Administrador de competiciones

CICLO FORMATIVO DE GRADO SUPERIOR

Desarrollo de Aplicaciones Web

AUTORES

Álvaro González Sancha
Bárbara Jurado Sánchez
José Manuel Samper Muñoz

TUTOR

Rocío María Vicente Navas

DEDICATORIAS/AGRADECIMIENTOS

Queremos agradecer a todos los profesores del departamento de informática por todos los conocimientos adquiridos durante los cursos.

INDICE

1.- INTRODUCCIÓN

El proyecto de creación de una herramienta de administración para los campeonatos de gimnasia acrobática en el que nos embarcamos, nace con la idea principal de acercarnos lo máximo posible a la gestión y desarrollo de un proyecto real de software.

Partiendo de ello, trataremos de adaptar metodologías (ágiles) de trabajo y conocimientos adquiridos durante nuestra formación con las ideas y necesidades reales de un cliente potencial. Principalmente, nuestra intención es desarrollarlo con el objetivo de empezar a adaptar nuestra visión y forma de trabajo a lo que nos vamos a encontrar en nuestro futuro laboral.

Como propósito general del software, nos gustaría indicar que, este desarrollo nace con la intención de proporcionar al usuario final, en este caso los árbitros de competición, jueces y la propia federación, una herramienta para la gestión integral de las acciones llevadas a cabo durante los campeonatos de gimnasia acrobática que tengan lugar, en este caso dentro del ámbito de la Federación Madrileña. Es nuestra intención que gracias a esta herramienta que desarrollamos, se pueda automatizar toda acción que se deba llevar a cabo, facilitándola y haciéndola más amigable para el usuario final.

Teniendo todo ello en cuenta, y con la base formativa que hemos adquirido durante el periodo lectivo, planteamos el proyecto como una Arquitectura de Nivel 3, en la que, los diferentes tipos de usuario existentes, a través de una interfaz gráfica situada en el lado cliente, puedan acceder, pasando por la parte servidora, a los datos alojados en la base de datos correspondiente.

Para ello, la base de datos será creada con la idea de contener toda la información relativa a la administración y gestión de este tipo de campeonatos. Como tal, se le dotará de integridad relacional y coherencia para que el acceso a los datos de la misma pueda ser automatizado y lo más expedito posible.

En ella, se contendrán datos tales como: información de los clubes de la federación; información de las inscripciones realizadas en cada campeonato; información sobre los componentes que han accedido a podio en cada campeonato, etc. Dado que el nivel de sensibilidad de los datos, sin ser muy alto, es netamente privado, trataremos de dotar, tanto en lado cliente como en lado servidor de los medios para restringir el acceso a los mismos desde elementos externos que escapen a nuestro control, así como de crear diferentes perfiles, que permitan diferenciar el acceso de distintos usuarios.

Los diferentes actores:

- Pantalla: Es la parte que se va a proyectar y sirve de marcador que visualice las puntuaciones de los participantes en la competición. Pretendemos que alterne la vista entre la pantalla de puntuaciones propiamente dicha y el “streaming”.

- Enlace: Es la persona designada por los clubes para estar con los competidores. Tendrán acceso a las puntuaciones y éste estará limitado por contraseña que se generará con cada inscripción.

- Juez de Mesa: Es el encargado de toda la gestión de la aplicación en el momento de la competición. Tiene acceso al registro de campeonatos, inscripciones, pódiums, rotaciones y deportistas.
- Administrador: Es el súper-usuario de la aplicación. Tiene acceso a todos los formularios y registros, aunque el acceso al contenido dentro del campeonato es meramente simbólico, por si hubiera alguna incidencia que solucionar en un momento puntual.

Por su parte, la interfaz del usuario se desarrollará con la idea de ser lo más sencilla y accesible posible para el usuario, pero dotándola de personalidad propia y un aspecto cuidado y atrayente. Para ello nos serviremos de las siguientes tecnologías:

Más allá del lado cliente, trataremos de dotar a la parte servidora de entidad suficiente para conseguir facilitar la conexión y visualización de los datos que queden alojados en la base de datos. Como hemos visto en las actividades que hemos venido desarrollando, siguiendo el patrón MVC (Modelo, Vista, Controlador), que creemos se adapta fielmente a la naturaleza de nuestro proyecto, el servidor actúa y actuará como elemento de enlace entre dichos datos y su visualización en la parte cliente, anteriormente mencionada. Para conseguir estos propósitos, utilizaremos las siguientes tecnologías:

Nos gustaría incluir dentro del desarrollo temas tales como: la carga automática de datos de clubes y participantes; la formalización de inscripciones de las participantes; la gestión de las puntuaciones de las participantes así como su visualización; la conformidad de los podios de cada categoría; Así como aquellas funcionalidades solicitadas por el cliente potencial, llegado el caso.

También es nuestra intención añadir un servicio de “streaming”, utilizando alguna de las tecnologías o plataformas ya disponibles en Internet, con el objetivo de profesionalizar la aplicación, si bien no es nuestra prioridad.

