Aplicación para la Gestión de Entrenamientos de Natación

Autor: Cristian Quevedo Suárez

Grado en Ingeniería Informática

Trabajo de Fin de Grado

Curso: 2018 - 2019

Tutor 1: Agustín Salgado de la Nuez

Tutor 2: Nelson Monzón López

Las Palmas de Gran Canaria 3 de junio de 2019

Agradecimientos

Antes de continuar, me gustaría aprovechar la oportunidad para agradecer a mis padres, a mis abuelos y a mi tío que siempre han estado apoyándome y dándome ánimos en los momentos más difíciles, ya que han hecho este camino mucho más llano.

Agradecer también a mis amigos que me han acompañado durante este periodo, los cuáles me ayudan a crecer día a día como persona.

Agradecer a mis tutores Agustín Salgado de la Nuez y Nelson Monzón López que me han ayudado durante todo el desarrollo del proyecto.

Muchas gracias a todos.

Resumen

En la actualidad se dispone de una gran cantidad de aplicaciones, tanto para móviles como para escritorio, para realizar un seguimiento de actividades deportivas, como el ciclismo o carreras de montaña. Sin embargo, no existen tantas aplicaciones para natación.

En este TFG se propone el desarrollo de una aplicación que permita a un entrenador o nadador llevar la gestión de los entrenamientos de su alumnado. Cada nadador o entrenador podrá ver el entrenamiento planificado, así como cronometrar las actividades que posteriormente podrá analizar mediante gráficas y valorar la evolución de su entrenamiento.

Abstract

Nowadays there are available many mobile and desktop applications, to make a follow up of sports activities, such as cycling or mountain tracks. However, nowadays does not exist any application with these features for swimming.

For this reason, it is proposed as Final Project of the Degree, the development of an application that allow a trainer to maintain the management of their students' trainings. Furthermore, the students can see their trainings, the time of those trainings and many graphics of those times.

Índice

Índice de Figuras

Índice de Tablas

1. Introducción

En la actualidad se dispone de una gran cantidad de dispositivos y aplicaciones para hacer un seguimiento de cualquier actividad deportiva. Las actividades como el ciclismo, carreras de asfalto o de montaña disponen de aplicaciones, tanto para móviles como para escritorio, que registran los entrenamientos y que facilitan la publicación e interacción de los datos recopilados.

Sin embargo, no existen aplicaciones de este tipo para el mundo de la natación, ya que es un deporte que por sus diferentes características dificultan que las aplicaciones anteriores puedan adaptarse fácilmente para el registro y gestión de los entrenamientos. Entre esas particularidades se destacan la amplitud de disciplinas como crol, espalda, mariposa, braza y estilos y las distancias para cada disciplina, que abarcan desde los 50 metros hasta los 1500 metros tanto en piscina corta (25 metros) o larga (50 metros).

Esta aplicación pretende abarcar estas tres etapas de forma que un entrenador o nadador pueda llevar la gestión de los entrenamientos. Cada nadador o entrenador podrá ver el entrenamiento planificado, así como cronometrar las actividades de dicho entrenamiento para posteriormente analizar, mediante gráficas, valorar la evolución de su entrenamiento, en aras de alcanzar unos objetivos que el mismo se haya propuesto.

1.1 Contexto y Motivación

Esta idea de proyecto surge debido a la poca informatización que posee este deporte en el mundo actual, en comparación con otros. Esto se refleja claramente cuando un nadador de un club de natación recibe sus entrenamientos a través de aplicaciones de mensajería como *WhatsApp*, siendo estas fotos, tomadas de la propia libreta del entrenador.

Por otro lado, la propia Federación Canaria de Natación a pesar de que dispone de una aplicación para la toma de tiempos, la cual se explicará más adelante, emite estos tiempos a través de un fichero PDF, teniendo los propios entrenadores tomar los tiempos a mano con un cronometro manual.

Debido a la situación actual que hemos descrito anteriormente, surge una necesidad de contar con una aplicación que permita a un entrenador gestionar los planes de entrenamiento, así como el cronometraje de las distintas actividades, haciendo más fácil la gestión en el día a día.

Otra aportación personal de este proyecto es el uso de tecnologías como *Docker, Jenkins* y *Amazon Web Services*, las cuales no las había utilizado previamente en la titulación y tenía un gran interés en aprenderlas para mejorar mi capacitación profesional.

2. Objetivos

2.1 Objetivos Académicos

El objetivo académico consiste en que el alumno realice un proyecto informático en el que se refleje los conocimientos y competencias que ha adquirido durante el transcurso del Grado en Ingeniería Informática. Este proyecto preparará al estudiante en su futuro como profesional, además de enriquecer su currículum.

Por otro lado, el estudiante ha seleccionado a un tutor que ha asesorado desde la elección del proyecto hasta la exposición del trabajo ante un tribunal universitario.

Estos objetivos se han consultado en este enlace

2.2 Objetivos Generales del Proyecto

Los objetivos generales que se han planteado en este proyecto son:

- Aplicar la metodología de ingeniería de software para documentar el proceso de desarrollo del proyecto, especificando el análisis, diseño, programación e implantación del sistema.
- Implementar un servicio de gestión que permita recopilar los entrenamientos y analizar la evolución de un conjunto de nadadores.
- Diseñar e implementar una interfaz de usuario que permita relacionar los conjuntos de datos de entrenamientos, añadiendo tanto los tiempos y objetivos alcanzados.
- Implementar un sistema de interacción del usuario con el graficado, que le permita visualizar la evolución del nadador o de los nadadores.
- Verificación y prueba del sistema sobre datos reales.
- Generar una herramienta que sea fácilmente modificable y ampliable en el futuro.

3. Competencias

A continuación, se expondrá las distintas competencias específicas que se encuentran presentes en este trabajo.

"IS01: Capacidad para desarrollar, mantener y evaluar servicios y sistemas software que satisfagan todos los requisitos del usuario y se comporten de forma fiable y eficiente, sean asequibles de desarrollar y mantener y cumplan normas de calidad, aplicando las teorías, principios, métodos y prácticas de la ingeniería del software"

Esta competencia se puede apreciar en el apartado de "*Desarrollo"*, donde se ha hecho uso del patrón MVC para realizar una REST API. Además, se ha hecho uso puntual de TDD y de BDD para poder mantener más fácil el código y cumplir con unos criterios mínimos de calidad.

"IS02 Capacidad para valorar las necesidades del cliente y especificar los requisitos software para satisfacer estas necesidades, reconciliando objetivos en conflicto mediante la búsqueda de compromisos aceptables dentro de las limitaciones derivadas del coste, del tiempo, de la existencia de sistemas ya desarrollados y de las propias organizaciones"

Esta competencia se cumple durante el análisis del proyecto, donde el alumno se ha informado de dichas necesidades al exponer a los distintos nadadores y entrenadores sobre su idea de proyecto, y estos indicando una serie de sugerencias e ideas sobre los requerimientos que les resultara útiles como usuarios de la aplicación.

"*IS03. Capacidad de dar solución a problemas de integración en función de las estrategias, estándares y tecnologías disponibles"*

Esta competencia se puede reflejar en el apartado de "*Tecnologías Utilizadas*", en las cuáles se puede apreciar el uso de tecnologías actuales como *Jenkins, Docker, Vue*, etc.

"IS04: Capacidad de identificar y analizar problemas y diseñar, desarrollar, implementar, verificar y documentar soluciones software sobre la base de un conocimiento adecuado de las teorías, modelos y técnicas actuales"

Esta competencia se refleja en el apartado de "*Metodología*", en el cual se ha utilizado un ciclo de vida incremental, aplicando un enfoque ágil tomando algunas prácticas de *Scrum* y *Extreme Programming*.

"IS06. Capacidad para diseñar soluciones apropiadas en uno o más dominios de aplicación utilizando métodos de la ingeniería del software que integren aspectos éticos, sociales, legales y económicos"

Esta competencia se puede observar durante todo el transcurso del proyecto, ya que esta aplicación busca saciar una necesidad social para un conjunto de la población que

practique un deporte como la natación y quiera llevar una gestión y seguimiento de su entrenamiento.

"TFG01: Ejercicio original a realizar individualmente y presentar y defender ante un tribunal universitario, consistente en un proyecto en el ámbito de las tecnologías específicas de la Ingeniería en Informática de naturaleza profesional en el que se sinteticen e integren las competencias adquiridas en las enseñanzas"

Esta competencia se cubre una vez el estudiante finalice el Trabajo de Fin de Grado con la presentación del proyecto ante los miembros del tribunal.

4. Estado del Arte

En esta sección abarcaremos las distintas aplicaciones en el mercado relacionadas con la gestión de entrenamientos y la natación. Además, se mostrará distintas API de grandes marcas de *relojes inteligentes*, que incluyen funcionalidades para la toma de tiempos de distintas actividades.

Antes de nada, he de resaltar que la natación es uno de los deportes con más variedad y dificultad que existe, debido a la gran multitud de pruebas que posee. Esto se debe en gran medida a la existencia de 5 tipo de pruebas (crol, espalda, braza, mariposa, estilos).

Estas pruebas abarcan los 50, 100 y 200 metros para todos los estilos, existiendo a su vez pruebas que van desde los 400 metros hasta los 1500 metros, solo para el estilo de *Crol*. He de resaltar que en las pruebas que incluyen los 4 estilos se realizan pruebas de 200 y 400 metros.

4.1 Aplicaciones en el Mercado

4.1.1 Strava

En primer lugar, empezaremos explicando lo que es Strava [1], a pesar de que es una aplicación que no está muy ligada a la natación, es un gran referente en cuanto a aplicaciones que se dedican a la gestión de tiempos para el ciclismo y las carreras de montaña.

Ilustración 1. Gráficos de Matej Mohoric en la aplicación Strava

Strava es una aplicación pensada para corredores y ciclistas, donde estos pueden subir las actividades que han realizado a través de relojes digitales, véase Garmin, Polar, Suunto, etc. De esta manera podrán llevar un control de los entrenamientos que han realizado durante los últimos meses, así como visualizarlos en gráficas, como se muestra en la ilustración1. Hay que destacar que dichos entrenamientos se pueden compartir con otros usuarios, ya que esta aplicación posee ligeras características sociales.

Por último, cabe destacar que Strava permite subir toda clase de deportes, como la natación. Sin embargo, solo permite subir cuánto tiempo y metros has nadado, por lo que no llega a satisfacer todas las necesidades de un nadador.

Ilustración 2. Entrada manual de datos en la aplicación Strava

4.1.2 Splash Meet Manager

Meet Manager [2] es un software de escritorio para la gestión de datos en competiciones de natación. Esta aplicación ofrece funciones comunes como crear listas de entrada de las series, una lista de resultados y otra de estadísticas, como por ejemplo el número de medallas por clubs, o records que se hayan batido, así como exportar dichos resultados en formato .csv o.xsl. Dicha aplicación es utilizada tanto por la Federación Canaria de Natación como por la Real Federación de Española de Natación.

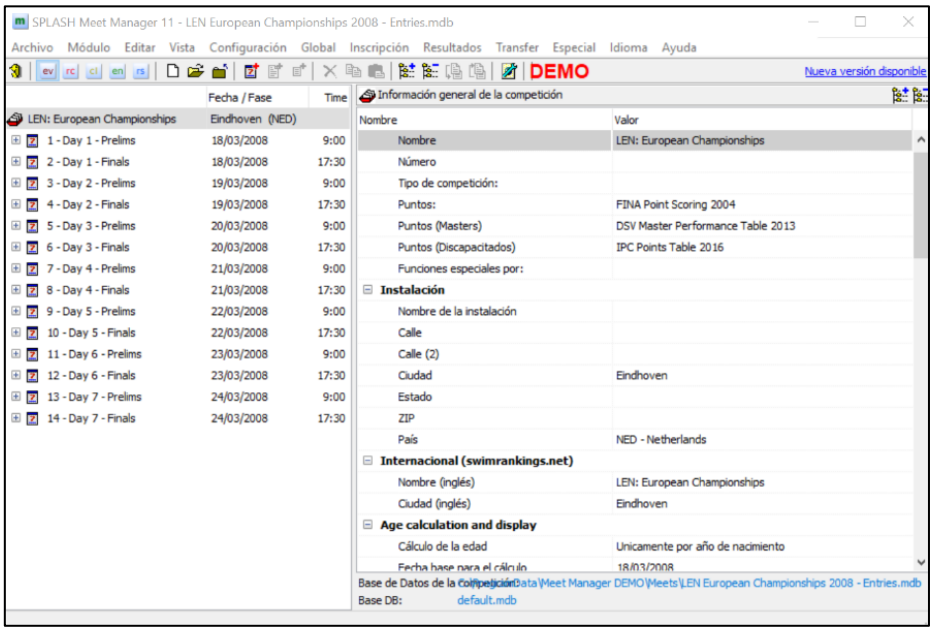


Ilustración 3. Splash Meet Manager

4.1.3 Meet Mobile

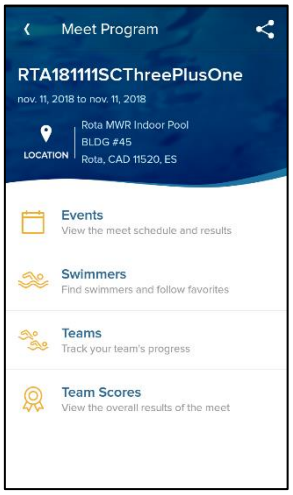


Ilustración 4. Meet Mobile

Meet Mobile es una aplicación móvil que se utiliza para la consulta de competiciones deportivas en la natación, en la cual se puede ver los nadadores que nadan dicha competición, así como los tiempos que realizan. Para poder visualizar dichos tiempos es necesario pagar una suscripción de 7€ al año.

Esta aplicación no ha tenido éxito en España ya que la última competición registrada data de noviembre de 2018. En países como EEUU, China y Australia la utilizan frecuentemente y prácticamente son los únicos países que registran competiciones en esta aplicación.

4.1.4 MySwimPro

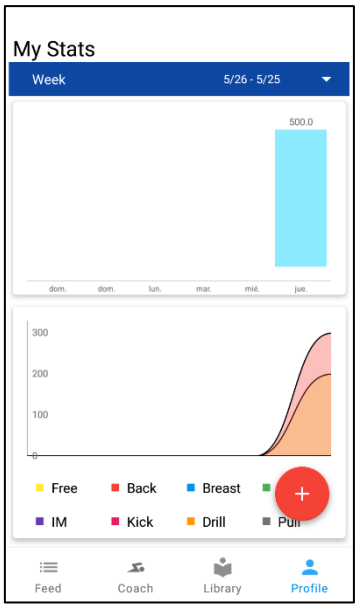


Ilustración 5. MySwimPro

MySwimPro es una aplicación móvil donde su principal funcionalidad consiste en crear tu propia rutina de ejercicios. También se puede visualizar, en tu perfil, la distancia total nadada y por estilos, las cuales se pueden filtrar por semana, mes y año. Asimismo, se puede registrar los mejores tiempos por cada prueba.

Posee vistas para la publicación de planes de entreno o la inclusión de material multimedia; para enseñar a mejorar la técnica de los distintos estilos o mejorar las cualidades de resistencia y velocidad. Sin embargo, para poder acceder a estas últimas funcionalidades se requiere de una suscripción premium con un coste de 7€/mes.

En la tabla 1 presentamos una comparativa de las funcionalidades que ofrece cada una de las aplicaciones anteriores respecto a la desarrollada en este TFG.

Funcionalidades	Strava	Splash Meet Manager	Meet Mobile	MySwimPro	TFG
Gestionar Plan de entrenamiento	-	-	-	-	✓
Gestionar Entrenamiento	✓	-	-	✓	✓
Gestionar Actividad	✓	-	-	✓	✓
Consultar gráficos	✓	-	-	✓	✓
Cronómetro	✓	-	-	-	✓
Calendario	✓	-	-	-	✓

Tabla 1. Cuadro-resumen de las distintas aplicaciones

4.2 API's en el Mercado

Durante el desarrollo del proyecto se planteó la posibilidad de implementar un mecanismo para la automatización de la toma de tiempos utilizando las API's que ofrecen algunas compañías de *relojes inteligentes.* En este apartado se comentarán algunas de las distintas API's analizadas y el motivo por el cuál no se han utilizado. Entre esas API's nos encontramos con:

4.2.1 Garmin

Es el fabricante líder en cuanto a relojes multideporte se refiere. Posee numerosas API's, donde cada una de ellas tiene una finalidad específica. En nuestro caso estábamos interesados en la *Garmin Connect API*, ya que permitía subir los entrenos que realizaba el usuario. Sin embargo, obtener esta API costaba unos 5,000$ obtenerla, por lo que era inviable el uso de esta.

Sin embargo, a mayo de 2019, parece ser que Garmin ha cambiado su política con respecto al precio de su API, y lo ha quitado de su página web dicho precio, pudiendo obtenerla gratis si se tiene una idea de negocio, que ellos consideren adecuada [3].

Hoy en día ya tienen visible su API, cuando el día que se realizó el análisis de mercado, no estaba públicamente accesible [4].

4.2.2 Suunto

Esta marca de relojes digitales posee su propia API llamada *movesense*. Esta API no tiene ninguna funcionalidad que satisface las necesidades de un nadador, ya que solo se limita a monitorizar temas de salud, como realizar un electrocardiograma [5].

4.2.3 Polar

Para finalizar, otros de los gigantes de la industria de los relojes multideporte, posee una API denominada *Polar Accesslink*, la cual se puede obtener de manera gratuita si se registra en *Polar Flow* y rellena un formulario indicando cual es la idea de su aplicación [6].

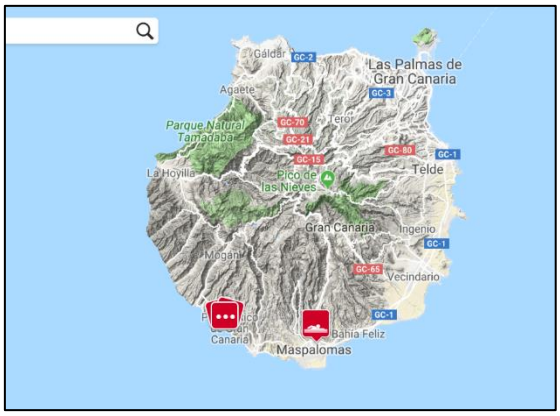


Ilustración 6. Polar Flow

Polar Flow es una especie de red social para usuarios de un reloj Polar donde se pueden subir los actividades y tiempo de los últimos seis meses. En la Ilustración 6 se puede ver una imagen de las últimas actividades relacionadas con la natación durante los últimos 6 meses en la isla de Gran Canaria. Ante la poca información recibida y la escasez de usuarios se decidió de prescindir de esta API.

5. Aportaciones

Después de realizar todo el análisis del mercado en el apartado anterior, podemos concluir que la natación a nivel amateur o profesional (sin incluir los CAR, Centros de Alto Rendimiento) dispone de muy pocas herramientas tecnologías que les facilite el día a día de su entrenamiento.

La aplicación desarrollada en este TFG pretende facilitar la gestión de entrenamientos en el deporte de la natación, así como la toma de tiempos y su posterior visualización en gráficos. De esta forma, se quiere impulsar el desarrollo de la natación a nivel amateur suponiendo una notable aportación en el plano social. Esto se puede apreciar cuando en una competición, los entrenadores de los clubes tienen que tomar los tiempos de su alumnado con un cronometro manual y apuntarlos a mano, debido a que, a pesar de que la Federación Canaria dispone de una aplicación como Splash Meet Manager que exporta los resultados en .csv, no se encuentra dispuesta a enviar dichos datos a los clubes, conformándose estos con un pdf publicado en la página oficial de la federación.

En el día a día de un entrenador, estos crean los entrenamientos a mano, para luego enviárselos a sus nadadores por una aplicación de mensajería, como WhatsApp. Además, durante los entrenamientos, los entrenadores suelen tomar tiempos con un cronometro. Normalmente no los suelen anotar por lo que se pierde una gran fuente de información muy valiosa para ambas partes.

Lo que se pretende aportar esta aplicación al mundo de la natación, es darle una herramienta a los entrenadores, donde estos puedan gestionar sus planes de entrenamiento, permitiendo a los entrenadores organizar su trabajo de manera mucho más eficiente y cómoda.

Los nadadores solo podrán ver los planes de entrenamientos definidos por sus entrenadores. Tanto los nadadores como los entrenadores podrán cronometrarse, con la particularidad de que el entrenador podrá tomar además los tiempos de los nadadores. El sistema de cronometraje permite tomar parciales durante el desarrollo de una prueba.

Por último, los datos recogidos se podrán observar en gráficas permitiendo que los usuarios puedan llevar un seguimiento de sus resultados.

6. Tecnologías Utilizadas

En ese apartado se especificará y explicará las distintas tecnologías, servicios y recursos que se han utilizado para el desarrollo de este proyecto.

6.1 Recursos de Hardware

Para desarrollar este proyecto se ha utilizado un portátil ASUS el cuál posee las siguientes características técnicas:

- Procesador: Intel i7-6500U @ 2.50GHz
- RAM: 8,00 GB
- Almacenamiento: 256 GB SSD

Por otro lado, cabe destacar que se ha utilizado un *Elastic Compute Cloud (EC2)*, el cual se procederá a explicar más adelante en esta sección a que posee las siguientes características:

- Modelo: t2. small
- CPU virtual: 1 Intel Xeon de alta frecuencia
- RAM: 2 GB
- Almacenamiento: Un EBS con una capacidad de 8 GB [7].

6.2 Amazon Web Services

En este apartado procederemos a explicar que es Amazon Web Services, así como los distintos servicios que se han utilizado de esta plataforma, durante el transcurso del proyecto.

6.2.1 ¿Qué es AWS?

Amazon Web Services se puede definir como una plataforma en la nube que posee entorno a más de 165 servicios, desde Almacenamiento, Machine Learning, Base de Datos, Informática, etc. Actualmente es la empresa con mayor uso con respecto a su competencia. [8]

Respecto a la infraestructura que posee AWS, esta dispone de 65 zonas de disponibilidad en 21 regiones geográficas situadas alrededor del mundo, como se puede observar en la Ilustración 7. [9]

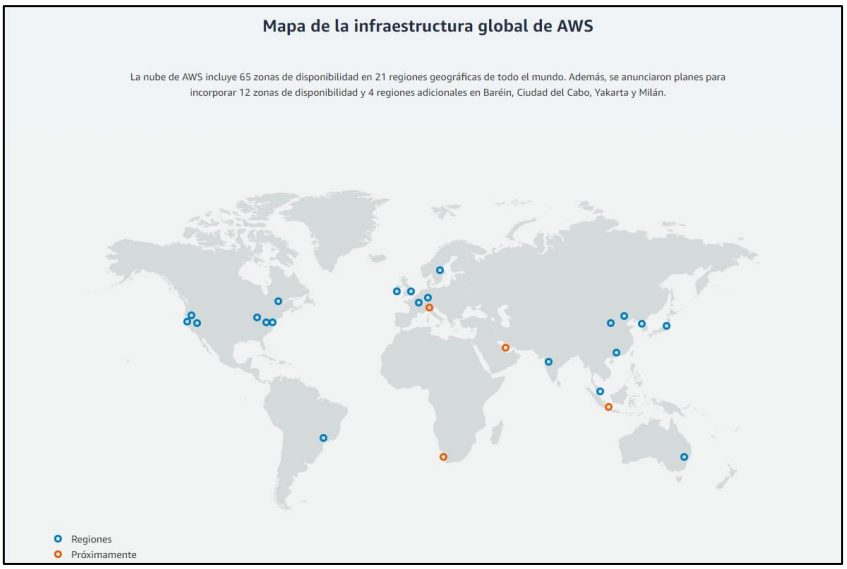


Ilustración 7. Mapa de la Infraestructura de AWS [9]

6.2.2 Elastic Compute Cloud

"Amazon EC2 reduce el tiempo necesario para obtener e iniciar nuevas instancias de servidor en cuestión de minutos, lo que permite escalar rápidamente la capacidad, ya sea aumentándola o reduciéndola, en función de sus necesidades. Amazon EC2 cambia el modelo económico de la informática al permitir pagar solo por la capacidad que utiliza realmente. Amazon EC2 les brinda a los desarrolladores las herramientas necesarias para crear aplicaciones resistentes a errores y para aislarlas de los casos de error comunes." [10]

Como bien se explica en la cita anterior, debido a la gran escalabilidad que posee un ec2, Amazon ofrece una gran variedad de instancias, las cuales poseen diferentes características. A continuación solo se hablará de las instancias tipo t2, ya que durante este proyecto se ha utilizado la instancia t2.small, debido a que es la que ofrecía la capa gratuita.

