

**U**NIVERSIDAD

DE **V**IGO

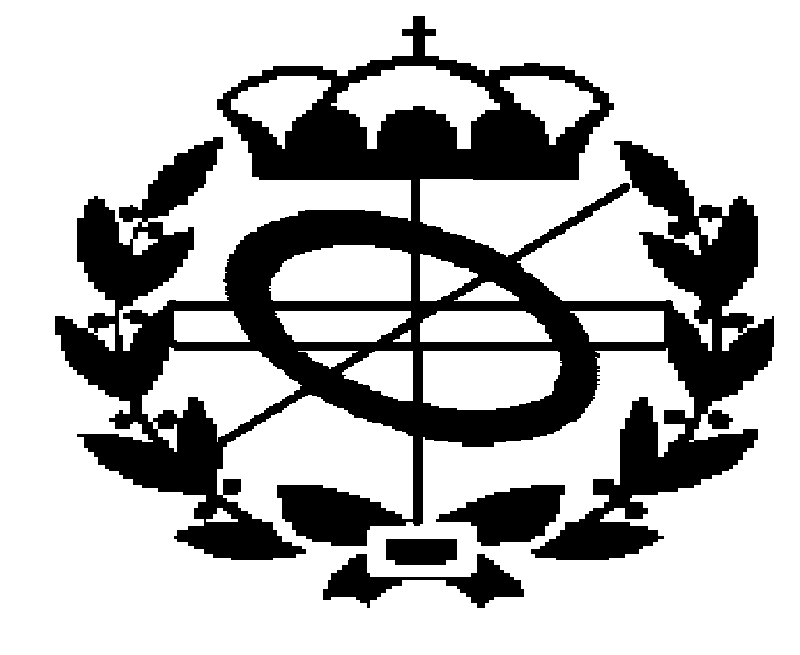
**E**SCUELA **S**UPERIOR DE **I**NGENIERÍA **I**NFORMÁTICA

Memoria del Trabajo de Fin de Grado que presenta

**D. Edgar Conde Nóvoa**

para la obtención del Título de Graduado en Ingeniería Informática

**Aplicación para el Seguimiento de Pruebas de Atletismo**

Enero, 2020

**Trabajo de Fin de Grado Nº**: EI 18/19-077

**Tutor:** Enrique Barreiro Alonso

**Cotutor**: N/A

**Área de conocimiento:** Linguaxes e Sistemas Informáticos

**Departamento:** Informática

# Agradecimientos

*A la comunidad de desarrolladores, la más desinteresada.*

*A mi pareja, por estar siempre ahí.*

*Al tiempo, por poner todo en su sitio.*

Índice de contenidos

[Índice de tablas 6](#_Toc28530341)

[Índice de ilustraciones 6](#_Toc28530342)

[1. Introducción 8](#_Toc28530343)

[2. Objetivos 9](#_Toc28530344)

[3. Resumen de la solución propuesta 9](#_Toc28530345)

[3.1 Solución propuesta 9](#_Toc28530346)

[3.2 Metodología empleada 9](#_Toc28530347)

[4. Planificación y seguimiento 11](#_Toc28530348)

[4.1 Planificación inicial 11](#_Toc28530349)

[4.2 Seguimiento 11](#_Toc28530350)

[5. Arquitectura 13](#_Toc28530351)

[5.1 Servidor 14](#_Toc28530352)

[5.2 Cliente Web 14](#_Toc28530353)

[5.3 Cliente móvil 15](#_Toc28530354)

[6. Tecnologías e integración de terceros 16](#_Toc28530355)

[6.1 Tecnologías 16](#_Toc28530356)

[6.2 Herramientas 17](#_Toc28530357)

[6.3 Librerías 18](#_Toc28530358)

[6.4 Productos de terceros 18](#_Toc28530359)

[7. Especificación y análisis de requisitos 19](#_Toc28530360)

[Gestión de usuarios 19](#_Toc28530361)

[Gestión de atletas 19](#_Toc28530362)

[Gestión de clubes 19](#_Toc28530363)

[Gestión de tipos de prueba 20](#_Toc28530364)

[Gestión de competiciones 20](#_Toc28530365)

[Gestión de pruebas 20](#_Toc28530366)

[Cámara de llamadas 20](#_Toc28530367)

[Gestión de inscripciones 20](#_Toc28530368)

[Gestión de alturas 21](#_Toc28530369)

[Gestión de marcas 21](#_Toc28530370)

[Gestión de clasificaciones 21](#_Toc28530371)

[Autenticación de usuario 21](#_Toc28530372)

[8. Diseño del software 22](#_Toc28530373)

[8.1 Vista estática 22](#_Toc28530374)

[8.2 Vista dinámica 26](#_Toc28530375)

[9 Gestión de datos e información 28](#_Toc28530376)

[9.1 Modelo de datos 28](#_Toc28530377)

[9.2 URIs de los recursos 28](#_Toc28530378)

[10 Pruebas 31](#_Toc28530379)

[Gestión de usuarios 31](#_Toc28530380)

[Gestión de atletas 32](#_Toc28530381)

[Gestión de clubes 33](#_Toc28530382)

[Gestión de tipos de prueba 33](#_Toc28530383)

[Gestión de competiciones 34](#_Toc28530384)

[Gestión de pruebas 35](#_Toc28530385)

[Gestión de inscripciones 35](#_Toc28530386)

[Gestión de alturas 36](#_Toc28530387)

[Gestión de marcas 36](#_Toc28530388)

[Gestión de clasificaciones 37](#_Toc28530389)

[Autenticación de usuario 37](#_Toc28530390)

[11 Manual de usuario 38](#_Toc28530391)

[11.1 Requisitos mínimos 38](#_Toc28530392)

[11.2 Instalación 38](#_Toc28530393)

[11.3 Funcionamiento 39](#_Toc28530394)

[12 Principales aportaciones 48](#_Toc28530395)

[13 Conclusiones 48](#_Toc28530396)

[13.1 Conclusiones técnicas 48](#_Toc28530397)

[13.2 Conclusiones personales 48](#_Toc28530398)

[14 Vías de trabajo futuro 49](#_Toc28530399)

[15 Referencias 49](#_Toc28530400)

Índice de tablas

[Tabla 1. Dedicación al proyecto por Sprint 12](#_Toc28526955)

Índice de ilustraciones

[Ilustración 1. Arquitectura 13](#_Toc28526956)

[Ilustración 2. VIsta estática del Back-End 22](#_Toc28526957)

[Ilustración 3. Diagrama de clases del Back-End 23](#_Toc28526958)

[Ilustración 4. Vista estática del Front-End 24](#_Toc28526959)

[Ilustración 5. Diagrama de módulos del Front-End 24](#_Toc28526960)

[Ilustración 6. LayoutModule 25](#_Toc28526961)

[Ilustración 7. UsuariosModule 26](#_Toc28526962)

[Ilustración 8. Diagrama de secuencia 27](#_Toc28526963)

[Ilustración 9. Modelo de datos 28](#_Toc28526964)

[Ilustración 10. Interfaz web - Listado de pruebas 40](#_Toc28526965)

[Ilustración 11. Interfaz web - Crear prueba 40](#_Toc28526966)

[Ilustración 12. Interfaz web - Editar prueba 41](#_Toc28526967)

[Ilustración 13. Interfaz web - Eliminar prueba 41](#_Toc28526968)

[Ilustración 14. Interfaz web - Hoja de campo 42](#_Toc28526969)

[Ilustración 15. Interfaz web - Anotar intento 42](#_Toc28526970)

[Ilustración 16. Interfaz web - Añadir alturas 43](#_Toc28526971)

[Ilustración 17. Interfaz web - Editar altura 43](#_Toc28526972)

[Ilustración 18. Interfaz web - Eliminar marca 44](#_Toc28526973)

[Ilustración 19. Interfaz web - Ayuda marcas 44](#_Toc28526974)

[Ilustración 20. Interfaz web - Ayuda acciones 45](#_Toc28526975)

[Ilustración 21. Interfaz web - Finalizar prueba 45](#_Toc28526976)

[Ilustración 22. Interfaz móvil - Inicio de sesión y listado de competiciones 46](file:///C:\Users\Edgar\Repositorios\ASPA\aspa-documentacion\TFG%20-%20Aplicación%20para%20el%20Seguimiento%20de%20Pruebas%20de%20Atletismo.docx#_Toc28526977)

[Ilustración 23. Interfaz móvil - Listado de pruebas, clasificación de prueba y marcas de atleta 46](file:///C:\Users\Edgar\Repositorios\ASPA\aspa-documentacion\TFG%20-%20Aplicación%20para%20el%20Seguimiento%20de%20Pruebas%20de%20Atletismo.docx#_Toc28526978)

[Ilustración 24. Interfaz móvil - Filtro de competiciones e información de una prueba 47](file:///C:\Users\Edgar\Repositorios\ASPA\aspa-documentacion\TFG%20-%20Aplicación%20para%20el%20Seguimiento%20de%20Pruebas%20de%20Atletismo.docx#_Toc28526979)

[Ilustración 25. Interfaz móvil - Inscripción en una prueba 47](file:///C:\Users\Edgar\Repositorios\ASPA\aspa-documentacion\TFG%20-%20Aplicación%20para%20el%20Seguimiento%20de%20Pruebas%20de%20Atletismo.docx#_Toc28526980)

# Introducción

Hoy en día vemos que el software, hardware y la tecnología en general son los responsables de la modernización en cualquier ámbito. Uno de ellos es el deporte, aunque por desgracia y como en cualquier campo, no todas las disciplinas tienen el mismo protagonismo. Una de ellas es el atletismo de competición, un deporte que practica más gente de la que alguien abstraído de ese mundo puede imaginar y del que casi únicamente se escuchan noticias cuando se están celebrando los Juegos Olímpicos o alguna competición internacional de prestigio.

El atletismo es un deporte que engloba numerosas disciplinas que se pueden practicar al aire libre o a cubierto. Éstas son el conjunto de todos los tipos de carreras (maratones, pruebas de velocidad, relevos…) y de todos los tipos de concursos (salto de altura, lanzamiento de peso, salto con pértiga, etc.). Estas pruebas acostumbran ser celebradas durante los fines de semana por todas las delegaciones de Galicia y de España.

Desde el año 2018, la ciudad de Ourense dispone de una pista cubierta de atletismo, y la celebración de pruebas a cubierto es algo frecuente durante la temporada desde entonces. Esta nueva pista dispone de las mejores instalaciones actualmente en Galicia y es por eso que ahora es el núcleo dónde han pasado a reunirse los atletas para exhibirse en las competiciones de la Federación Gallega de Atletismo.

Pero la cosa no ha quedado ahí, durante el pasado mes de marzo de 2019 se ha celebrado el Campeonato de España de Atletismo de la categoría Máster (atletas de 30 o más años). Ese campeonato supuso una prueba de fuego para las instalaciones, la organización y los jueces, pues se prevé que en 2020 se celebre el Campeonato de España de categoría Absoluta. Un campeonato que reunirá a los mejores atletas del país, así como a los mejores jueces y a los medios de comunicación. Sin embargo, pese a ser competiciones de tanta relevancia, hay aspectos que no están a la altura.

Los jueces son los encargados de que se cumplan los horarios de las pruebas, de aplicar y revisar que se aplique el reglamento durante las mismas y de registrar los resultados. Actualmente los listados de inscripciones que manejan estos jueces, así como los horarios y las hojas de campo, se imprimen en papel. En el caso de las hojas de campo, una vez cubiertas deben ser revisadas y posteriormente entregadas a alguien de la organización que se encargue de introducir los datos de esa hoja de forma manual en la base de datos.

En este Trabajo de Fin de Grado (TFG) se propone la creación de una herramienta que facilite y ayude a agilizar la gestión, así como el registro y consulta de resultados de pruebas de atletismo correspondientes a los concursos. Con la intención de dar una experiencia fluida y teniendo presente la evolución actual en tecnologías web, se ha considerado crear un cliente Web como Single Page Application (SPA) y una pequeña aplicación móvil híbrida, utilizando también tecnologías web.

# Objetivos

Este TFG tiene como objetivo principal el desarrollo de una aplicación web para gestionar pruebas de atletismo y cubrir las hojas de campo de las mismas, así como una pequeña aplicación móvil para consultar los resultados de las pruebas y desde la cual un atleta pueda inscribirse en ellas. Estas dos aplicaciones aglutinarán las siguientes funciones:

* Gestión de usuarios(jueces).
* Gestión de clubes y atletas.
* Creación de competiciones y asignación de pruebas(concursos).
* Relleno de las hojas de campo de las pruebas aplicando sus reglamentos.
* Inscripción de atletas y control de la cámara de llamadas.
* Consulta de clasificaciones y resultados de las pruebas.

También se establece como objetivo que la interfaz sea lo más intuitiva posible, sobre todo la del cliente web, ya que se establece como propósito que sea sencilla para que un juez de atletismo, que puede tener o no agilidad en el uso de aplicaciones, se desenvuelva sin ningún problema.

# Resumen de la solución propuesta

## 3.1 Solución propuesta

La solución propuesta consiste en construir las aplicaciones que cumplan con los objetivos definidos en el apartado anterior, en concreto haciendo uso de las siguientes tecnologías principales:

* **Front-end**: Se utiliza el framework Angular para crear una SPA basada en Componentes Web. El desarrollo de esta SPA debe estar enfocado a su uso en tabletas, que serían el dispositivo más indicado para el propósito de este proyecto, presuponiendo que esto implica que también pueda ser usada en pantallas más grandes. Se propone desarrollar también un pequeño cliente móvil haciendo uso de la herramienta Ionic, que proporciona facilidades para el desarrollo de aplicaciones móviles híbridas basadas en tecnologías Web. De este modo se pretende generar dos aplicaciones para distintas plataformas usando la misma tecnología: Angular.
* **Back-end**: Se ha decidido utilizar el framework Spring Boot para la creación de una API REST que consumirán ambos clientes. En cuanto a la persistencia de datos, se usará Hibernate para implementar la API de Persistencia de Java (JPA) y manipular una base de datos relacional MySQL.

La elección de estas tecnologías ha sido motivada por la intención del alumno en conocer nuevas tecnologías de desarrollo web front-end así como por las ventajas que supone implementar clientes y servidor de una forma tan desacoplada.

## Metodología empleada

En cuanto al proceso de desarrollo llevado a cabo para este proyecto, se ha implementado una metodología ágil por los siguientes motivos:

* Para aumentar el tiempo de implementación. Existe una cantidad de horas estimadas a invertir en el proyecto (300 horas) que supondrían una limitación a la hora de querer entregar un producto funcional si se invierte demasiado tiempo en documentación, teniendo en cuenta que el propio producto no es de una complejidad que requiera de documentación exhaustiva para la comprensión de su implementación.
* Para disponer de flexibilidad y capacidad de adaptación ante cambios.

La metodología ágil utilizada ha sido inspirada por Scrum, dónde el proceso se organiza en sprints caracterizados por lo siguiente:

* Tienen una duración fija, entre una semana y un mes.
* Deben incluir la planificación previa de dicho sprint y un grupo de reuniones de demostración y retrospectivas vitales para el correcto funcionamiento de la metodología.

Por otro lado, los artefactos que se manejan durante el desarrollo en Scrum son:

* **Product Backlog**: es una lista en la que se organizan y priorizan todos los requisitos. Proporciona una vista general del proyecto y evoluciona a lo largo del mismo.
* **Sprint Backlog**: recoge los requisitos a completar en el siguiente sprint y es habitual que se subdividan en tareas.
* **Incremento del producto:** es el resultado de un sprint, que debería corresponderse con una versión potencialmente publicable del producto.

Finalmente, los roles que forman un equipo Scrum:

* **Product Owner**: decide qué desarrollar, cuándo y en qué orden, siempre teniendo en cuenta las necesidades del cliente.
* **Scrum Master**: ayuda al equipo en la creación y seguimiento del proceso de desarrollo propio basado en Scrum, solucionando cualquier problema que surja en cuanto a productividad y rendimiento.
* **Equipo de desarrollo**: equipo responsable de determinar cómo producir lo que solicite el Product Owner.

Como se ha mencionado anteriormente, en este TFG se ha implementado una metodología ágil inspirada por Scrum (**no** es Scrum), definiendo un **marco de trabajo** regido por lo siguiente:

* El proyecto:
  + Su duración estimada es de 300 horas.
  + La disponibilidad del equipo de desarrollo es de unas 12-15 horas semanales.
* Los artefactos:
  + Como en Scrum, se ha trabajado con el Backlog del producto y con los Sprint Backlog, definidos al inicio de cada sprint.
  + En el marco de trabajo que se ha definido y a diferencia de Scrum, el incremento en cada sprint no supone necesariamente una versión publicable del producto, aunque se establece como propósito que se corresponda con la adición de funcionalidades que no dependan de desarrollos posteriores.
* El equipo
  + Alumno: realiza el papel equivalente tanto al de Product Owner como al de Equipo de desarrollo en Scrum. Es el conocedor de los requisitos y al mismo tiempo el encargado de desarrollar la solución.
  + Tutor: realiza un papel similar al de Scrum Master pero, por restricciones de localización, sin la posibilidad de reunirse periódicamente con el equipo de desarrollo. Ejerce de mero revisor del estado del producto y de los sprints.