2.- ALCANCE DEL PROYECTO Y ANÁLISIS PREVIO

Construir una aplicación web para gestionar en tiempo real competiciones de gimnasia acrobática. La infraestructura de la aplicación está pensada para poder tanto alojarse en un dominio web como para poder desplegarse sobre un ordenador en la misma zona de la competición sin necesidad de acceso a internet.

3.- ESTUDIO DE VIABILIDAD

La aplicación que hemos creado tiene una alta capacidad de uso en lso entornos de las competiciones, tanto a nivel profesional como amateur. Al ser una aplicación web hecha en PHP su despliegue es sencillo y fácil pudiendo incluso llevarse portable en un dispositivo externo (USB, Disco duro ,etc...) y mantener todos los datos en él. Además al no requerir internet podemos desplegarlo con un ordenador y un router para tener acceso a la aplicación en la zona de competición.

Por otro lado, es fácil de llevar a la web en un servicio de hosting y disponer de ello en cualquier punto con acceso a internet.

3.1Estado actual del sistema

El sistema gestiona correctamente competiciones y almacena los datos de la misma. También permite tener una pantalla que muestre las puntuaciones y los resultados de manera muy visual para poder proyectarlo.

3.2 Resumen de requisitos del cliente

El cliente quiere un sistema de puntuaciones para la Federación Madrileña de Gimnasia, en el cuál se vayan realizando las rotaciones de cada deportista automáticamente. A su vez, un usuario con rol de juez, podrá asignar puntuaciones a cada rotación, para más tarde crear los pódiums.

3.3 Posibles soluciones

Existen aplicaciones que gestionan las competiciones de manera mas o menos general, pero ninguna específica de este tipo de deporte y que además funcione como una aplicación de marcador.

3.4 Solución elegida

Decidimos tomar esta idea de servicio web que sea accesible en la zona de competición y permita gestionar todos los aspectos básicos de la competición de manera simple y fácil para cualquier persona sin conocimientos específicos de informática

3.5 Planificación temporal de las tareas del proyecto Administración de competiciones

Las tareas realizadas en este proyecto están accesibles mediante el siguiente enlace https://tree.taiga.io/project/alvarinchin-gestor-competiciones/

En total ha llevado un tiempo total de aproximadamente 3 meses.

3.6 Planificación de los recursos a utilizar

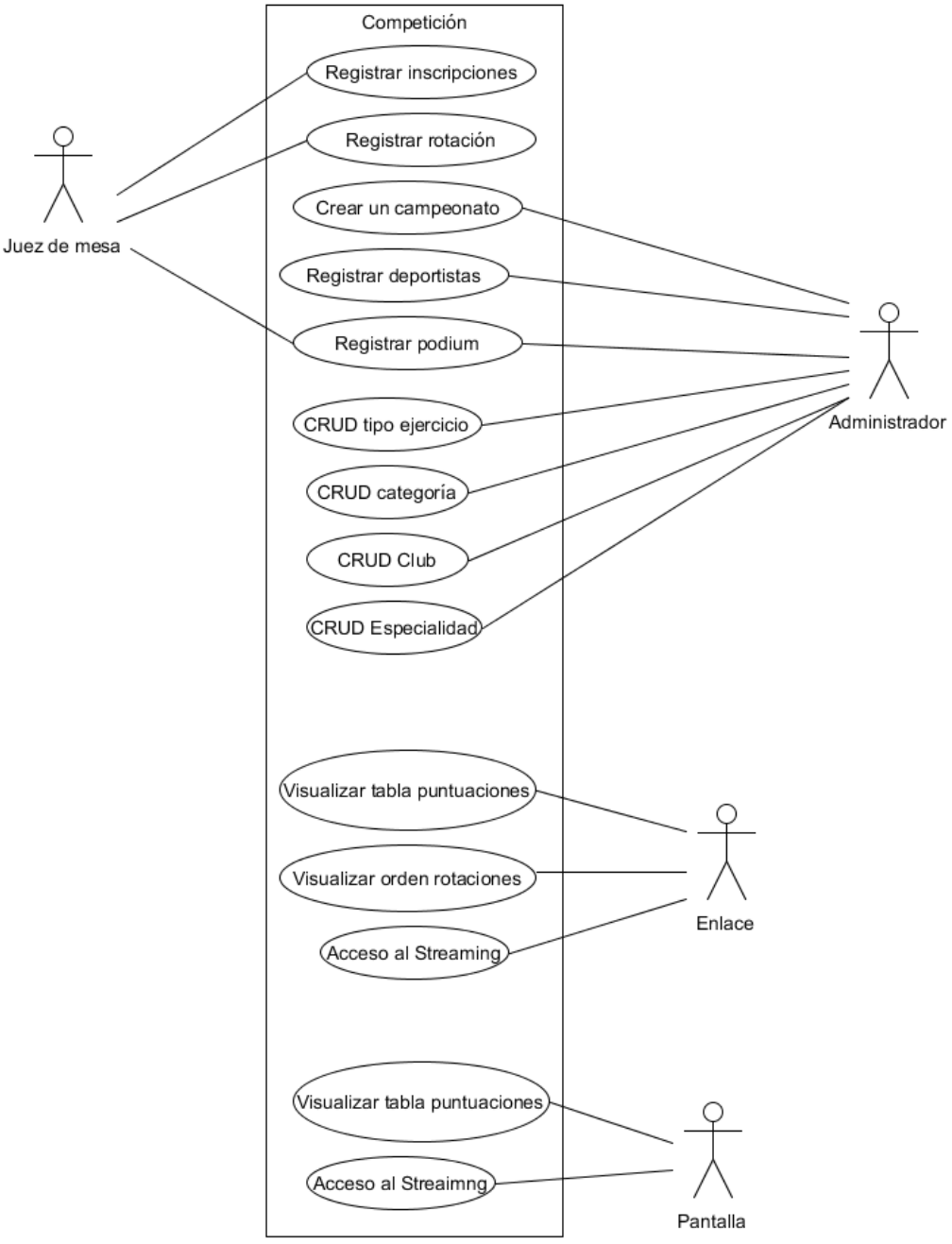
En este proyecto se utiliza una metodología ágil SCRUM, que ayuda a planificar las tareas que debe realizar el equipo.