"Las instancias T2 son instancias de rendimiento ampliable que proporcionan un nivel base de rendimiento de la CPU con la posibilidad de ampliarse por encima del nivel básico" [7]

6.2.3 Amazon Relational Database Service

"Amazon Relational Database Service (Amazon RDS) es un servicio administrado que facilita las tareas de configuración, operación y escalado de una base de datos relacional en la nube. Proporciona capacidad rentable y de tamaño modificable y, al mismo tiempo, administra las arduas tareas de administración de las bases de datos, lo que le permite centrarse en sus aplicaciones y su negocio." [11]

Por otro lado, cuando se creó esta base de datos de este servicio, este nos pide seleccionar el tipo de motor de la base de datos, la cuales son Amazon Aurora, MySQL, MariaDB, Oracle, SQL Server y PostgreSQL.

En nuestro caso elegiremos PostgreSQL, debido a sus funciones de escritura y lectura en paralelo, debido a que es probable que nuestra aplicación realice muchas consultas en un poco periodo de tiempo, sobre todo cuando el usuario quiera consultar los tiempos en los gráficos. [12]

6.3 Recursos de Software

6.3.1 Docker

"Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos. Docker utiliza características de aislamiento de recursos del kernel Linux, tales como cgroups y espacios de nombres (namespaces) para permitir que "contenedores" independientes se ejecuten dentro de una sola instancia de Linux, evitando la sobrecarga de iniciar y mantener máquinas virtuales" [13] [14]

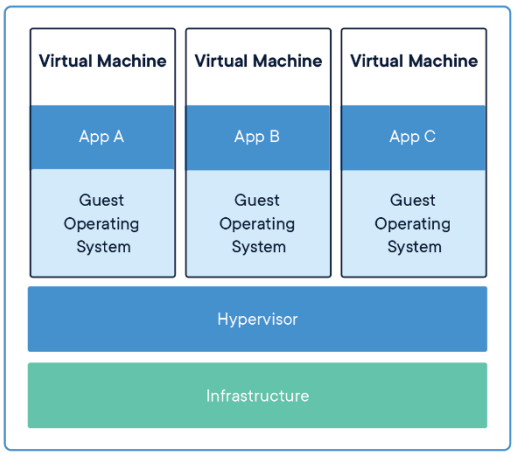


Ilustración 9. Estructura de una Máquina Virtual [13]

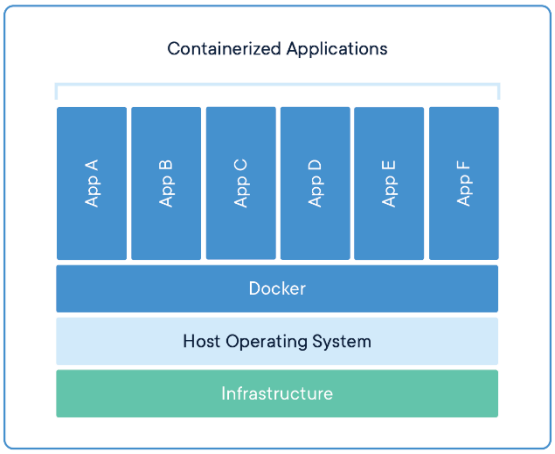


Ilustración 8. Estructura de un Contenedor Docker [13]

6.3.2 Jenkins

Jenkins es un servidor de automatización de código abierto que está escrito en Java. Su principal funcionalidad es la automatización de parte del proceso de desarrollo de software, aplicando la integración continua y el despliegue continuo.

Por otro lado, Jenkins presenta una nueva forma de trabajar, la cuál es a través de Pipelines. Un pipeline se puede definir como un conjunto de instrucciones automatizadas en la que podemos definir el ciclo de vida de un proceso de integración continua. Estos procesos se realizarán automáticamente una vez se haya realizado un *"push"* a una plataforma de control de versiones, como Github o BitBucket.

Jenkins propone como flujo de trabajo de pipeline (Ilustración 10), primero descargar o actualizar el código que se encuentra en el repositorio remoto, luego se realizará un "*build*" del proyecto, para posteriormente testear todo el código que se ha descargado. [15] [16]

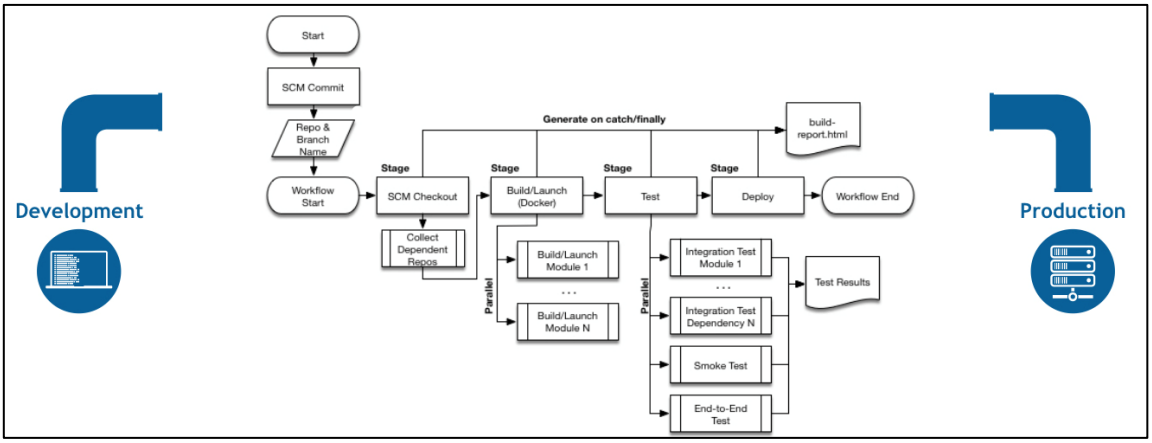


Ilustración 10. Flujo de Jenkins [15]

Por último, se desplegará la aplicación en el caso que se hayan completado todas las fases del ciclo de vida.

6.3.3 Vue.js

Es un framework progresivo basado en Javascript que se utiliza para desarrollar interfaces de usuario, asimismo como crear Single Page Application (SPA). Utiliza el patrón MVVM, y es orientado a componentes.

6.3.4 Vuex

Es una librería de Vue.js que sigue el patrón de gestión de estado, para el desarrollo de grandes aplicaciones. Su principal función se puede comparar con la de un almacén centralizado que está disponible para todos los componentes.

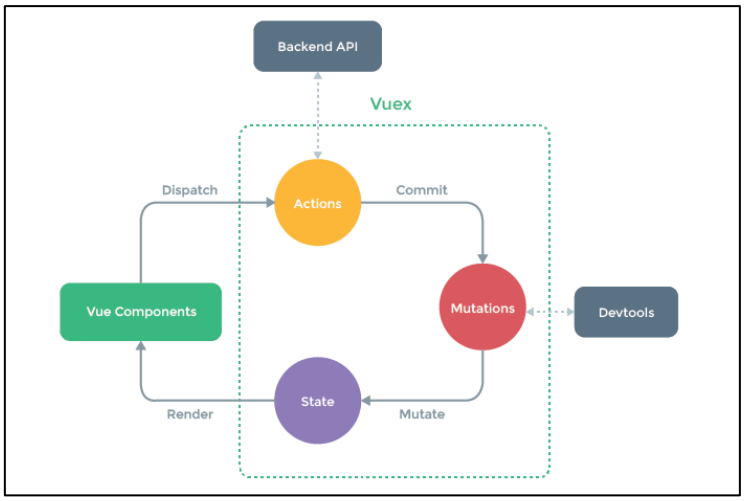


Ilustración 11. Flujo de Vuex [32]

6.3.5 Vue- Routing

Es una librería de Vue.js que se encarga de gestionar el enrutamiento de dicho framework, desde indicar que componentes son las vistas de la aplicación, los nombres de dichas rutas, hasta métodos como beforeRouteEnter o beforeRouteLeave que se encargan de proteger las rutas de la aplicación.

6.3.6 Vuetify

"Vuetify es un framework que combina la potencia del popular VueJs con la estética de Material Design. Permite acelerar el desarrollo de aplicaciones web complejas, incorporando una gran cantidad de componentes «listos para usar».

Vuetify se basa en el habitual sistema tipo «grid» para la ordenación del layout de la página. Dispone de una enorme librería de componentes que incluye desde elementos de formulario sencillos como botones, combobox, inputs, sliders, a componentes más avanzados típicos en aplicaciones Android como «cards» o «snackbars»." [17]

6.3.7 Vue-test-utils

Es una librería de Vue.js que se utiliza para el testeo de dicho framework a través de librerías de testeo como Jest o Mocha.

6.3.8 Javascript

"Es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos,3 basado en prototipos, imperativo, débilmente tipado y dinámico" [18]

6.3.9 HighsChart

"Highcharts es una biblioteca de gráficos multiplataforma basada en SVG y probada en el campo de batalla que se ha desarrollado activamente desde 2009. Facilita la adición de gráficos interactivos optimizados para dispositivos móviles a sus proyectos web y móviles. Cuenta con documentación robusta, capacidad de respuesta avanzada y soporte de accesibilidad líder en la industria" [19]

6.3.10 Axios

Es una librería de Javascript que se utiliza para realizar peticiones HTTP, además de gestionar llamadas asíncronas. Dicha librería se ha utilizado para realizar las llamadas a la API que hemos creado.

6.3.11 Django

"Django es un framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como Modelo–vista–template. Django pone énfasis en el re-uso, la conectividad y extensibilidad de componentes, el desarrollo rápido y el principio No te repitas (DRY, del inglés Don't Repeat Yourself)" [20]

6.3.12 Django REST-framework

Django REST Framework es una herramienta que nos permite construir una API REST API de una manera rápida y sencilla.

6.3.13 Pyjwt

Es una librería que permite el uso de JSON Web Token (JWT) en Python. Su principal uso, es de dar más seguridad al sistema de login de la aplicación, devolviendo al usuario un token que validará al usuario durante toda la aplicación.

6.3.14 Python

"Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico y es multiplataforma" [21]

6.3.15 Gherkin

Es un lenguaje que ayuda a la creación de tests de comportamiento. Entre los términos más importantes de su sintaxis nos encontramos lo siguiente:

- Feature: Se define una funcionalidad.
- Background: Se predefine los prerrequisitos de esa funcionalidad.
- Scenario: Se plantea un posible escenario de dicha funcionalidad.
- Given: Se presentan unas condiciones predefinidas de una funcionalidad.
- When: Cuando se realiza una determinada acción.
- Then: Resultado final de la funcionalidad [22].

6.3.16 Behave

Es una librería de Python que facilita la utilización de BDD (Behaviour-Driven Development).

6.3.17 Selenium

Es una librería que permite simular de forma automática la interacción de un usuario con una aplicación web.

6.3.18 Git

"Es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. Su propósito es llevar registro de los cambios en archivos de computadora y coordinar el trabajo que varias personas realizan sobre archivos compartidos" [23]

6.3.19 Github

GitHub es una plataforma donde se alojan una gran variedad de proyectos informáticos a través del uso del sistema de control de versiones Git.

6.3.20 Gitkraken

Es una aplicación de escritorio que sirve para la gestión tanto de repositorios que se encuentran tanto en local y remoto. Se ha utilizado sobre todo para resolver los distintos

merges que han ocurrido en la aplicación, ya que su interfaz gráfica deja claro que fichero se está aceptando.

6.3.21 Postman

Es una aplicación de escritorio que se utiliza para realizar peticiones HTTP, permitiendo modificar las cabeceras o insertar un objeto JSON en el cuerpo de la petición. Su principal función ha sido para realizar llamadas a las API.

7. Metodología

A continuación, pasaremos a explicar cuál ha sido la metodología empleada durante el desarrollo del proyecto. El proyecto también se ha basado en un modelo de ciclo de vida incremental e iterativo, incluyendo algunas prácticas que proponen las metodologías de desarrollo ágil, como *SCRUM* o *Extreme Programming*. A continuación, procederemos a comentar aspectos generales de estas metodologías, para posteriormente explicar cuáles son las características que se han escogido.

7.1 SCRUM

7.1.1 ¿Qué es?

"Scrum es un proceso de gestión que reduce la complejidad en el desarrollo de productos para satisfacer las necesidades de los clientes. La gerencia y los equipos de Scrum trabajan juntos alrededor de requisitos y tecnologías para entregar productos funcionando de manera incremental usando el empirismo" [24]

7.1.2 Roles

"Los roles principales en Scrum son el 'Scrum Master, que procura facilitar la aplicación de scrum y gestionar cambios, el Product Owner, que representa a los stakeholders (interesados externos o internos), y el Team (equipo) que ejecuta el desarrollo y demás elementos relacionados con él" [25]

7.1.3 Flujo de Trabajo

El desarrollo se realiza a través de iteraciones, denominadas Sprints, que poseen una duración de entre una y cuatro semanas donde se va desarrollando las distintas Historias de Usuario (funcionalidades del producto software).

Este conjunto de funcionalidades proviene del Sprint Backlog, que a su vez proviene del Product Backlog. El Product Backlog posee todas las historias de usuario a implementar,

mientras que en el Sprint Backlog se encuentran las historias de usuario que se deben desarrollar durante cada iteración.

Con lo explicado anteriormente, cabe añadir que durante este proyecto hemos obviado el apartado de roles, ya que es un proyecto individual y carece de sentido incluirlos,

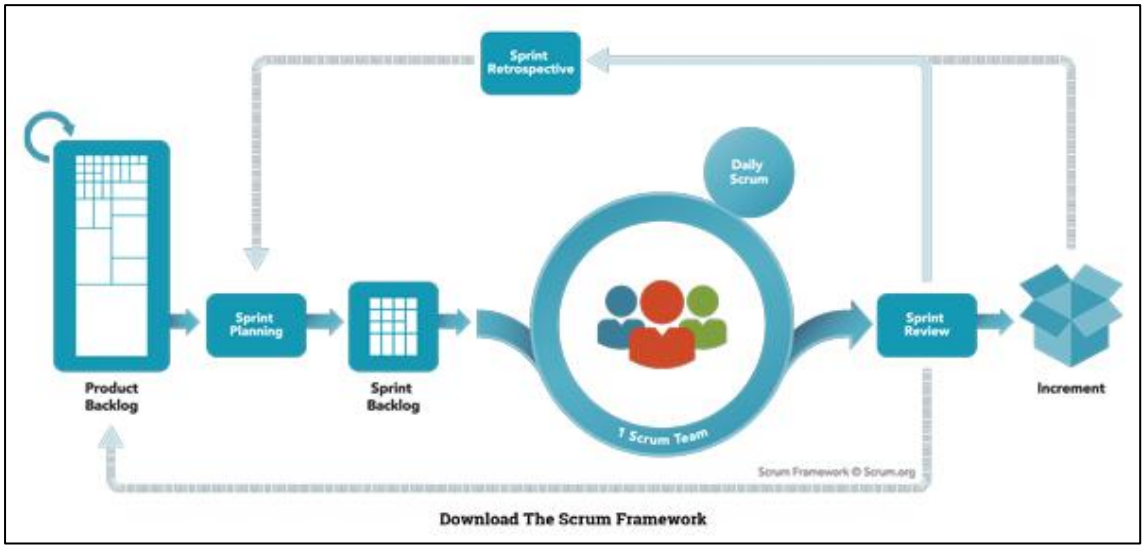


Ilustración 12. Flujo de Scrum [24]

quedándonos con los conceptos del Product Backlog y de los Sprints, con su correspondiente Sprint Backlog.

En el Product Backlog se reflejan las distintas historias de usuarios que se han identificado durante la fase de análisis. A continuación, se muestra el formato que seguirá dicha Pila. Para verla en su totalidad, consulte el anexo 2: Pila de producto.

Nombre	Estimación (horas)	Descripción	Criterio de Validación
Log in	12	Como usuario quiero poder iniciar sesión en la aplicación a través de la web	El usuario puede identificarse y acceder a la página principal de la web. En caso de fallo, le aparecerá los campos de inicio de sesión en rojo

Tabla 2. Estructura del Product Backlog

El desarrollo del proyecto se divide en tres Sprints. A continuación, se explicará brevemente el objetivo de cada Sprint:

- Sprint 0: Se realizó toda la parte de infraestructura de la aplicación, creando una instancia EC2, el cuál contendría un Jenkins y un Docker para realizar integración y despliegue continuo, así como crear y configurar los entornos del *front-end* y *back-end*.

- Sprint 1: En esta iteración se ha desarrollado el sistema de inicio de sesión y de registro del usuario, así como la gestión de los planes de entrenamiento, de los entrenamientos y de las actividades.

- Sprint 2: En esta parte se implementó la funcionalidad del cronómetro y la visualización de las gráficas. Además, también se utilizó este sprint para realizar ciertas mejoras a la interfaz gráfica.

En el apartado de "*Desarrollo*" se explicará más detenidamente cada Sprint.

7.2 Extreme Programming

Entre todas las prácticas que incluye se ha utilizado la de integración continua, la cual consiste en realizar frecuentemente integraciones en un proyecto para poder detectar lo más rápido posible los fallos que pueda tener éste, ya bien sea por un fallo de compatibilidad, o porque no se haya superado algún test.

Se han creado test unitarios, aplicando Test Driven Development en el *back-end*, mientras que en el *front-end* se ha realizado Behaviour Driven Development de manera puntual [26].

8. Planificación

A continuación, se expondrá una tabla ilustrativa de la planificación prevista para el desarrollo del proyecto, así como la duración real de las mismas.

FASES	*Duración estimada (horas)*	*Duración real (horas)*	*ACTIVIDADES*
Creación del Product Backlog	20	30	Análisis de la situación actual. (10 horas) Primera toma de requisitos de usuario y priorización de necesidades. (10 horas)
Sprint Zero	40	30	Definición de requisitos del software. (10 horas) Diseño de la base de datos. (15 horas) Diseño de la interfaz de usuario. (15 horas)
Fase de desarrollo de las distintas *releases*	200	252	Se estime el desarrollo de tres *versiones* que sucederán fases de implementación de la interacción con la base de datos, acceso de distintos usuarios en cada fase e implementación iterativa de la interfaz de usuario según las necesidades que se estimen. La estimación de puesta a producción es la siguiente: • Release 1 (80 horas) • Release 2 (60 horas) • Release 3 (60 horas)
Documentación / Presentación	40	60	Documentación de la aplicación desarrollada Edición de la memoria del proyecto realizado Preparación de la defensa pública

Tabla 3. Planificación

9. Desarrollo del Proyecto

En este apartado explicamos cómo se ha desarrollado la aplicación desde sus inicios, hasta su fase final, siguiendo la metodología que se ha comentada en el apartado anterior. A continuación, explicaremos en cada Sprint los pasos más destacables.

9.1 Análisis de la aplicación

Antes del desarrollo de la aplicación se llevó a cabo un análisis de las aplicaciones similares en el mercado para identificar las principales funcionalidades que debía tener nuestro producto. Este análisis abarca desde la creación de los casos de usos, para delimitar cuales serían las funcionalidades de la aplicación, para luego realizar una pila de producto. A partir de esta información se realizó un primer diseño de la interfaz de usuario, teniendo en cuenta que la aplicación también debe ser **responsive**, es decir, que la vista se adapte a todo tipo de dispositivos.

9.1.1 Pila de Producto

En este apartado se exponen las historias de usuario implementadas en el proyecto. Para verla en su totalidad consulte el Anexo 2: Pila de Producto.

	Nombre	Estimación (horas)	Descripción	Criterio de Validación
1	Puesta a punto del proyecto	8	Como **desarrollador,** juntaré tanto el Frontend como el Backend, para evitar tener problemas al final del proyecto al unir las dos partes	Que cuando ejecute tanto el Frontend como el Backend a la vez, la conexión de las dos partes no de ningún tipo de error
2	Log in	12	Como usuario quiero poder iniciar sesión en la aplicación a través de la web	El usuario puede identificarse y acceder a la página principal de la web. En caso de fallo, le aparecerá los campos de inicio de sesión en rojo
3	Sign Up	10	Como usuario quiero poder crearme una cuenta en la aplicación.	El usuario se crea una cuenta y accede a la página principal. Si no rellena los campos obligatorios, se mostrará los campos que fallan en rojo y no se creará ningún usuario.
4	Log out	8	Como usuario quiero poder cerrar sesión en la aplicación a través de la web	El usuario sale de la aplicación web y lo lleva a la página de Login
5	Ver perfil	10	Como usuario quiero poder ver mis datos personales.	El usuario puede ver satisfactoriamente los datos de su perfil.

6	Editar perfil	10	Como usuario quiero poder editar mi perfil de la aplicación, para actualizarlo o corregir erratas.	El usuario puede modificar satisfactoriamente los datos de su perfil. Si algún campo que es obligatorio se deja vació, se mostrará en rojo y no se dejará modificar
7	Ver Calendario	12	Como nadador quiero poder ver un calendario en el que aparezca los distintos entrenamientos	El nadador podrá ver un calendario con los entrenamientos que ha realizado o va a realizar. Si no posee ningún entrenamiento, vera el calendario vacío.
10	Iniciar Cronómetro	8	Como nadador quiero iniciar un crono para cronometrar un tiempo de mi marca personal o el de otro nadador	El nadador le da a iniciar cronometro y este se pone en marcha
11	Parar Cronómetro	0.5	Como nadador quiero parar de cronometrar	El nadador le da a parar el cronometro y este deja de contar el tiempo
12	Reiniciar Cronómetro	0.5	Como nadador quiero reiniciar el tiempo que marca en ese momento el cronometro	El cronometro obtiene un valor de 0:0.0 cuando el nadador le da al botón de reiniciar
13	Tomar sector en el Cronómetro	4	Como nadador quiero tomar el tiempo de un sector tantas veces como quiera	Cuando le dé al botón para coger el sector, me debe tomar el tiempo para ese sector
14	Subir Tiempo de Cronometro	12	Como nadador quiero subir mi tiempo a la plataforma, para luego poder consultarlo	Que cuando el nadador le dé al botón de subir el crono, se suba dicho tiempo
15	Visualizar tiempo registrado de una actividad	6	Como nadador quiero poder visualizar el tiempo que se ha registrado de una actividad y sus parciales	Cuando el nadador le dé al botón de visualizar, lleve al usuario a una página en la que muestre el tiempo y los parciales que ha realizado ese usuario de dicha actividad. En caso de que no posea un tiempo se le mostrará un mensaje de error.
17	Ver Gráficos de mis tiempos	6	Como nadador quiero poder ver mis tiempos para evaluar mi rendimiento a través de Gráficos	Cuando el nadador vaya a la sección de gráficos, vea sus tiempos en una gráfica
19	Crear una Actividad	8	Como entrenador quiero crear una actividad para	Cuando un entrenador termine el proceso de creación, se vea la nueva actividad creada. Si algún campo que es obligatorio se deja vació, se mostrará en rojo y no se podrá crear

20	Ver una Actividad	8	Como nadador quiero ver una actividad	Que le aparezca al nadador una lista con las distintas actividades del entrenamiento que he seleccionado
21	Modificar una Actividad	3	Como entrenador quiero modificar una actividad	Cuando un entrenador ha modificado la actividad se debe ver que el cambio se ha producido. En caso de que algún campo que es obligatorio se deja vació, se mostrará en rojo y no se dejará modificar.
22	Eliminar una Actividad	2	Como entrenador quiero eliminar una actividad	Cuando un entrenador le dé a eliminar, se elimina de la base de datos y además no debe aparecer más
23	Clonar una Actividad	2	Como entrenador quiero clonar una actividad	Cuando un entrenador le dé al botón de clonar se me debe crear una nueva actividad en la base de datos con los datos del elemento clonado. Además, me debe aparecer en la vista una vez clonado.
24	Crear un Plan de entrenamiento	8	Como entrenador quiero crear un plan de entrenamiento donde guardar mis entrenos	Que un entrenador se haya credo un nuevo plan de entrenamiento correctamente. Si algún campo que es obligatorio se deja vació, se mostrará en rojo y no se podrá crear
25	Ver un Plan de entrenamiento	8	Como nadador quiero ver mis distintos planes de entrenamiento	Que a un nadador le aparezca una lista con los distintos planes de entrenamiento que posee ese nadador
26	Modificar un Plan de entrenamiento	2	Como entrenador quiero modificar mi plan de entrenamiento	Cuando un entrenador haya hecho una modificación el cambio que se ha realizado se haya guardado y se muestre. En caso de que algún campo que es obligatorio se deja vació, se mostrará en rojo y no se dejará modificar.
27	Eliminar un Plan de entrenamiento	1	Como entrenador quiero eliminar un plan de entrenamiento	Cuando un entrenador confirme que quiero eliminar un Plan de entrenamiento, este se debe eliminar de la base de datos y no se debe mostrar
28	Clonar un Plan de entrenamiento	8	Como entrenador quiero poder clonar un plan de entrenamiento	Cuando un entrenador le dé al botón de clonar se me debe crear un nuevo plan de entrenamiento que contenga los entrenamientos y las actividades del elemento clonado, todo esto en la base de datos. Además, me debe aparecer en la vista una vez clonado.