# Planificación y seguimiento

En este apartado se realiza una comparativa entre la planificación inicial y el tiempo real de ejecución invertido en el desarrollo del proyecto. Se ilustrará mediante tablas y gráficas.

## 4.1 Planificación inicial

La planificación inicial del proyecto se realizó acorde a dos restricciones. La primera, que la dedicación máxima al TFG debería ser de 300 horas. La segunda, la disponibilidad del alumno. Por razones de carácter personal y laboral, ésta se estimó en unas 12-15 horas semanales. Por tanto, se estableció una planificación consistente en 8 sprints de 3 semanas cada uno, lo que implica 12.5 horas de dedicación semanal y 37.5 horas por sprint. De este modo, a la hora de definir cada sprint backlog se han seleccionado tareas para una dedicación estimada de entre 35 y 40 horas durante ese sprint.

Al inicio del proyecto se contaba con una batería de historias de usuario/tareas en el product backlog, que ha ido incrementando a lo largo del proyecto. Al inicio de cada sprint se han seleccionado tareas para una dedicación estimada de entre 35 y 40 horas.

Dadas las características de la metodología de trabajo empleada, es imposible ilustrar una planificación inicial de tareas para todo el proyecto puesto que ésta no existía. Al inicio del proyecto no se conocían la totalidad de tareas a realizar y por supuesto tampoco en qué sprint se iban a desarrollar. Se conocían las funcionalidades del sistema, y los requisitos de éstas se han ido analizando de forma continua durante el proyecto para definir tareas al inicio de cada sprint.

En definitiva, la planificación de este proyecto ha sido estrictamente temporal para la distribución de horas de trabajo.

## Seguimiento

Pese a haberse invertido tiempo en investigación acerca de las tecnologías empleadas, la necesidad de aprendizaje sobre algunas de sus características no ha cesado durante el transcurso del proyecto, y en ocasiones ha conllevado retrasos por diversos motivos:

* Angular: Pese a conocimientos previos en Angular, se han tenido que asimilar los cambios de las últimas versiones de este framework que evoluciona tan rápidamente.
* Spring: Inversión del Control e Inyección de Dependencias.
* Hibernate: Lenguaje de consulta (HQL), anotaciones.
* Despliegues: Al terminar la implementación de una nueva funcionalidad, se ha efectuado un despliegue que implica una serie de tareas que conllevan tiempo extra sobre el de desarrollo.
* Contexto del proyecto: Revisión de reglamentos de las pruebas de atletismo para la correcta implementación de algunas historias de usuario.

En un inicio, se consideró implantar un sistema de puntos de historia para dar valor a las tareas realizadas en los sprint y hacer así un seguimiento del alcance del proyecto. Esta opción fue descartada por dos motivos:

* Para poder asignar puntos de historia a tareas, primero es necesario contar con un histórico de tareas ya terminadas en el proyecto que, conociendo su dificultad y el tiempo que se invirtió en realizarlas, nos sirvan para identificar tareas que se puedan usar como “pivote” para estimar tareas futuras. Si un equipo con 6 desarrolladores (240 horas semanales de trabajo del equipo) necesita un par de sprints para encontrar un ritmo de trabajo y comenzar a seleccionar esas tareas “pivote” con cierta precisión…en un proyecto corto de 300 horas como el que se ha llevado a cabo, ¿realmente merece la pena? Definitivamente no. Si se implantase ese sistema en este proyecto desde sus inicios, no sería eficaz. Si se implantase cuando se dispusiese de trabajo realizado suficiente para ejecutarlo con precisión, probablemente ya se hubiesen consumido las horas de trabajo de las que se disponen para este TFG.
* Los sistemas de puntos de historia están pensados para equipos que quieren encontrar su ritmo de trabajo, y que independientemente del desarrollador que vaya a realizar una tarea, ésta esté valorada de forma que todos los miembros del equipo sean conscientes de su dificultad. En el proyecto actual, el equipo está conformado por un único desarrollador y esto jamás va a cambiar, por lo que no compensaría el esfuerzo invertido en implantar ese sistema, y todavía menos en un entorno que se presupone ágil.

Dadas las argumentaciones anteriores, el seguimiento del proyecto se ilustra únicamente con el registro de horas invertidas en las tareas de cada sprint, recordando que a cada sprint se asignó la carga de trabajo estimada con anterioridad de 35-40 horas (Tabla 1).

Tabla 1. Dedicación al proyecto por Sprint

|  |  |
| --- | --- |
| **Sprint 1** 21/01 – 08/02 | |
| Creación proyecto API REST (12h) | Crear proyecto Angular web app (6h) |
| Maquetación del layout de la aplicación (18h) | Login y control de sesión (2h) |
| Dedicación estimada: 35 – 40 horas | Dedicación real: 38 horas |
| **Sprint 2** 11/02 – 01/03 | |
| Gestión de usuarios (15h) | Autenticación de usuarios mediante JWT (5h) |
| Autorización de usuarios API y Web (5h) | Configurar despliegue remoto (15h) |
| Dedicación estimada: 35 – 40 horas | Dedicación real: 40 horas |
| **Sprint 3** 04/03 – 22/03 | |
| Gestión de tipos de prueba (10h) | Gestión de pruebas (20h) |
| Gestión de competiciones (10h) |  |
| Dedicación estimada: 35 – 40 horas | Dedicación real: 40 horas |
| **Sprint 4** 01/04 - 12/04 & 22/04 - 26/04 | |
| Validación token almacenado en localStorage (1h) | Controlador CRUD para inscripciones (4h) |
| Controlador CRUD para marcas (8h) | Controlador CRUD para alturas (2h) |
| Cubrir hoja de campo (no SV) (40h) |  |
| Dedicación estimada: 35 – 40 horas | Dedicación real: 55 horas |
| **Sprint 5** 13/05 - 31/05 | |
| Cubrir hoja de campo (SV) (42h) | Controlador CRUD para clasificaciones (5h) |
| Dedicación estimada: 35 – 40 horas | Dedicación real: 47 horas |
| **Sprint 6** 03/06 - 07/06 & 17/06 - 28/06 | |
| Horarios de la cámara de llamadas (8h) | Inscripción de atleta en prueba (12h) |
| Confirmación de asistencia de atleta (6h) | Revisión documentación (10h) |
| Dedicación estimada: 35 – 40 horas | Dedicación real: 36 horas |
| **Sprint 7** 01/07 - 12/07 & 22/07 - 26/07 | |
| Gestión de clubes (6h) | Gestión de atletas (12h) |
| [Cliente móvil] Pantalla de login de atletas (6h) | [Cliente móvil] Pantalla de resultados de las pruebas (20h) |
| Dedicación estimada: 35 – 40 horas | Dedicación real: 44 horas |
| **Sprint 8** 09/09 - 27/09 | |
| [Cliente móvil] Inscripción de atleta en prueba (6h) | [Cliente móvil] Pantalla de resultados de las pruebas en curso (18h) |
| Recopilación documentación del proyecto para la redacción de los documentos a entregar. (20h) |  |
| Dedicación estimada: 35 – 40 horas | Dedicación real: 44 horas |

* **Horas estimadas para ejecutar el proyecto: 300 horas**
* **Horas de trabajo reales invertidas: 344 horas**

# Arquitectura

La arquitectura establecida (Ilustración 1) es en la que se basan todas las aplicaciones Web cuya interfaz es una SPA. Se trata de una arquitectura Cliente-Servidor dónde el cliente (en este caso dos) realiza peticiones de información al servidor y éste contesta enviando una respuesta que normalmente contiene una representación de un recurso.

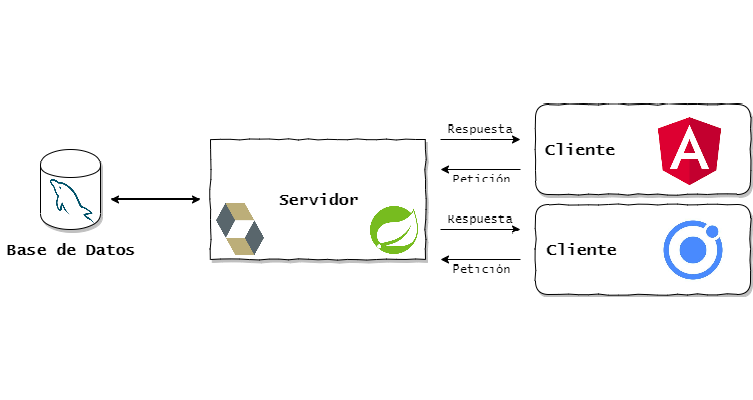


Ilustración 1. Arquitectura

Además de esto, cabe mencionar que tanto la parte Back-End como los clientes tienen su propia arquitectura. El Back-End dispone una arquitectura por capas mientras que los clientes una basada en módulos, entendiendo por módulo un contexto de compilación para un conjunto de componentes web.

Cada arquitectura será explicada en mayor detalle en los siguientes apartados.

## 5.1 Servidor

El intercambio de datos entre el servidor y los clientes se realiza mediante una API que expone recursos que pueden ser consumidos a través URLs y que siguen una serie de reglas. Se ha empleado el estilo arquitectónico REST (Representational State Transfer), el cual gira en torno a recursos, no acciones. Se apoya totalmente en el estándar del protocolo HTTP y proporciona un buen rendimiento, escalabilidad y confiabilidad.

En la Ilustración, se puede observar el funcionamiento de la arquitectura del Back-End. Con SpringBoot se ha creado una aplicación que es configurada para escuchar peticiones vía HTTPS en el puerto 8443. Para poder usar HTTPS, se debe obtener un certificado SSL (en este caso se ha obtenido de la Autoridad Certificadora Let’s Encrypt).

En esta aplicación se ha establecido una arquitectura por capas con el objetivo de separar las distintas responsabilidades y desacoplar lo máximo posible la interfaz que consumen los clientes y el acceso a datos. Estas son las capas:

* **Controllers**: Forman la capa que atiende a las peticiones que han pasado los filtros necesarios. Dada una petición con su verbo (GET, POST, PUT, DELETE…) y unos determinados datos de entrada, los controladores delegan en la capa de servicios para realizar las operaciones pertinentes.
* **Services**: Los objetos que forman parte de esta capa implementan el patrón singleton y son injectados en los controladores que delegan lógica en ellos. Las implementaciones de los servicios son registradas en el contenedor de dependencias de Spring, a través de un único fichero de configuración. De este modo, si se quiere utilizar una implementación de un servicio distinta en toda la aplicación, basta con modificar ese fichero.
* **Repositories**: Esta capa la conforman los objetos que tienen acceso a los datos. Extienden a la clase CrudRepository, que proporciona implementaciones de los métodos básicos de manipulación de datos. A partir de ahí, el desarrollador extiende la funcionalidad de estos repositorios haciendo uso del lenguaje de consulta de Hibernate (HQL).
* **Models**: Son las clases que representan a los modelos del dominio, es decir, los recursos con los que trabaja el API REST. A estos modelos se les conoce como entidades y son configuradas con anotaciones de Hibernate. Estas anotaciones proporcionan facilidades para configurar cómo se parsean las entidades a formato JSON, cómo se almacenan ciertos datos en la BBDD, cómo se relacionan unas entidades y otras…

## Cliente Web

Angular es un framework orientado a la creación de interfaces de usuario a través de componentes. Es una herramienta muy popular cuándo se trata de construir aplicaciones de una sola página y, aunque establece un marco de trabajo, deja flexibilidad al desarrollador a la hora de organizar el proyecto.

En este caso, se ha organizado el proyecto Angular en torno a módulos, distinguibles por sus propósitos:

* **Módulos que se corresponden con funcionalidades**: estos son los módulos de clubes, usuarios, atletas, competiciones y tipos de prueba. Cada uno de ellos aglutina componentes que se encargan de construir la interfaz de usuario correspondiente a una determinada sección de la aplicación.
* **Módulo Core**: este módulo importa los módulos y servicios del framework Angular que serán usados por el resto de módulos de la aplicación.
* **Módulo Shared**: aquí se importan todos los elementos que se usan por toda la aplicación y que no son parte del framework:
  + **Services**: los servicios que usan para realizar peticiones HTTP al API REST.
  + **Models**: los modelos de dominio de la aplicación web.
  + **Components**: componentes creados con propósitos específicos que se usan por toda la aplicación (diálogos de confirmación, spinners, popups...).
* **Módulo layout**: este módulo está formado por los componentes que juntos establecen una interfaz base de la aplicación y su enrutamiento.

Además, en el proyecto se han implementado las siguientes características:

* Restricción de acceso a las diferentes rutas en función de autenticación y autorización.
* Configuración para ser ejecutado en dos entornos diferentes, que denotan principalmente hacia qué servidor se harán las peticiones de recursos:
  + Localhost
  + Servidor remoto (DigitalOcean)

## Cliente móvil

El cliente móvil desarrollado se trata de una aplicación muy sencilla y con arquitectura muy similar al del cliente web, ya que ha sido desarrollado con el mismo framework. En este caso sólo cabe mencionar los módulos correspondientes a las tres funcionalidades que se han implementado:

* Consulta de clasificaciones y resultados de pruebas de todas las competiciones
* Inscripción de atleta en pruebas
* Login de atletas

# Tecnologías e integración de terceros

En este apartado se explican las diferentes tecnologías, librerías, herramientas y productos de terceros que se han utilizado para la realización del proyecto.

## 6.1 Tecnologías

### Angular

Framework de Front-End basado en componentes usado para crear aplicaciones de una sola página.

Utilizado para: desarrollo del cliente web

### Ionic

Herramienta que permite la utilización de tecnologías web, en este caso Angular, en el desarrollo de aplicaciones móviles híbridas.

Utilizado para: desarrollo del cliente móvil

### Typescript

Lenguaje de programación libre y de [código abierto](https://es.wikipedia.org/wiki/C%C3%B3digo_abierto) desarrollado y mantenido por [Microsoft](https://es.wikipedia.org/wiki/Microsoft). Es un superconjunto de [JavaScript](https://es.wikipedia.org/wiki/JavaScript), que esencialmente añade tipos estáticos y objetos basados en clases.

Utilizado para: desarrollo de los clientes web y móvil.

### NPM

Administrador de paquetes JavaScript y el registro de software más grande del mundo.

Utilizado para: gestión de dependencias en proyectos Angular e Ionic.

### Java

Lenguaje de programación orientado a objetos.

Utilizado para: desarrollo de la API.

### Spring Boot

Tecnología dentro del mundo de Spring que agiliza la creación de aplicaciones Spring con estructura Maven totalmente configurada.

Utilizado para: desarrollo de la API.

### Hibernate

Herramienta de [mapeo objeto-relacional](https://es.wikipedia.org/wiki/Mapeo_objeto-relacional) (ORM) para la [plataforma Java](https://es.wikipedia.org/wiki/Plataforma_Java) que facilita el mapeo de atributos entre una [base de datos](https://es.wikipedia.org/wiki/Base_de_datos) relacional tradicional y el modelo de [objetos](https://es.wikipedia.org/wiki/Objetos_(programaci%C3%B3n_orientada_a_objetos)) de una aplicación, mediante archivos declarativos ([XML](https://es.wikipedia.org/wiki/XML)) o anotaciones en las entidades que permiten establecer esas relaciones.

Utilizado para: desarrollo de la API, persistencia de datos.

### Maven

Herramienta de software para la gestión de dependencias y construcción de proyectos [Java](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_Java).

Utilizado para: gestión de dependencias del proyecto API REST.

### MySQL

Sistema de gestión de bases de datos relacionales.

Utilizado para: persistencia de datos.

## Herramientas

### Visual Studio Code

Editor de código fuente potente y ligero desarrollado por Microsoft. Dispone de la mayoría de las características esperadas de un entorno de desarrollo, incluido el control de versiones.

Utilizado para: desarrollo de proyectos Angular e Ionic.

### Eclipse IDE

Entorno de desarrollo muy completo que permite la realización de todo tipo de tareas necesarias para el desarrollo de software. Se distribuye en numerosas versiones distintas, en este caso se ha usado la versión para desarrollo con Java.

Utilizado para: desarrollo del API REST

### MySQL Workbench

Software que facilita la gestión de bases de datos MySQL. También permite la generación del modelo de datos mediante diagramas.

Utilizado para: gestión de la base de datos.

### Git

Software de control de versiones. Lleva el registro de los cambios en los archivos locales y facilita la coordinación del trabajo de varias personas sobre archivos compartidos.

Utilizado para: control de versiones del repositorio, que incluye todos los proyectos.

### GitHub

Herramienta web que complementa a Git. Permite crear un repositorio donde alojar y administrar el código fuente de aplicaciones.

Utilizado para: almacenamiento de código fuente.

### Postman

Herramienta que permite realizar diversas tareas dentro del mundo API REST, como la realización de peticiones o la elaboración de tests para validar el comportamiento de APIs.

Utilizado para: testing de la API REST.

### MobaXterm

Cliente SSH para Widows.

Utilizado para: conexión al servidor remoto para realizar despliegues manuales.

### Microsoft Word

Herramienta de procesamiento de texto y documentos desarrollado por Microsoft, que forma parte de la suite ofimática Office.

Utilizado para: elaboración de la documentación del proyecto.

### Draw.io

Herramienta web que permite el diseño de gráficos y diagramas de una forma sencilla y gratuita.