Dentro de las tecnologías que usamos solo AngularJs y JWT requería de una especialización añadida a la formación que ya teníamos.

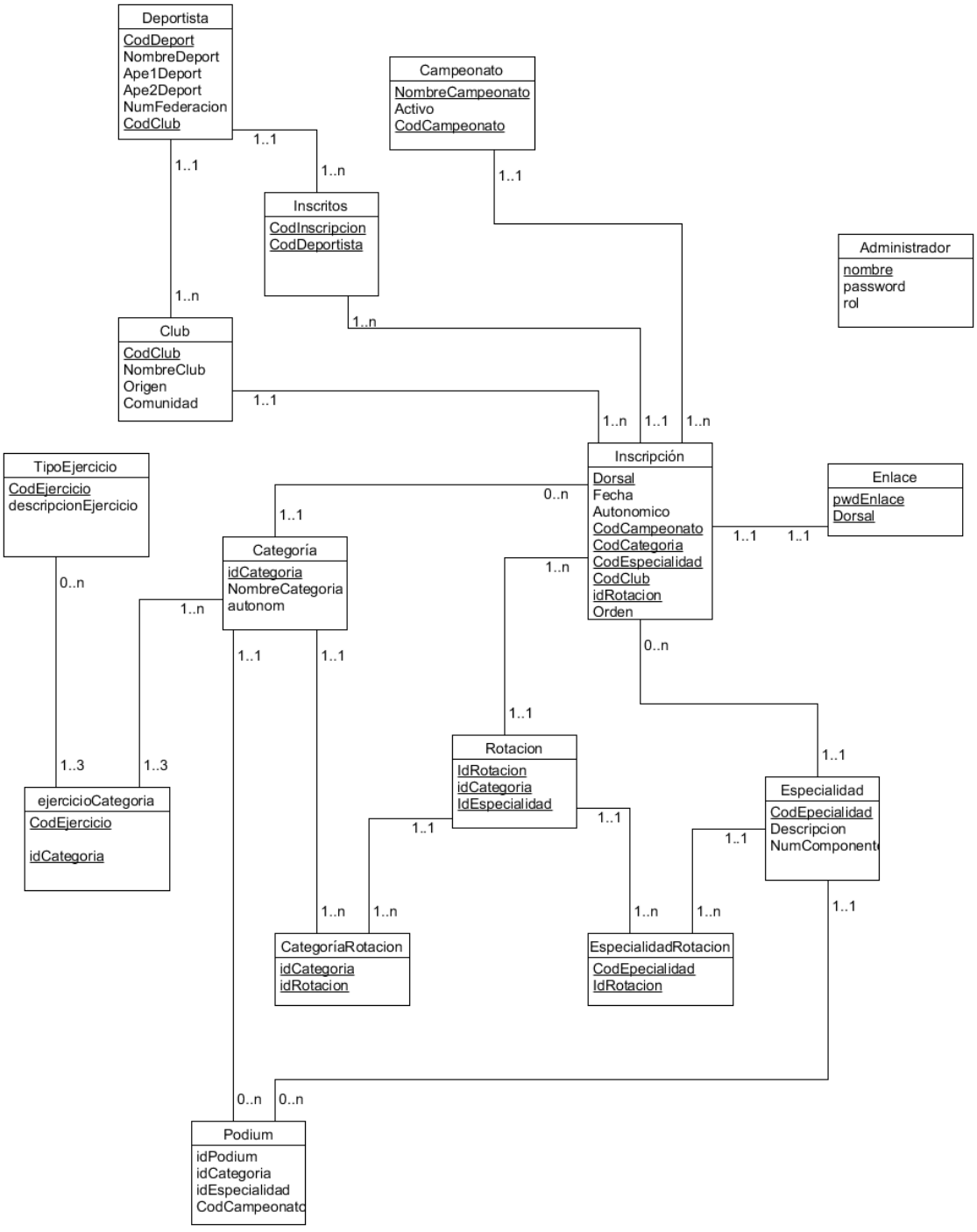
Todo el software usado es de acceso libre y está a disposición de todos bajo licencia openSource. El único gasto añadido sería el de contratación del dominio para mantener la aplicación en internet.