29	Crear un Entrenamiento	8	Como entrenador quiero crear un entrenamiento para mi alumnado	Que un entrenador que haya credo un nuevo entrenamiento, donde se pueda crear actividades. Si algún campo que es obligatorio se deja vació, se mostrará en rojo y no se podrá crear
30	Ver un Entrenamiento	5	Como nadador quiero ver un entrenamiento	Que a un nadador le aparezca una lista con los distintos entrenamientos del plan de entrenamiento que he seleccionado
31	Modificar un Entrenamiento	3	Como entrenador quiero modificar mi entrenamiento	Que a un entrenador se le permita modificar las características de un entrenamiento. En caso de que algún campo que es obligatorio se deja vació, se mostrará en rojo y no se dejará modificar.
32	Eliminar un Entrenamiento	1	Como entrenador quiero eliminar un entrenamiento	Cuando un entrenador le dé al botón de eliminar, dicho entrenamiento debe de haber sido eliminado de la base de datos y no debe aparecer en la interfaz
33	Clonar un entrenamiento	4	Como entrenador quiero clonar un entrenamiento	Cuando un entrenador le dé al botón de clonar se me debe crear un nuevo entrenamiento que contenga las actividades del elemento clonado, todo esto en la base de datos. Además, me debe aparecer en la vista una vez clonado.
47	Implementar IU general de la web	20	Como **desarrollador,** quiero implementar una interfaz de usuario que se ajuste su contenido en cualquier dispositivo	Que el usuario pueda ver el contenido de la aplicación de manera que la aplicación se ajuste al ancho y altura del dispositivo que se encuentra
48	Desplegar Aplicación en AWS	20	Como **desarrollador** quiero que mi página web este en la red, para ello utilizaremos AWS para desplegar dicho producto	Cualquier usuario con acceso a internet debe poder acceder a la aplicación
	Horas totales	**216**		

Tabla 4. Pila de Producto

9.1.2 Casos de usos

A continuación, expondremos los casos de uso de los distintos roles que tiene la aplicación.

Casos de uso de Usuario

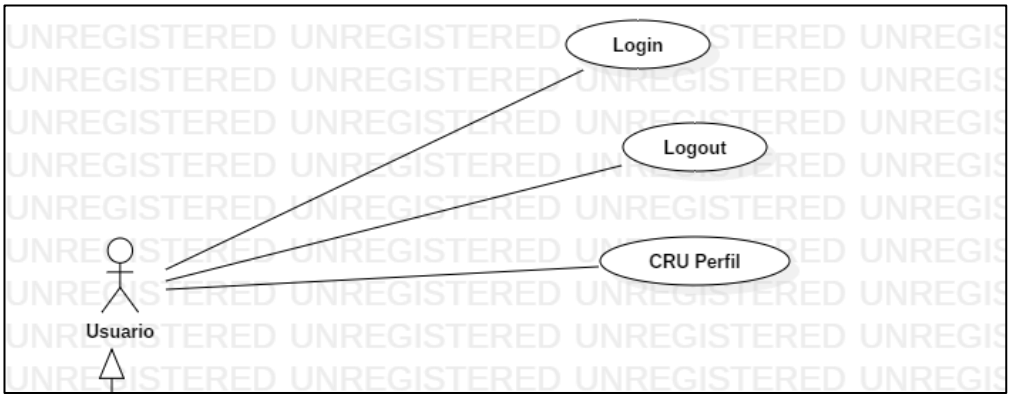


Ilustración 13. Diagrama de Casos de Uso para todos los usuarios

Nadador

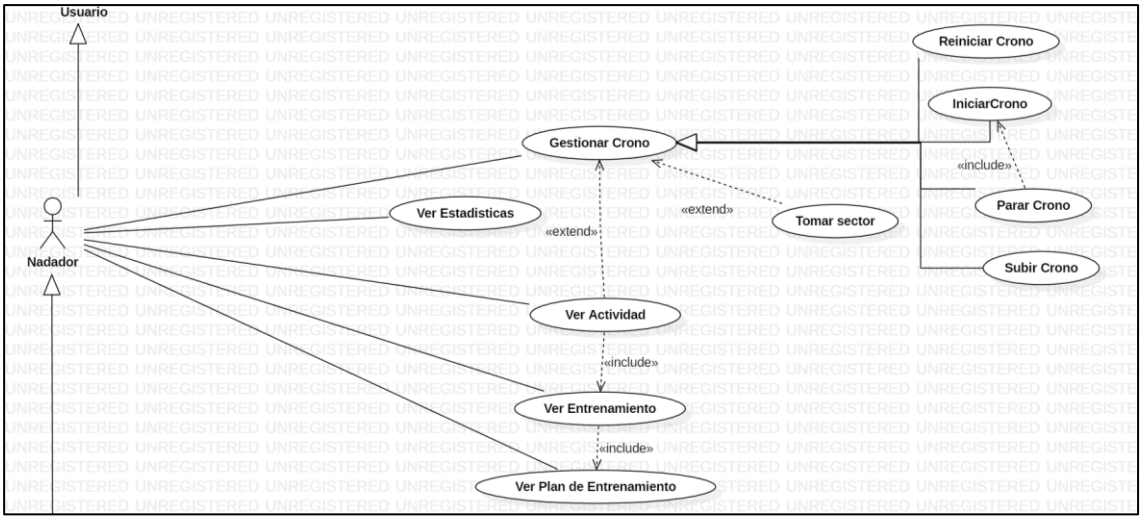


Ilustración 14. Diagrama de Casos de Uso para todos los nadadores

Entrenador

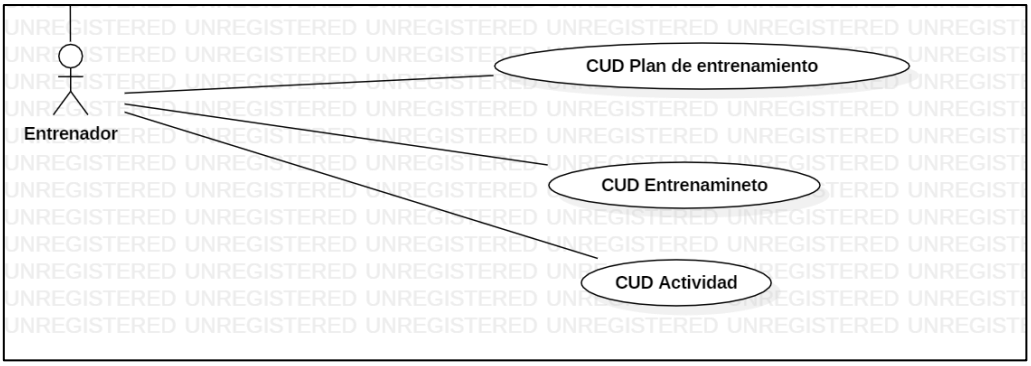


Ilustración 15. Diagrama de Casos de Uso para todos los entrenadores

Para ver todos los casos de usos, consulte el Anexo 1: Casos de Uso.

9.1.3 Diagrama de servicios de Amazon Web Services

A continuación, se muestra la infraestructura de la aplicación. En el EC2 tenemos implementado Jenkins y Docker, los cuales se encargan de todo el proceso de integración y despliegue continuo. Por otro lado, se encuentra el RDS, la base de datos de la aplicación que utiliza PostgreSQL. Para finalizar, se puede apreciar un Elastic Block Store (EBS) que es la unidad de almacenamiento de nuestro EC2.

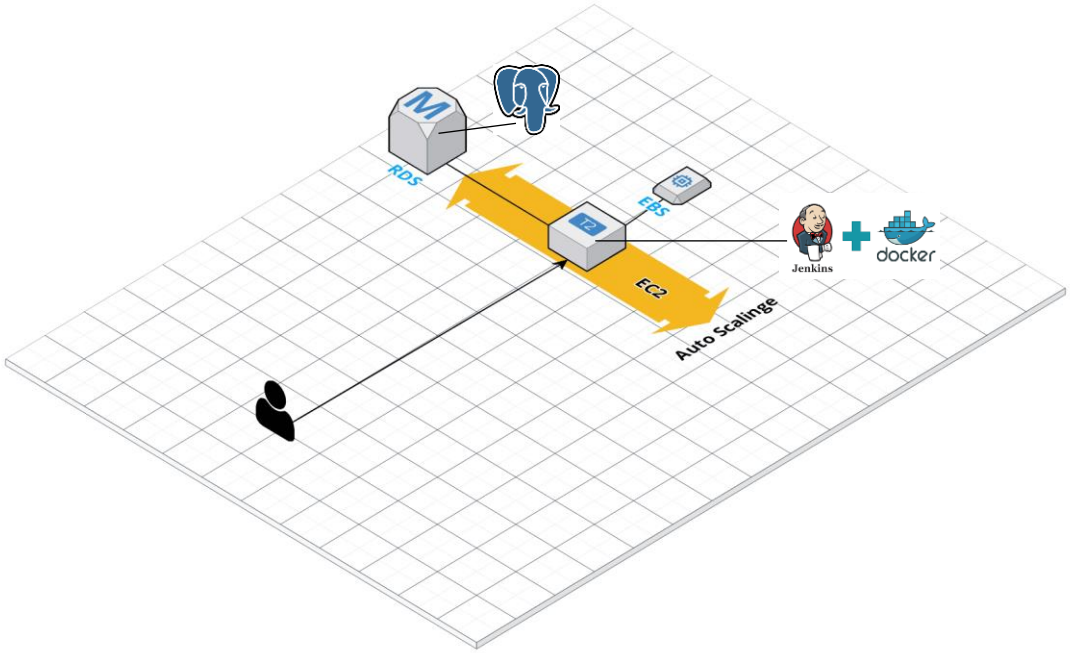


Ilustración 16. Diagrama de Infraestructura de AWS

9.1.4 Estructura de la Base de Datos

En este apartado explicaremos como se ha estructurado el modelo de la base de datos de cada recurso de la aplicación, explicando los campos de cada tabla.

User: Esta tabla contiene a los distintos usuarios de la aplicación, los cuales tiene por parámetro un **nombre** de usuario, una **contraseña**, una dirección de **correo electrónico**, una **descripción** (opcional), el **tipo** de usuario (nadador o entrenador) y el **club** al que pertenece.

PlanTrainings: Esta tabla contiene todos los planes de entrenamiento. En ella existe el campo **user** que define a los usuarios que pueden verla, luego posee el campo **nombre** y **descripción** siendo este último opcional.

Trainings: En esta tabla se define un entrenamiento que está relacionado con un "*PlanTraining*"**,** así como el **nombre** del entrenamiento, la **fecha** del entrenamiento y la **descripción** siendo este último opcional.

Activities: En esta tabla se define una actividad que está relacionado con un "*PlanTraining*" y "*Training*". *Por otro lado, en una actividad se debe definir el número de* ***series*** *y* ***metros***, el ejercicio (punto muerto, piernas, palas, etc.), el estilo de natación, el tipo (calentamiento, entreno, vuelta a la calma) y el ritmo.

Chronos: En esta tabla se definirán los cronos que ha hecho cada usuario, por ello habrá un campo **usuario** indicándolo. Asimismo, se registra el **tiempo** que ha realizado el usuario, a que **actividad** pertenece dicho tiempo y la **fecha** en el que se registró el tiempo

Phases: En esta tabla se registran las fases de un cronómetro. En ella se encontrará el **tiempo** de esa fase, a que "*Chrono*" pertenece, así como los **metros** que se han realizado cuando se registrado dicha fase. Hay que recalcar que los metros tienen valores que son múltiplo 100, debido a que por tónica general se cogen los tiempos cada 100 metros.

Para un mayor detalle puede consultar el Anexo 3: Diagrama BBDD.

9.1.5 Diseño de Interfaz de Usuario

Antes de iniciar el desarrollo de la aplicación, se realizó unos bocetos de la interfaz de usuario para tener un primer concepto. Hay que recalcar que esta interfaz estuvo sujeta a cambios durante todo el desarrollo del proyecto, debido a la mejora de ésta y a la aparición de nuevas historias de usuario.

Para verla en su totalidad consulte el Anexo 4: Mockup.

9.2 Sprint 0

El objetivo de este Sprint consistía en montar toda la infraestructura en AWS, además de crear y configurar tanto el *front-end* como el *back-end*. Para saber más sobre este proceso consulte los siguientes anexos:

- Anexo 5: Implementación de la Infraestructura en AWS.
- Anexo 6: Creación de los contenedores Docker.
- Anexo 7: Estructura y configuración de los proyectos.

9.3 Sprint 1

En este Sprint comienza el desarrollo de la aplicación. Se realizaron las historias de usuario relacionadas con el acceso y registro de usuario, así como la gestión del plan de entrenamiento, entrenamiento y actividades.

9.3.1 Login

En este apartado solo comentaremos el uso de JWT para realizar el *login* de la aplicación. Esta tecnología brinda más seguridad a nuestra aplicación, ya que verifica una autenticidad de los datos, velando en nuestro caso de la autenticidad del usuario que se ha registrado.

A continuación, se explicará con más en detalle la estructura de un token y el diagrama de flujo a la hora de enviar un token.

Un usuario se registra en la aplicación a través de una petición POST. Esta petición le llega al servidor, el cual comprobará si existe el usuario con los datos que le ha proporcionado éste. Una vez comprobado que el usuario existe, el servidor crea el token (string) y se lo devuelve al usuario, pudiendo este ya acceder a la aplicación.

Cada vez que el usuario quiera hacer una consulta al servidor, envía en la cabecera el token que le ha dado el servidor, comprobando el servidor la autenticidad del token. Si el token es válido, el servidor suministrará la información que ha pedido; en caso contrario, caducará la sesión del usuario.

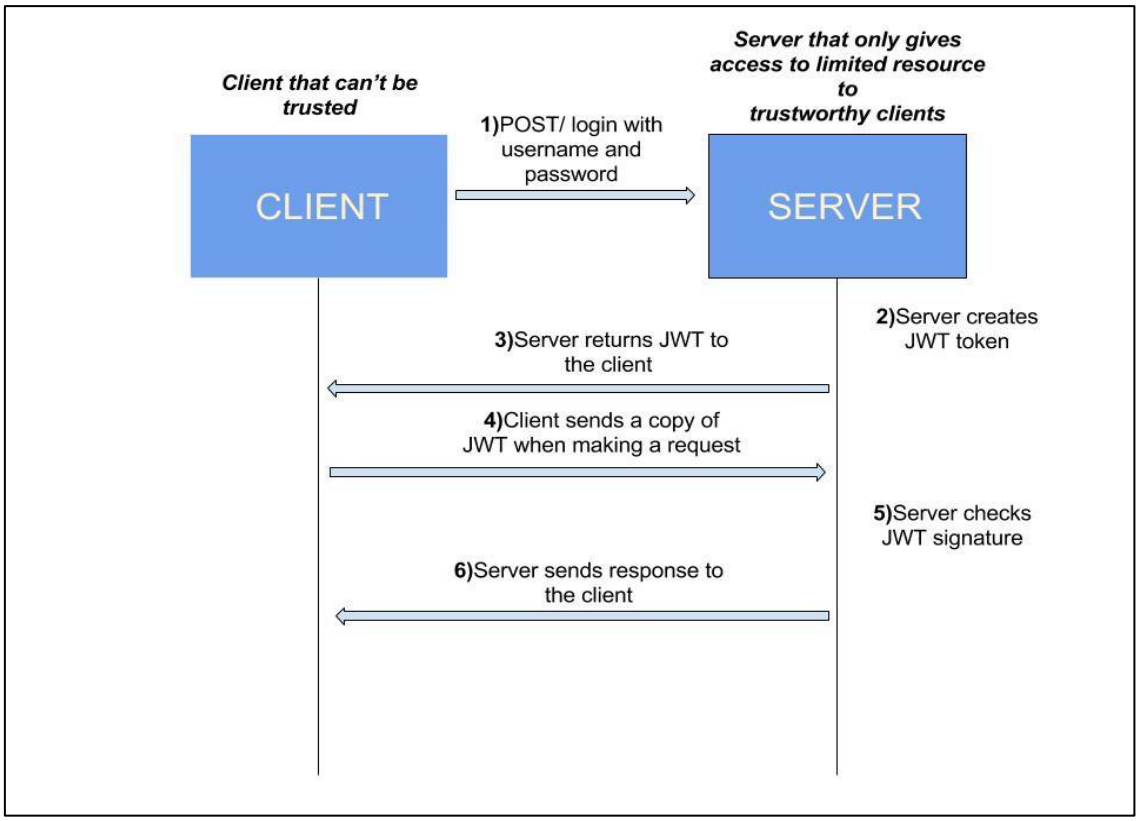


Ilustración 17. Flujo de JWT [33]

Un token se encuentra divido en tres partes por un punto (Ilustración 34), que pasaremos a definir a continuación:

Ilustración 18. Estructura de un token [33]

Header: Es un objecto JSON que contiene el tipo del token, el cuál va a ser siempre JWT y por otro lado el algoritmo de codificación, como SHA256 o SHA512.

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

Ilustración 19. Token header [27]

Payload: Es un objecto JSON que contiene los atributos de la entidad, que en general suele ser el del usuario.

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "admin": true
}
```

Ilustración 20. Token payload [27]

Signature: Para crear el signatura con esta librería, necesitaremos llamar a la función encode(), la cual nos pide como parámetros el payload que hemos generado previamente, y un valor secreto, un objeto String que puede poseer cualquier valor.

jwt.encode({'sub': 1234567890, 'name': 'John Doe', 'admin': true}, 'secret')

Hay que destacar que para decodificar el token basta con utilizar la función jwt.decode(), al cual solo deberemos pasarle el token por una petición HTTP. [27]

9.3.2 Gestión del Plan de entrenamiento

En este apartado solo expondremos como se ha realizado la implementación de la gestión de un plan de entrenamiento, ya que existe una gran similitud entre esta estructura con respecto a la de un entrenamiento y actividad.

En la **Tabla 11** explicaremos los distintos permisos que posee cada usuario, así como los botones que podrá ver cada uno con respecto a las funcionalidades que tienen acceso.

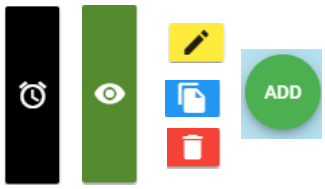
Funcionalidad	Nadador	Entrenador
Plan de entrenamientos	R	CRUD
Entrenamientos	R	CRUD
Actividades	R	CRUD
Cronometraje	CRD	CRD
Guía	**Botones**	
C = Create R = Read U = Update D = Delete		

Tabla 5. Guía de Funcionalidades por tipo de usuario

Para ver los distintos planes de entrenamiento se creó un componente modular llamado TrainingCard, en el cual a través de la etiqueta reservada de Vue.js llamada *slot*, podemos insertar componentes, dentro de ese componente, como se observa en la Ilustración 37.

```
<training-card v-for="plantraing in plantraings"
  class="trainingCard"
  :key="plantraing.id"
  :url="url"
  :params="{idPlan:plantraing.id}"
  :methods="methods"
  :training="plantraing"
  :newUpdate="params"
  :message="'Modify Plan Training'"
  >
  <template v-slot:modify>
    <modify-form
    @updateInstance="updateInstance"
    :plantraining="plantraing"/>
  </template>
</training-card>
```

Ilustración 21. Template mostrando Training Card

A este componente se le pasa por parámetros, denominados *props, que* son los que poseen dos puntos con el nombre de la variable (*:training),* los métodos clonar, modificar y eliminar. Estos métodos se importan a través del mapActions de Vuex.

```
export default {
  name: 'Tabs',
  components: {
    TrainingCard,
    ModifyForm,
    draggable,
    Chart
  },
  data () {
    return {
      url: 'TRAININGS',
      loading: false,
      drag: false,
      params: {},
      methods: {
        clone: (plantraining) => this.clonePlanTrainings(plantraining),
        update: (params) => this.modifyPlanTraining(params),
        delete: (id) => this.deletePlanTrainings(id)
      }
    }
  },
  methods: {
    ...mapActions({
      getPlanTrainings: GET_PLANTRAININGS,
      clonePlanTrainings: CLONE_TRAINING_PLAN,
      deletePlanTrainings: DELETE_TRAINING_PLAN,
      modifyPlanTraining: MODIFY_TRAINING_PLAN
    }),
    updateInstance (value) {
      this.params = value
    }
  },
```

Ilustración 22. Código Javascript del componente Tab

A continuación, se mostrará los métodos más importantes que se han implementado:

Clonar

El método **clonar** realiza una copia en cascada de un elemento (ya sea plan de entrenamiento, entrenamiento o actividad). En la Ilustración 38 podemos observar el código que clona un plan de entrenamiento. No solo se copia los datos del plan de entrenamiento, sino también los entrenamientos y ejercicios que éste contenga.

```
[CLONE_TRAINING_PLAN]: async ({ commit }, params) => {
  try {
    let post = {
      name: params.name,
      description: params.description,
      user: params.user
    }

    let planTrainingCloned = await planTrainingClient.addPlanTraining(post)
    let trainings = await trainingClient.getTrainings(params.id).catch(() => {
      commit(ADD_TRAINING_PLAN, planTrainingCloned.data)
    })

    trainings.data.forEach(async (training) => {
      let request = {
        plantraining: planTrainingCloned.data.id,
        name: training.name,
        description: training.description,
        timetraining: training.timetraining
      }

      let clonedTraining = await trainingClient.addTraining(planTrainingCloned.data.id, request)
      let activities = await activityClient.getActivities(training.plantraining_id, training.id).catch((error) => {
        return error
      })
      if (activities.data !== undefined) {
        cloneActivities(planTrainingCloned.data.id, clonedTraining.data.id, activities.data)
      }
    })
    commit(ADD_TRAINING_PLAN, planTrainingCloned.data)
  } catch (error) {
    return error
  }
```

Ilustración 23. Acción de clonación (Vuex)

Añadir

Para el método añadir hemos tenido que crear un diálogo, el cual se activa una vez se ha realizado un click en el botón añadir. Lo que se muestra no es más que el componente AddDialog, el cual es otro componente modular que nos permite insertar el componente que posee el formulario del plan de entrenamiento.