Utilizado para: elaboración de gráficos y diagramas para la documentación del proyecto.

### Gliffy

Aplicación web que permite la creación de diversos tipos de diagramas.

Utilizado para: elaboración de diagramas para la documentación del proyecto.

### Trello

Herramienta web que permite que equipos realicen una gestión de tareas de manera sencilla, pensada para proyectos regidos por metodologías ágiles.

Utilizado para: gestión de tareas y planificación.

## Librerías

Son numerosas las librerías utilizadas en el proyecto, por lo que únicamente se mencionarán las más relevantes.

### Angular Material

Librería de componentes Material Design para ser usados junto con el framework Angular.

### Apache Cordova

Framework Open Source que permite usar contenido HTML, CSS y JavaScript para crear una aplicación nativa para una variedad de plataformas móviles.

### Moment

Librería para la gestión de fechas y horas en JavaScript.

### Jwt-decode

Librería que ayuda a la descodificación de los JWT de autenticación.

### Io.jsonwebtoken

Librería que agiliza la construcción de JWT.

### Spring-boot-starter-mail

Librería que simplifica el envío de mails.

## 6.4 Productos de terceros

### DigitalOcean

Es una plataforma en la nube que permite crear, entregar, monitorear, escalar y desplegar aplicaciones. Permite a los que la usan no preocuparse por cuestiones de infraestructura y enfocarse únicamente en el desarrollo.

Utilizado para: despliegue de las aplicaciones.

# Especificación y análisis de requisitos

En este apartado se exponen los roles y las distintas funcionalidades a desarrollar durante el proyecto. La implementación de una de estas funcionalidades implica la realización de varias tareas del Product Backlog. Por último, se citarán los requisitos no funcionales de este.

Por una parte, los roles identificados son los siguientes:

* **Público**: usuario no registrado que utiliza la aplicación móvil para consultar resultados y horarios.
* **Atleta**: usuario registrado que utiliza la aplicación móvil para inscribirse/des-inscribirse en pruebas.
* **Juez**: usuario registrado que utiliza la aplicación web para cubrir las hojas de campo de las pruebas y pasar lista de los atletas que se presentan a una prueba.
* **Administrador**: usuario registrado que utiliza la aplicación web y tiene acceso a todas las funcionalidades.

Por otra, las historias de usuarios se describen siguiendo una estructura con estos elementos:

* *Nombre breve.*
* *Descripción de la funcionalidad.*
* *Criterios de aceptación* o la lista de requisitos que tiene que cumplir la historia para que se considere completada.

## Gestión de usuarios

|  |  |
| --- | --- |
| Como | Administrador |
| Quiero | Poder crear, modificar y eliminar usuarios (jueces o administradores). |
| Para | Gestionar las cuentas con roles “Juez” y “Administrador”. |
| Pruebas de aceptación | * Se debe poder crear, modificar y eliminar usuarios. * Si cualquier acción se realizó correctamente, se debe mostrar un mensaje indicándolo. En caso contrario, también. |

## Gestión de atletas

|  |  |
| --- | --- |
| Como | Administrador |
| Quiero | Poder crear, modificar y eliminar atletas. |
| Para | Gestionar las cuentas con rol “Atleta”. |
| Pruebas de aceptación | * Se debe poder crear, modificar y eliminar atletas. * Si cualquier acción se realizó correctamente, se debe mostrar un mensaje indicándolo. En caso contrario, también. |

## Gestión de clubes

|  |  |
| --- | --- |
| Como | Administrador |
| Quiero | Poder crear, modificar y eliminar clubes. |
| Para | Gestionar los clubes de atletismo. |
| Pruebas de aceptación | * Se debe poder crear, modificar y eliminar clubes. * Si cualquier acción se realizó correctamente, se debe mostrar un mensaje indicándolo. En caso contrario, también. |

## Gestión de tipos de prueba

|  |  |
| --- | --- |
| Como | Administrador |
| Quiero | Poder crear y eliminar tipos de prueba. |
| Para | Gestionar los tipos de prueba. |
| Pruebas de aceptación | * Se debe poder crear y eliminar tipos de prueba. * Si cualquier acción se realizó correctamente, se debe mostrar un mensaje indicándolo. En caso contrario, también. |

## Gestión de competiciones

|  |  |
| --- | --- |
| Como | Administrador |
| Quiero | Poder crear, modificar y eliminar competiciones. |
| Para | Gestionar las competiciones de atletismo. |
| Pruebas de aceptación | * Se debe poder crear, modificar y eliminar competiciones. * Si cualquier acción se realizó correctamente, se debe mostrar un mensaje indicándolo. En caso contrario, también. |

## Gestión de pruebas

|  |  |
| --- | --- |
| Como | Administrador |
| Quiero | Poder crear, modificar y eliminar pruebas. |
| Para | Gestionar las pruebas de atletismo. |
| Pruebas de aceptación | * Se debe poder crear, modificar y eliminar pruebas. * Si cualquier acción se realizó correctamente, se debe mostrar un mensaje indicándolo. En caso contrario, también. |

## Cámara de llamadas

|  |  |
| --- | --- |
| Como | Administrador |
| Quiero | Poder marcar como asistentes o no los atletas inscritos en una prueba. |
| Para | Confirmar la participación en una prueba de los atletas inscritos en ella. |
| Pruebas de aceptación | * Se debe poder marcar como asistentes o no a los atletas inscritos en una prueba. * Si cualquier acción se realizó correctamente, se debe mostrar un mensaje indicándolo. En caso contrario, también. |

## Gestión de inscripciones

|  |  |
| --- | --- |
| Como | Administrador / Atleta |
| Quiero | Poder crear y eliminar inscripciones. |
| Para | Inscribir/des-inscribir un atleta en una prueba. |
| Pruebas de aceptación | * Se debe poder crear y eliminar inscripciones. * Si el usuario tiene rol “Atleta”, sólo podrá crear o eliminar sus inscripciones. * Si cualquier acción se realizó correctamente, se debe mostrar un mensaje indicándolo. En caso contrario, también. |

## Gestión de alturas

|  |  |
| --- | --- |
| Como | Administrador / Juez |
| Quiero | Poder crear, modificar y eliminar alturas. |
| Para | Poder añadir, modificar o eliminar alturas en una prueba de tipo SV (salto vertical). |
| Pruebas de aceptación | * Se debe poder crear, modificar y eliminar alturas. * Si cualquier acción se realizó correctamente, se debe mostrar un mensaje indicándolo. En caso contrario, también. |

## Gestión de marcas

|  |  |
| --- | --- |
| Como | Administrador / Juez |
| Quiero | Poder crear, modificar y eliminar marcas. |
| Para | Añadir, editar y eliminar marcas para poder cubrir la hoja de campo de una prueba. |
| Pruebas de aceptación | * Se debe poder crear, modificar y eliminar marcas. * Si cualquier acción se realizó correctamente, se debe mostrar un mensaje indicándolo. En caso contrario, también. |

## Gestión de clasificaciones

|  |  |
| --- | --- |
| Como | Administrador |
| Quiero | Poder crear clasificaciones. |
| Para | Generar las clasificaciones de pruebas de atletismo. |
| Pruebas de aceptación | * Se debe poder generar la clasificación de una prueba según las marcas de los atletas en la misma y cumpliendo con los criterios para los desempates del reglamento. * Si cualquier acción se realizó correctamente, se debe mostrar un mensaje indicándolo. En caso contrario, también. |

## Autenticación de usuario

|  |  |
| --- | --- |
| Como | Público |
| Quiero | Iniciar sesión |
| Para | Hacer uso de las funcionalidades restringidas de la aplicación. |
| Pruebas de aceptación | * El usuario debe proporcionar un email y contraseña y el sistema debe crear un JWT de sesión con información del rol e identidad del usuario. * Si ya existe un token de sesión válido almacenado en el localstorage, la aplicación debe redirigir al usuario directamente a la parte privada. |

En cuanto a *requisitos no funcionales*, se destaca únicamente la *Usabilidad*. La interfaz se ha diseñado para ser lo más sencilla posible, orientando este diseño a un uso en tabletas y teniendo en cuenta que cualquier usuario, familiarizado con estos dispositivos o no, debería poder interactuar sin ninguna dificultad con el sistema.

# Diseño del software

En este apartado se expone el diseño software del proyecto con un enfoque simplificado. Al haber aplicado una metodología ágil el diseño no se ha documentado de forma exhaustiva, sino que éste se ha ido madurando a lo largo del proyecto y se ilustra a continuación a través de dos perspectivas del sistema: una estática para entender la estructura y otra dinámica para comprender cómo se comporta.

## Vista estática

La estructura y organización de archivos es una parte muy importante en el desarrollo de software. Una correcta disposición de los archivos de un proyecto surge casi involuntariamente si se aplican buenos principios de desarrollo software, y es algo que favorece el desarrollo sobre todo si las aplicaciones son susceptibles de escalar y se pretende reducir el acoplamiento y aumentar la cohesión.

A continuación, se muestran el diagrama de clases y la estructura de archivos correspondientes tanto a la parte Back-End como al cliente Web, omitiéndose la del cliente móvil al tener la misma estructura que este último. En el caso del cliente Web, el diagrama de clases es substituido por un diagrama de módulos y componentes.

### Back-End

La parte servidora de este proyecto es un proyecto Spring organizado en paquetes (Ilustración 2). Principalmente cada uno de estos paquetes se corresponden con una capa de la aplicación (modelos, repositorios, servicios y controladores).

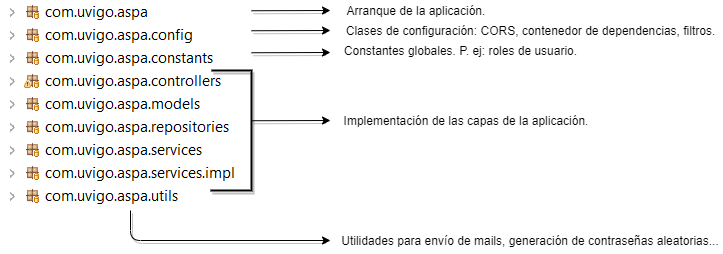


Ilustración 2. VIsta estática del Back-End

A continuación, se muestra un diagrama de clases (Ilustración 3) de esta parte del proyecto. Nos abstraemos por un momento del contexto del proyecto para mostrarlo de una forma genérica y así simplificar su entendimiento y evitar redundancias:

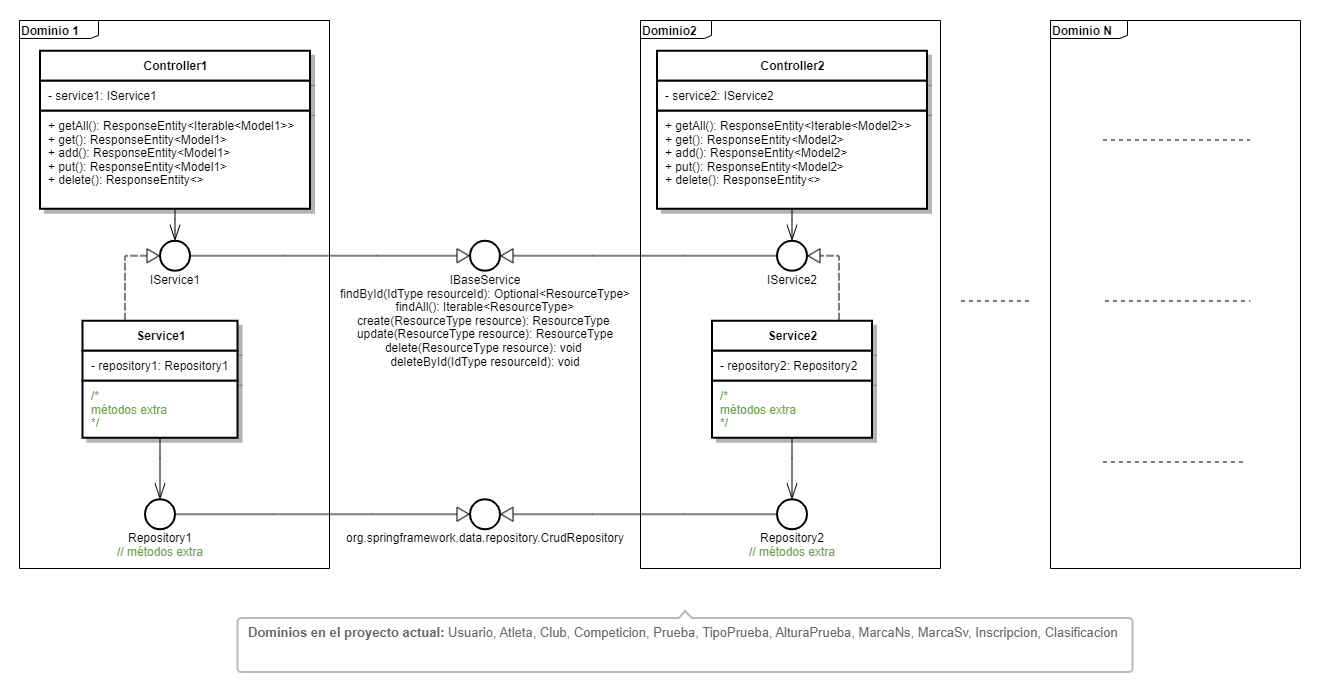


Ilustración 3. Diagrama de clases del Back-End

### Cliente Web

Como se ha explicado anteriormente en el apartado de Arquitectura, el cliente Web de este proyecto está organizado en módulos que se corresponden en su mayoría con una funcionalidad de la aplicación. La estructura de archivos es la que se puede ver en la siguiente ilustración:

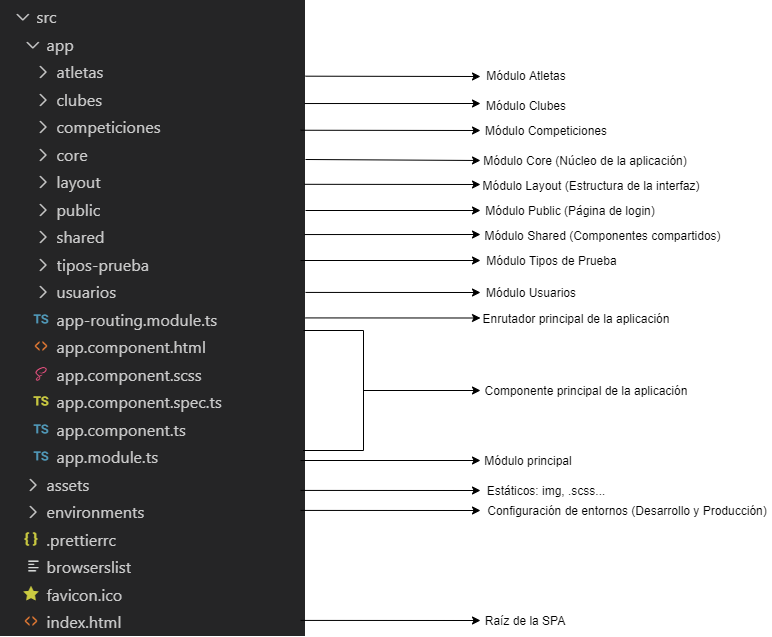


Ilustración 4. Vista estática del Front-End

A continuación, se muestra un diagrama de módulos y componentes (Ilustración 5) que refleja cómo se organiza la aplicación:

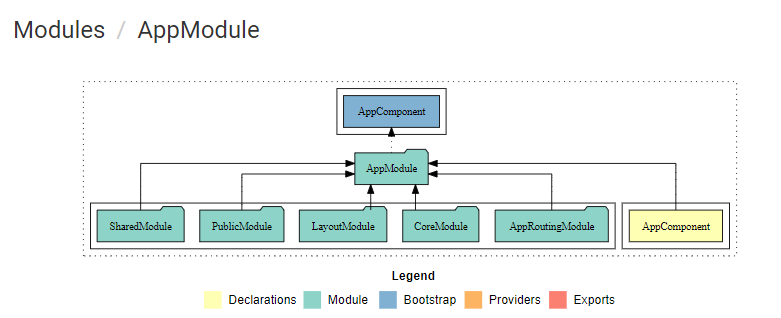


Ilustración 5. Diagrama de módulos del Front-End

Como se ha comentado anteriormente, el LayoutModule (Ilustración 6) es el que define la estructura de la interfaz y del que cuelgan los módulos correspondientes a los dominios de la aplicación:

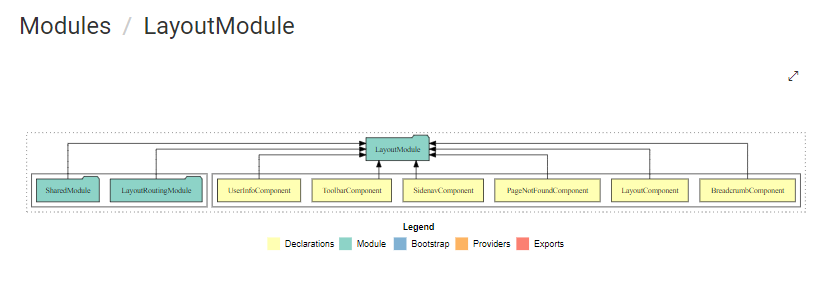


Ilustración 6. LayoutModule

Finalmente, se ilustra la estructura de un módulo de dominio sencillo, en este caso el de usuarios:

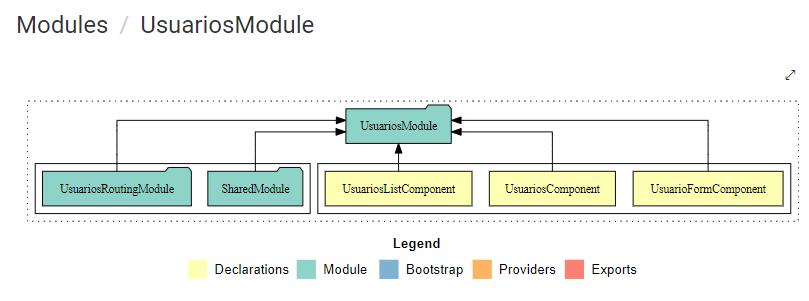


Ilustración 7. UsuariosModule

La documentación completa en html de los módulos de la aplicación se puede encontrar en el directorio **/aspa-frontend/documentation** del código fuente entregado junto con el presente documento.

