Damas_Open		CHILE	05:40:47.562	OK	0	0	0	12:00:00.000
6	Ignacia belen	Salas Bañados	Damas_Open	Troylee	CHILE	05:40:47.930	OK	0	0	0	12:00:00.000
7	Constanza	Salgado Orue	Damas_Open		55988374403	05:40:48.275	OK	0	0	0	12:00:00.000
8	José	Gelcich Vergara	E_Baik_Open		CHILE	05:40:48.633	OK	0	0	0	12:00:00.000
9	Álvaro Nicolás	Montecinos Suazo	E_Baik_Open	MTB San Javier	CHILE	05:40:48.982	OK	0	0	0	12:00:00.000
10	Sergio	Palacios Pavez	E_Baik_Open	Momberg School	CHILE	05:40:50.001	OK	0	0	0	12:00:00.000
11	Adolfo Andres	Abarca Muñoz	Elite_Varones	Ridebike constitucion	CHILE	05:40:50.857	OK	0	0	0	12:00:00.000
12	Joaquin	Abrigo Martinez	Elite_Varones	Club Deportivo SakaSaka Racing Team	CHILE	05:40:51.403	OK	0	0	0	12:00:00.000
13	Pedro	Burns Contreras	Elite_Varones	Cardeluka racing team	CHILE	05:40:51.933	OK	0	0	0	12:00:00.000
14	Benjamín	Corral Gallardo	Elite_Varones	Propain / Club bmx las condes	CHILE	05:40:52.513	OK	0	0	0	12:00:00.000
15	Jose Ignacio	Feller Guzman	Elite_Varones		CHILE	05:40:53.052	OK	0	0	0	12:00:00.000
16	Ignacio	Pasten Román	Elite_Varones	Bicisport	CHILE	05:40:53.637	OK	0	0	0	12:00:00.000
17	Fernando	Riquelme Cardemil	Elite_Varones	Beclycles	CHILE	05:40:54.259	OK	0	0	0	12:00:00.000
18	Paulo	Avila Parra	Enduro_Open		CHILE	05:40:54.837	OK	0	0	0	12:00:00.000

Figura I.1: Interfaz de edición de cronometraje de la versión inicial

Figura I.2: Modal de edición de cronometraje de la versión inicial

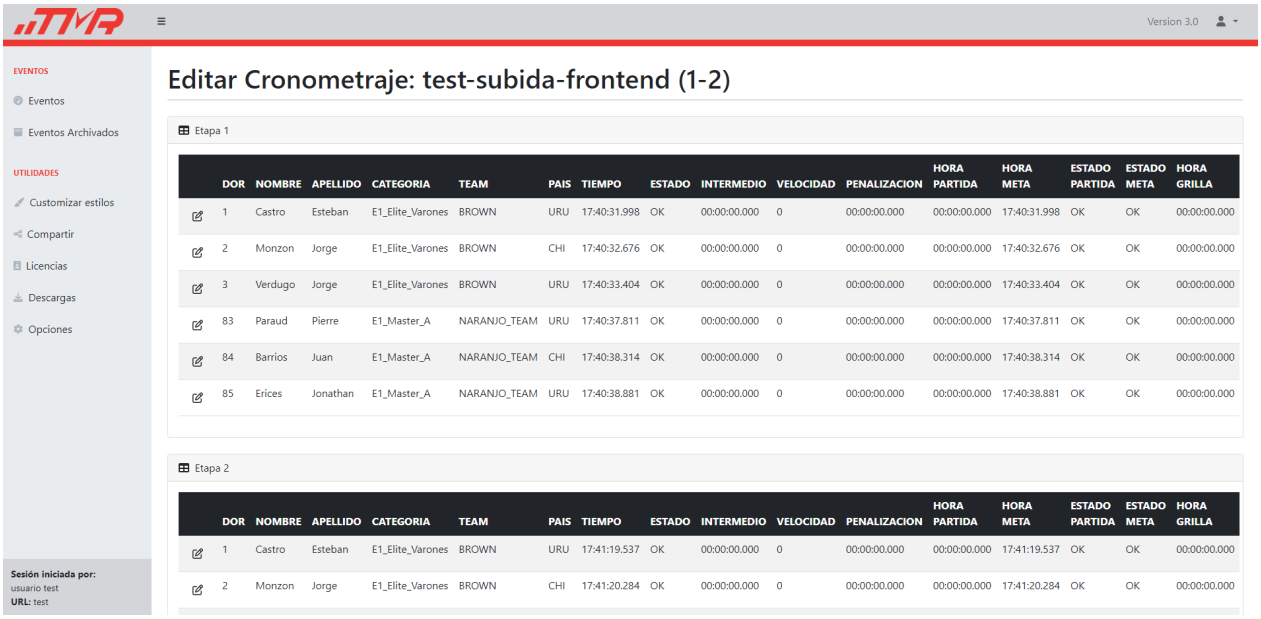


Figura I.3: Interfaz de edición de cronometraje de la versión final

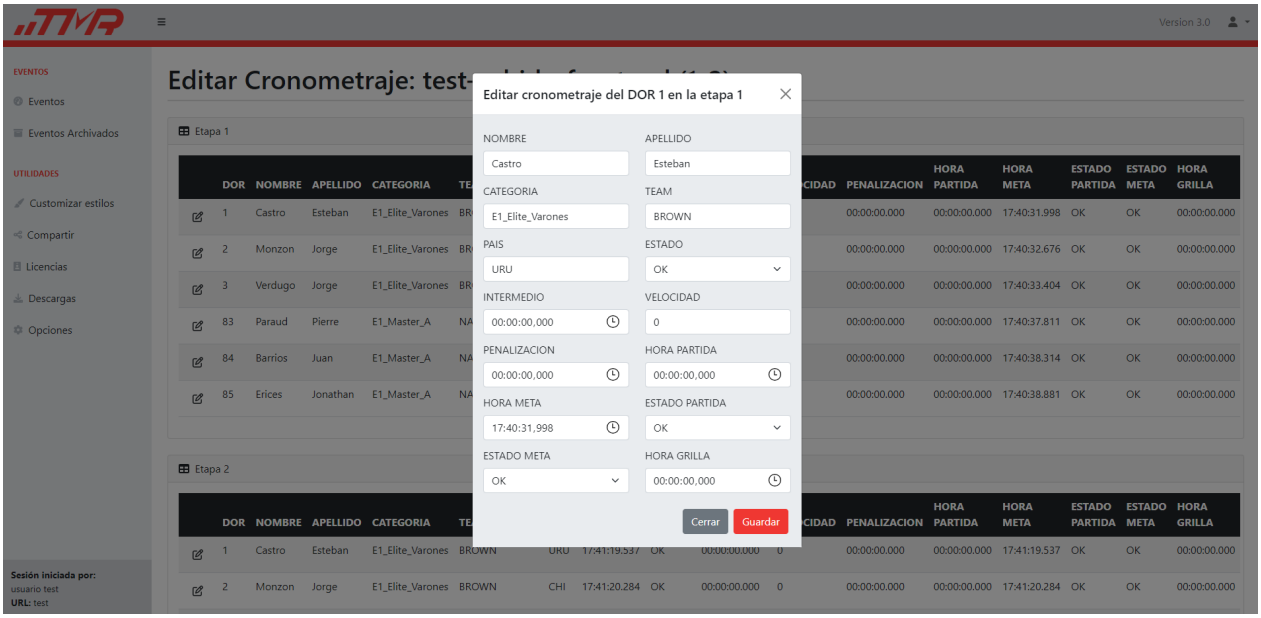


Figura I.4: Modal de edición de cronometraje de la versión final