4.- ANÁLISIS

4.1 Diagrama de casos de uso.



4.2 Modelo de datos



4.3 Requisitos funcionales

- Es necesario que a la aplicación puedan acceder usuarios con distintos tipos de roles para poder separar funciones y tareas.
- Tener una vista que sirva de pantalla de puntuaciones que al añadir una nueva puntuación esta muestre en primer plano los detalles de esta inserción
- Permitir el acceso a través de wifi a la aplicación a los enlaces de los equipos para poder ver el orden de realización de los ejercicios y los detalles de los mismos

4.4 Requisitos no funcionales

- Debe tener un diseño intuitivo a la par que atractivo.
- Ser fácil de usar para gente no acostumbrada a aplicaciones específicas
- Ser estable y ligera para poder llevarla en el menos espacio posible

5.- DISEÑO

Crearemos una seria de servicios REST que serán accesibles desde la vista de nuestra aplicación, aunque esta puede estar separada por completo de los servicios hemos decidido mantenerlo juntos y aprovechar los recursos de php en las vistas y el enrutamiento de url.

Para la vista usaremos AngujarJs que facilita el acceso de los datos desde el lado del cliente y hace más cómodo el acceso a servicios AJAX.

5.1 Estructura de la aplicación

Front End

La estructura de las vistas de la aplicación se basa en la tecnología de AngularJs, permitiendo tener dentro de cada página una aplicación que conecta mediante servicios REST para la realizar todas las acciones de un CRUD.

Estructura de Angular JS:

Angular Js es un framework de desarrollo que se aplica sobre JavaScript, la principal ventaja frente a usar javascript puro es la inclusión de metodos y servicios que facilitan el acceso a datos como AJAX, intervals, etc.

Las aplicaciones de AngularJs, tienen una estructura de MVC, esto quiere decir que cada página de html que hay en este proyecto es en si mismo una aplicación funcional completa, solo requiere una fuente de datos y esto viene dado por la aplicación REST.

Otras tecnologías del Front End:

Jquery -> Nos permite tener un fácil acceso al DOM y usar recursos como los de JqueryUI que añaden funcionalidades.

BootStrap -> Este framework para html nos permite tener una aplicación funcional y visualmente atractiva de manera rápida y además añade la compatibilidad con otros formatos como puede ser Smartphone haciendo todo el contenido responsive.

Back End

Toda la aplicación está desarrollada en PHP. Elegimos esta tecnología por ser muy cómoda y rápida, además con las tecnologías que aplicamos sobe esta el desarrollo se pudo hacer en menos tiempo y de manera más profesional.

Sobre PHP usamos Codeiginter, un framework de MVC muy sencillo pero potente, este permite el uso de helpers que facilitan el trabajo enormemente y la inclusión de librerias de terceros para añadir nuevas funcionalidad, estas mediante la configuración se cargan automaticamente permitiendo que esten accesibles sin necesidad de importarlas manualmente.

Librerias externas:

Template -> Una pequeña librería que editamos para poder tener un servicio de templating, pudiendo crear las páginas web por un lado en pequeños trozos y poder separar entre los diferentes roles que tenemos.

JWT -> JSON web token, como nuestra aplicación iba a conformarse por servicios web necesitábamos una manera de validar el acceso a estos, la mejor manera y más cómoda es el uso de JWT, que codifica en una cadena de texto en tres partes los datos que nosotros le pasemos, en nuestro caso el usuario con el que estamos logueados, y así mantener el menor numero de datos en sesión evitando posibles ataques de XSS(cross site script).

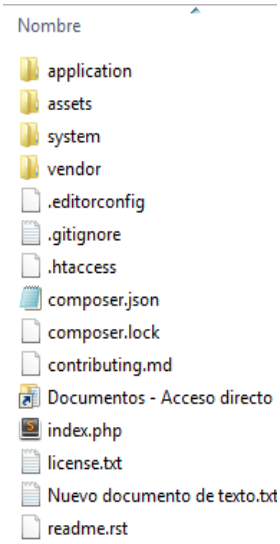
JSON -> Los servicios REST que levantamos funcionan con tecnología JSON, esto permite un rápido intercambio de información ya que son texto plano, pero junto a angularJS tienen funcionalidades para recorrerlos y mostrarlos en pantalal muy potentes.

Persistencia de objetos -> Para facilitar el acceso a los datos y mejorar la seguridad de la aplicación usamos RedBeans php, un framework para trabajar con acceso a base de datos, un ORM, es cómodo, sencillo y no es necesario crear entidades como clases.

Librerías personalizadas -> Durante el desarrollo y con la finalidad de que el código sea más claro y evitar repetirnos sacamos varios métodos útiles a una librería externa.

Servicios REST

La idea de la aplicación es que sea los más escalable posible, esto lo conseguimos convirtiendo la aplicación en un servicio REST, esto es, en esencia, separar al máximo el front del back. En nuestro caso pese a estar bastante separado podríamos haber estructurado la aplicación aun más ajena una de otra, haciendo que Front y back no compartieran Framework como hacen ahora. Gracias a angular JS podríamos tener esta aplicación sin apenas cambiar nada de esta, pero preferimos usar las facilidades que nos brindaba php a la hora de trabajar con páginas web y el sistema de templating que teniamos y editamos. En cualquier caso si por ejemplo esta aplicación se quisiera llevar a una app movil, es desarrollo sería mucho más fácil, ya que solo habría que desarrollar la parte visual de la aplicación y usar las URs de las que ya disponemos para realizar las acciones CRUD.