### Cliente Móvil

Dado que se ha desarrollado utilizando las mismas tecnologías que el cliente web y, por ende, estableciendo la misma arquitectura y organización de ficheros, se omite por redundante la documentación en este apartado para este artefacto.

La documentación completa en html de los módulos de esta aplicación se puede encontrar en el directorio **/aspa-mobile/documentation** del código fuente entregado junto con el presente documento.

## Vista dinámica

Con la intención de evitar la creación de un diagrama de secuencia para cada una de las operaciones que implementa el sistema, se ha optado por una única representación que ilustre de un modo genérico cómo se realizan estas operaciones.

Se utiliza como ejemplo una acción que es iniciada por la interacción del usuario con la interfaz del cliente web del sistema. La aplicación web se comunica con el API para solicitarle la ejecución de alguna operación. La API responde y en función de la respuesta el cliente web actúa de un modo u otro, siempre informando al usuario.

A continuación, se muestra un diagrama (Ilustración 8) que representa el flujo en el cliente y en el servidor y que conjuntamente ilustran el ejemplo genérico de acción descrito con anterioridad:

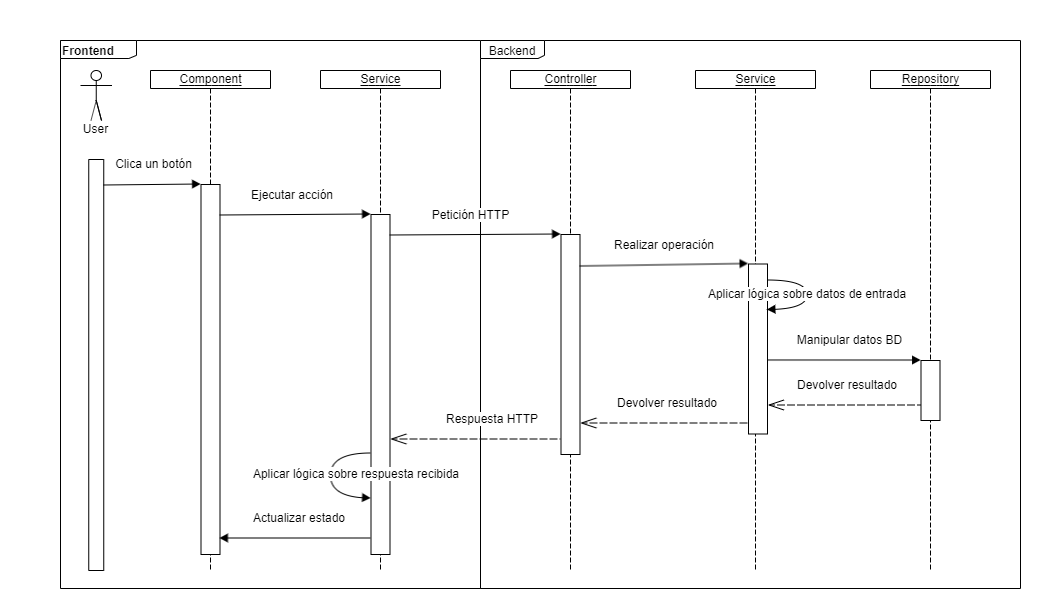


Ilustración 8. Diagrama de secuencia

# Gestión de datos e información

En este proyecto se ha utilizado una base de datos relacional MySQL para la gestión de los datos de la aplicación. En la capa de persistencia de datos de la API implementada en este proyecto se ha usado Hibernate para implementar la API de persistencia de Java (JPA), que se integra rápidamente y a la perfección con bases de datos MySQL.

El origen de los datos manipulados en este proyecto es totalmente ficticio, aunque el modelo de datos (Ilustración 9) es posible que se acerque bastante a la realidad dado el conocimiento del desarrollador de este proyecto acerca del contexto de la información.

## 9.1 Modelo de datos

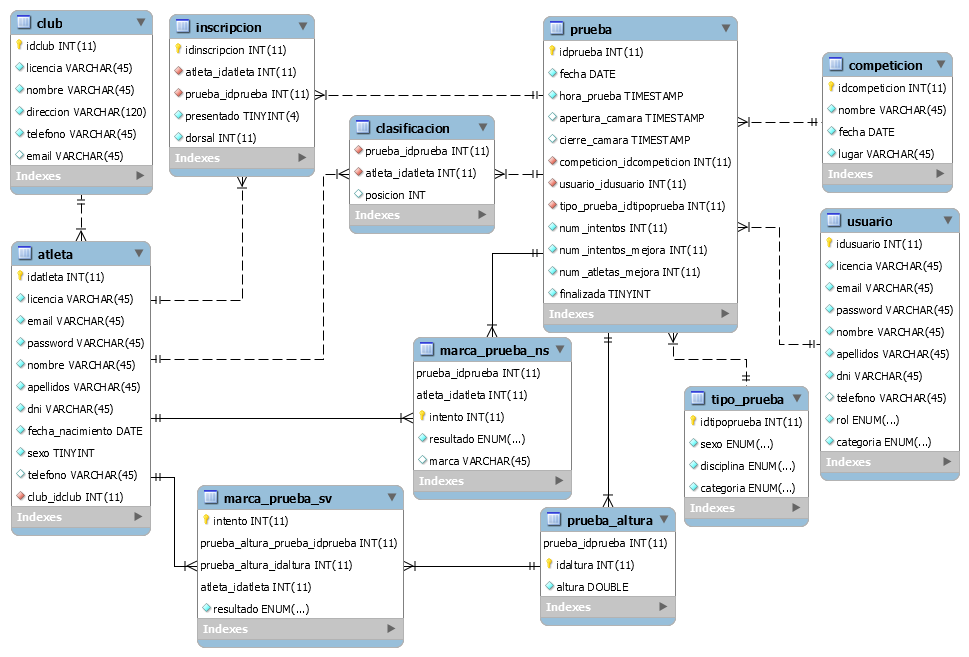


Ilustración 9. Modelo de datos

## 9.2 URIs de los recursos

En este apartado se muestra una tabla informativa con las URI de los recursos REST, compuesta por una breve descripción y los métodos HTTP aceptados por cada URL según el mecanismo de autenticación (AUTH) y autorización (ROL) implementado en la API.

### Usuarios

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| GET | /api/usuarios | Obtener colección de usuarios | Sí | ADMIN |
| POST | /api/usuarios | Crear un usuario | Sí | ADMIN |
| GET | /api/usuarios/{usuarioId} | Obtener un usuario | Sí | ADMIN |
| PUT | /api/usuarios/{usuarioId} | Modificar un usuario | Sí | ADMIN |
| DELETE | /api/usuarios/{usuarioId} | Eliminar un usuario | Sí | ADMIN |

### Atletas

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| GET | /api/atletas | Obtener colección de atletas | Sí | ADMIN, JUEZ |
| POST | /api/atletas | Crear un atleta | Sí | ADMIN |
| GET | /api/atletas/{atletaId} | Obtener un atleta | Sí | ADMIN |
| PUT | /api/atletas/{atletaId} | Modificar un atleta | Sí | ADMIN |
| DELETE | /api/atletas/{atletaId} | Eliminar un atleta | Sí | ADMIN |

### Clubes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| GET | /api/clubes | Obtener colección de clubes | Sí | ADMIN |
| POST | /api/clubes | Crear un club | Sí | ADMIN |
| PUT | /api/clubes/{clubId} | Modificar un club | Sí | ADMIN |
| DELETE | /api/clubes/{clubId} | Eliminar un club | Sí | ADMIN |

### Tipos de prueba

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| GET | /api/tipos-prueba | Obtener colección de tipos de prueba | No | N/A |
| POST | /api/ tipos-prueba | Crear un tipo de prueba | Sí | ADMIN |
| DELETE | /api/tipos-prueba/{tipoPruebaId} | Eliminar un tipo de prueba | Sí | ADMIN |

### Competiciones

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| GET | /api/competiciones | Obtener colección de competiciones | No | N/A |
| GET | /api/competiciones/hoy | Obtener competiciones del día actual | No | N/A |
| POST | /api/ competiciones | Crear una competición | Sí | ADMIN |
| GET | /api/competiciones/{competicionId} | Obtener una competición | No | N/A |
| PUT | /api/competiciones/{competicionId} | Modificar una competición | Sí | ADMIN |
| DELETE | /api/competiciones/{competicionId} | Eliminar una competición | Sí | ADMIN |

### Pruebas

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| GET | /api/competiciones/{competicionId}/pruebas | Obtener pruebas de competición | No | N/A |
| POST | /api/competiciones/{competicionId}/pruebas | Crear una prueba | Sí | ADMIN |
| GET | /api/competiciones/{competicionId}/pruebas/{pruebaId} | Obtener una prueba | No | N/A |
| PUT | /api/competiciones/{competicionId}/pruebas/{pruebaId} | Modificar una prueba | Sí | ADMIN |
| DELETE | /api/competiciones/{competicionId}/pruebas/{pruebaId} | Eliminar una prueba | Sí | ADMIN |

### Inscripciones

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| GET | /api/competiciones/{competicionId}/pruebas/{pruebaId}/inscripciones | Obtener inscripciones de prueba | No | N/A |
| POST | /api/competiciones/{competicionId}/pruebas/{pruebaId}/inscripciones | Crear una inscripción | Sí | ADMIN, ATLETA |
| GET | /api/competiciones/{competicionId}/pruebas/{pruebaId}/inscripciones/confirmadas | Obtener inscripciones confirmadas prueba | No | N/A |
| PUT | /api/competiciones/{competicionId}/pruebas/{pruebaId}/inscripciones/{inscripcionId} | Modificar una inscripción | Sí | ADMIN, JUEZ |
| DELETE | /api/competiciones/{competicionId}/pruebas/{pruebaId}/inscripciones/{inscripcionId} | Eliminar una inscripción | Sí | ADMIN, ATLETA |

### Alturas (en las pruebas de tipo “salto vertical”: pértiga y altura)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| GET | /api/competiciones/{competicionId}/pruebas/{pruebaId}/alturas | Obtener alturas de prueba de tipo sv | No | N/A |
| POST | /api/competiciones/{competicionId}/pruebas/{pruebaId}/alturas | Crear una o más alturas | Sí | ADMIN, JUEZ |
| PUT | /api/competiciones/{competicionId}/pruebas/{pruebaId}/alturas | Modificar una altura | Sí | ADMIN, JUEZ |
| DELETE | /api/competiciones/{competicionId}/pruebas/{pruebaId}/alturas | Eliminar una altura | Sí | ADMIN, JUEZ |

### Marcas Sv (marca en las pruebas de tipo “salto vertical”: pértiga y altura)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| GET | /api/pruebas/{pruebaId}/marcas/sv | Obtener marcas de prueba de tipo sv | No | N/A |
| POST | /api/pruebas/{pruebaId}/marcas/sv | Crear una marca | Sí | ADMIN, JUEZ |
| PUT | /api/pruebas/{pruebaId}/marcas/sv | Modificar una marca | Sí | ADMIN, JUEZ |
| DELETE | /api/pruebas/{pruebaId}/marcas/sv | Eliminar una marca | Sí | ADMIN, JUEZ |

### Marcas Ns (marca para el resto de tipos de prueba)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| GET | /api/pruebas/{pruebaId}/marcas/ns | Obtener marcas de prueba de tipo ns | No | N/A |
| POST | /api/pruebas/{pruebaId}/marcas/ns | Crear una marca | Sí | ADMIN, JUEZ |
| PUT | /api/pruebas/{pruebaId}/marcas/ns | Modificar una marca | Sí | ADMIN, JUEZ |
| DELETE | /api/pruebas/{pruebaId}/marcas/ns | Eliminar una marca | Sí | ADMIN, JUEZ |

### Clasificaciones

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| GET | /api/competiciones/{competicionId}/clasificaciones | Obtener clasificaciones de todas las pruebas de una competición | No | N/A |
| GET | /api/competiciones/{competicionId}/pruebas/{pruebaId}/clasificaciones | Obtener clasificaciones de una prueba | No | N/A |
| POST | /api/competiciones/{competicionId}/pruebas/{pruebaId}/clasificaciones | Crear una o más clasificaciones | Sí | ADMIN, JUEZ |
| PUT | /api/competiciones/{competicionId}/pruebas/{pruebaId}/clasificaciones | Modificar una o más clasificaciones | Sí | ADMIN, JUEZ |
| DELETE | /api/competiciones/{competicionId}/pruebas/{pruebaId}/clasificaciones | Eliminar una clasificación | Sí | ADMIN, JUEZ |

### Endpoints relacionados con la autenticación

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÉTODO | URL | Descripción | AUTH | ROL |
| POST | /api/token/usuarios | Obtener un token de sesión para usuarios con rol ADMIN o JUEZ | No | N/A |
| POST | /api/token/atletas | Obtener un token de sesión para usuarios con rol ATLETA | No | N/A |
| POST | /api/password/usuarios | Generar una nueva contraseña para usuarios con rol ADMIN o JUEZ | No | N/A |
| POST | /api/password/atletas | Generar una nueva contraseña para usuarios con rol ATLETA | No | N/A |

# Pruebas

Con el fin de garantizar el buen funcionamiento del sistema, a medida que éste se fue desarrollando se realizaron un gran número de pruebas funcionales. Se han tenido en cuenta los criterios de aceptación previamente definidos en cada una de las historias de usuario y comprobando si éstas efectivamente se cumplían de manera exitosa. Este tipo de prueba se centra únicamente en las salidas generadas en respuesta a las entradas seleccionadas y a las condiciones de ejecución.

## Gestión de usuarios

|  |  |
| --- | --- |
| **Funcionalidad** | **Consultar usuarios** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Usuarios” de la aplicación. |
| Ejecución | - |
| Resultado | Debe mostrar el listado de usuarios (jueces) registrados en el sistema. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Crear usuario** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Usuarios” de la aplicación y ha pulsado el botón “Nuevo usuario”. |
| Ejecución | El usuario cubre el formulario y pulsa el botón “Crear”. |
| Resultado | Si los datos son correctos, el usuario creado debe guardarse en la base de datos y aparecer reflejado en la interfaz. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Modificar usuario** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Usuarios” de la aplicación y ha pulsado el botón “Modificar usuario”. |
| Ejecución | El usuario cubre el formulario y pulsa el botón “Modificar”. |
| Resultado | Si los datos son correctos, el usuario modificado debe guardarse en la base de datos y aparecer reflejado en la interfaz. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Eliminar usuario** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Usuarios” de la aplicación. |
| Ejecución | El usuario pulsa el botón “Eliminar” de un usuario y pulsa “Sí” en el diálogo de confirmación. |
| Resultado | El usuario debe eliminarse de la base de datos y esto debe reflejarse en la interfaz. |
| Evaluación | Superada correctamente. |

## Gestión de atletas

|  |  |
| --- | --- |
| **Funcionalidad** | **Consultar atletas** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Atletas” de la aplicación. |
| Ejecución | - |
| Resultado | Debe mostrar el listado de atletas registrados en el sistema. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Crear atleta** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Atletas” de la aplicación y ha pulsado el botón “Nuevo atleta”. |
| Ejecución | El usuario cubre el formulario y pulsa el botón “Crear”. |
| Resultado | Si los datos son correctos, el atleta creado debe guardarse en la base de datos y aparecer reflejado en la interfaz. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Modificar atleta** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Atletas” de la aplicación y ha pulsado el botón “Modificar atleta”. |
| Ejecución | El usuario cubre el formulario y pulsa el botón “Modificar”. |
| Resultado | Si los datos son correctos, el atleta modificado debe guardarse en la base de datos y aparecer reflejado en la interfaz. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Eliminar atleta** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Atletas” de la aplicación. |
| Ejecución | El usuario pulsa el botón “Eliminar” de un atleta y pulsa “Sí” en el diálogo de confirmación. |
| Resultado | El atleta debe eliminarse de la base de datos y esto debe reflejarse en la interfaz. |
| Evaluación | Superada correctamente. |