```
<floating-button v-if="user.userType === 2"
  @click.native="dialog = !dialog">
  add
</floating-button>
<add-dialog
  :message="'Add Plan Training'"
  :dialog="dialog"
  @isActivated="isDialogActivated">
    <template v-slot:text>
      <plan-training-form
        v-if="dialog"
        @plantraining="newPlanTraining"
        :userPlan="user"
        />
    </template>
    <template v-slot:buttons>
      <v-btn color="blue darken-1" flat @click="closeDialog()">Close</v-btn>
      <v-btn color="blue darken-1" flat @click="saveDialog()">Save</v-btn>
    </template>
```

Ilustración 24. Template del botón añadir

9.4 Sprint 2

En este último Sprint se realizaron las historias de usuario relacionadas con el Cronómetro y la Visualización de los datos en una gráfica. A continuación, se expondrán como se han implementado.

9.4.1 Cronómetro

Para realizar el cronómetro hemos tenido que crear dos componentes por separado: (1) se encarga de la lógica de los botones y (2) apariencia del cronómetro, junto con la funcionalidad de subir el tiempo del cronómetro y sus fases.

Para realizar la interfaz del cronómetro nos hemos basado en el css en este enlace.

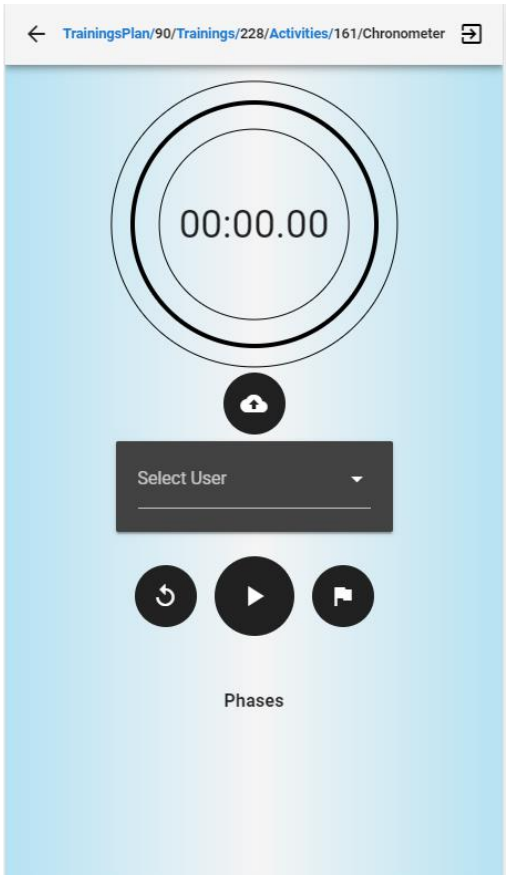


Ilustración 25. Vista Cronómetro

Lo más destacable de esta sección es la lógica que presenta la subida de un tiempo a la base de datos, ya que primero se comprueba que el usuario ya dispone de un tiempo en esa actividad, si lo posee le aparecerá un diálogo de confirmación advirtiéndole de que para subir ese tiempo debe borrar el anterior.

```
async uploadChrono () {
  try {
    this.loading = true
    if (this.user.userType === 2 && this.userChrono == 0) {
      alert('Select a User')
      this.loading = false
      return
    }
    let params = {
      user: this.user.id,
      time: this.chronoString,
      activity: this.$route.params.idActivity,
      timechrono: Date.now()
    }
    if (this.user.userType === 2) {
      params.user = this.userChrono
      let otherUserChrono = this.chronos.filter(chrono => chrono.user_id === this.userChrono)
      let adviceOther
      if (otherUserChrono.length > 0) {
        adviceOther = confirm('The User have a time for this activity.\nIf you continue you will delete it\n' +
        'Are you sure?')
        if (adviceOther) {
          await this.deleteChrono(otherUserChrono[0].id)
        } else {
          this.loading = false
          return
        }
      }
```

Ilustración 26. Lógica de subida de tiempos I

Esta comprobación se realiza por ambos lados, pero cambia una condición a la hora de comprobarlo debido a que un entrenador puede elegir a quién pertenece dicho tiempo.

En esta última fase se procede a añadir el tiempo del cronómetro, así como sus fases.

```
    let response = await this.addChrono(params)
    if (this.phases.length > 0) {
      this.phases.forEach((phase, index) => {
        let request = {
          timePhase: phase,
          chrono: response.id,
          meters: index * 100 + 100
        }
        this.addPhase(request)
      })
    }
    this.loading = false
  } catch (error) {
    this.loading = false
    return error
  }
```

Ilustración 27. Lógica de subida de tiempos II

9.4.2 Visualización de Datos

Para visualizar los datos se ha creado un componente llamado Chart, en el que a través de radiobuttons, le damos la opción al usuario de cambiar de tipo de gráfico, diferenciando los metros realizados durante el entrenamiento de los tiempos que ha realizado los nadadores durante las pruebas 100 Crawl y 100 Backstroke.

Por otro lado, se ha instalado la librería HighchartsVue, que proporciona por defecto un componente que se encarga de realizar las gráficas dependiendo de los parámetros que le pasemos a través de props. [28]

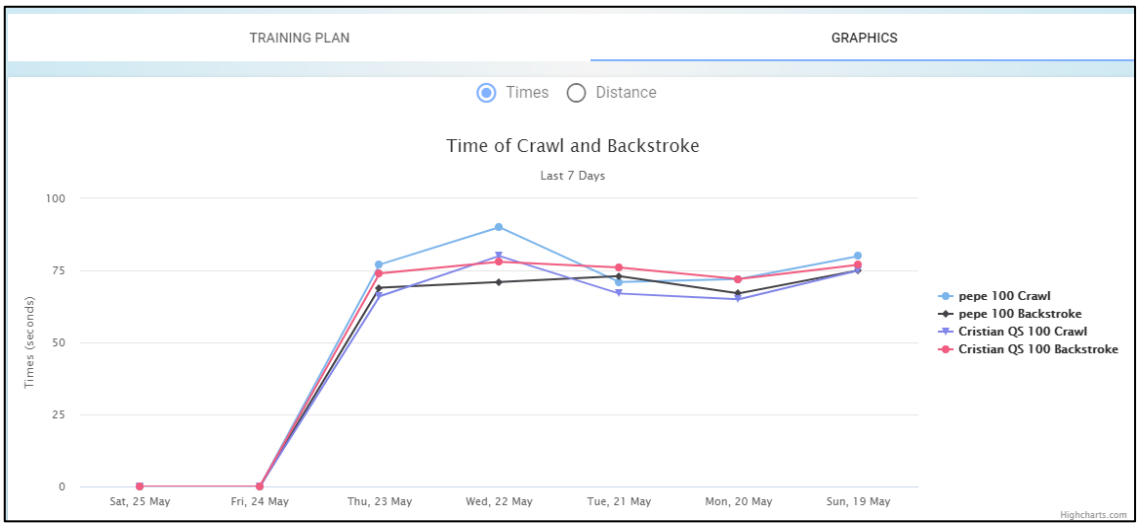


Ilustración 28. Gráfica de Tiempos

Ilustración 29. Gráfica de Metros

Hay que puntualizar que debido a la gran diversidad de datos a representar y dificultad a la horas de interpretar un amplio volumen de datos se ha decidido que estos gráficos sólo aparezcan dos usuarios: cris@gmail.com y pepe@gmail.com.

Cada vez que se crea el componente, obtenemos todos los tiempos de las actividades por el identificador del usuario. Una vez realizado esto, filtramos estos tiempos, seleccionando aquellos que se encuentren en un intervalo de 7 días, tomando como referencia el día en el que se encuentra el usuario.

```
async created () {
  this.loading = true
  let users = this.usersClub.filter(user => user.value === 1 | user.value === 2)
  for (let i = 0; i < users.length; i++) {
    let today = Date.now()
    let weekBefore = today - 604800 * 1000
    let weekAfter = today + 604800 * 1000
    let chronos = await this.getChronoActivity({ userid: users[i].value })
    chronos = chronos.filter(chrono => weekBefore <= chrono.timechrono && chrono.timechrono <= weekAfter)
    let chronosCrawl = chronos.filter(chrono => chrono.activity__style == 'Crawl' &&
    chrono.activity__meters == 100 && chrono.activity__series == 1 && chrono.activity__exercise == 'Normal')

    let chronosBack = chronos.filter(chrono => chrono.activity__style == 'Backstroke' &&
    chrono.activity__meters == 100 && chrono.activity__series == 1 && chrono.activity__exercise == 'Normal')

    let metersChartData = this.getMetersChart(chronos, today, users[i].name)
    let crawlChronosData = this.getChornosTimesChart(chronosCrawl, today, users[i].name)
    let backChronosData = this.getChornosTimesChart(chronosBack, today, users[i].name)

    this.dataLine.push(crawlChronosData)
    this.dataLine.push(backChronosData)

    this.dataColumn.push(metersChartData)
  }
  this.loading = false
}
```

Ilustración 30. Lógica en la creación del componente Chart

Después de este filtro, se realiza un segundo filtro para conseguir los tiempos de las pruebas de 100 Crawl y de 100 Backstroke. Después de todo este proceso se ha llamado a dos métodos, uno encargado de recoger los tiempos de cada prueba y otro destinado a la captación de los metros que se ha realizado durante cada entrenamiento.

```
  getMetersChart (chronos, today, user) {
    let columnMeters = { name: user, data: [] }
    let day1 = 0; let day2 = 0; let day3 = 0; let day4 = 0; let day5 = 0; let day6 = 0; let day0 = 0
    for (let j = 0; j < chronos.length; j++) {
      let date = new Date(chronos[j].timechrono)
      let meters = chronos[j].activity__meters * chronos[j].activity__series

      if (date.getDay() === 1) day1 += meters
      if (date.getDay() === 2) day2 += meters
      if (date.getDay() === 3) day3 += meters
      if (date.getDay() === 4) day4 += meters
      if (date.getDay() === 5) day5 += meters
      if (date.getDay() === 6) day6 += meters
      if (date.getDay() === 0) day0 += meters
    }
    for (let i = 0; i < 7; i++) {
      let timestampDay = today - 86400 * i * 1000
      let finalIndex = new Date(timestampDay).getDay()
      columnMeters.data.push(eval('day' + finalIndex))
    }
    return columnMeters
  }
},
```

Ilustración 31. Método getMetersChart

```
getChornosTimesChart (chronos, today, user) {
  let lineChrono = {
    name: '',
    data: []
  }
  let timeChrono = 0
  let chrono1 = Number.MAX_SAFE_INTEGER; let chrono2 = Number.MAX_SAFE_INTEGER
  let chrono3 = Number.MAX_SAFE_INTEGER; let chrono4 = Number.MAX_SAFE_INTEGER
  let chrono5 = Number.MAX_SAFE_INTEGER; let chrono6 = Number.MAX_SAFE_INTEGER
  let chrono0 = Number.MAX_SAFE_INTEGER

  for (let j = 0; j < chronos.length; j++) {
    lineChrono.name = user + ' ' + chronos[j].activity__meters + ' ' + chronos[j].activity__style
    let date = new Date(chronos[j].timechrono)
    timeChrono = this.timechrono(chronos[j].time)
    if (date.getDay() === 1) chrono1 = timeChrono
    if (date.getDay() === 2) chrono2 = timeChrono
    if (date.getDay() === 3) chrono3 = timeChrono
    if (date.getDay() === 4) chrono4 = timeChrono
    if (date.getDay() === 5) chrono5 = timeChrono
    if (date.getDay() === 6) chrono6 = timeChrono
    if (date.getDay() === 0) chrono0 = timeChrono
  }
  for (let i = 0; i < 7; i++) {
    let timestampDay = today - 86400 * i * 1000
    let finalIndex = new Date(timestampDay).getDay()
    lineChrono.data.push(eval('chrono' + finalIndex) === Number.MAX_SAFE_INTEGER ? 0 : eval('chrono' + finalIndex))
  }
  return lineChrono
},
```

Ilustración 32. Método getChronosTimesChart

Para finalizar cabe destacar que en la carpeta *constant* hemos creado dos contantes, una que obtiene las configuraciones de la gráfica de tiempos y otra que tiene las configuraciones de la gráfica de metros. A continuación, se mostrará la configuración de esta última.

```
export const column = (data) => {
  return {
    chart: {
      type: 'column'
    },
    title: {
      text: 'Total Distance'
    },
    subtitle: {
      text: 'Last 7 Days'
    },
    xAxis: {
      categories: getXaxis(),
      crosshair: true
    },
    yAxis: {
      min: 0,
      title: {
        text: 'Meters (m)'
      }
    },
    tooltip: {
      headerFormat: '<span style="font-size:10px">{point.key}</span><table>',
      pointFormat: '<tr><td style="color:{series.color};padding:0">{series.name}: </td>' +
          '<td style="padding:0"><b>{point.y:.1f} mm</b></td></tr>',
      footerFormat: '</table>',
      shared: true,
      useHTML: true
    },
    plotOptions: {
      column: {
        pointPadding: 0.2,
        borderWidth: 0
      }
    },
    series: data
  }
}