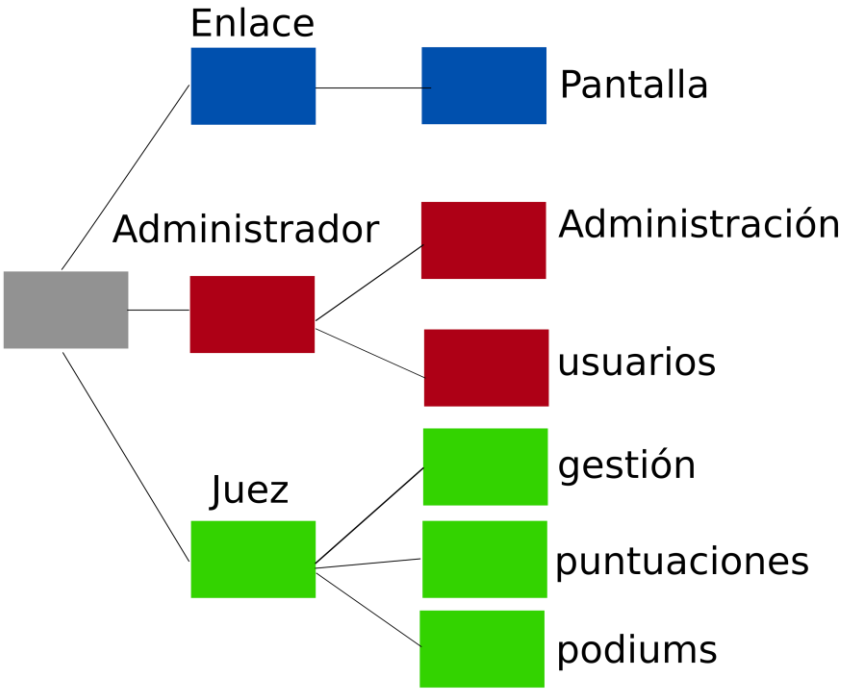

Nombre
application
assets
system
vendor
.editorconfig
.gitignore
.htaccess
composer.json
composer.lock
contributing.md
Documentos - Acceso directo
index.php
license.txt
Nuevo documento de texto.txt
readme.rst

5.2 Componentes del sistema / arquitectura de red

Paquete XAMPP v3.2.2 de aplicaciones

La estructura de archivos es el siguiente:Es el estandar establecido por Codeigniter para usar el framework.

Estructura de páginas:



5.3 Herramientas y tecnologías utilizadas.

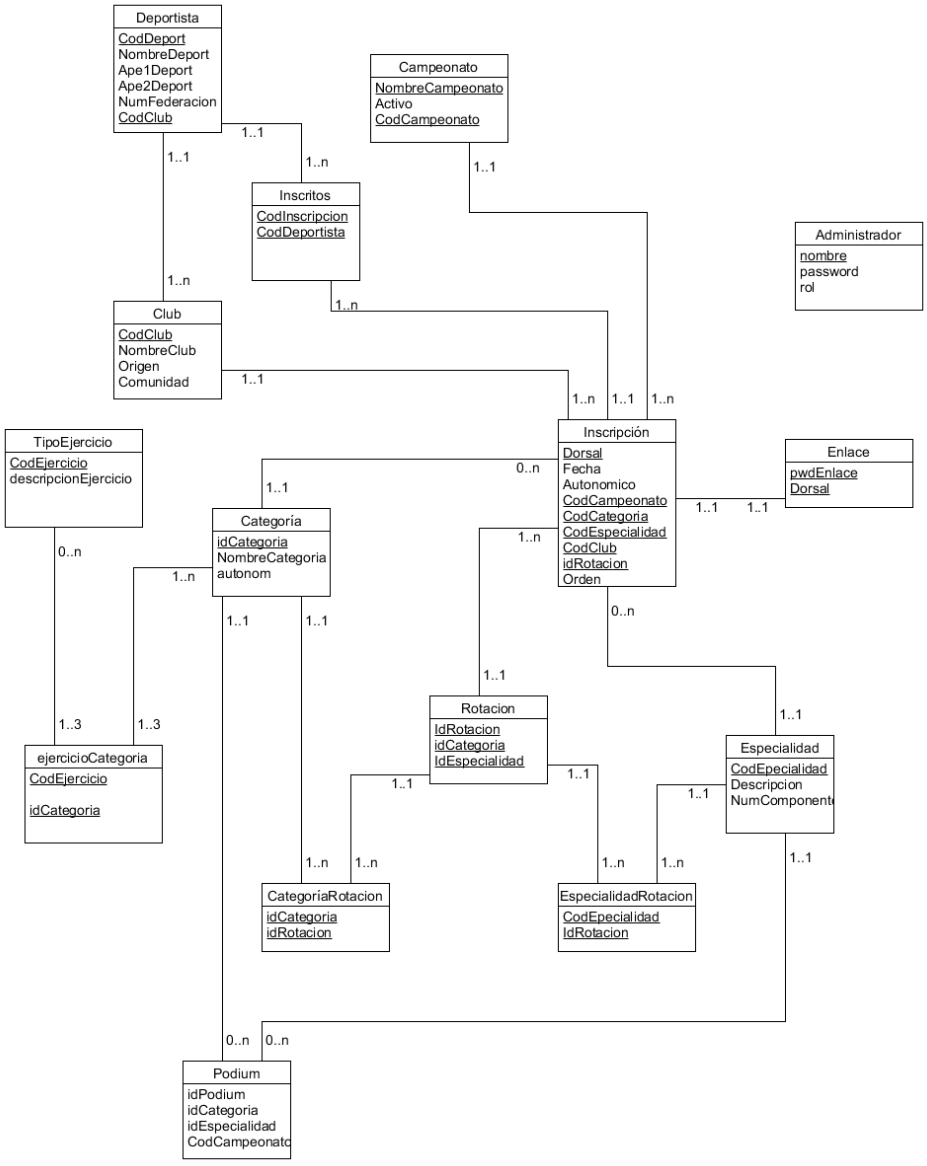
- Back-end: PHP
- Framework: CodeIgniter
- ORM: Red beans PHP
- Front-end: HTML, JavaScript, AngularJS, Bootstrap
- Base de datos: MySQL
- Servidor: Apache