## Gestión de clubes

|  |  |
| --- | --- |
| **Funcionalidad** | **Consultar clubes** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Clubes” de la aplicación. |
| Ejecución | - |
| Resultado | Debe mostrar el listado de clubes registrados en el sistema. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Crear club** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Clubes” de la aplicación y ha pulsado el botón “Nuevo club”. |
| Ejecución | El usuario cubre el formulario y pulsa el botón “Crear”. |
| Resultado | Si los datos son correctos, el club creado debe guardarse en la base de datos y aparecer reflejado en la interfaz. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Modificar club** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Clubes” de la aplicación y ha pulsado el botón “Modificar club”. |
| Ejecución | El usuario cubre el formulario y pulsa el botón “Modificar”. |
| Resultado | Si los datos son correctos, el club modificado debe guardarse en la base de datos y aparecer reflejado en la interfaz. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Eliminar club** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Clubes” de la aplicación. |
| Ejecución | El usuario pulsa el botón “Eliminar” de un club y pulsa “Sí” en el diálogo de confirmación. |
| Resultado | El club debe eliminarse de la base de datos y esto debe reflejarse en la interfaz. |
| Evaluación | Superada correctamente. |

## Gestión de tipos de prueba

|  |  |
| --- | --- |
| **Funcionalidad** | **Consultar tipos de prueba** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Tipos de prueba” de la aplicación. |
| Ejecución | - |
| Resultado | Debe mostrar el listado de tipos de prueba registrados en el sistema. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Crear tipo de prueba** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Tipos de prueba” de la aplicación y ha pulsado el botón “Nuevo tipo de prueba”. |
| Ejecución | El usuario cubre el formulario y pulsa el botón “Crear”. |
| Resultado | Si los datos son correctos, el tipo de prueba creado debe guardarse en la base de datos y aparecer reflejado en la interfaz. |
| Evaluación | Superada correctamente. |

|  |  |
| --- | --- |
| **Funcionalidad** | **Eliminar tipo de prueba** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Tipos de prueba” de la aplicación. |
| Ejecución | El usuario pulsa el botón “Eliminar” de un tipo de prueba y pulsa “Sí” en el diálogo de confirmación. |
| Resultado | El tipo de prueba debe eliminarse de la base de datos y esto debe reflejarse en la interfaz. |
| Evaluación | Superada correctamente. |

## Gestión de competiciones

|  |  |
| --- | --- |
| **Funcionalidad** | **Consultar competiciones** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Competiciones” de la aplicación. |
| Ejecución | - |
| Resultado | Debe mostrar el listado de competiciones registradas en el sistema. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Crear competición** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Competiciones” de la aplicación y ha pulsado el botón “Nueva competición”. |
| Ejecución | El usuario cubre el formulario y pulsa el botón “Crear”. |
| Resultado | Si los datos son correctos, la competición creada debe guardarse en la base de datos y aparecer reflejado en la interfaz. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Modificar competición** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Competición” de la aplicación y ha pulsado el botón “Modificar competición”. |
| Ejecución | El usuario cubre el formulario y pulsa el botón “Modificar”. |
| Resultado | Si los datos son correctos, la competición modificada debe guardarse en la base de datos y aparecer reflejado en la interfaz. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Eliminar competición** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Competiciones” de la aplicación. |
| Ejecución | El usuario pulsa el botón “Eliminar” de una competición y pulsa “Sí” en el diálogo de confirmación. |
| Resultado | La competición debe eliminarse de la base de datos y esto debe reflejarse en la interfaz. |
| Evaluación | Superada correctamente. |

## Gestión de pruebas

|  |  |
| --- | --- |
| **Funcionalidad** | **Consultar pruebas** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Competiciones” y ha pulsado el botón “Pruebas” de una competición. |
| Ejecución | - |
| Resultado | Debe mostrar el listado de pruebas de la competición correspondiente. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Crear prueba** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Competiciones”, ha consultado las pruebas correspondientes a una competición y ha pulsado el botón “Nueva prueba”. |
| Ejecución | El usuario cubre el formulario y pulsa el botón “Crear”. |
| Resultado | Si los datos son correctos, la prueba creada debe guardarse en la base de datos y aparecer reflejado en la interfaz. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Modificar prueba** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Competiciones”, ha consultado las pruebas correspondientes a una competición y ha pulsado el botón “Modificar” de una prueba. |
| Ejecución | El usuario cubre el formulario y pulsa el botón “Modificar”. |
| Resultado | Si los datos son correctos, la prueba modificada debe guardarse en la base de datos y aparecer reflejado en la interfaz. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Eliminar prueba** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Competiciones” y ha pulsado el botón “Pruebas” de una competición. |
| Ejecución | El usuario pulsa el botón “Eliminar” de una prueba y pulsa “Sí” en el diálogo de confirmación. |
| Resultado | La prueba debe eliminarse de la base de datos y esto debe reflejarse en la interfaz. |
| Evaluación | Superada correctamente. |

## Gestión de inscripciones

|  |  |
| --- | --- |
| **Funcionalidad** | **Consultar inscripciones** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Competiciones”, ha consultado las pruebas correspondientes a una competición y ha pulsado el botón “Inscripciones” de una prueba. |
| Ejecución | - |
| Resultado | Debe mostrar un listado de atletas de la misma categoría que la prueba, indicándose para cada uno si está inscrito o no en ella. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Inscribir / des-inscribir atleta** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la sección “Competiciones”, ha consultado las pruebas correspondientes a una competición y ha pulsado el botón “Inscripciones” de una prueba. |
| Ejecución | El usuario pulsa el checkbox para inscribir/desinscribir a un atleta. |
| Resultado | La inscripción se crea/elimina en base de datos y se refleja en la interfaz. |
| Evaluación | Superada correctamente. |

## Gestión de alturas

|  |  |
| --- | --- |
| **Funcionalidad** | **Crear altura** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la pantalla donde se cubre una hoja de campo de una prueba de tipo “salto vertical” y ha pulsado el botón “Añadir alturas”. |
| Ejecución | El usuario introduce las alturas a crear y pulsa el botón “Añadir”. |
| Resultado | Las alturas se han creado en la base de datos y se reflejan en la interfaz como nuevas columnas a cubrir en la hoja de campo. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Modificar altura** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la pantalla donde se cubre una hoja de campo de una prueba de tipo “salto vertical” y ha pulsado sobre la cabecera de una columna correspondiente a una altura. |
| Ejecución | El usuario modifica el valor de la altura y pulsa el botón “Guardar”. |
| Resultado | La altura se ha modificado en la base de datos y se refleja el cambio en la interfaz. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Eliminar altura** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la pantalla donde se cubre una hoja de campo de una prueba de tipo “salto vertical” y ha pulsado sobre la cabecera de una columna correspondiente a una altura. |
| Ejecución | El usuario pulsa sobre el icono “Eliminar” y a continuación pulsa “Sí” en el diálogo de confirmación. |
| Resultado | La altura se ha eliminado en la base de datos y se refleja en la interfaz, desapareciendo la columna de la hoja. |
| Evaluación | Superada correctamente. |

## Gestión de marcas

|  |  |
| --- | --- |
| **Funcionalidad** | **Crear marca Sv/Ns** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la pantalla donde se cubre una hoja de campo de una prueba y ha pulsado sobre una celda correspondiente a un intento no cubierto de un atleta. |
| Ejecución | El usuario cubre el formulario de marca y pulsa el botón “Guardar”. |
| Resultado | La marca se almacena en la base de datos y se refleja en la interfaz, actualizándose los puestos, descalificaciones y mejores marcas de los atletas. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Modificar marca Sv/Ns** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la pantalla donde se cubre una hoja de campo de una prueba y ha pulsado sobre una celda correspondiente a un intento cubierto de un atleta. |
| Ejecución | El usuario modifica el formulario de marca y pulsa el botón “Guardar”. |
| Resultado | La marca se modifica en la base de datos y se refleja en la interfaz, actualizándose los puestos, descalificaciones y mejores marcas de los atletas. |
| Evaluación | Superada correctamente. |

|  |  |
| --- | --- |
| **Funcionalidad** | **Eliminar marca Sv/Ns** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la pantalla donde se cubre una hoja de campo de una prueba y ha mantenido pulsado (o ha hecho click derecho) sobre una celda correspondiente a un intento cubierto de un atleta. |
| Ejecución | El usuario pulsa “Sí” en el diálogo de confirmación de eliminación. |
| Resultado | La marca se elimina en la base de datos y se refleja en la interfaz, actualizándose los puestos, descalificaciones y mejores marcas de los atletas. |
| Evaluación | Superada correctamente. |

## Gestión de clasificaciones

|  |  |
| --- | --- |
| **Funcionalidad** | **Generar clasificación** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la pantalla donde se cubre una hoja de campo de una prueba y ha pulsado el botón “Finalizar prueba”. |
| Ejecución | El usuario pulsa “Sí” en el diálogo de confirmación. |
| Resultado | La prueba pasa a tener estado “Finalizada” en base de datos y se genera la clasificación correspondiente a la prueba. |
| Evaluación | Superada correctamente. |
| **Funcionalidad** | **Consultar clasificación** |
| Precondiciones | El usuario (autenticado y autorizado) ha accedido a la pantalla de visualización de la clasificación de una prueba de una competición. |
| Ejecución | - |
| Resultado | Debe mostrar el listado de atletas inscritos en la prueba, ordenados según su puesto en la prueba y con la información de sus marcas en la misma. |
| Evaluación | Superada correctamente. |

## Autenticación de usuario

|  |  |
| --- | --- |
| **Funcionalidad** | **Iniciar sesión** |
| Precondiciones | El usuario ha introducido correctamente su email y contraseña en la página de login de la aplicación. |
| Ejecución | El usuario pulsa el botón “Entrar”. |
| Resultado | Si los datos son correctos, el usuario es dirigido a la parte privada de la aplicación. |
| Evaluación | Superada correctamente. |

# Manual de usuario

En este apartado se explican de manera detallada todos los pasos para realizar la instalación del proyecto disponible en el soporte de almacenamiento adjunto con este documento, así como un breve manual de usuario que explica cómo utilizar el sistema.

## Requisitos mínimos

A continuación, se detallan los requerimientos mínimos tanto para el desarrollo como para el uso del sistema.

### Desarrollador

A nivel hardware, se estima que para el desarrollo del sistema es suficiente con disponer de una máquina con características similares a las siguientes:

* Procesador: de 2 núcleos a 2GHz.
* Disco duro: unos 10 GB de espacio libre.
* RAM: 4GB.
* Sistema operativo: Windows 10 o similares.

A nivel de software:

* Versión 1.8 del JDK de Java
* Versión 10.15.1 + de Node y versión 6.10.2 + de npm
* Versión 5.7.25 de MySQL Community Server

### Usuario

Para usar el cliente web, el único requisito es disponer de un navegador web como Google Chrome, que fue el utilizado durante el desarrollo.

Para usar el cliente móvil, únicamente hace falta disponer de un dispositivo móvil con sistema operativo Android.

## Instalación

### Desarrollador

* Base de datos:
  + Es necesario que el servicio *mysql* esté en ejecución en la máquina de desarrollo. Por defecto MySQL utiliza el puerto 3306.
  + Crear el esquema de la base de datos ejecutando el script *aspa\_entidades.sql* adjunto con este documento.
  + (Opcional) Poblar la base de datos ejecutando el script *aspa\_datos.sql* adjunto con este documento.
* API:
  + Importar el proyecto de la API como proyecto Maven desde el IDE eclipse.
  + Crear las siguientes *run configurations* para Maven:
    - *clean install*: para instalar las dependencias del proyecto.
    - *spring-boot:run*: para ejecutar la aplicación Spring.
    - *package*: para empaquetar el proyecto en un archivo .jar ejecutable.
* Cliente Web
  + Ejecutar *npm install* desde el directorio del proyecto para instalar los paquetes necesarios indicados en el fichero *package.json* del proyecto.
  + Comandos útiles para el desarrollo:
    - *ng serve –o*: para ejecutar el proyecto en modo desarrollo. La opción –o hará que automáticamente se abra en nuestro navegador web por defecto.
    - *npm run production*: para compilar el proyecto. Generará el empaquetado de archivos html, css y js que podremos desplegar en un servidor.
* Cliente móvil
  + Ejecutar *npm install* desde el directorio del proyecto para instalar los paquetes necesarios indicados en el fichero *package.json* del proyecto.
  + Comandos útiles para el desarrollo:
    - *ionic serve:* para ejecutar la aplicación en modo desarrollo. Se abrirá automáticamente nuestro navegador web por defecto.
    - *cordova run android –device*: para ejecutar la aplicación en modo desarrollo en un dispositivo físico. Es necesario disponer de un smartphone Android conectado a la máquina dónde se está desarrollando.
    - *ionic cordova build android*: para generar el archivo .apk que se podrá instalar en un smartphone con sistema operativo Android.

### Usuario

* Cliente Web: No existe ningún proceso de instalación ya que la aplicación está alojada en un servidor externo, por lo que simplemente hay que acceder a la URL que tenga definida. En este caso: <http://aspafga.ga/>
* Cliente Móvil: El archivo .apk no está publicado en la Play Store de Android, por lo que es el desarrollador el encargado de su distribución para su instalación.

## Funcionamiento

### Cliente web

El uso por parte de un usuario del cliente web es muy sencillo. En este apartado se ilustrará el funcionamiento con imágenes capturadas en una tableta con resolución de 1024x768 píxeles.

En primer lugar, se muestra cómo se crean, modifican y eliminan pruebas de atletismo. El proceso sería similar para el caso de competiciones, usuarios, atletas, clubes y tipos de prueba.