function getXaxis () {
  var today = Date.now()
  let result = []
  for (let i = 0; i < 7; i++) {
    var h = today - 86400 * i * 1000
    result.push(new Date(h).toUTCString().substring(0, 12))
  }
  return result
}
```

Ilustración 33. Constante para la generación de una gráfica de la distancia

10. Presupuesto

A continuación, procederemos a calcular el presupuesto que tendría desarrollar esta aplicación, sin tener en cuenta las capas gratuitas de un año que ofrece AWS y las licencias gratuitas de JetBrains que ofrece la universidad. Además, se comentará los métodos de financiación que se podrían emplear.

10.1 Costes Hardware

En este apartado procederemos a calcular el precio total del equipo hardware que se ha utilizado durante el desarrollo.

Hardware	Cantidad	Precio
Portátil Asus	1	1100€
Ratón	1	20 €
Monitor Philips 22 pulgadas	1	147 €
Total		1267 €

Tabla 6. Costes Hardware

10.2 Costes Software

En primer lugar, calcularemos el coste mensual de los productos de AWS, que hemos utilizado durante el desarrollo de la aplicación, los cuáles se encuentran en la región de EU (Ireland).

Software	Precio
Amazon EC2 Service	15,08€
Amazon RDS Service	15.37 €
Total	30.45€

Tabla 7. Costes Software

Por lo tanto, el coste de utilizar estos servicios durante el desarrollo del proyecto, empezando desde febrero hasta mayo sería de:

$$4\ meses\ x\ 30.45\ € = 121.8\ €$$

Para calcular dichos costes se ha utilizado la calculadora que ofrece Amazon para calcular dichos costes [29].

Por otro lado, también tenemos el IDE Pycharm Professional de JetBrains, el cuál cuesta **199€** al año la licencia. Por lo que, los costes anuales en este apartado ascienden a:

$$12\ \text{meses x}\ 30.45€ + 199€ = 564.4€/año$$

10.3 Costes por Personal

En este apartado, a pesar de que fue un trabajo individual, se asumirá que hemos ejercido los distintos roles que vamos a exponer a continuación:

Perfil	Precio por hora	Participación	Coste total
Diseñador Gráfico	30€	20%	1800€
Analista	40€	10%	1200€
Desarrollador	35€	50%	5250€
Product Owner	50€	20%	1800€
Total			10050€

Tabla 8. Coste personal

10.4 Coste Total

En definitiva, si sumamos los costes totales de todas las partes involucradas en el proyecto, el coste ascendería a:

$$1267€ + 10050€ + 97.76€ = 11414.76\,€$$

10.5 Coste de Mantenimiento

Esta aplicación será *open source* por lo que cualquier club que desee ponerla en funcionamiento para su propio uso, deberá costear la infraestructura donde se aloje pero no una licencia de uso. Por ello, procederemos a calcular cuál sería el coste anual de dichos servicios para algunos tipos de instancias t2 (Tabla 7) y el coste del servicio RDS (Tabla 6).

Servicio	Almacenamiento	Precio	Total (€/año)
Amazon RDS Service	20GB	15.37 €	188.44€/año

Tabla 9. Coste anual RDS

Instancia	CPU	RAM	Precio	Total (€/año)
Linux on T2.micro	1	1GB	7,63€	91.56€/año
Linux on T2.small	1	2GB	15,08€	180.6€/año
Linux on T2.medium	2	4GB	30,50€	366€/año
Linux on T2.large	2	8GB	60,99€	731.88€/año

Tabla 10. Coste anual instancias t2

A continuación, se calculará el precio total de cada instancia con el precio total del servicio RDS.

Instancia	Total (€/año)	RDS	Coste Total
Linux on T2.micro	91.56€/año	188.44€/año	280€/año
Linux on T2.small	180.6€/año		369.04€/año
Linux on T2.medium	366€/año		554.44€/año
Linux on T2.large	731.88€/año		920.32€/año

Tabla 11. Costes Totales de Mantenimiento

Como conclusión, la instancia t2.small es la más adecuada para el estado actual de la aplicación, debido a que por sus especificaciones es capaz de ejecutar sin ningún tipo de problema la aplicación. En la Ilustración 13 se puede observar el rendimiento de la aplicación cuando se realiza un despliegue a través de *Jenkins*.

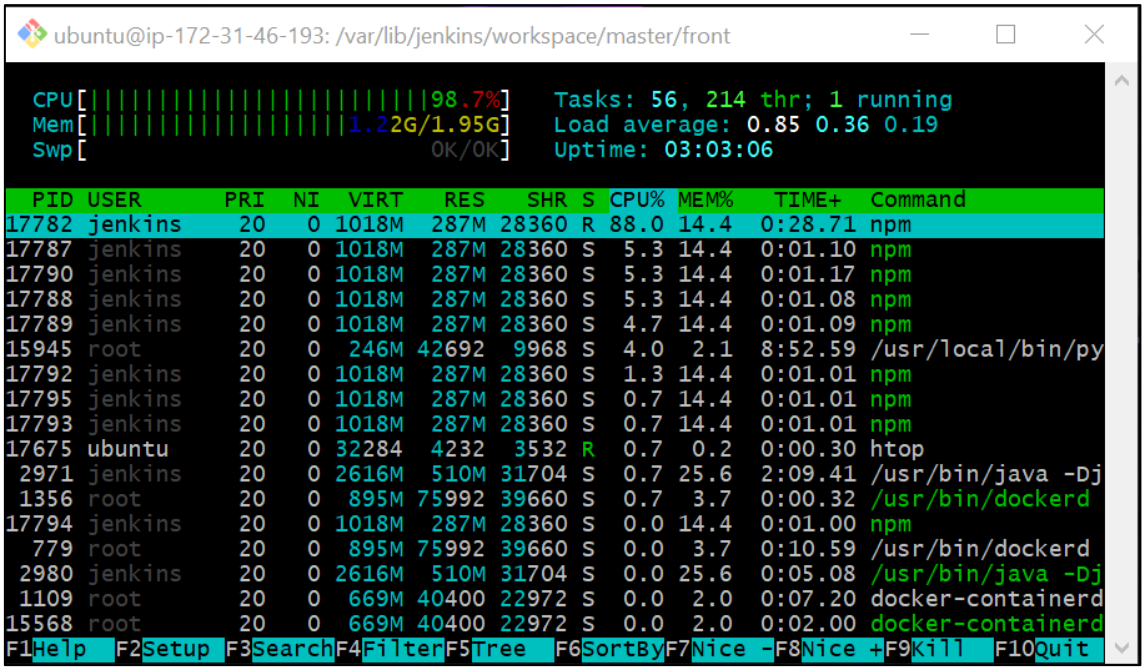


Ilustración 34. Rendimiento de un t2small en un despliegue

10.6 Financiación

En este apartado se abordarán distintos métodos para poder financiar el proyecto en el caso de que se quisiera seguir desarrollando en un futuro:

• Publicidad: Se hablaría con empresas relacionadas con la venta de material deportivo para la natación, y con fabricantes de relojes multideporte para dejarles un espacio publicitario en la web.

• Crowdfunding: Se expondría el proyecto en una plataforma de crowdfunding como Kickstarter, para convencer a las empresas y a los usuarios de que inviertan en este proyecto.

11. Conclusiones y Trabajo Futuro

11.1 Conclusiones

Una vez concluido el desarrollo de este proyecto, podemos destacar que se ha alcanzado los objetivos propuestos. La aplicación desarrollada supone un beneficio para los distintos clubes de natación, tanto a nivel profesional como amateur. Esto supone un impacto socioeconómico ante estas entidades ya que tienen a su alcance una herramienta que les facilita el desarrollo de su actividad. Dado que la aplicación será *open-source* sólo tendrán que costear la infraestructura tecnológica donde se implante. Como hemos visto en el apartado de presupuesto el coste de la plataforma de AWS es abordable por un club de natación.

La aplicación desarrollada cubre las principales funcionalidades que requiere cualquier club de natación ofreciendo:

- un sistema de gestión de planes de entrenamiento, que permitirá a los entrenadores llevar toda la gestión de los entrenamientos que deben realizar, mientras que sus nadadores podrán consultar dichos entrenos.

- un sistema de cronometraje para permitir a los nadadores que puedan registrar sus tiempos en las distintas pruebas.

- la visualización gráfica de los tiempos recogidos para que los usuarios puedan analizar y valorar la evolución de su entrenamiento.

- una aplicación que utiliza servicios como AWS, para un club ya que supone un precio asequible en base a tener un servidor local propio.

- un sistema automatizado que emplee Integración y Despliegue Continuo, para facilitar a los desarrolladores la integración del código y asegurando que se cumple con unos mínimos de calidad. Por otro lado, permite a los clientes tener disponible las mejoras en la aplicación en un menor tiempo.

Además, como objetivo personal, este proyecto me ha servido sobre todo para aprender a utilizar Jenkins y Docker, dos tecnologías que son muy usadas en el mundo de la informática y muy demandas en el mercado actual.

11.2 Trabajo Futuro

Tras haber finalizado el proyecto existe una serie de funcionalidades que nos hubiera gustado incluir en la aplicación y otras mejoras de las ya existentes. Por cuestiones de tiempo, 300 horas en el contexto del desarrollo del TFG, no ha sido posible abordarlas.

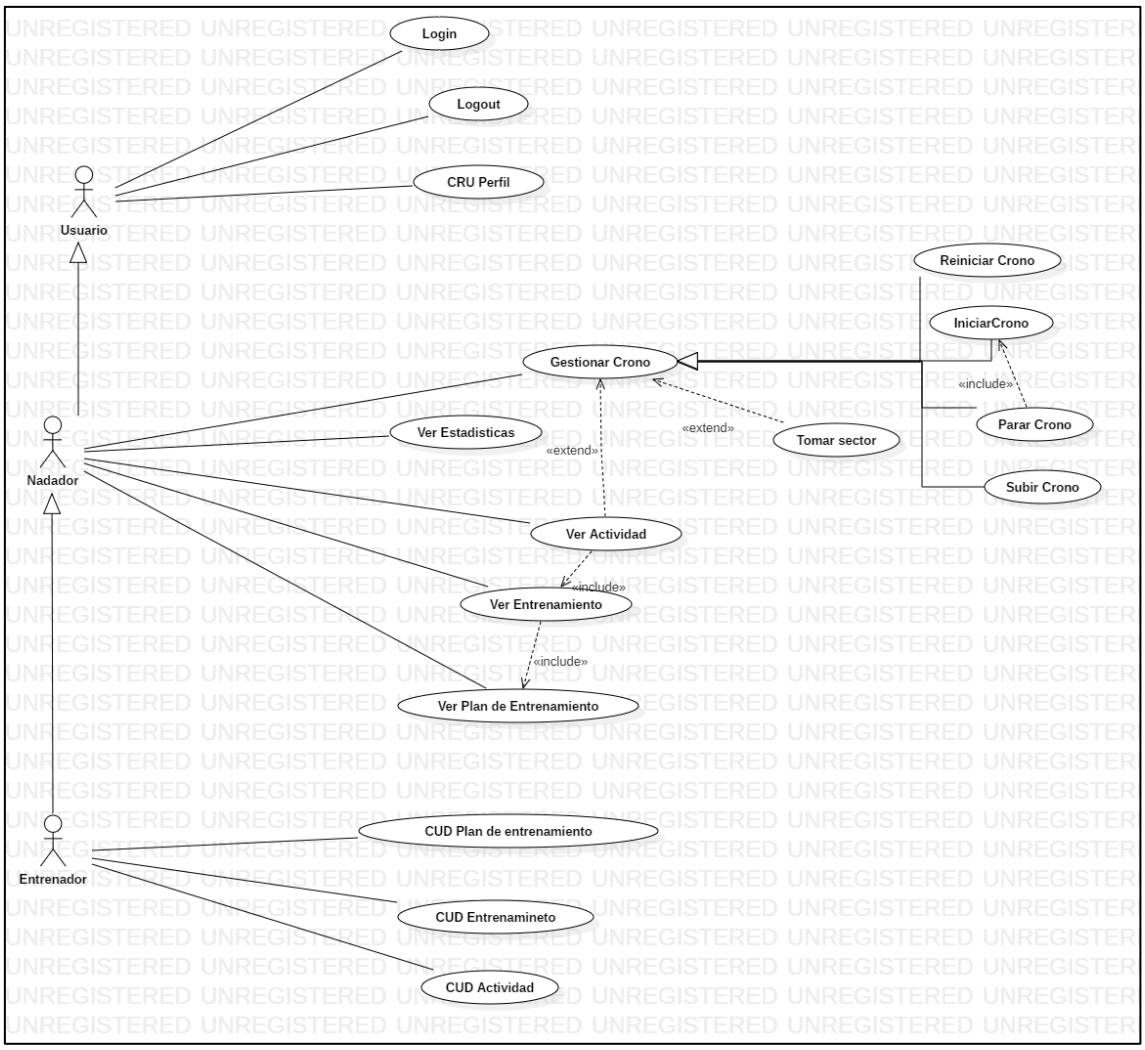
Actualmente la visualización de las gráficas de tiempos y volumen de entrenamiento están limitadas a dos ejercicios y dos usuarios. Nos gustaría ampliar dicha visualización a un número ilimitado de usuarios (tanto con especifique el entrenador) y de ejercicios. Para facilitar la interpretación de dichas gráficas, esta funcionalidad incluiría un filtro de ejercicios y usuarios. La implementación de otro tipo de filtrado de datos como puede ser serie temporales, tiempos de un día, última semana, por meses y años sería también una funcionalidad de gran utilidad para los usuarios de esta aplicación.

Asimismo, cuando un entrenador tiene a su cargo un gran número de nadadores la gestión individualizada de los entrenamientos se puede volver una tarea bastante engorrosa. La inclusión de un buscador de usuarios facilitaría dicha función. La publicación e intercambio de datos de los entrenamientos, como se hace en una red social, podría favorecer la participación activa de los usuarios de la aplicación.

Para finalizar, una vez realizado la funcionalidad extra comentada en el apartado anterior, se debería dar la posibilidad de crear un sistema de mensajería dentro de la aplicación, para que dichos usuarios puedan hablar y compartir sus entrenos y tiempos.

Anexo 1: Casos de Uso

En este apartado se pueden observar los distintos casos de uso que se han implementado en la aplicación.



Anexo 2: Pila de Producto

En este apartado se puede observar las distintas historias de usuarios que se han generado durante el proceso de análisis, destacar que las que se encuentran con un fondo grisáceo son historias de usuario que no se han implementado en la aplicación.

	Nombre	Estimación (horas)	Descripción	Criterio de Validación
1	Puesta a punto del proyecto	8	Como **desarrollador,** juntaré tanto el Frontend como el Backend, para evitar tener problemas al final del proyecto al unir las dos partes	Que cuando ejecute tanto el Frontend como el Backend a la vez, la conexión de las dos partes no de ningún tipo de error
2	Log in	12	Como usuario quiero poder iniciar sesión en la aplicación a través de la web	El usuario puede identificarse y acceder a la página principal de la web. En caso de fallo, le aparecerá los campos de inicio de sesión en rojo
3	Sign Up	10	Como usuario quiero poder crearme una cuenta en la aplicación.	El usuario se crea una cuenta y accede a la página principal. Si no rellena los campos obligatorios, se mostrará los campos que fallan en rojo y no se creará ningún usuario.
4	Log out	8	Como usuario quiero poder cerrar sesión en la aplicación a través de la web	El usuario sale de la aplicación web y lo lleva a la página de Login
5	Ver perfil	10	Como usuario quiero poder ver mis datos personales.	El usuario puede ver satisfactoriamente los datos de su perfil.
6	Editar perfil	10	Como usuario quiero poder editar mi perfil de la aplicación, para actualizarlo o corregir erratas.	El usuario puede modificar satisfactoriamente los datos de su perfil. Si algún campo que es obligatorio se deja vació, se mostrará en rojo y no se dejará modificar
7	Ver Calendario	12	Como nadador quiero poder ver un calendario en el que aparezca los distintos entrenamientos	El nadador podrá ver un calendario con los entrenamientos que ha realizado o va a realizar. Si no posee ningún entrenamiento, vera el calendario vacío.
8	Eliminar Perfil	2	Como usuario quiero poder eliminar mi perfil de la aplicación, ya que no quiero tener una cuenta en la web	El usuario no puede volver acceder a la app

9	Buscar Usuario	10	Como usuario quiero buscar a otros usuarios en la aplicación para informarme de sus resultados	El usuario busca a otro usuario que exista dentro de la aplicación y lo encuentra
10	Iniciar Cronómetro	8	Como nadador quiero iniciar un crono para cronometrar un tiempo de mi marca personal o el de otro nadador	El nadador le da a iniciar cronometro y este se pone en marcha
11	Parar Cronómetro	0.5	Como nadador quiero parar de cronometrar	El nadador le da a parar el cronometro y este deja de contar el tiempo
12	Reiniciar Cronómetro	0.5	Como nadador quiero reiniciar el tiempo que marca en ese momento el cronometro	El cronometro obtiene un valor de 0:0.0 cuando el nadador le da al botón de reiniciar
13	Tomar sector en el Cronómetro	4	Como nadador quiero tomar el tiempo de un sector tantas veces como quiera	Cuando le dé al botón para coger el sector, me debe tomar el tiempo para ese sector
14	Subir Tiempo de Cronometro	12	Como nadador quiero subir mi tiempo a la plataforma, para luego poder consultarlo	Que cuando el nadador le dé al botón de subir el crono, se suba dicho tiempo
15	Visualizar tiempo registrado de una actividad	6	Como nadador quiero poder visualizar el tiempo que se ha registrado de una actividad y sus parciales	Cuando el nadador le dé al botón de visualizar, lleve al usuario a una página en la que muestre el tiempo y los parciales que ha realizado ese usuario de dicha actividad. En caso de que no posea un tiempo se le mostrará un mensaje de error.
16	Habilitar un modo MultiCronómetro	12	Como nadador quiero tomar varios cronos a la vez para medir y comparar los tiempos de dos o más entrenadores	Que el nadador al cambiar de modo tenga varios cronos simultáneos que me permitan realizar todas las funcionalidades que hace un cronómetro
17	Ver Gráficos de mis tiempos	6	Como nadador quiero poder ver mis tiempos para evaluar mi rendimiento a través de Gráficos	Cuando el nadador vaya a la sección de gráficos, vea sus tiempos en una gráfica
18	Filtrar para obtener diferentes gráficos de distintas pruebas	4	Como nadador quiero poder filtrar en las gráficas en base a las pruebas, para que pueda ver los tiempos de dichas pruebas	Que cuando un nadador filtre, se muestre una gráfica con los datos filtrados

19	Crear una Actividad	8	Como entrenador quiero crear una actividad para	Cuando un entrenador termine el proceso de creación, se vea la nueva actividad creada. Si algún campo que es obligatorio se deja vació, se mostrará en rojo y no se podrá crear
20	Ver una Actividad	8	Como nadador quiero ver una actividad	Que le aparezca al nadador una lista con las distintas actividades del entrenamiento que he seleccionado
21	Modificar una Actividad	3	Como entrenador quiero modificar una actividad	Cuando un entrenador ha modificado la actividad se debe ver que el cambio se ha producido. En caso de que algún campo que es obligatorio se deja vació, se mostrará en rojo y no se dejará modificar.
22	Eliminar una Actividad	2	Como entrenador quiero eliminar una actividad	Cuando un entrenador le dé a eliminar, se elimina de la base de datos y además no debe aparecer más
23	Clonar una Actividad	2	Como entrenador quiero clonar una actividad	Cuando un entrenador le dé al botón de clonar se me debe crear una nueva actividad en la base de datos con los datos del elemento clonado. Además, me debe aparecer en la vista una vez clonado.
24	Crear un Plan de entrenamiento	8	Como entrenador quiero crear un plan de entrenamiento donde guardar mis entrenos	Que un entrenador se haya credo un nuevo plan de entrenamiento correctamente. Si algún campo que es obligatorio se deja vació, se mostrará en rojo y no se podrá crear
25	Ver un Plan de entrenamiento	8	Como nadador quiero ver mis distintos planes de entrenamiento	Que a un nadador le aparezca una lista con los distintos planes de entrenamiento que posee ese nadador
26	Modificar un Plan de entrenamiento	2	Como entrenador quiero modificar mi plan de entrenamiento	Cuando un entrenador haya hecho una modificación el cambio que se ha realizado se haya guardado y se muestre. En caso de que algún campo que es obligatorio se deja vació, se mostrará en rojo y no se dejará modificar.
27	Eliminar un Plan de entrenamiento	1	Como entrenador quiero eliminar un plan de entrenamiento	Cuando un entrenador confirme que quiero eliminar un Plan de entrenamiento, este

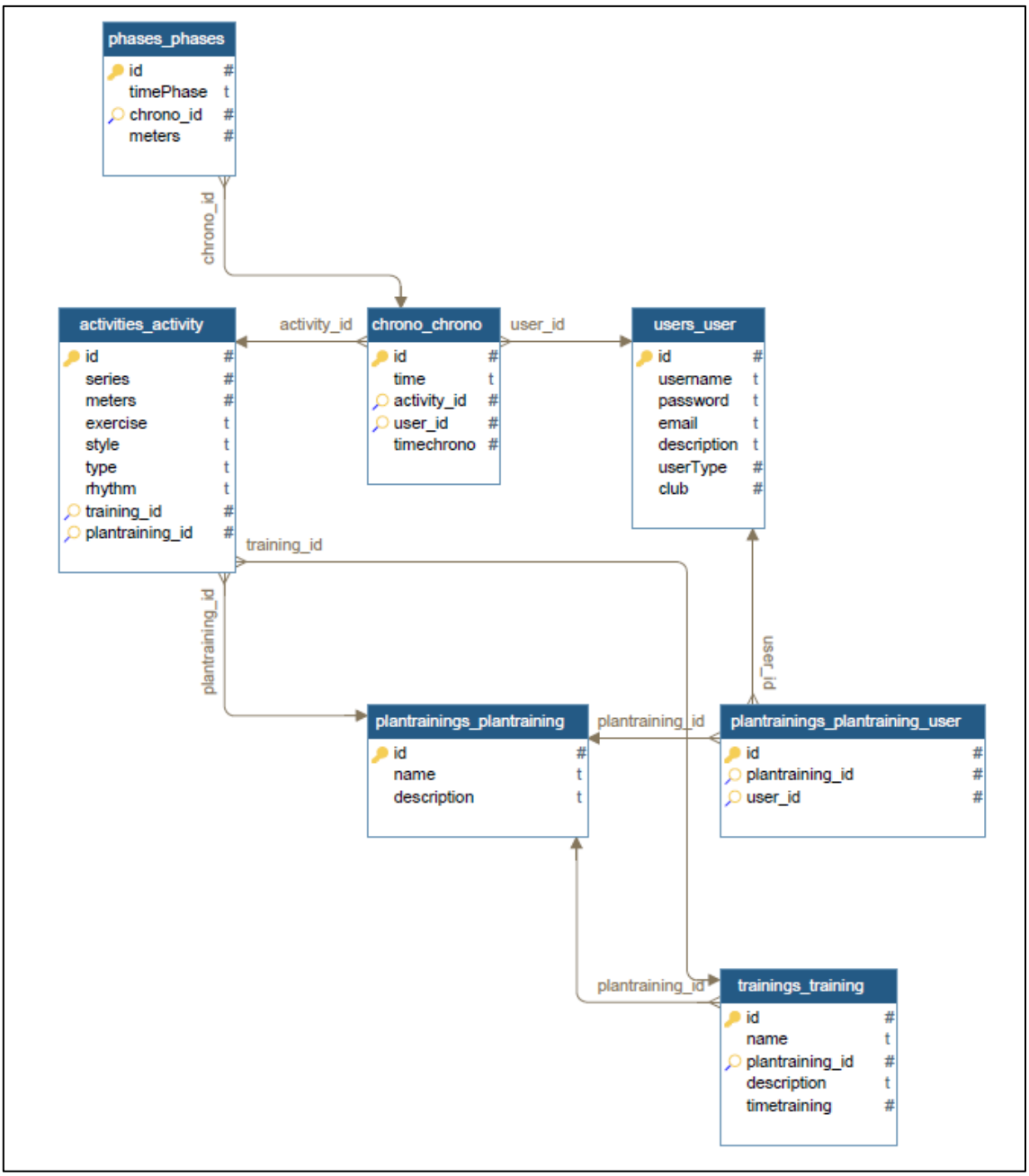
				se debe eliminar de la base de datos y no se debe mostrar
28	Clonar un Plan de entrenamiento	8	Como entrenador quiero poder clonar un plan de entrenamiento	Cuando un entrenador le dé al botón de clonar se me debe crear un nuevo plan de entrenamiento que contenga los entrenamientos y las actividades del elemento clonado, todo esto en la base de datos. Además, me debe aparecer en la vista una vez clonado.
29	Crear un Entrenamiento	8	Como entrenador quiero crear un entrenamiento para mi alumnado	Que un entrenador que haya credo un nuevo entrenamiento, donde se pueda crear actividades. Si algún campo que es obligatorio se deja vació, se mostrará en rojo y no se podrá crear
30	Ver un Entrenamiento	5	Como nadador quiero ver un entrenamiento	Que a un nadador le aparezca una lista con los distintos entrenamientos del plan de entrenamiento que he seleccionado
31	Modificar un Entrenamiento	3	Como entrenador quiero modificar mi entrenamiento	Que a un entrenador se le permita modificar las características de un entrenamiento. En caso de que algún campo que es obligatorio se deja vació, se mostrará en rojo y no se dejará modificar.
32	Eliminar un Entrenamiento	1	Como entrenador quiero eliminar un entrenamiento	Cuando un entrenador le dé al botón de eliminar, dicho entrenamiento debe de haber sido eliminado de la base de datos y no debe aparecer en la interfaz
33	Clonar un entrenamiento	4	Como entrenador quiero clonar un entrenamiento	Cuando un entrenador le dé al botón de clonar se me debe crear un nuevo entrenamiento que contenga las actividades del elemento clonado, todo esto en la base de datos. Además, me debe aparecer en la vista una vez clonado.
34	Importar un Entrenamiento	10	Como entrenador quiero importar un entrenamiento para ahorrarme tiempo en crear uno nuevo	Que como entrenador cuando le dé click al botón importar, se importe un entrenamiento, este se cree sin ningún tipo de problema, pudiendo modificarlo, eliminarlo y verlo

35	Exportar un entrenamiento	2	Como entrenador quiero exportar un entrenamiento para poder tener almacenado en mi dispositivo	Que como entrenador cuando le de click al botón exportar, se exporte un fichero json o csv con el contenido del entrenamiento
36	Crear un Grupo	10	Como entrenador quiero crear un grupo para poder hablar y compartir el entreno con mi alumnado	Como entrenador cuando haya rellenado los campos obligados para crear un grupo, que se cree dicho grupo, del cual soy administrador.
37	Ver un Grupo	5	Como nadador puedo ver un grupo en el que estoy asignado	Como nadador puedo ver los grupos en los cuales estoy incluido
38	Modificar un Grupo	1	Como entrenador puedo modificar las características del grupo	Cuando un entrenador realice una modificación esta se haya hecho sin ningún tipo de problema
39	Eliminar un Grupo	1	Como nadador puedo eliminar un grupo	Cuando un nadador le dé al botón de eliminar, dicho grupo debe de haber desaparecido
40	Añadir Usuarios	2	Como entrenador puedo añadir a un nadador al grupo	Cuando un entrenador haya añadido a un usuario, este se haya añadido y pueda ver el grupo
41	Eliminar Usuarios	1	Como entrenador puedo eliminar a un nadador al grupo	Cuando un entrenador le da a eliminar a un nadador o entrenador, que dicho usuario se elimine, no se encuentre en el grupo
42	Enviar mensaje por un grupo	20	Como nadador quiero enviar un mensaje por el grupo	Que cuando un nadador envié un mensaje, este se hay enviado correctamente
43	Crear Publicación	20	Como Club quiero poder crear una publicación para informar a mis nadadores de las últimas noticias	Cuando un club ha rellenado los campos para crear una publicación, esta se haya creado. Si algún campo que es obligatorio se deja vació, se mostrará en rojo y no se podrá crear
44	Ver una Publicación	10	Como Club y nadador quiero poder ver una publicación	Que un club pueda ver un listado de las publicaciones que ha publicado.
45	Modificar una Publicación	5	Como Club quiero poder modificar una publicación para corregir los datos de la publicación	Cuando un club ha modificado una publicación, se debe ver que el cambio se ha producido
46	Eliminar una Publicación	1	Como Club quiero poder crear una publicación para informar a mis nadadores de las últimas noticias	Cuando un club le dé al botón de eliminar, esta tiene eliminarse, y no se puede volver a seguir visualizando

47	Implementar IU general de la web	20	Como **desarrollador,** quiero implementar una interfaz de usuario que se ajuste su contenido en cualquier dispositivo	Que el usuario pueda ver el contenido de la aplicación de manera que la aplicación se ajuste al ancho y altura del dispositivo que se encuentra
48	Desplegar Aplicación en AWS	20	Como **desarrollador** quiero que mi página web este en la red, para ello utilizaremos AWS para desplegar dicho producto	Cualquier usuario con acceso a internet debe poder acceder a la aplicación
	Horas totales	333		

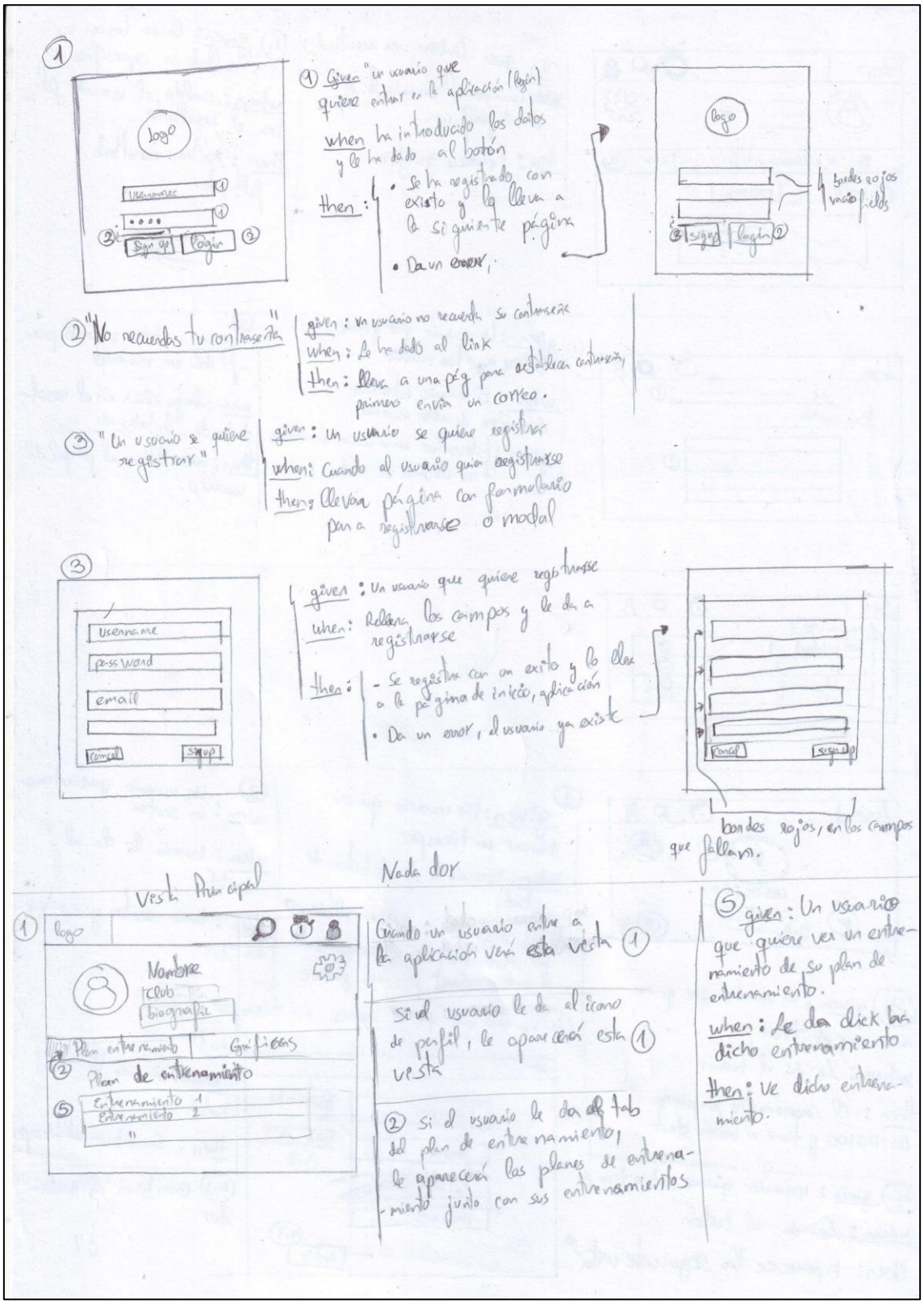
Anexo 3: Diagrama de la Base de Datos

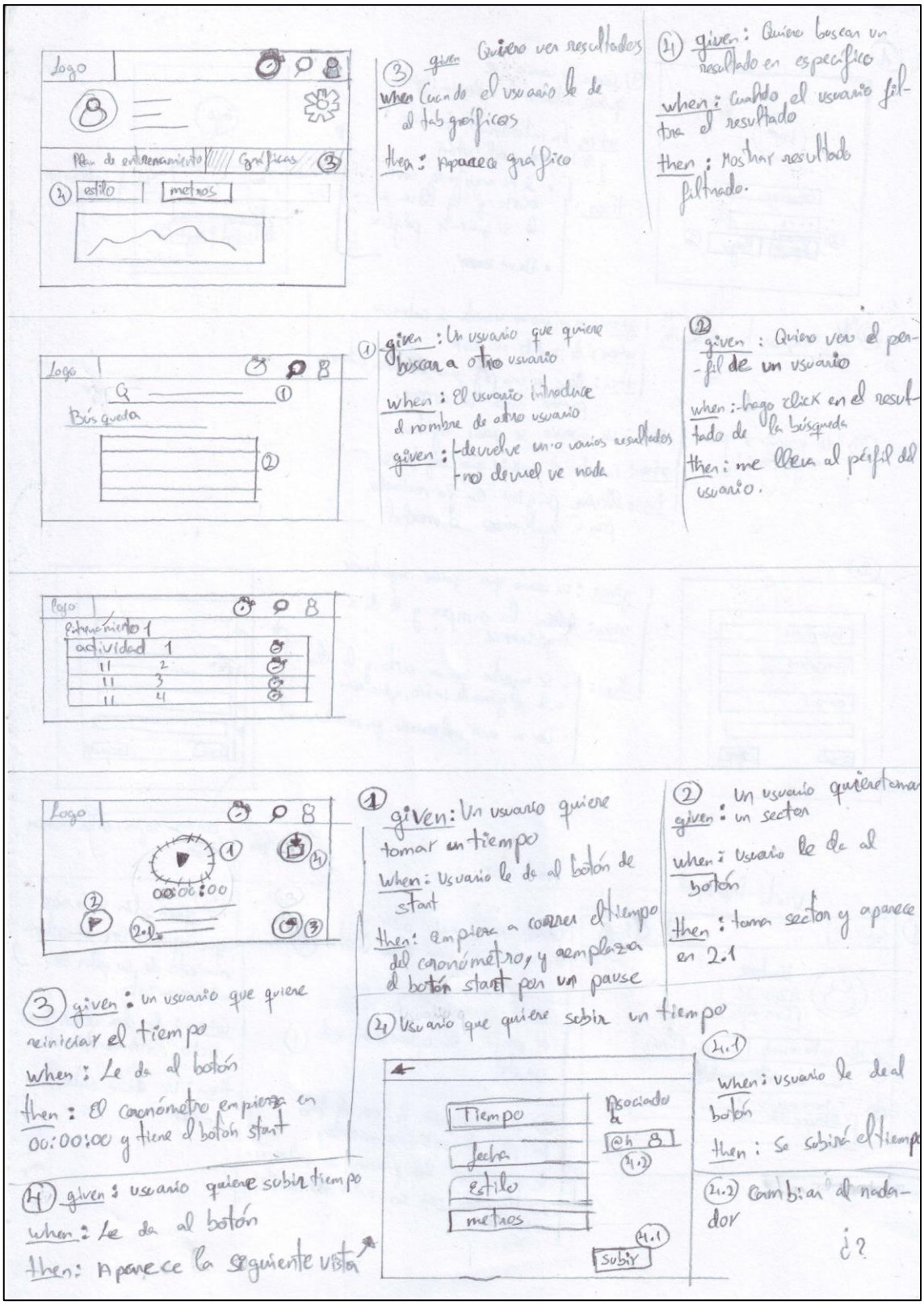
En este apartado se puede visualizar el diagrama de la base de datos que se ha desarrollado en el proyecto.



Anexo 4: Mockup

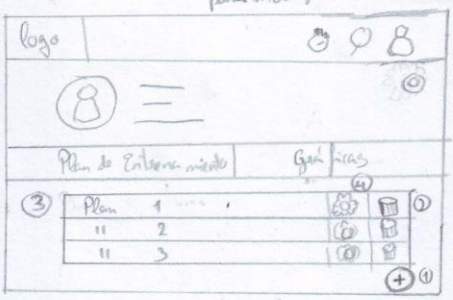
En este apartado se podrá visualizar el primer diseño que se realizó de la interfaz de usuario, no siendo esta la versión final del proyecto.




Logo
Plan de entrenamiento
Gráficas
3
4
estilo
metros
3 given Quiero ver resultados
when Cuando el usuario le de al tab gráficos
then: Aparece gráfico
2.1 given: Quiero buscar un resultado en específico
when: Cuando el usuario filtra el resultado
then: Mostrar resultado filtrado.
Logo
Búsqueda
1
2
1 given: Un usuario que quiere buscar a otro usuario
when: El usuario introduce el nombre de otro usuario
given: devuelve uno o varios resultados
no devuelve nada
2 given: Quiero ver el perfil de un usuario
when: hago click en el resultado de la búsqueda
then: me lleva al pérfil del usuario.
Logo
Entrenamiento 1
actividad 1
2
3
4
Logo
00:00:00
1
2
2.1
3
4
1 given: Un usuario quiere tomar un tiempo
When: Usuario le da al botón de start
then: empieza a correr el tiempo del cronómetro, y reemplaza el botón start por un pause
2 given: Un usuario quiere tomar un secton
when: Usuario le da al botón
then: toma secton y aparece en 2.1
3 given: un usuario que quiere reiniciar el tiempo
When: Le da al botón
then: El cronómetro empieza en 00:00:00 y tiene el botón start
4 given: usuario quiere subir tiempo
When: Le da al botón
then: Aparece la siguiente vista
4 Usuario que quiere subir un tiempo
Tiempo
fecha
estilo
metros
Asociado a
@h 8
4.2
Subir
4.1
2.1.1
When: usuario le da al botón
then: Se subirá el tiempo
2.1.2 Cambia al nadador
¿?

④ Modificar (aparece Modal con formulario para modificar)

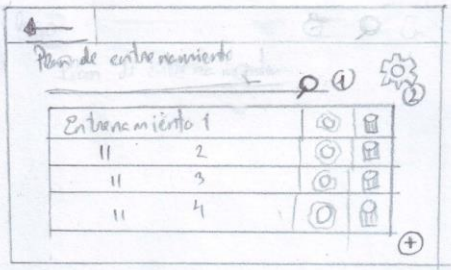
Entrenador



① given: un entrenador puede crear un plan de entrenamiento
when: le da al botón
then: Aparece un modal con un formulario

② given: Un entrenador quiere eliminar un plan
when: pulsa el botón
then: aparece modal para confirmación, luego se elimina.

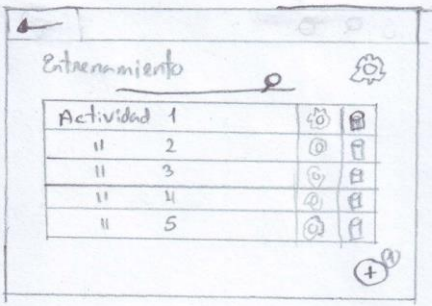
③ given: una entrenadora quiere ver los planes de entrenamiento
when: le da al tab de planes
then: se ve una lista de planes de entrenamiento, puedo picar en ellos y que me lleva a otra vista



Lo mismo que en la vista de planes de entrenamientos pero aplicado a los entrenamientos

① given: quiero buscar un entrenamiento
when: introduzco un nombre
then: aparece el resultado de la búsqueda

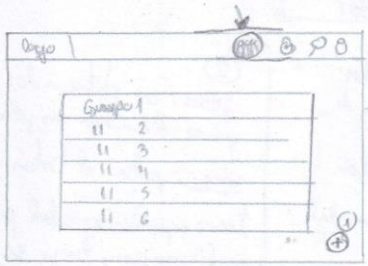
② se puede modificar el plan de entrenamiento.



Lo mismo que arriba pero para actividad

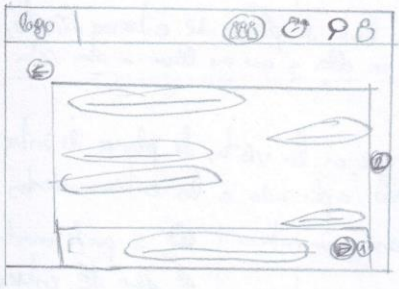
① then: aparece un modal para crear una actividad

Grupos



- Ver lista de grupos

① Crear un grupo nuevo, donde se añade a los participantes.



① given: Un usuario que quiere enviar un mensaje
where: Escribe un mensaje y lo envía
there: se envía un mensaje y se muestra

② Un usuario quiere ver los mensajes de un grupo.

Anexo 5: Implementación de la Infraestructura en AWS

En este apartado lo primero que hicimos fue iniciar sesión con nuestra cuenta de Amazon, y crearnos una instancia EC2, para ello se tuvo que realizar los siguientes pasos:

Paso 1: Selección del Sistema Operativo de la instancia

Amazon nos ofrece una gran diversidad de AMI's (Amazon Machine Image, Ilustración 17), en nuestro caso hemos seleccionado, entre las de uso gratuito, Ubuntu Server 18.04

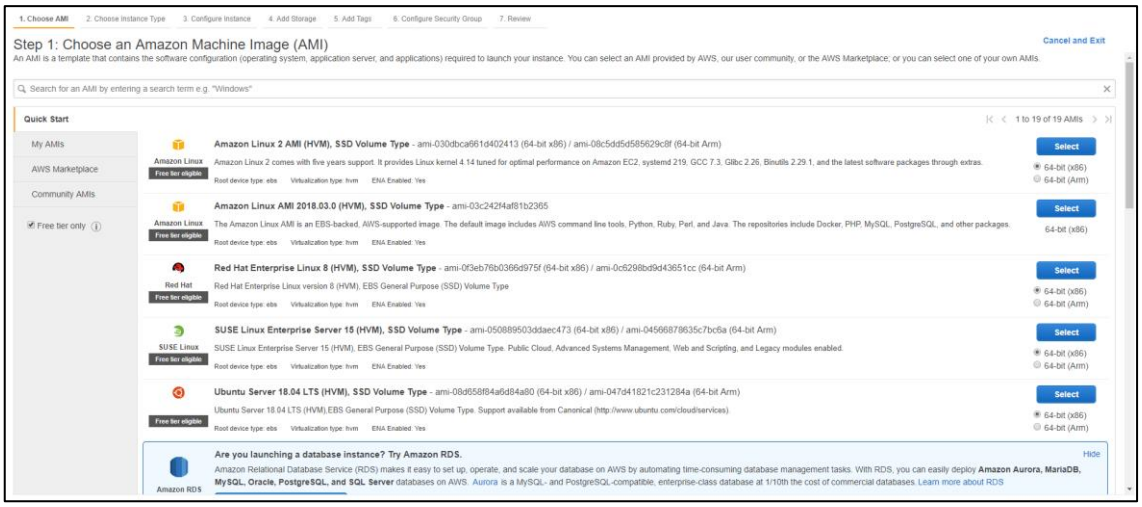


Ilustración 35. Selección de Sistema AMI

Paso 2: Selección del tipo de instancia

En este apartado debemos elegir el tipo de instancia que tendrá nuestro EC2. En nuestro seleccionaremos la *t2.small* debido a que con las prestaciones que posee cumplía con los requisitos mínimos para poder realizar el despliegue.

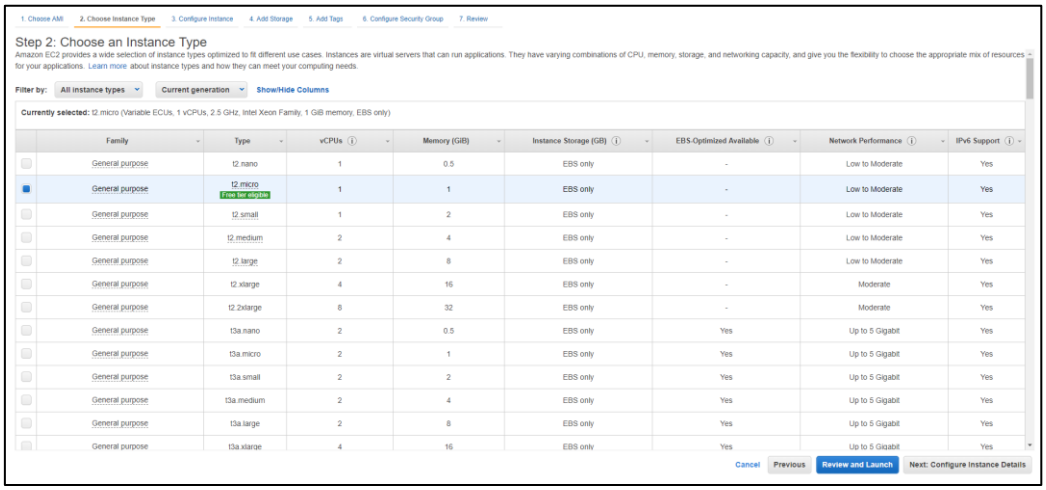


Ilustración 36. Selección del tipo de instancia

Una vez realizado esto, confirmamos en todas las pestañas hasta obtener disponible nuestra instancia EC2, la cuál por defecto ya habrá incluido un EBS de 8 GB de almacenamiento. Una vez terminado este proceso, Amazon nos pedirá crear un fichero con una key privada, la cual es jenkins.pem, que utilizaremos para conectarnos via ssh al EC2, a través del siguiente comando:

```
ssh -i "jenkins.pem" ubuntu@ec2-34-254-248-9.eu-west-1.compute.amazonaws.com
```

Paso 3: Instalando Jenkins

Para descargar Jenkins, primero necesitamos añadir la clave del repositorio al sistema, para luego

```
wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add –
```

```
sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
```

Una vez hecho esto procedemos a instalar Jenkins con las siguientes instrucciones

```
sudo apt update
sudo apt install jenkins
```

Paso 4: Iniciando Jenkins

Una vez realizado el paso anterior esto iniciamos Jenkins

```
sudo systemctl start Jenkins
```

Luego accedemos por nuestro navegador a la instancia ec2 a través del puerto 8080, ya que Jenkins tiene por defecto este puerto para su uso.

http://ec2-34-254-248-9.eu-west-1.compute.amazonaws.com:8080

Una vez hecho esto, nos aparecerá un textfield en el que tendremos que introducir la contraseña que nos proporciona Jenkins para continuar con la instalación, para ello debemos realizar el siguiente comando:

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

Ilustración 37. Desbloquear Jenkins

Una vez realizado esto, instalaremos los plugins que nos propone Jenkins. Después nos crearemos un usuario administrador, para luego acceder a nuestro Jenkins con el usuario creado anteriormente. [30]

Paso 5: Creando una tarea

En este paso sea creará un pipeline una tarea para la rama máster. Para ello le damos al botón Nueva tarea para crear una nueva tarea. Como se observa en la Ilustración 20 Jenkins nos ofrece múltiples opciones para crear una tarea. En nuestro caso escogeremos pipeline Project.

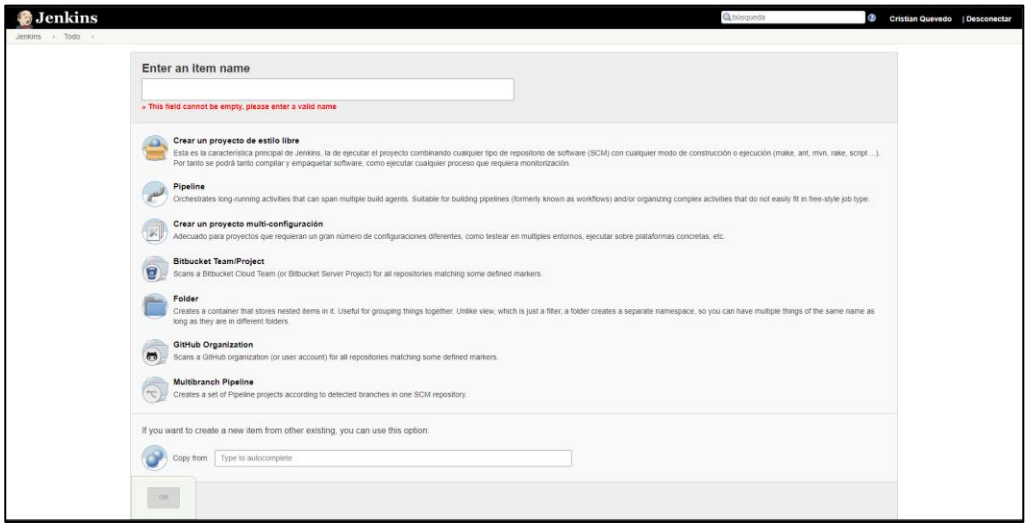


Ilustración 38. Selección de nueva tarea

Como se puede observar en la sección de builds, aceptamos la opción de que se ejecute la tarea cada vez que se haya realizado un push a nuestro repositorio.

En la sección de pipeline indicamos que nuestro control de versiones es git y a continuación especificamos la url de nuestro repo. Además, deberemos crearnos una nueva credencial para poder realizar la petición de github, en la que insertaremos nuestro usuario y contraseña de github.

Para finalizar diremos que realice un build de la rama master y que lo que debe ejecutar es el fichero Jenkinsfile. Cabe destacar que para que funcione todo correctamente se debe ir al botón de configuración de nuestro repo en github y añadir un webhook, con la url del ec2.

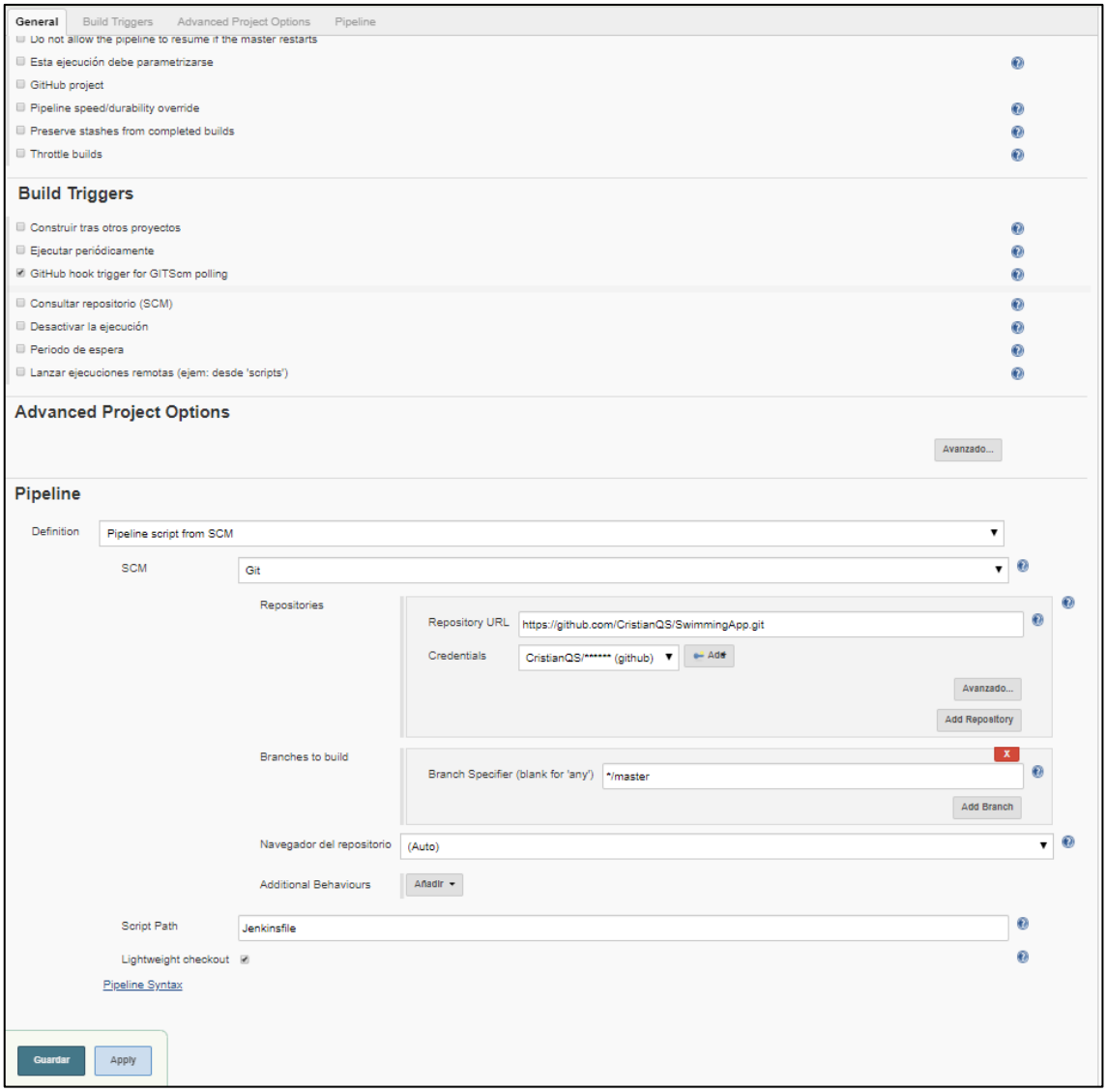


Ilustración 39. Creación de una nueva tarea del tipo pipeline

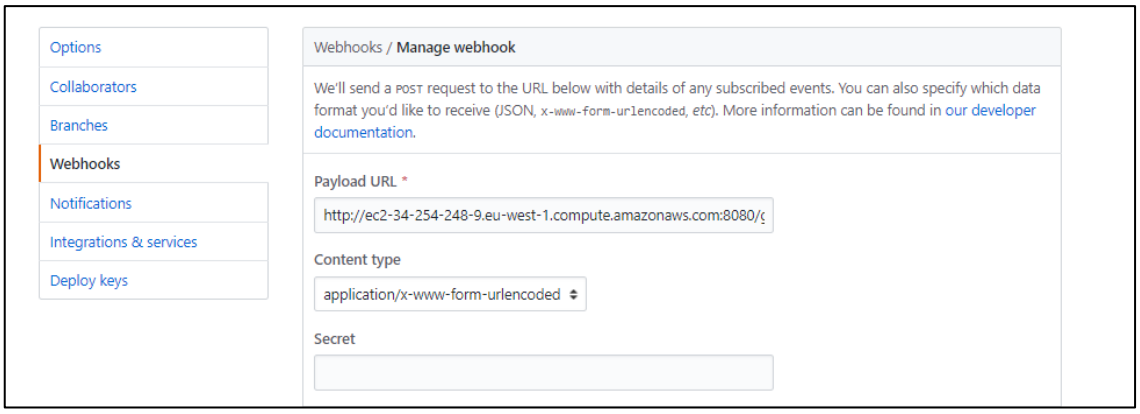


Ilustración 41. Inserción de la URL en un Webkooks

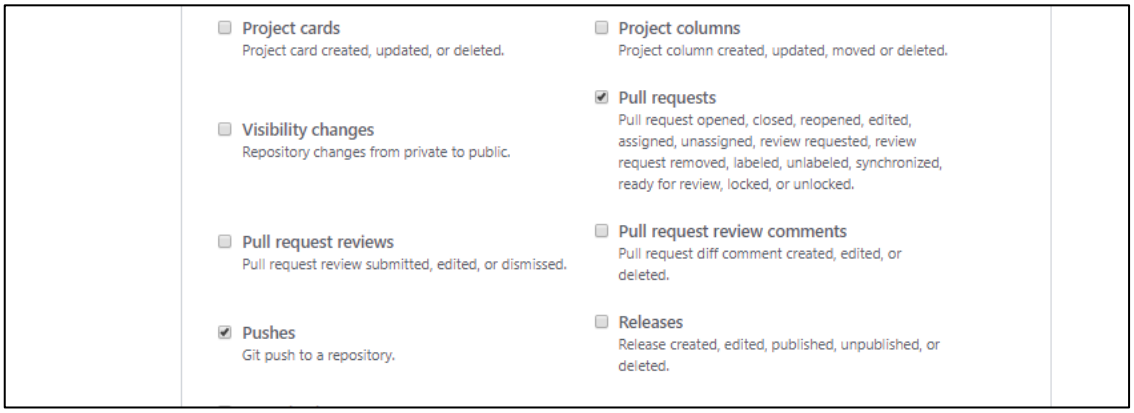


Ilustración 40. Selección de las acciones que ejecutará el Webhook

Una vez realizado esto, tendremos que crear nuestro fichero Jenkinsfile en el primer nivel del proyecto, que es el fichero que ejecutará Jenkins cuando se realice un push a github.

En este fichero definiremos las distintas etapas del ciclo de vida de nuestra tarea y lo que se debe hacer en cada una como se aprecia en el siguiente anexo.

Hay que recalcar que la sintaxis que aparece en dicha foto significa:

- stage: Crea una nueva etapa del ciclo de vida
- steps: Zona de ejecución de una etapa
- dir: se cambia al directorio especificado
- sh: instrucción a realizar
- post: Zona de ejecución después de que finalice el pipeline ya sea por éxito o fallo

Hay que destacar que se ha integrado con Slack, para que cada vez que se realice una tarea, esta aplicación nos notifique si se ha producido con éxito o fracaso.

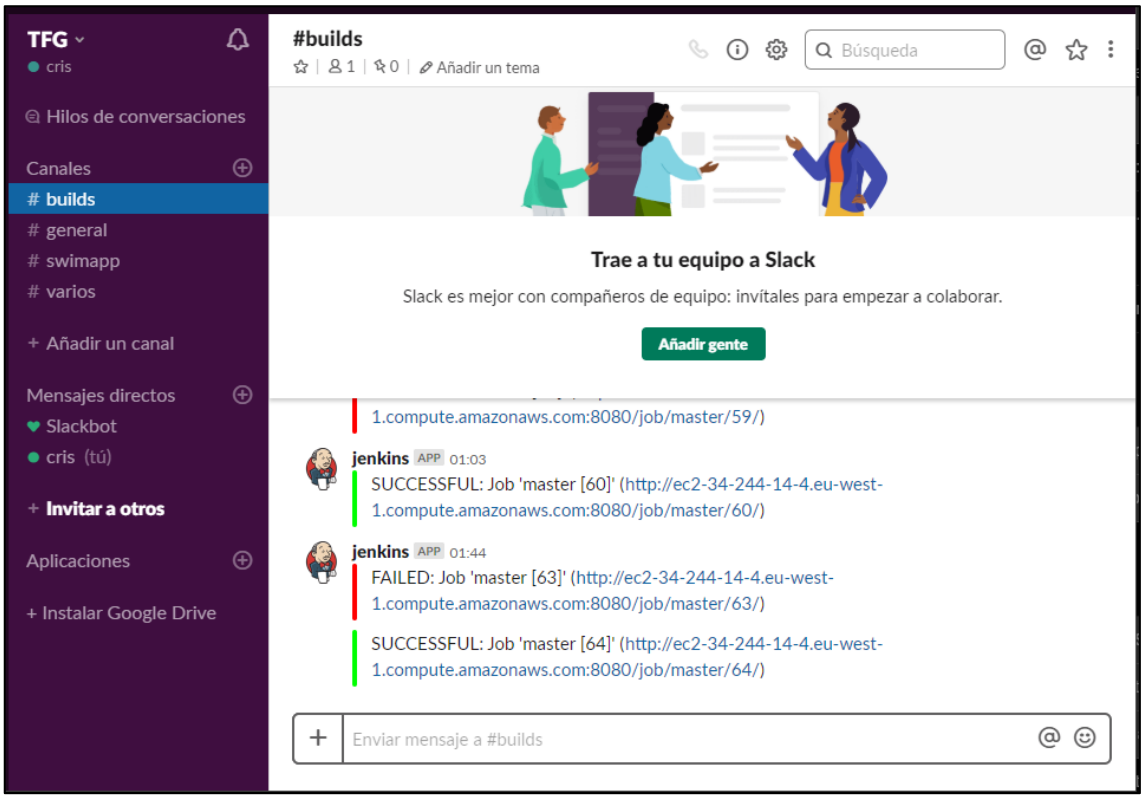


Ilustración 42. Notificaciones de Slack

Anexo 6: Creación de los contenedores Docker

Por otro lado, para poder desplegar la aplicación y poder ejecutar el back-end en local, se ha hecho uso de Docker-compose, una herramienta de Docker que permite la creación de nuevas imágenes por parte del usuario, pudiendo este configurar como crea conveniente. En este apartado mostraremos en el que se ha creado para la parte del back-end.

En primer lugar, deberemos crear un Dockerfile (Ilustración 25), en el cuál indicaremos la versión de Python que queremos que tenga el contenedor, luego crearemos una carpeta donde instalaremos todas las librerías o frameworks que se necesite (Ilustración 26).