- Tecnologías adicionales: Git(control de versiones), Composer(gestor de dependencias),JWT(control de acceso)

6.- IMPLEMENTACIÓN

Se lleva a cabo la implementación de la página web en PHP. La creación de las tablas es generada automáticamente en la base de datos.

6.1 Implementación del modelo de datos



6.2 Carga de datos

Inicialmente la base de datos contendrá una tabla con un usuario administrador.

Éste usuario podrá hacer login en la aplicación y crear otros usuarios con rol de administrador o juez.

6.3 Configuraciones realizadas en el sistema

Ya que se trata de una aplicación web, ésta no necesita muchos recursos.

Debe ejecutarse en un sistema operativo de Windows que tenga XAMPP instalado.

6.4 Implementaciones de código realizadas

A través de JWT hemos implementado que dependiendo del rol que tenga el usuario con el que nos hemos loqueado, nos redirija a la página de administración (en caso de que el usuario con el que nos hemos logueado tenga un rol de administrador), a la página de gestión (en el caso de que estemos logueado con un usuario con un rol de juez), o a la página de enlace (en el caso de que seamos un usuario invitado)

7.- PRUEBAS

Son muchas las pruebas que pueden realizarse en un proyecto para eliminar los posibles errores y garantizar su correcto funcionamiento. Los casos de prueba establecen las condiciones/variables que permitirán determinar si los requisitos establecidos se cumplen o no.

A continuación se detallan algunos de los casos de prueba que se ejecutarán para comprobar la correcta construcción de este proyecto.

7.1 Casos de pruebas

Caso de prueba crear nueva competición

•**Autor**: Bárbara

•**Caso de prueba**: Crear una nueva competición

•**Identificador del caso de prueba:** Crear competición.

•**Descripción**: el usuario administrador puede crear una nueva competición.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de crear una nueva competición, asignando un nombre y una fecha para ésta.

•**Condiciones de ejecución**: debe introducir un nombre de competición que no exista previamente y una fecha válida.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente.

•**Datos necesarios para poder ejecutar la prueba**: nombre de una competición y la fecha de ésta.

•**Resultado esperado:** se crea la nueva competición y aparece la competición creada en una lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse un nombre de competición válido. La fecha de la competición debe ser igual o superior a la actual.

•**Resultado obtenido:** crea correctamente la nueva competición y agrega ésta a una lista con las competiciones que se vayan generando.

•**Valor de salida al ejecutar el caso de prueba:** aparece la competición creada en una lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la creación de la nueva competición en la base de datos y su inserción en la lista, y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Crear una nueva competición

•**Comparación del valor esperado y obtenido para concluir**: no se ha creado la competición.

•**Posible causa de error:**

- El nombre de la competición ya existía previamente.
- La fecha introducida es anterior a la actual

•**Posible corrección**: asignar un nuevo nombre de competición, asignar una fecha válida.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido crearse la nueva competición, es que el nombre de ésta ya existiera previamente o que la fecha no sea válida. Hay que introducir un nombre válido y una fecha válida.

Caso de prueba modificar una competición

•**Autor**: Bárbara

•**Caso de prueba**: Modificar una competición

•**Identificador del caso de prueba:** Modificar competición.

•**Descripción**: el usuario administrador puede modificar una competición.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de modificar una competición, asignando un nuevo nombre y/o una fecha para ésta.

•**Condiciones de ejecución**: debe introducir un nombre de competición que no exista previamente y una fecha válida.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creada una competición con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: nombre de una competición y la fecha de ésta.

•**Resultado esperado:** se modifica competición y aparece la competición modificada en la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse un nombre de competición válido. La fecha de la competición debe ser igual o superior a la actual.

•**Resultado obtenido:** modifica correctamente la competición y aparece la modificación en la lista de competiciones.