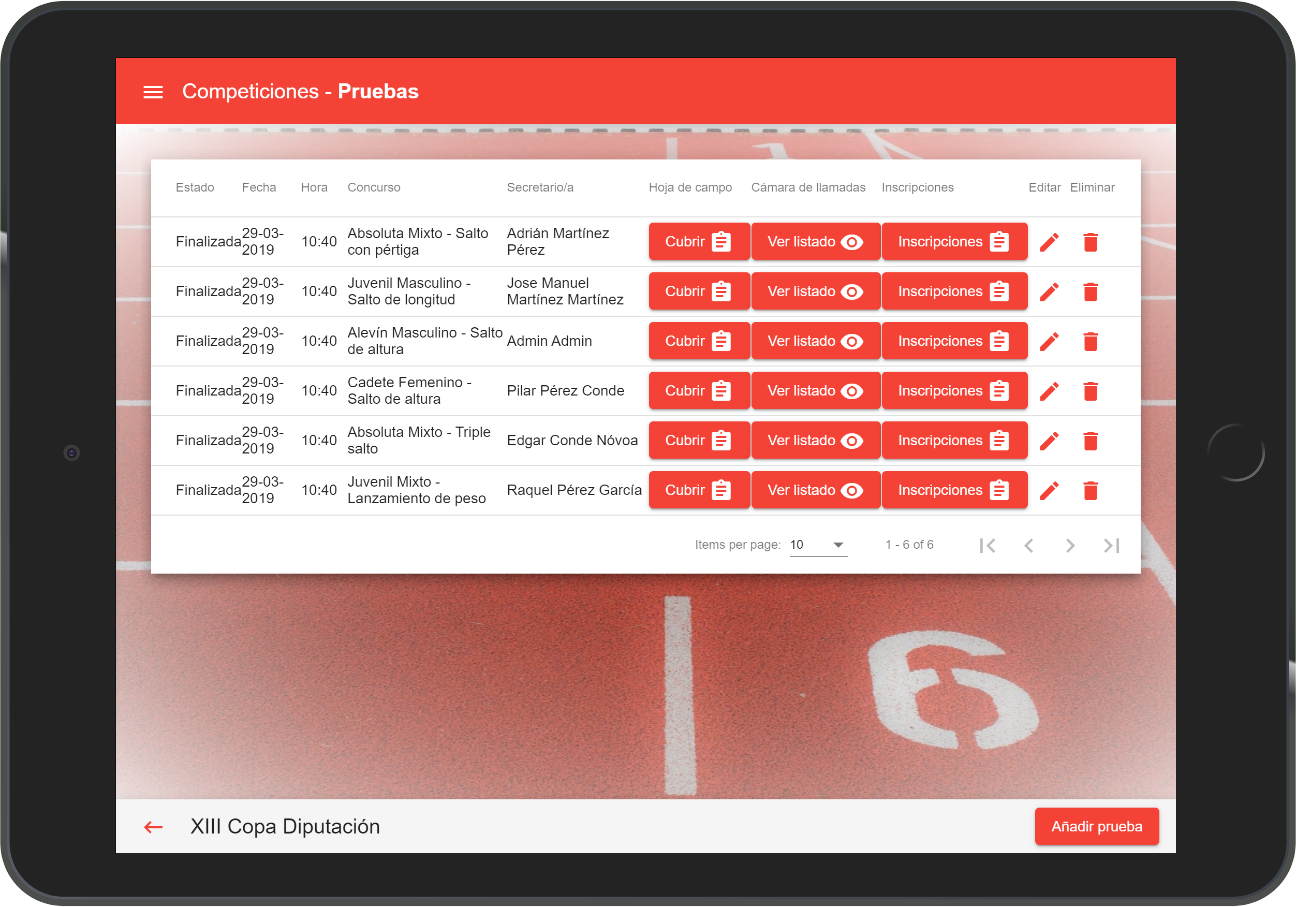


Ilustración 10. Interfaz web - Listado de pruebas

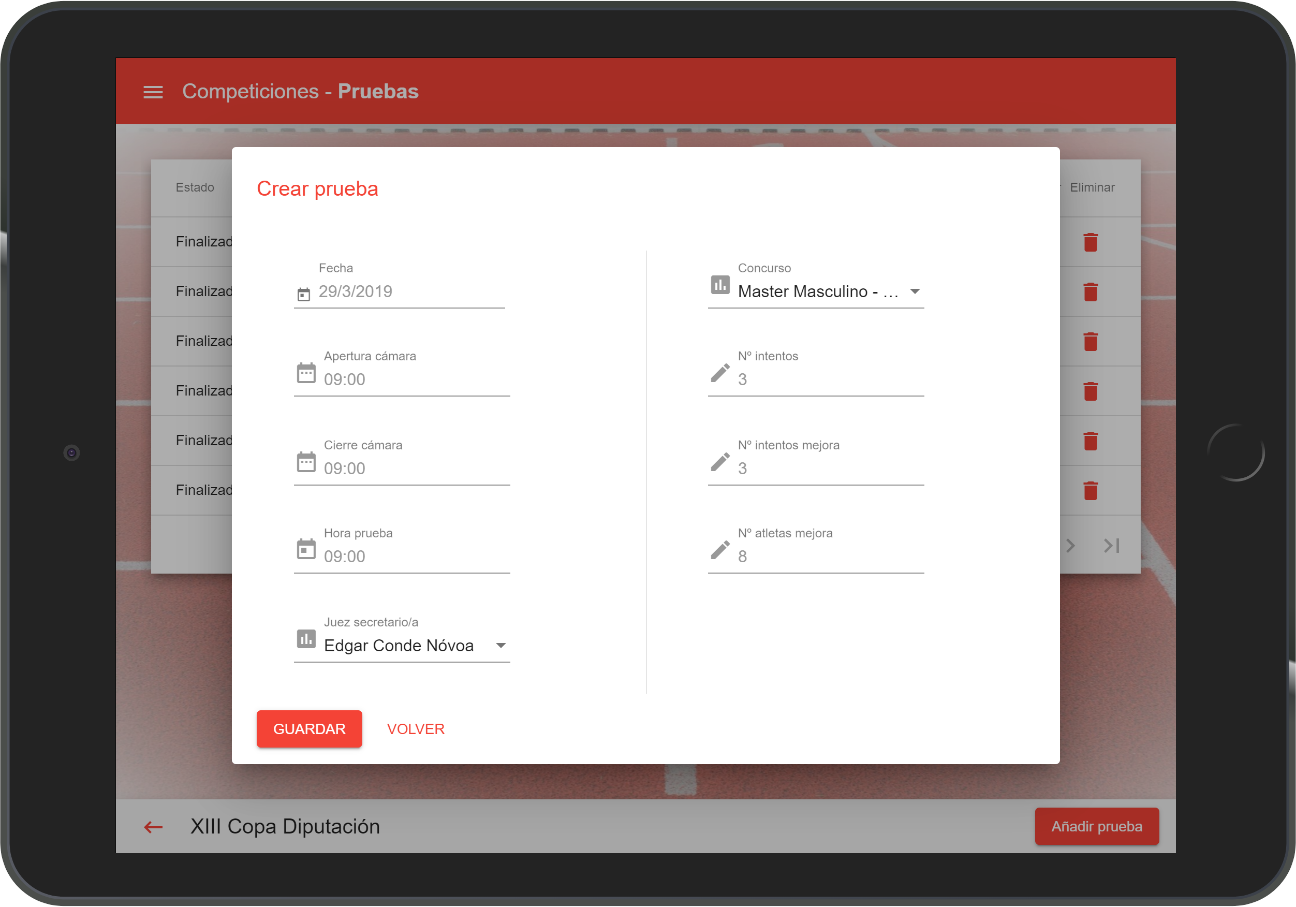


Ilustración 11. Interfaz web - Crear prueba

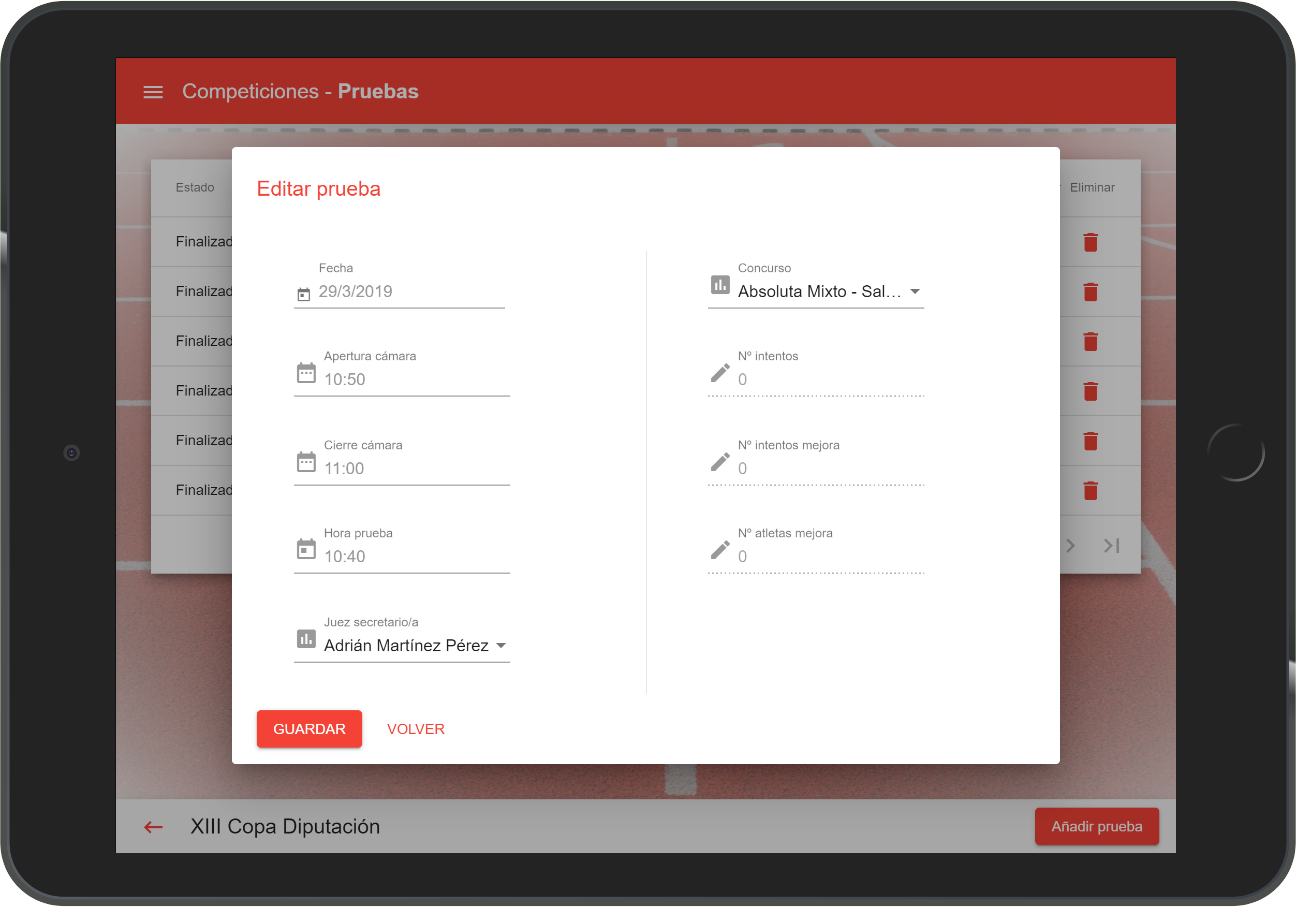


Ilustración 12. Interfaz web - Editar prueba

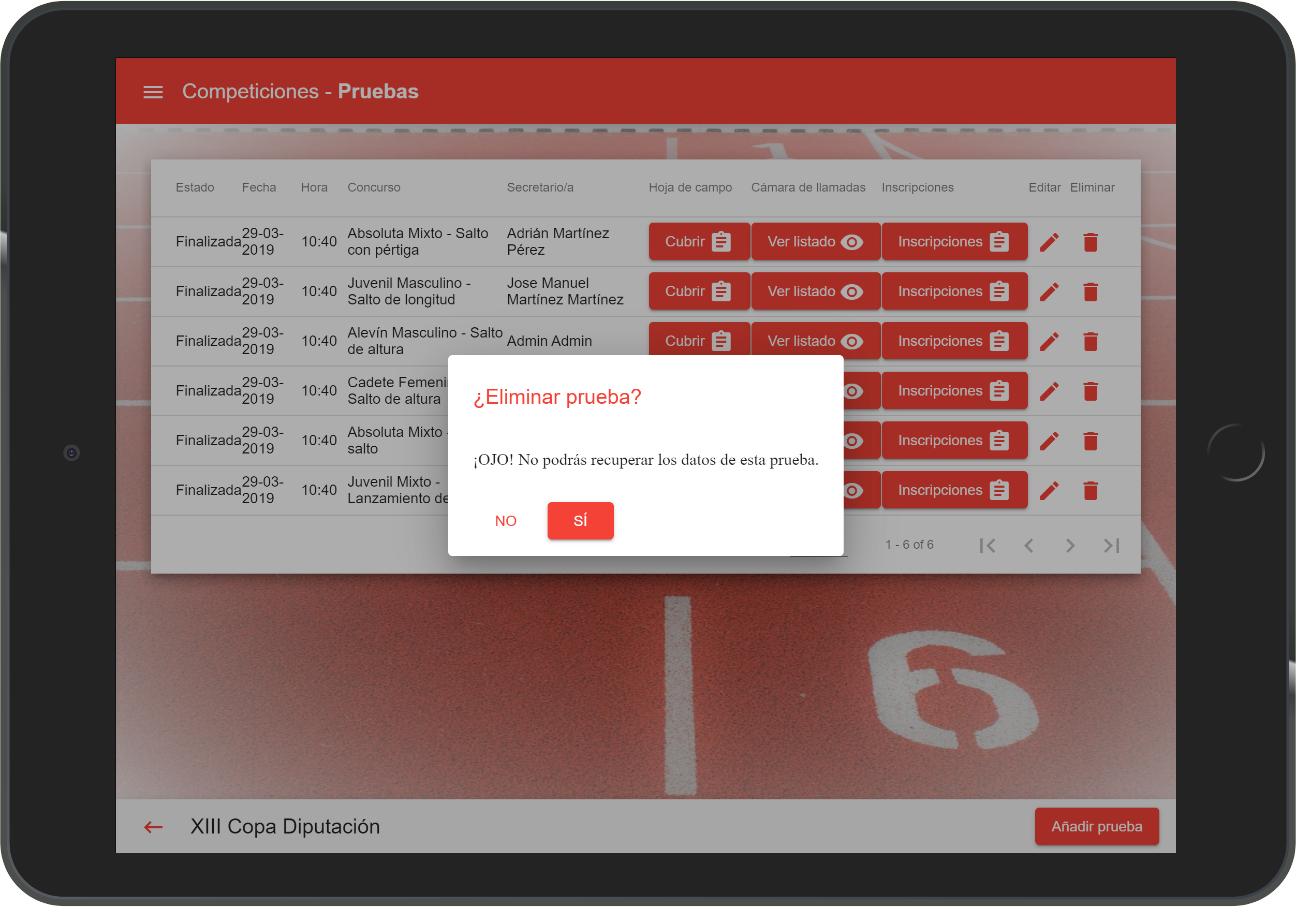


Ilustración 13. Interfaz web - Eliminar prueba

Finalmente, se muestra la pantalla donde se realiza el caso de uso más importante del proyecto, cubrir la hoja de campo de una prueba. Existen dos tipos de prueba: salto vertical (salto con pértiga y salto de altura) y el resto (salto de longitud, triple salto, lanzamiento de peso, lanzamiento de martillo y lanzamiento de jabalina). Mostramos la pantalla correspondiente a la hoja de campo de una prueba de tipo salto vertical, por ser más completa:

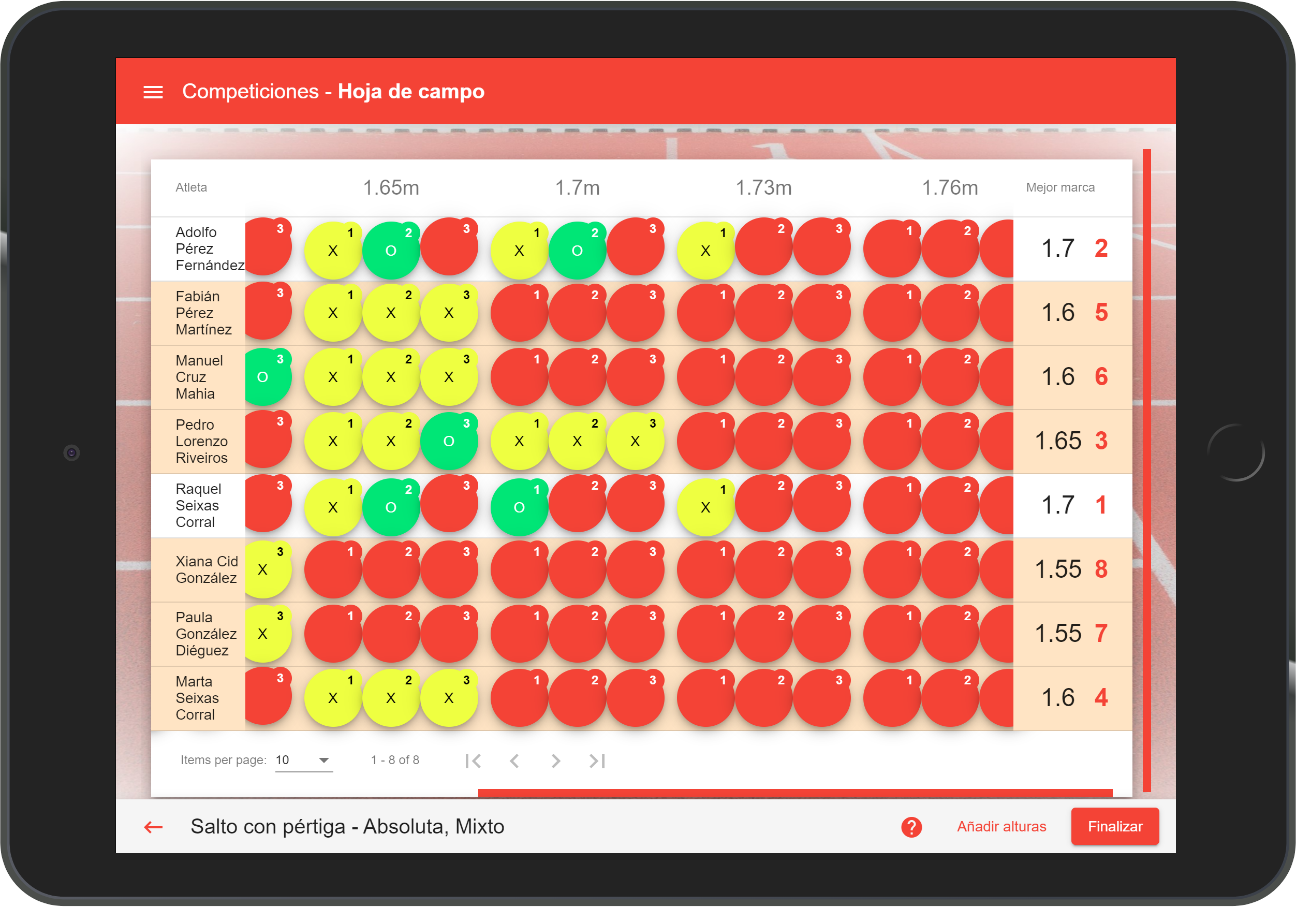


Ilustración 14. Interfaz web - Hoja de campo

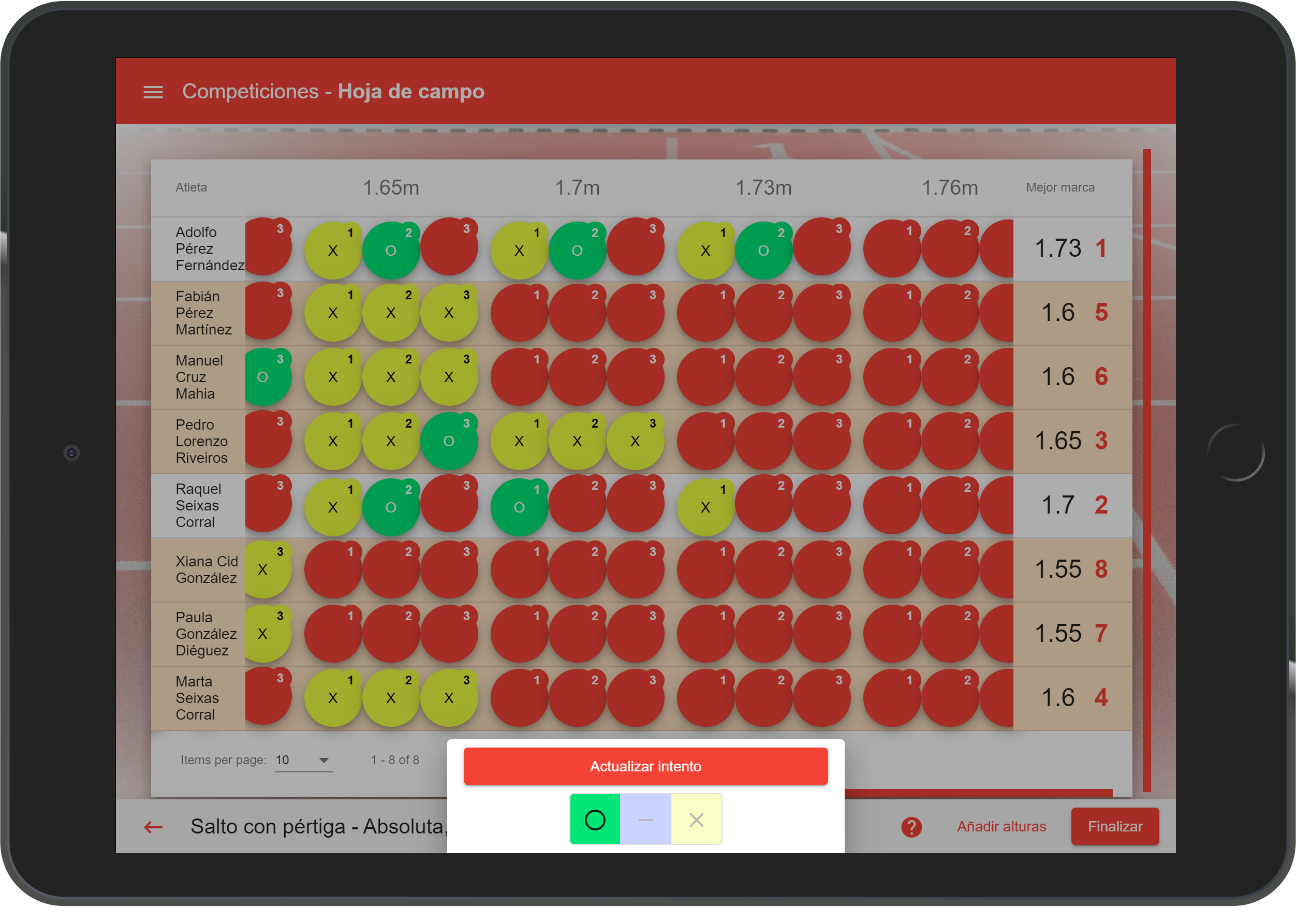


Ilustración 15. Interfaz web - Anotar intento

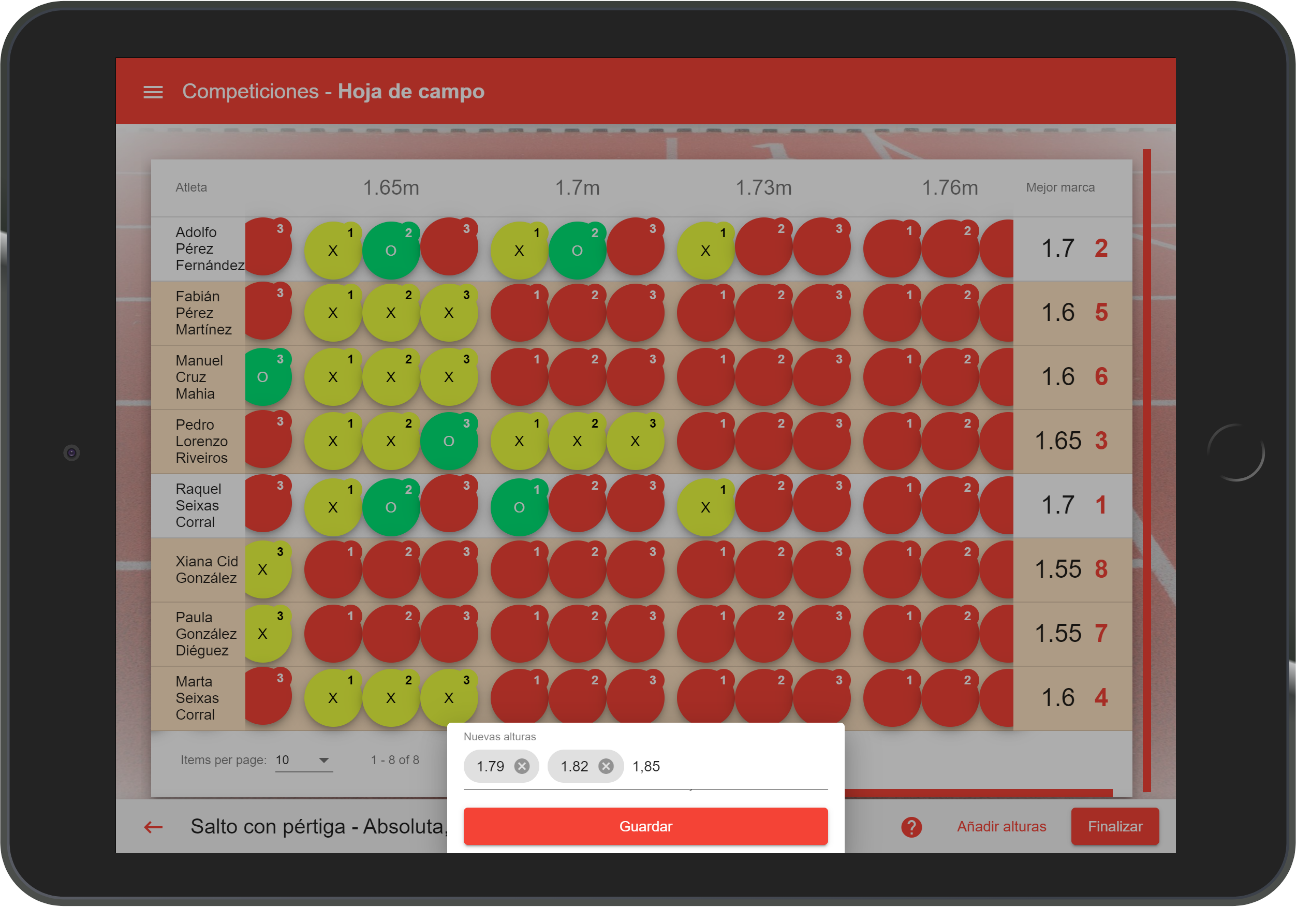


Ilustración 16. Interfaz web - Añadir alturas

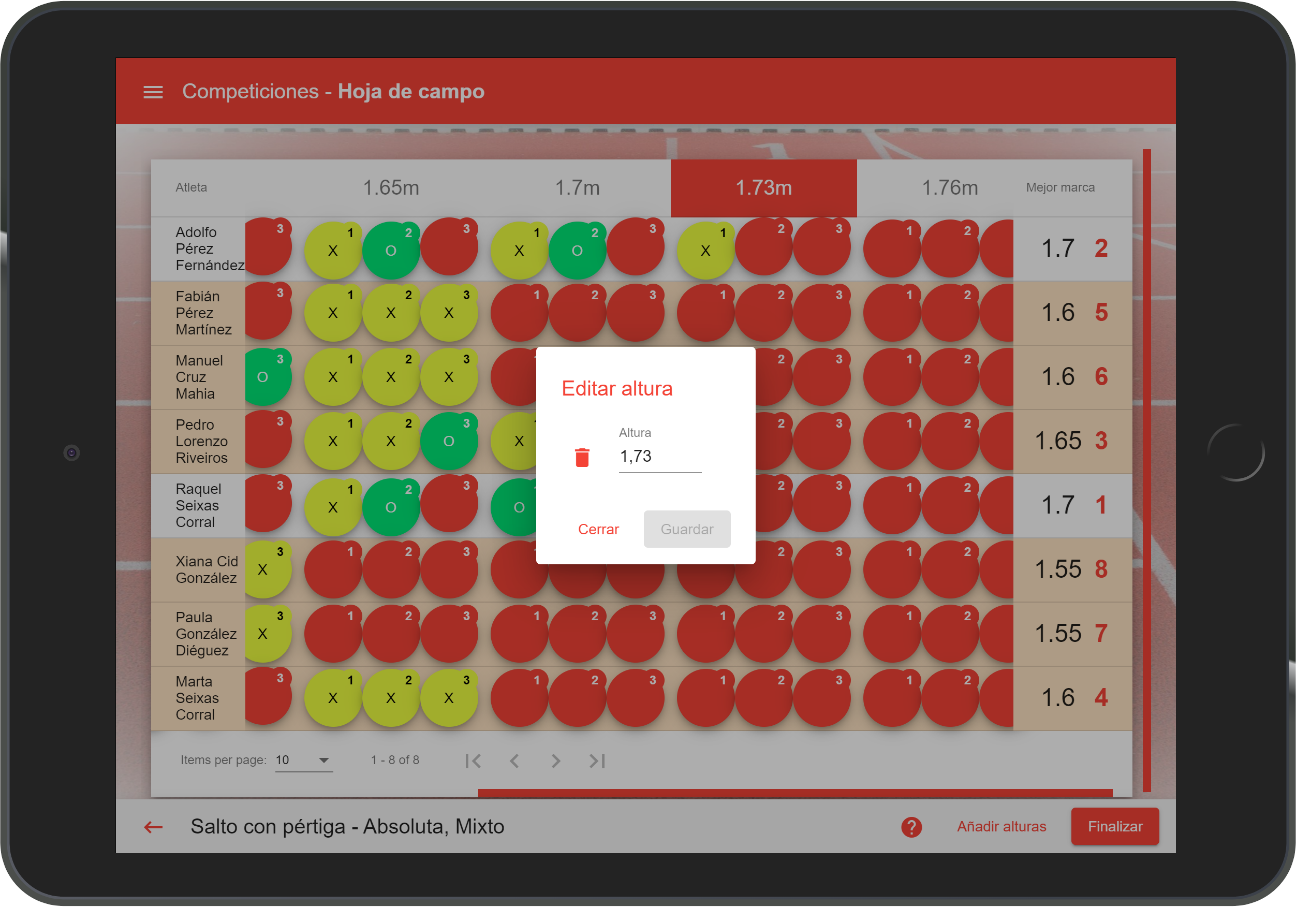


Ilustración 17. Interfaz web - Editar altura

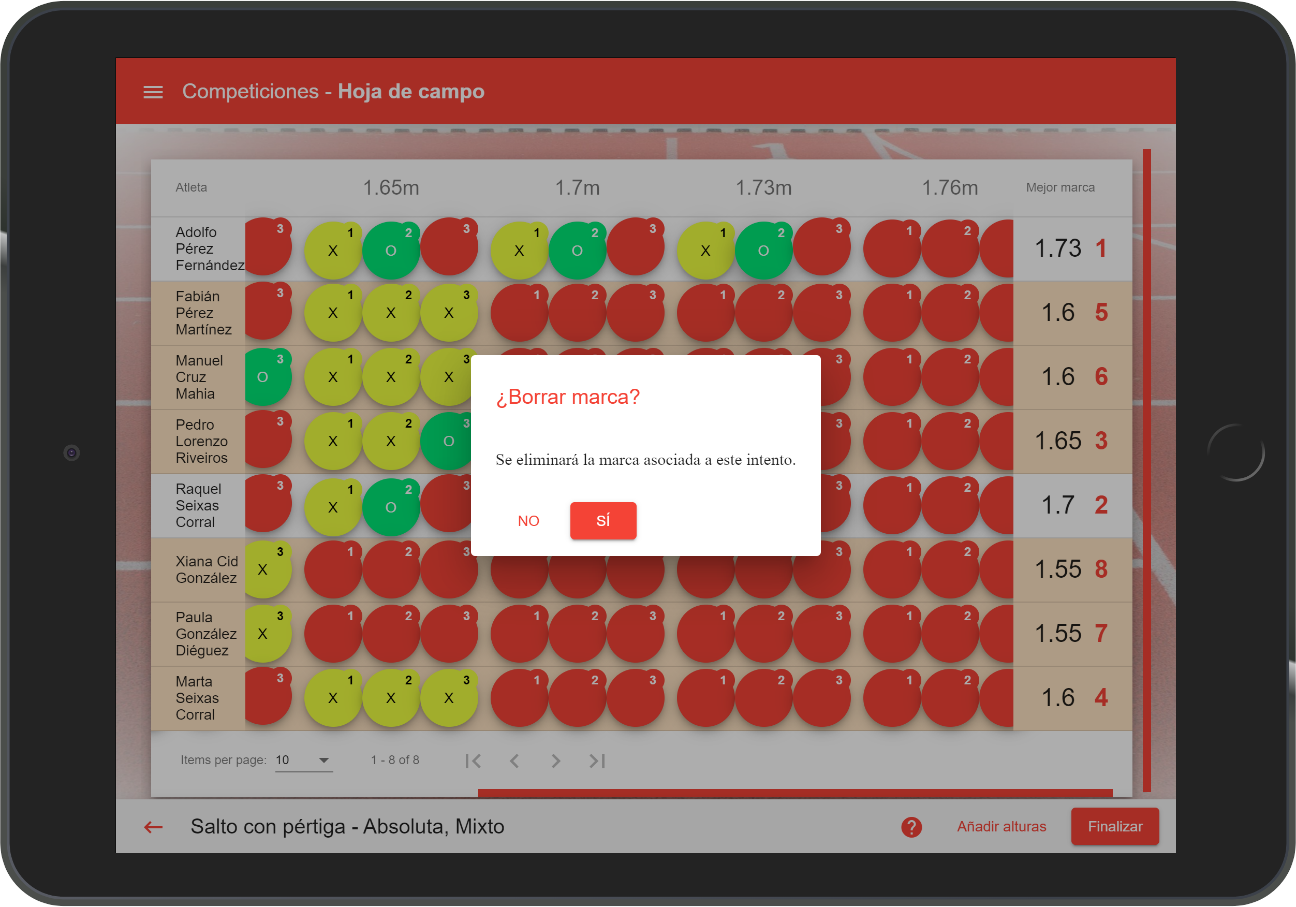


Ilustración 18. Interfaz web - Eliminar marca

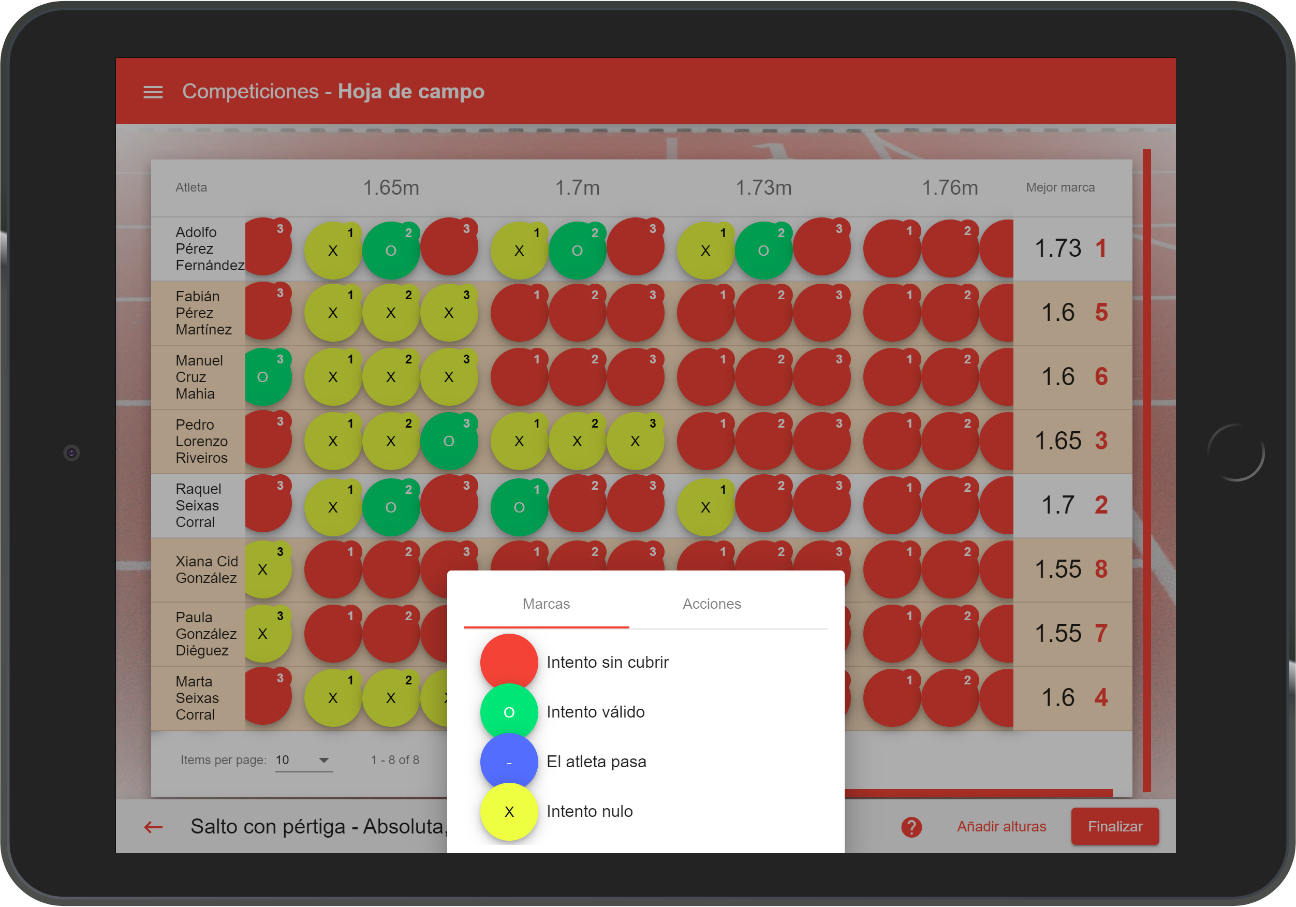


Ilustración 19. Interfaz web - Ayuda marcas

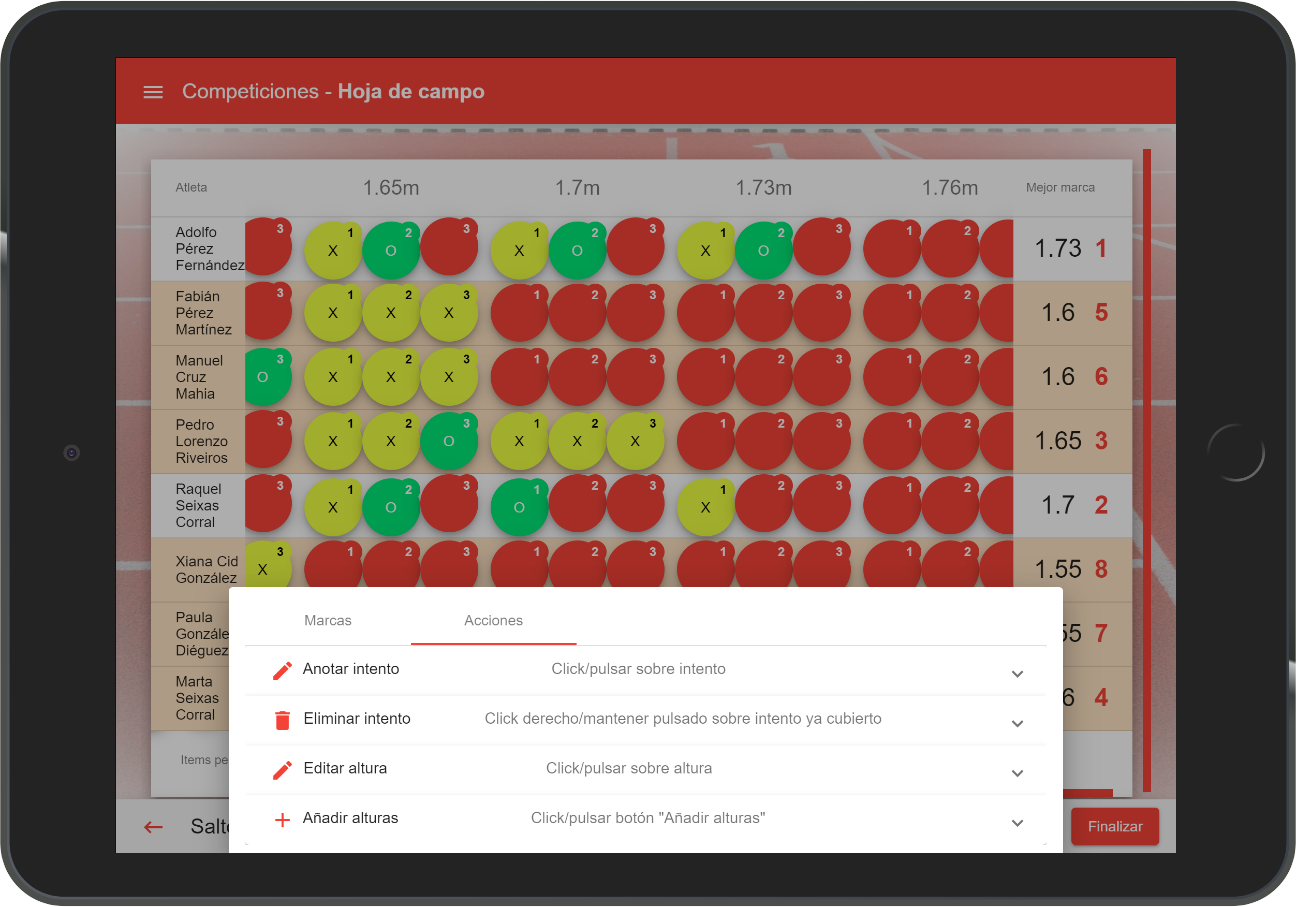


Ilustración 20. Interfaz web - Ayuda acciones

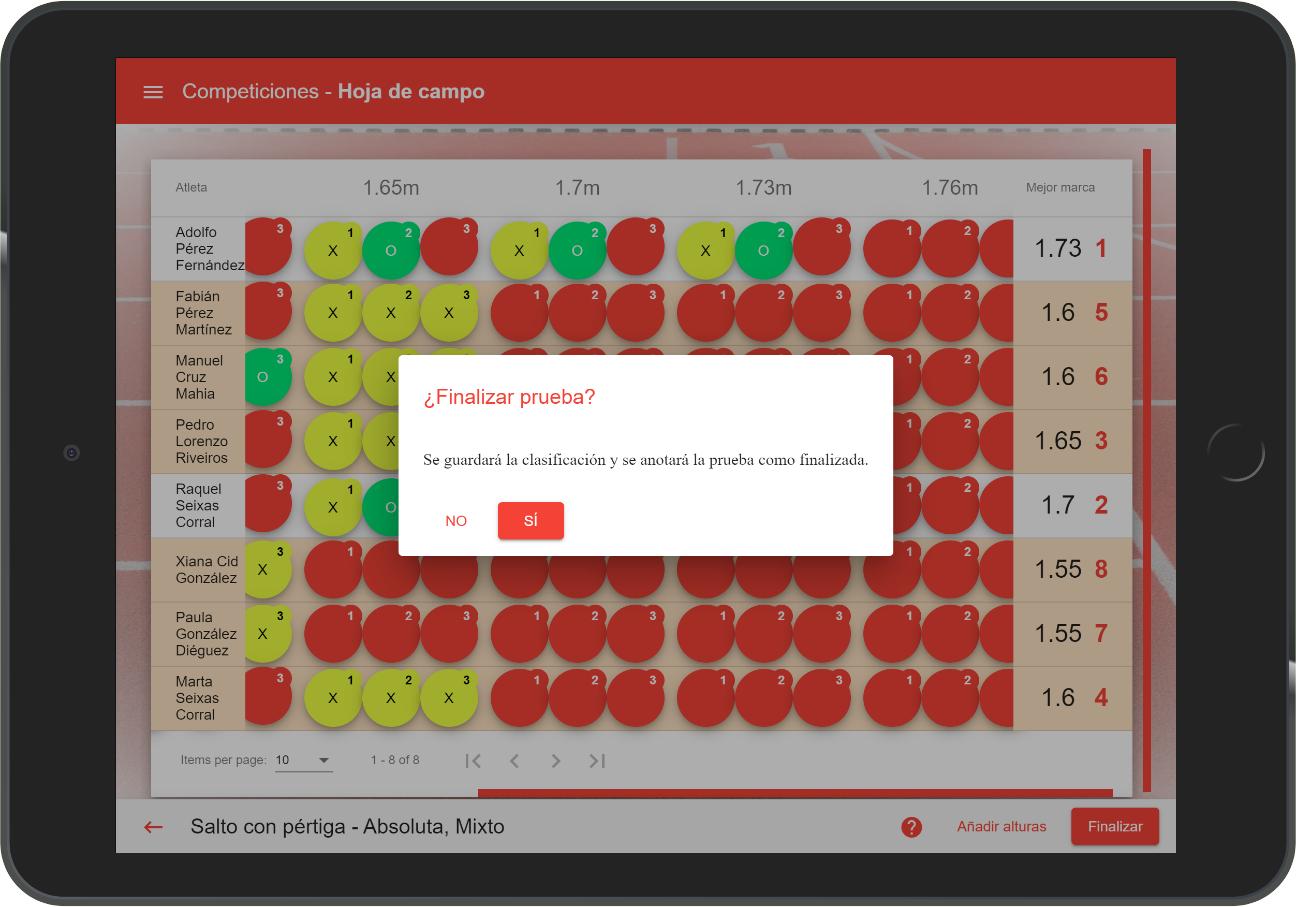


Ilustración 21. Interfaz web - Finalizar prueba

### Cliente móvil

En este apartado se ilustran las funcionalidades implementadas en la aplicación móvil mediante imágenes capturadas en un smartphone Android de 6 pulgadas.



Ilustración 22. Interfaz móvil - Inicio de sesión y listado de competiciones

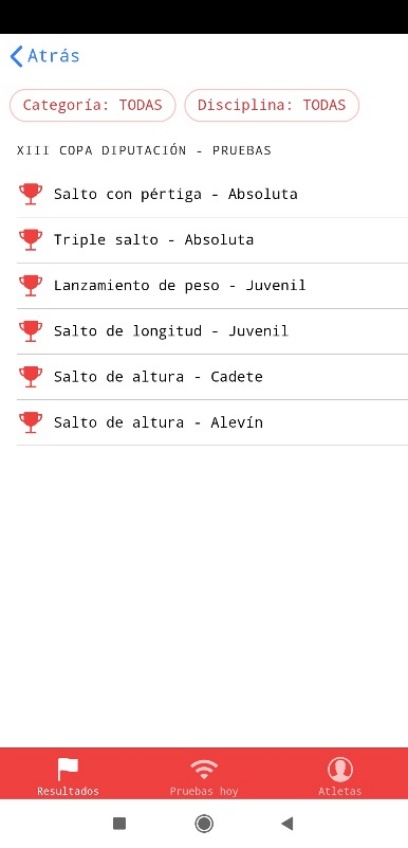


Ilustración 23. Interfaz móvil - Listado de pruebas, clasificación de prueba y marcas de atleta

Ilustración 24. Interfaz móvil - Filtro de competiciones e información de una prueba



Ilustración 25. Interfaz móvil - Inscripción en una prueba



# Principales aportaciones

Las principales aportaciones de este Trabajo de Fin de Grado se pueden resumir en las siguientes:

* Las aplicaciones web y móvil están protegidas con un sistema de autenticación mediante JWT. El cliente web se ha desarrollado con el objetivo de ser utilizado en tabletas mientras que cliente móvil es una aplicación Android.
* Un juez puede realizar un seguimiento de las pruebas de atletismo cubriendo sus hojas de campo, que están implementadas de forma que fuerzan el cumplimiento de los reglamentos de los concursos.
* Un administrador puede gestionar los datos de atletas, clubes, jueces, competiciones y pruebas.
* Un atleta puede inscribirse en pruebas a través del cliente móvil.
* La API implementa mecanismos de autorización de acceso a los recursos.
* Se aporta con este trabajo una vía para modernizar y agilizar tareas en un contexto que lo necesita.

# Conclusiones

## 13.1 Conclusiones técnicas

A nivel técnico, el objetivo de la realización de este proyecto era el de conocer nuevas tecnologías, aprender nuevos conceptos y profundizar en otros. Este objetivo se ha cumplido con creces ya que se han podido conocer frameworks modernos y con mucha aceptación en el mundo del desarrollo web profesional, como son Angular y Spring.

Se han adquirido conocimientos acerca del desarrollo web basado en componentes, de los frameworks de mapeo objeto-relacional y de mecanismos como la inyección de dependencias.

Además, el proyecto ha servido para conocer más a fondo tareas con las que quizá estaba menos familiarizado, como puede ser el despliegue de aplicaciones.

## 13.2 Conclusiones personales

Una vez finalizado proyecto, personalmente puedo decir que ha sido duro. Duro no por su complejidad, sino por el contexto. Compaginar vida laboral y académica se vuelve una odisea cuando toca abordar un proyecto que requiere bastante implicación. Sin embargo, opino que todo el esfuerzo ha merecido la pena.

Como se ha mencionado anteriormente, la asimilación de nuevas tecnologías y la profundización en algunos conceptos han sido parte del proyecto. Esto ha propiciado que algunas tareas se hayan ido de horas, y estos hechos son los que aportan la experiencia para ir mejorando en aspectos como la estimación de tareas.

Finalmente, reconozco que me hubiera gustado desarrollar alguna funcionalidad más, pero las 300 horas asignadas a esta materia acaban siendo insuficientes si se quiere tanto cantidad como calidad. Los objetivos iniciales se han cumplido y se han aprendido lecciones que van más allá de lo técnico.

# Vías de trabajo futuro

Dadas las restricciones de tiempo para este proyecto, se ha desarrollado un número limitado de funcionalidades seleccionadas en función de su prioridad. A continuación, se mencionan aspectos susceptibles de ser tratados en un futuro:

* **Mejoras en la implementación actual**