```
FROM python:3
ENV PYTHONUNBUFFERED 1
RUN mkdir /code
WORKDIR /code
COPY requirements.txt /code/
RUN pip install -r requirements.txt
COPY . /code/

```

Ilustración 44. Dockerfile

```
behave==1.2.6
Django==2.2
django-cors-headers==2.5.2
djangorestframework==3.9.2
parse==1.11.1
parse-type==0.4.2
pytz==2018.9
six==1.12.0
PyJWT==1.7.1
selenium==3.141.0
psycopg2-binary>=2.7,<3.0
```

Ilustración 43. Librerías que instalar en la aplicación

Una vez realizado lo anterior, configuramos en el docker-compose (Ilustración 27) como será nuestra imagen. Para ello en el aparatado de build, al poner un punto, indicaremos que ejecute el Dockerfile que se ha realizado anteriormente, así como el puerto por el que se ejecutará el contenedor una vez se haya puesto a ejecutar el docker-compose.

```
version: '3'

services:
  db:
    image: postgres
  web:
    build: .
    command: python manage.py runserver 0.0.0.0:8000
    volumes:
      - .:/code
    ports:
      - "8000:8000"
    depends_on:
      - db
```

Ilustración 45. Docker-compose

Para ejecutarlo, primero haremos un docker-compose build, que creará la imagen del contenedor, para luego realizar un docker-compose up, el cual empezará a ejecutar dicho contenedor.

Anexo 7: Estructura y configuración de los proyectos

En esta parte se mostrará tanto la estructura y configuración del front-end como del back-end.

Front-end

Para crear un proyecto con Vue.js deberemos tener instalado previamente Vue-Cli 3, el cual nos proporciona una serie de comandos para la creación y gestión de proyectos con Vue.js. En nuestro caso, utilizaremos el comando *vue create* para la creación del proyecto.

Hay que recalcar que Vue.js incluye tecnologías como Webpack, la cual recoge todos nuestros ficheros de un determinado tipo, ya sea .vue o .js y los agrupa en un único fichero de ese tipo. [31]

A continuación, se aprecia el resultado final después de crear el proyecto:

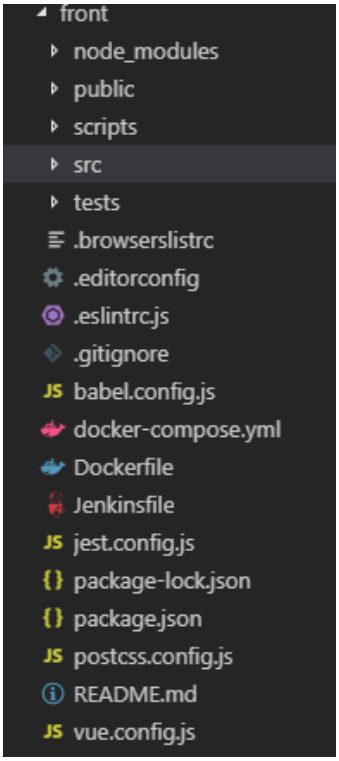


Ilustración 46.
Estructura de la carpeta front

Cabe destacar que todo nuestro código se encuentra en la carpeta src y posee la siguiente estructura:

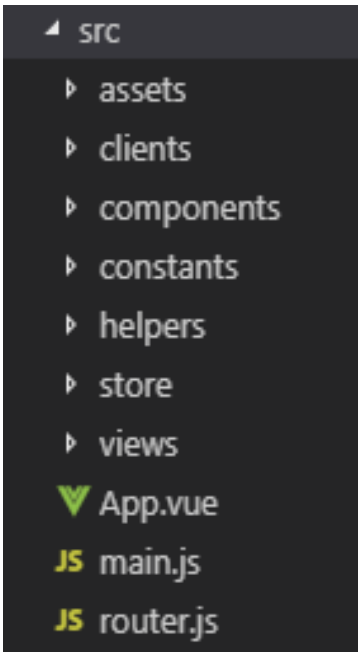


Ilustración 47. Estructura de la carpeta src

- En la carpeta assets se guardan ficheros estáticos como css, html y fotos.

- En la carpeta clients, se han creado los clientes de cada recurso de la API, los cuales tienen como función realizar la llamada a la API.

- En la carpeta componentes, se encontrarán todos los componentes que hemos creado para la creación de la aplicación.

- En la carpeta constants, se guardan ficheros que almacenan variables constantes de la aplicación, como el nombre de las rutas de la aplicación, y las rutas de la API que se ha creado.

- En la carpeta store se utiliza para gestionar nuestro estado a través de la librería Vuex.

En la carpeta views, se encuentran los ficheros .vue encargados de mostrar las vistas de la aplicación.

Back-end

Para esta parte hemos creado un proyecto en Django, para ello hemos ejecutado el comando *docker-compose run web django-admin startproject back*, luego se han creado durante el desarrollo de la aplicación los distintitos recursos de la API, realizando el siguiente comando *docker-compose run web python manage.py startapp <recurso>.*

A continuación, procederemos a comentar dicha estructura, la cuál será la misma para todas las carpetas, menos para la carpeta back, la cual es una carpeta de configuración de la aplicación.

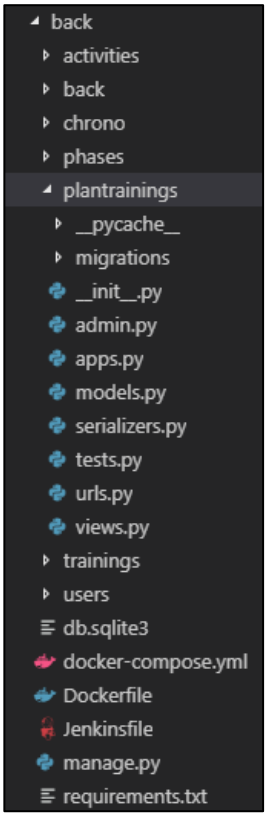


Ilustración 49. Estrcutura de la carpeta back

- En el fichero models.py, se define los atributos que tendrá nuestro recurso, una vez definido deberemos ejecutar el comando *docker-compose run web python manage.py makemigrations, para aplicar dichos cambios* y posteriormente *realizaremos docker-compose run web python manage.py migrate* para guardar dichos cambios.

- El fichero views, actuará de controller.

- En el fichero urls se definirán las rutas de ese recurso, que posteriormente se añadirán en el fichero urls.py de la carpeta back.

Asimismo destacar que en el fichero settings,py de la carpeta *back*, es donde realizamos todas las configuraciones de la API, desde incluir los diferentes recursos creados, decidir qué host pueden utilizarla, así como decidir la instancia de la base de datos que utilizará.

Para facilitar la legibilidad del código algunas de las capturas que pondremos a continuación serán recortadas. Este recorte no elimina información esencial para la correcta comprensión de dicho código.

```
ALLOWED_HOSTS = ['ec2-34-254-248-9.eu-west-1.compute.amazonaws.com','localhost']

CORS_ORIGIN_ALLOW_ALL = True

CORS_ALLOW_METHODS = (
    'DELETE',
    'GET',
    'OPTIONS',
    'PATCH',
    'POST',
    'PUT',
)

# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'rest_framework',
    'corsheaders',
    'users',
    'trainings',
    'activities',
    'plantrainings',
    'chrono',
    'phases'
]
```

Ilustración 48. Parte de la configuración del fichero settings.py

Anexo: Manual de Usuario

1. Requisitos Previos

Para poder acceder a la aplicación se necesita:

- Acceso a Internet.
- Navegador web.

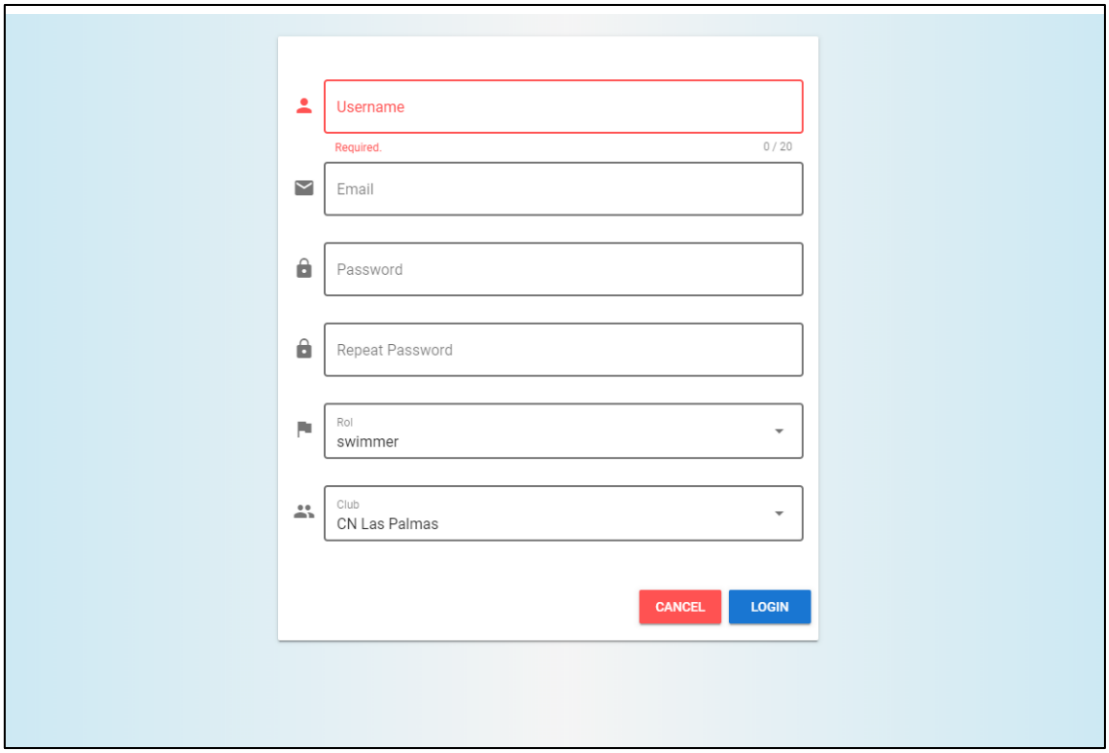
2. Guía de Usuario

A continuación, se procederá a explicar las diferentes funcionalidades de la aplicación como un usuario con el rol de **entrenador**, debido a que éste posee los máximos privilegios. En la tabla 11 (apartado 10.3.2) se describe las distintas roles y privilegios que tienen los usuarios de la aplicación.

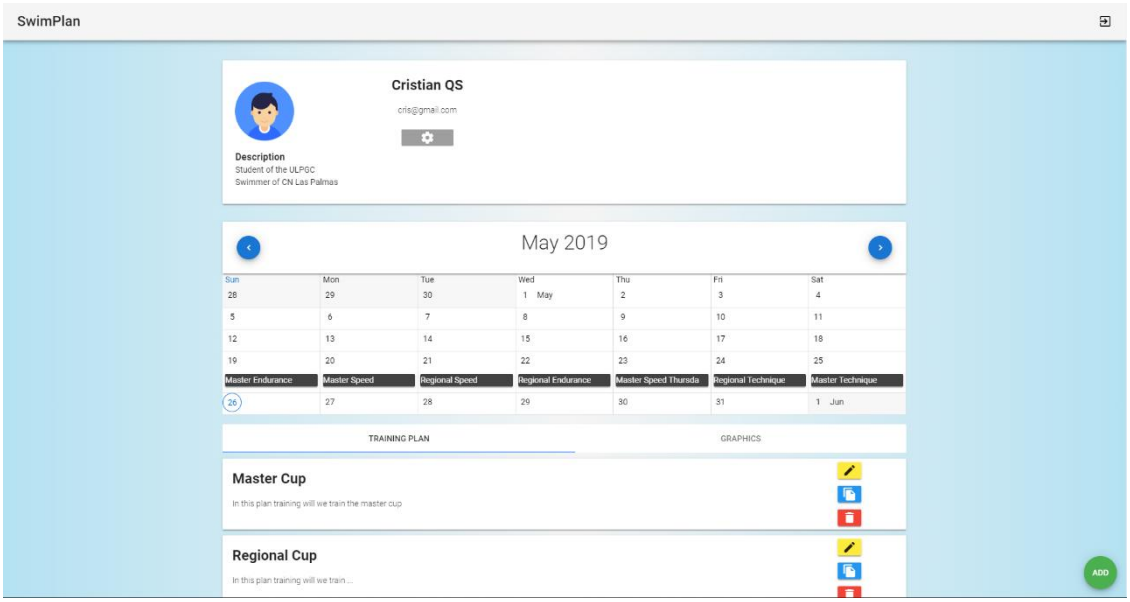
1) Acceder a la dirección de la aplicación. http://ec2-34-254-248-9.eu-west-1.compute.amazonaws.com:8081/#/auth/login

2) Introducir las credenciales del usuario. En caso de que no se haya registrado, puede crearse una nueva cuenta.

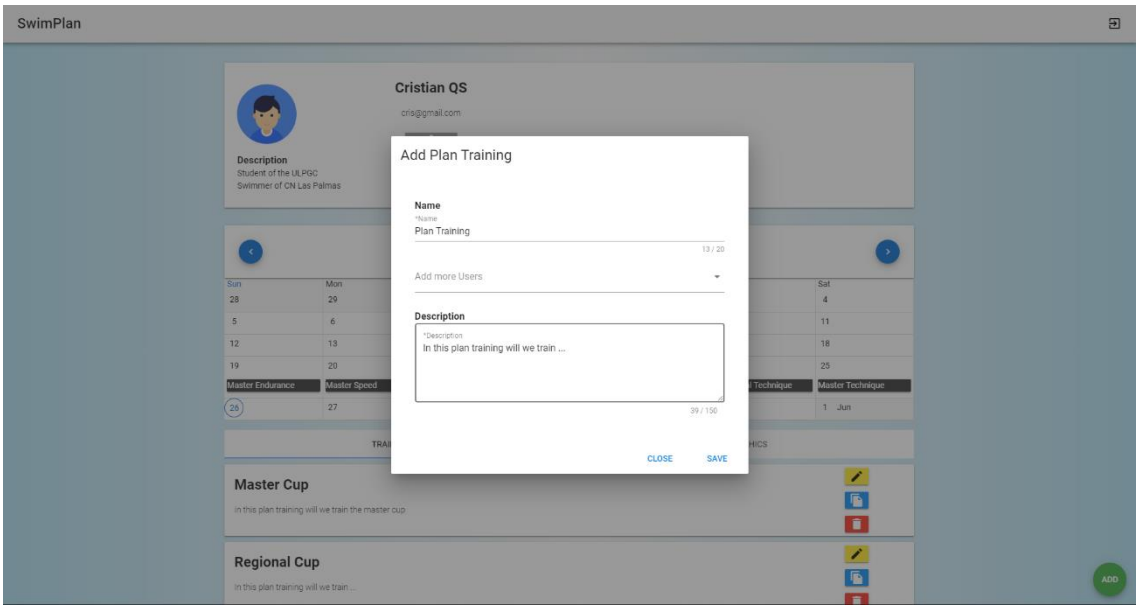
a. Si se va a crear una nueva cuenta, debe rellenar todos los campos del formulario



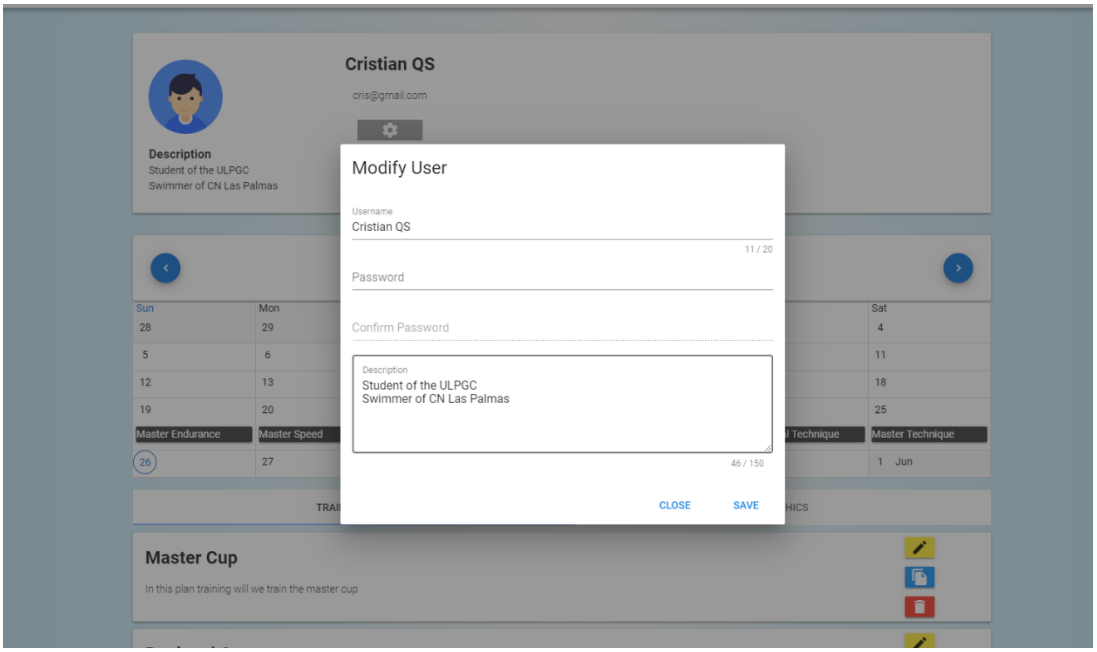
3) Una vez dentro de la aplicación podrá ver su perfil, planes de entrenamiento o un calendario con los distintos entrenamientos programados. Si hace *click* en el entrenamiento indicado en el calendario se redirigirá a las actividades de ese entrenamiento.



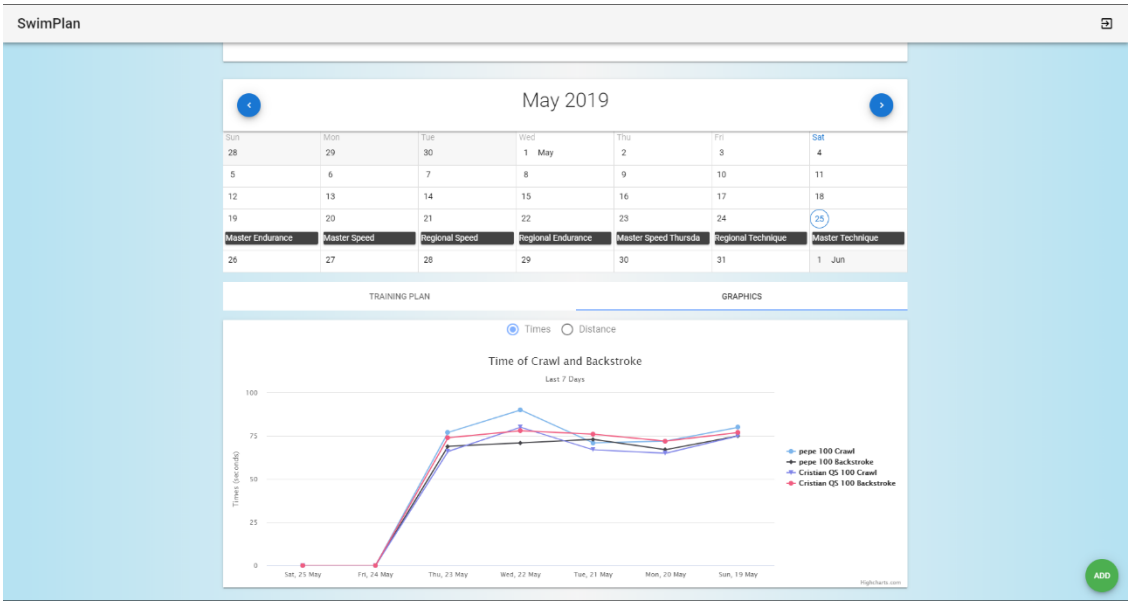
a) Si hace *click* en el botón *ADD*, le aparecerá una ventana en el que podrá introducir los datos de un plan de entrenamiento. Para facilitar la inserción de datos, la aplicación ofrece unos valores por defecto. Además, el plan de entrenamiento puede asignarse a uno o varios nadadores. Este formulario también aparece en el botón modificar.

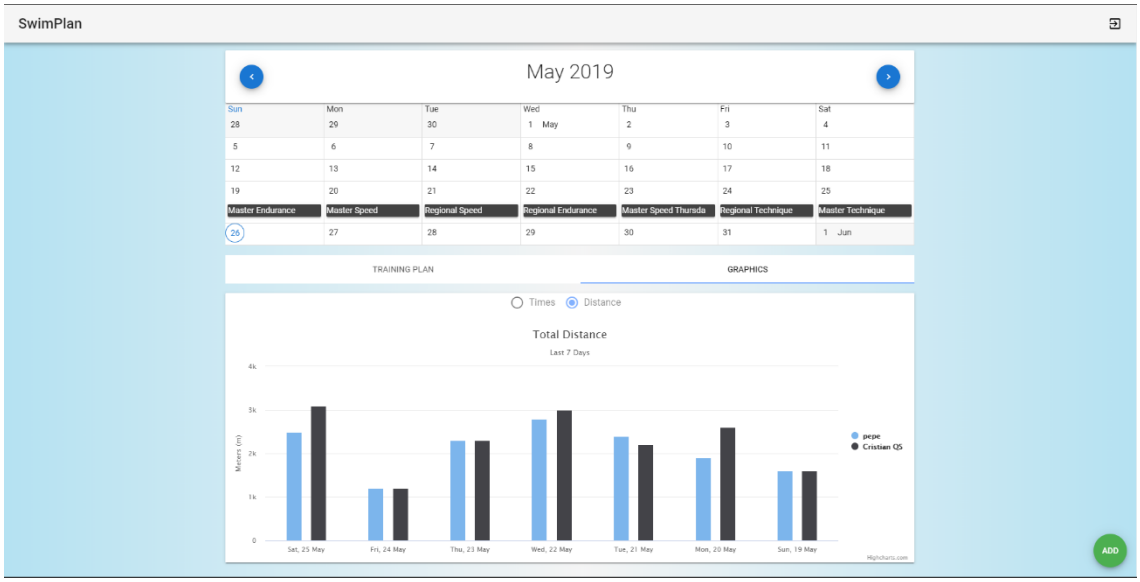


b) Por otro lado, si realizamos un click en el botón de ajustes que se encuentra debajo de nuestro email, podremos modificar nuestro usuario, a excepción del email.

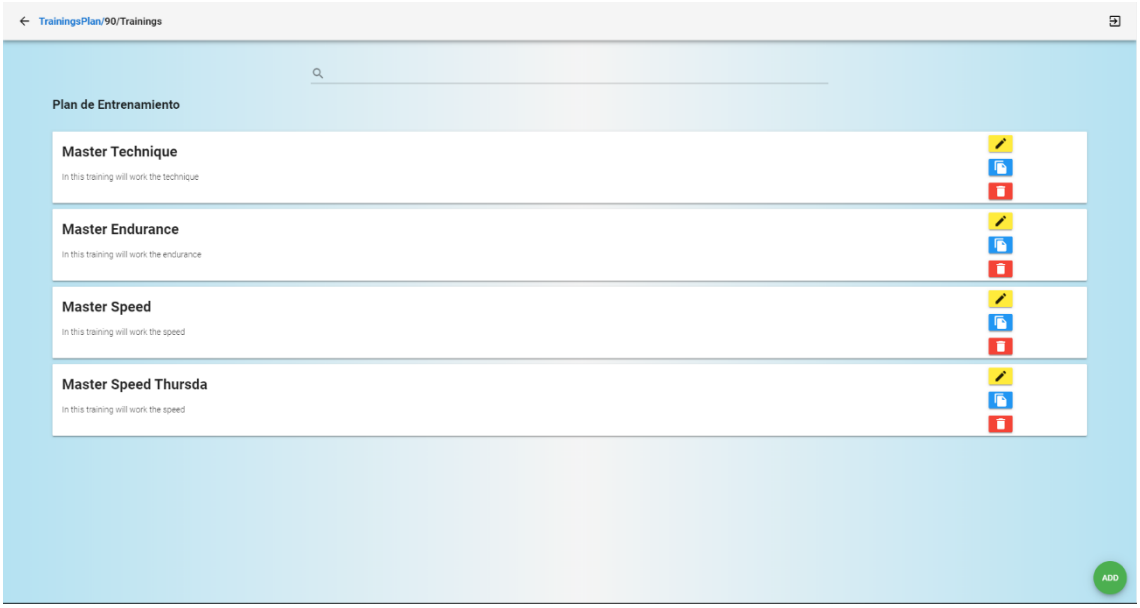


c) Podemos visualizar las gráficas cuando si seleccionamos la pestaña *Graphics*. Se puede elegir entre dos tipos de gráficas: la de distancia y la de tiempos. Cabe destacar que, si realizamos un *click* en la leyenda que aparece en el lado derecho de la gráfica, podemos añadir/quitar series de datos incluidos en el gráfico.

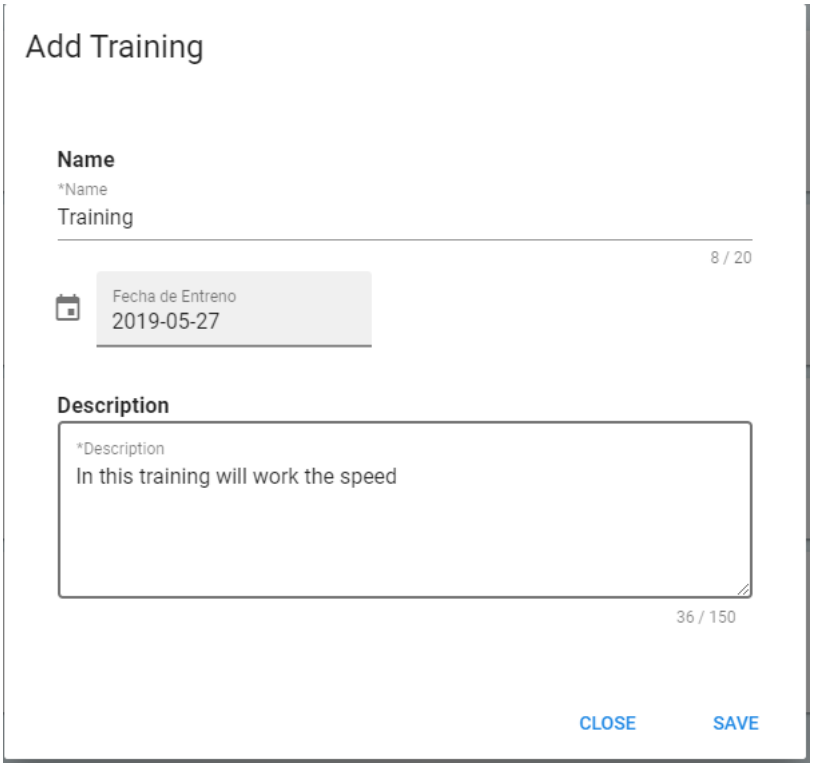




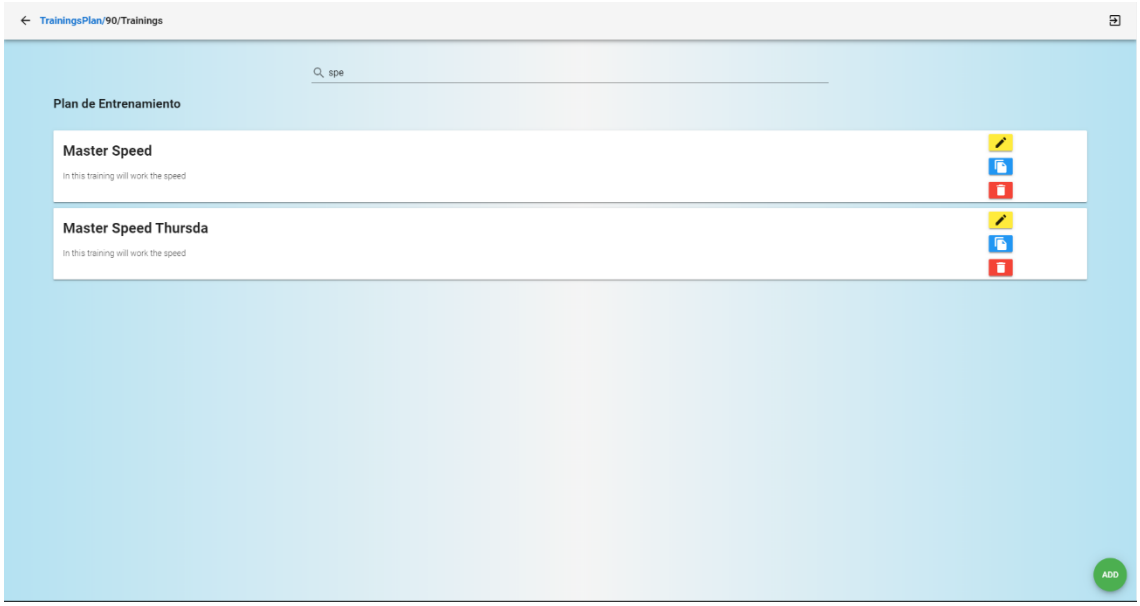
4) Si realizamos un click sobre el nombre del plan de entrenamiento, este nos llevará al listado de entrenamientos (trainings), en la cual aparecerá los distintos entrenamientos incluidos en dicho plan.



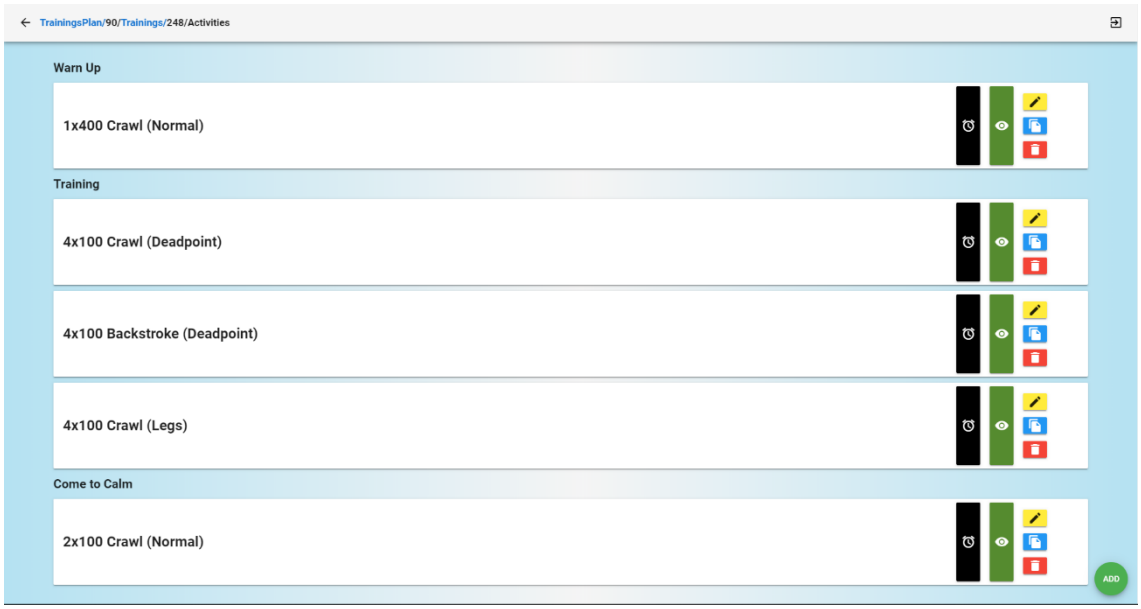
a) Resaltar que tanto el botón ADD como modificar muestran el siguiente formulario.



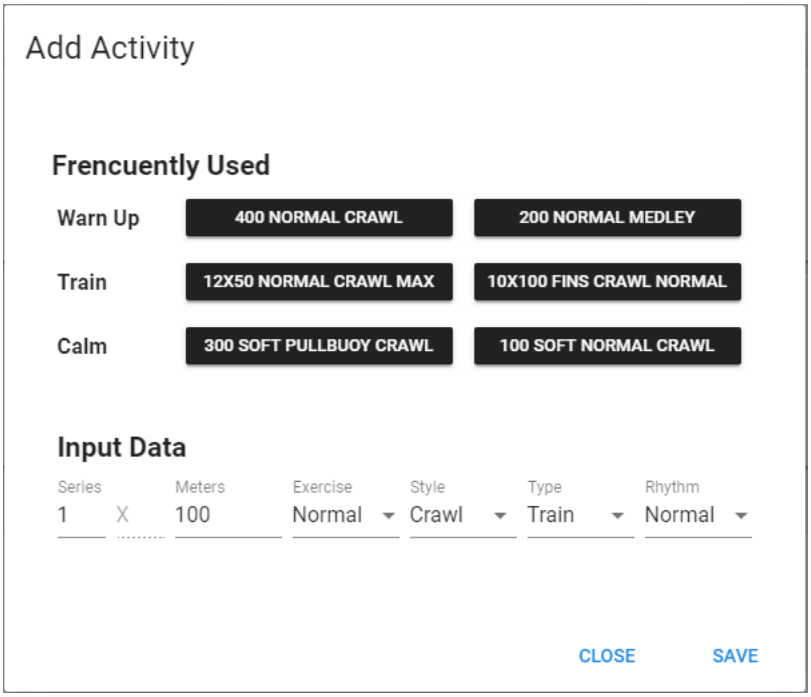
b) Hay que mencionar que en el buscador que se encuentra en la parte superior, si introducimos alguna letra o palabra que contenga el nombre del entrenamiento, este filtrará aquellos que la posean.



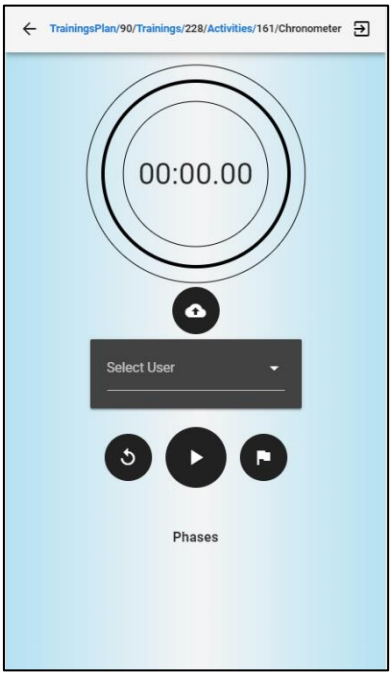
5) Si realizamos un click sobre el nombre del entrenamiento, este nos llevará a la vista de ejercicios (activities), en la cual aparecerá las distintas actividades de dicho entrenamiento



a) Resaltar que el botón ADD permite crear tanto un entrenamiento preestablecido o crear uno propio a mano; mientras que el modificar solo permite modificar los datos existentes, es decir, sin la sección ***Frecuently Used.***

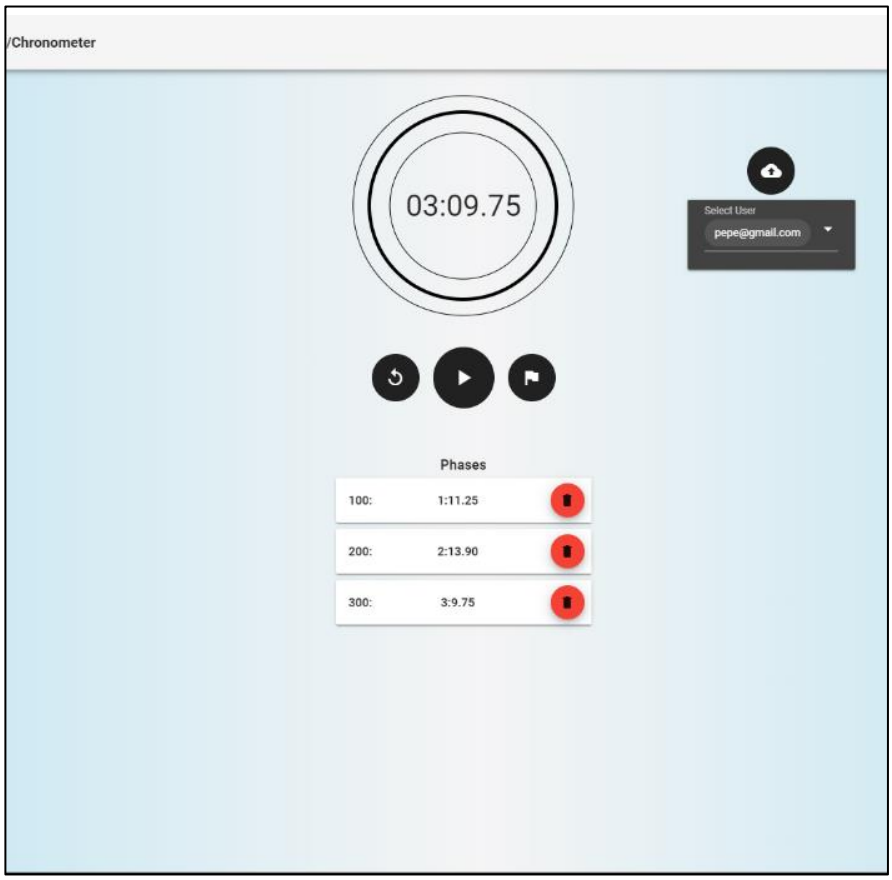


b) Si realizamos un click en el botón negro que posee un cronómetro, este nos llevará a la vista cronometro en el que se podrá iniciar la toma de tiempos, deternerlo o resetearlo. Además, se podrán guardar estos tiempos, así como seleccionar al usuario que ha realizado dicho tiempo.

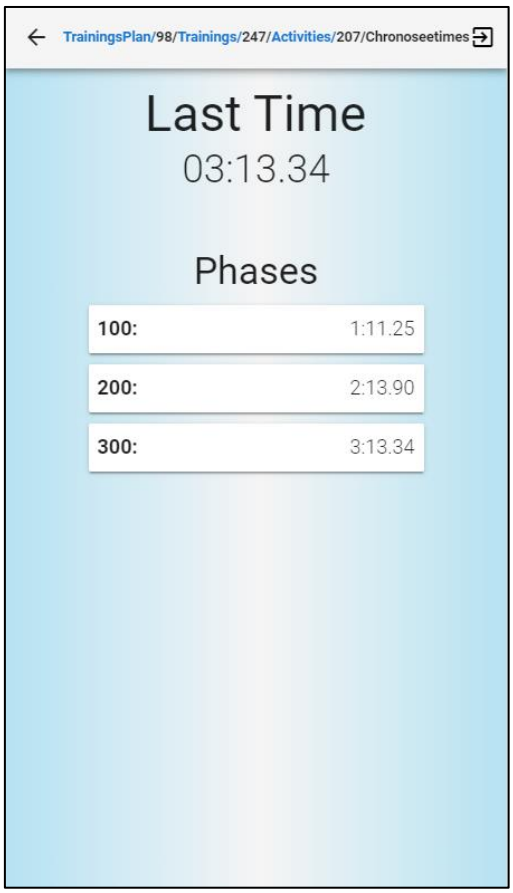


- Cuando la toma de tiempo la lleva a cabo un usuario con el rol de nadador, el tiempo se le asignará únicamente a él y no le aparecerá la lista de usuarios.

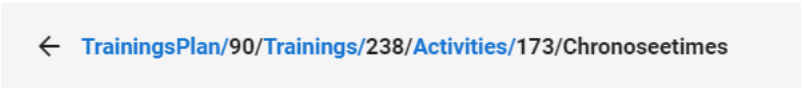
- Asimismo, los usuarios podrán tomar fases (parciales) en el cronómetro, las cuales se pueden ir visualizando, o eliminando. Hay que matizar que cuando se sube los tiempos, también se suben estas fases.



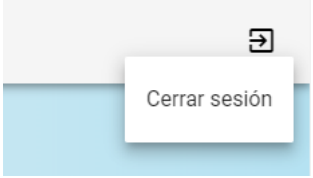
c) Por otro lado, también nos encontramos con el botón verde oscuro que posee un ojo, en el cuál podremos visualizar los tiempos y fases de dicha actividad.



6) En la barra de navegación de todas las vistas menos la de profile, siempre nos muestra la ruta en la que nos encontramos, en la que cada enlace en azul nos lleva a esa vista. Además, el botón con una flecha hacia atrás nos llevará a la página anterior que hemos visitado.



a) En todas las vistas nos encontraremos con el logo de *exit*, para poder cerrar sesión en cualquier momento.



Bibliografía

[1] Strava, «Strava,» [En línea]. Available: https://www.strava.com/. [Último acceso: 05 2019].

[2] Swimrankings, «Swimrankings,» [En línea]. Available: https://www.swimrankings.net/?page=meetmanager. [Último acceso: 05 2019].

[3] D. RAINMAKER, «Garmin Connect IQ Summit Day 2 Announcements | DC Rainmaker,» [En línea]. Available: https://www.dcrainmaker.com/2019/04/garmin-connect-iq-summit-day-2-announcements.html. [Último acceso: 05 2019].

[4] Garmin, «Garmin,» [En línea]. Available: https://developer.garmin.com/connect-iq/api-docs/. [Último acceso: 05 2019].

[5] Movesense, «bitbucket,» [En línea]. Available: https://bitbucket.org/suunto/movesense-device-lib/src/master/README.md.

[6] Polar, «Polar,» [En línea]. Available: https://www.polar.com/accesslink-api/#introduction. [Último acceso: 05 2019].

[7] A. I. Types, «AWS,» [En línea]. Available: https://aws.amazon.com/es/ec2/instance-types/. [Último acceso: 05 2019].

[8] A. W. is?, «AWS,» [En línea]. Available: https://aws.amazon.com/es/what-is-aws/. [Último acceso: 05 2019].

[9] A. Infrastructure, «AWS,» [En línea]. Available: https://aws.amazon.com/es/about-aws/global-infrastructure/. [Último acceso: 05 2019].

[10] A. EC2, «AWS,» [En línea]. Available: https://aws.amazon.com/es/ec2/. [Último acceso: 05 2019].

[11] A. RDS, «AWS,» [En línea]. Available: https://aws.amazon.com/es/rds/faqs/. [Último acceso: 05 2019].

[12] Santiago, «PostgreSQL vs MySQL – ¿Cuál es mejor?,» [En línea]. Available: https://guiadev.com/postgresql-vs-mysql/. [Último acceso: 05 2019].

[13] Docker, «Docker,» [En línea]. Available: https://www.docker.com/resources/what-container. [Último acceso: 05 2019].

[14] W. Docker, «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Docker_(software). [Último acceso: 05 2019].

[15] J. Pipeline, «jenkins,» [En línea]. Available: https://jenkins.io/doc/book/pipeline/. [Último acceso: 05 2019].

[16] E. Fernández, «La integración continua actual pasa por pipelines,» [En línea]. Available: https://sdos.es/blog/la-integracion-continua-actual-pasa-por-pipelines. [Último acceso: 05 2019].

[17] L. Llamas, «uetify, estética Material Design para tus Apps en VueJS,» [En línea]. Available: https://www.luisllamas.es/vuetify-estetica-material-design-para-tus-apps-en-vuejs/. [Último acceso: 05 2019].

[18] W. Javascript, «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/JavaScript. [Último acceso: 05 2019].

[19] Highcharts, «Highcharts,» [En línea]. Available: https://www.highcharts.com/blog/products/highcharts/. [Último acceso: 05 2019].

[20] W. Django, «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Django_(framework). [Último acceso: 05 2019].

[21] W. Python, «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Python. [Último acceso: 05 2019].

[22] Gherkin, «What is Gherkin? Write Gherkin Test in Cucumber,» [En línea]. Available: https://www.guru99.com/gherkin-test-cucumber.html. [Último acceso: 05 2019].

[23] W. Git, «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Git. [Último acceso: 05 2019].

[24] J. Francia, «¿Qué es Scrum?,» [En línea]. Available: https://www.scrum.org/resources/blog/que-es-scrum. [Último acceso: 05 2019].

[25] W. Scrum, «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Scrum_(desarrollo_de_software). [Último acceso: 05 2019].

[26] D. Wells, «Extreme Programming Rules,» [En línea]. Available: http://www.extremeprogramming.org/rules.html. [Último acceso: 05 2019].

[27] JWT, «JSON Web Token Introduction - jwt.io,» [En línea]. Available: https://jwt.io/introduction/. [Último acceso: 05 2019].

[28] H. Vue, «Github,» [En línea]. Available: https://github.com/highcharts/highcharts-vue. [Último acceso: 05 2019].

[29] A. Calculator, «Amazon Web Services Simple Monthly Calculator,» [En línea]. Available: https://calculator.s3.amazonaws.com/index.html. [Último acceso: 05 2019].

[30] M. A. &. K. Juell, «DigitalOcean,» [En línea]. Available: https://www.digitalocean.com/community/tutorials/how-to-install-jenkins-on-ubuntu-18-04. [Último acceso: 05 2019].

[31] J. M. Alarcón, «Webpack: qué es, para qué sirve y sus ventajas e inconvenientes,» [En línea]. Available: https://www.campusmvp.es/recursos/post/webpack-que-es-para-que-sirve-y-sus-ventajas-e-inconvenientes.aspx. [Último acceso: 05 2019].

[32] Vuex, «Vuex,» [En línea]. Available: https://vuex.vuejs.org/. [Último acceso: 05 2019].

[33] J. Gautam, «Medium,» [En línea]. Available: https://medium.com/python-pandemonium/json-web-token-based-authentication-in-django-b6dcfa42a332. [Último acceso: 05 2019].