•**Valor de salida al ejecutar el caso de prueba:** aparece la competición modificada en la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la modificación de la competición en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** modificar una competición

•**Comparación del valor esperado y obtenido para concluir**: no se ha modificado la competición.

•**Posible causa de error:**

- El nombre de la competición ya existía.
- La fecha introducida es anterior a la actual

•**Posible corrección**: asignar un nombre válido de competición, asignar una fecha válida.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido modificar una competición, es que el nombre de ésta ya existiera previamente o que la fecha no sea válida. Hay que introducir un nombre válido y una fecha válida.

Caso de prueba borrar una competición

•**Autor**: Bárbara

•**Caso de prueba**: Borrar una competición

•**Identificador del caso de prueba:** Borrar competición.

•**Descripción**: el usuario administrador puede borrar una competición.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de borrar una competición.

•**Condiciones de ejecución**: debe pulsar el botón de eliminar una competición.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creada una competición con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: debe haber creada al menos una competición.

•**Resultado esperado:** se elimina la competición y desaparece de la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** al pulsar el botón para eliminar la competición, ésta se elimina de la base de datos y desaparece de la lista de competiciones.

•**Resultado obtenido:** elimina la competición y desaparece de la lista de competiciones.

•**Valor de salida al ejecutar el caso de prueba:** desaparece la competición eliminada de la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la eliminación de la competición en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** borrar una competición

•**Comparación del valor esperado y obtenido para concluir**: no se ha eliminado la competición.

•**Posible causa de error:**

- Error de conexión con la base de datos.

•**Posible corrección**: comprobar que la base de datos funciona correctamente.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido eliminar una competición, es que no haya conectado correctamente con la base de datos.

Caso de prueba crear un nuevo deportista

•**Autor**: Bárbara

•**Caso de prueba**: Crear un nuevo deportista

•**Identificador del caso de prueba:** Crear deportista

•**Descripción**: el usuario administrador puede crear un nuevo deportista.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de crear un nuevo deportista.

•**Condiciones de ejecución**: un usuario con rol administrador, será capaz de crear un nuevo deportista, asignando un nombre, apellidos, fecha de nacimiento y número de federación.

•**Condiciones de ejecución**: se deben introducir uno datos válidos. El número de federación no debe haber sido asignado a otro deportista con anterioridad.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente.

•**Datos necesarios para poder ejecutar la prueba**: nombre, apellidos, fecha de nacimiento y número de federación.

•**Resultado esperado:** se crea el nuevo deportista y aparece en una lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse un nombre, apellidos, fecha de nacimiento y número de federación válidos. La fecha de nacimiento debe ser anterior a la actual.

•**Resultado obtenido:** crea correctamente el nuevo deportista y agrega éste a una lista con los deportistas que se vayan generando.

•**Valor de salida al ejecutar el caso de prueba:** aparece el deportista creado en una lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la creación del nuevo deportista en la base de datos y su inserción en la lista, y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Crear un nuevo deportista

•**Comparación del valor esperado y obtenido para concluir**: no se ha creado el deportista.

•**Posible causa de error:**

- La fecha introducida es igual o superior a la actual
- Número de federación repetido

•**Posible corrección**: asignar un número de federación válido, asignar una fecha válida.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido crear el nuevo deportista, es que el número de federación de éste ya existiera previamente o que la fecha no sea válida. Hay que introducir un número de federación válido y una fecha válida.

Caso de prueba modificar un deportista

•**Autor**: Bárbara

•**Caso de prueba**: Modificar un deportista

•**Identificador del caso de prueba:** Modificar deportista

•**Descripción**: el usuario administrador puede modificar un deportista.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de modificar un deportista.

•**Condiciones de ejecución**: un usuario con rol administrador, será capaz de modificar un deportista, asignando un nuevo nombre, apellidos, fecha de nacimiento o número de federación.

•**Condiciones de ejecución**: se deben introducir uno datos válidos. El número de federación no debe haber sido asignado a otro deportista con anterioridad.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente.

•**Datos necesarios para poder ejecutar la prueba**: deportista creado con anterioridad.

•**Resultado esperado:** se modifica el deportista y aparece en la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse un nombre, apellidos, fecha de nacimiento ó número de federación válidos. La fecha de nacimiento debe ser anterior a la actual.

•**Resultado obtenido:** modifica correctamente el deportista aparece modificado en la lista con los deportistas generados.

•**Valor de salida al ejecutar el caso de prueba:** aparece el deportista modificado en la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la modificación de un deportista en la base de datos y su aparición en la lista, y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Modificar un deportista

•**Comparación del valor esperado y obtenido para concluir**: no se ha modificado el deportista.

•**Posible causa de error:**

- La fecha introducida es igual o superior a la actual
- Número de federación repetido
- No ha conectado correctamente con la base de datos
- Hay algún dato vacío.

•**Posible corrección**: asignar un número de federación válido, asignar una fecha válida, comprobar que estés todos los campos rellenos.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido modificar un deportista, es que el número de federación de éste ya existiera previamente o que la fecha no sea válida. Hay que introducir un número de federación válido y una fecha válida.