  + **Renovación del certificado SSL**: los certificados emitidos por la Autoridad Certificadora Let’s Encrypt caducan a los tres meses y actualmente se renuevan manualmente. Se propone generar un script que se ejecute periódicamente para realizar esta tarea.
  + **Paginación en servidor**: En el cliente web, actualmente se utilizan componentes de Angular que permiten realizar paginación de los datos en la interfaz. Esto es suficiente actualmente, pero debe implementarse en el servidor para no encontrarnos con problemas de rendimiento cuando aumente el volumen de datos.
  + **Edición del perfil de usuario**: Actualmente los usuarios no tienen acceso a modificar su perfil, y si olvidan su contraseña únicamente pueden solicitar que se les envíe una nueva por correo. Se propone permitir que los usuarios puedan cambiar sus contraseñas en las aplicaciones.
  + **Multiidioma**: Aunque no es algo prioritario, permitir que la interfaz de las aplicaciones se presente en distintos idiomas siempre es un valor añadido.
* **Ideas para nuevas funcionalidades**:
  + **Comunicación con juez árbitro**: El juez árbitro es el supervisor de las competiciones de atletismo y a quién el resto de jueces reporta incidencias. Se sugiere implementar una funcionalidad que permita a los jueces enviarle mensajes.
  + **Soporte para carreras**: El proyecto desarrollado está pensado para agilizar en trabajo de los jueces de atletismo en concursos. Se propone investigar qué funcionalidades se podrían desarrollar para dar soporte también a las carreras.

# Referencias

|  |  |
| --- | --- |
| [1] | «Java 8 documentation» [En línea]. Available: <https://docs.oracle.com/javase/8/docs/api/>. [Último acceso: 27 Diciembre 2019]. |
| [2] | «Angular Framework» [En línea]. Available: <https://angular.io/>. [Último acceso: 27 Diciembre 2019]. |
| [3] | «Angular Material» [En línea]. Available: <https://material.angular.io/>. [Último acceso: 27 Diciembre 2019]. |
| [4] | «Material Design» [En línea]. Available: <https://material.io/>. [Último acceso: 27 Diciembre 2019]. |
| [5] | «Spring» [En línea]. Available: <https://spring.io/projects>. [Último acceso: 27 Diciembre 2019]. |
| [6] | «Spring Boot» [En línea]. Available: <https://start.spring.io/>. [Último acceso: 27 Diciembre 2019]. |
| [7] | «Hibernate ORM» [En línea]. Available <https://hibernate.org/orm/>. [Último acceso: 27 Diciembre 2019]. |
| [8] | «MySql» [En línea]. Available: <https://www.mysql.com/>. [Último acceso: 27 Diciembre 2019]. |
| [9] | «DigitalOcean» [En línea]. Available: <https://cloud.digitalocean.com/>. [Último acceso: 27 Diciembre 2019]. |
| [10] | «Guía para obtener y usar un certificado SSL» [En línea]. Available: <https://www.digitalocean.com/community/tutorials/how-to-secure-apache-with-let-s-encrypt-on-ubuntu-18-04>. [Último acceso: 27 Diciembre 2019]. |
| [11] | «Guía para incorporar certificado SSL en Spring Boot» [En línea]. Available: <https://dzone.com/articles/spring-boot-secured-by-lets-encrypt>. [Último acceso: 27 Diciembre 2019]. |
| [12] | «Guía para configurar y ejecutar servicios en Ubuntu» [En línea]. Available: <https://www.dynacont.net/documentation/linux/Useful_SystemD_commands/>. [Último acceso: 27 Diciembre 2019]. |
| [13] | «Guía para generar JWT en Spring Boot» [En línea]. Available: <https://www.devglan.com/spring-security/spring-boot-jwt-auth>. [Último acceso: 27 Diciembre 2019]. |
| [14] | «JWT» [En línea]. Available: <https://jwt.io/>. [Último acceso: 27 Diciembre 2019]. |
| [15] | «Ionic Framework» [En línea]. Available: <https://ionicframework.com/>. [Último acceso: 27 Diciembre 2019]. |
| [16] | «Trello» [En línea]. Available: <https://trello.com/>. [Último acceso: 27 Diciembre 2019]. |
| [17] | «Moment JS» [En línea]. Available: <https://momentjs.com/>. [Último acceso: 27 Diciembre 2019]. |
| [18] | «jwt-decode» [En línea]. Available: <https://www.npmjs.com/package/jwt-decode>. [Último acceso: 27 Diciembre 2019]. |
| [19] | «Spring-boot-starter-mail» [En línea]. Available: <https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-mail>. [Último acceso: 27 Diciembre 2019]. |
| [20] | «MySQL Workbench» [En línea]. Available: <https://www.mysql.com/products/workbench/>. [Último acceso: 27 Diciembre 2019]. |