Caso de prueba borrar un deportista

•**Autor**: Bárbara

•**Caso de prueba**: Borrar un deportista

•**Identificador del caso de prueba:** Borrar deportista.

•**Descripción**: el usuario administrador puede borrar un deportista.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de borrar un deportista.

•**Condiciones de ejecución**: debe pulsar el botón de eliminar un deportista.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creado un deportista con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: debe haber creado al menos un deportista.

•**Resultado esperado:** se elimina el deportista y desaparece de la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** al pulsar el botón para eliminar el deportista, éste se elimina de la base de datos y desaparece de la lista de deportistas.

•**Resultado obtenido:** elimina el deportista y desaparece de la lista de deportistas.

•**Valor de salida al ejecutar el caso de prueba:** desaparece el deportista eliminado de la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la eliminación de un deportista en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** borrar un deportista.

•**Comparación del valor esperado y obtenido para concluir**: no se ha eliminado el deportista.

•**Posible causa de error:**

- Error de conexión con la base de datos.

•**Posible corrección**: comprobar que la base de datos funciona correctamente.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido eliminar un deportista, es que no haya conectado correctamente con la base de datos.

Caso de prueba crear nuevo club

•**Autor**: Bárbara

•**Caso de prueba**: Crear un nuevo club

•**Identificador del caso de prueba:** Crear club.

•**Descripción**: el usuario administrador puede crear un nuevo club.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de crear un nuevo club, asignando un nombre, origen y rol.

•**Condiciones de ejecución**: debe introducir un nombre de club que no exista previamente y rellenar todos los campos.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente.

•**Datos necesarios para poder ejecutar la prueba**: nombre de un club, origen y rol.

•**Resultado esperado:** se crea el nuevo club y aparece en una lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse un nombre de club, origen y rol válidos.

•**Resultado obtenido:** crea correctamente el nuevo club y agrega éste a una lista con los clubs que se vayan generando.

•**Valor de salida al ejecutar el caso de prueba:** aparece el club creado en una lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la creación del nuevo club en la base de datos y su inserción en la lista, y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Crear un nuevo club

•**Comparación del valor esperado y obtenido para concluir**: no se ha creado el club.

•**Posible causa de error:**

- El nombre del club ya existía previamente.
- No se han rellenado todos los datos

•**Posible corrección**: asignar un nuevo nombre de club, rellenar todos los campos.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido crearse el nuevo club, es que el nombre de éste ya existiera previamente o que no se hayan rellenado los campos de origen ó rol. Hay que introducir un nombre válido y rellenar todos los campos.

Caso de prueba modificar un club

•**Autor**: Bárbara

•**Caso de prueba**: Modificar un club

•**Identificador del caso de prueba:** Modificar club.

•**Descripción**: el usuario administrador puede modificar un club.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de modificar un club, asignando un nuevo nombre, origen o rol para éste.

•**Condiciones de ejecución**: debe introducir un nombre de club que no exista previamente, origen y rol válidos.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creado un club con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: nombre de un club, origen y rol de éste.

•**Resultado esperado:** se modifica club y aparece modificado en la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse un nombre de club válido. Los campos origen y rol deben estar rellenos.

•**Resultado obtenido:** modifica correctamente un club y aparece la modificación en la lista de clubes.

•**Valor de salida al ejecutar el caso de prueba:** aparece un club modificado en la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la modificación de un club en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** modificar un club

•**Comparación del valor esperado y obtenido para concluir**: no se ha modificado un club

•**Posible causa de error:**

- El nombre del club ya existía.
- Los campos origen o rol no están rellenos

•**Posible corrección**: asignar un nombre válido de club, rellenar los campos vacíos.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido modificar un club, es que el nombre de éste ya existiera previamente o que campos origen o rol no están rellenos.

Caso de prueba borrar un club

•**Autor**: Bárbara

•**Caso de prueba**: Borrar un club

•**Identificador del caso de prueba:** Borrar club.

•**Descripción**: el usuario administrador puede borrar un club.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de borrar un club.

•**Condiciones de ejecución**: debe pulsar el botón de eliminar un club.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creado un club con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: debe haber creada al menos un club**.**

•**Resultado esperado:** se elimina el club y desaparece de la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** al pulsar el botón para eliminar el club, éste se elimina de la base de datos y desaparece de la lista de clubes.

•**Resultado obtenido:** elimina el club y desaparece de la lista de clubes.

•**Valor de salida al ejecutar el caso de prueba:** desaparece el club eliminado de la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la eliminación del club en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** borrar un club

•**Comparación del valor esperado y obtenido para concluir**: no se ha eliminado el club**.**

•**Posible causa de error:**

- Error de conexión con la base de datos.

•**Posible corrección**: comprobar que la base de datos funciona correctamente.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido eliminar el club, es que no haya conectado correctamente con la base de datos.

Caso de prueba crear nueva categoría

•**Autor**: Bárbara

•**Caso de prueba**: Crear una nueva categoría

•**Identificador del caso de prueba:** Crear categoría.

•**Descripción**: el usuario administrador puede crear una nueva categoría.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de crear una nueva categoría, asignando un nombre.

•**Condiciones de ejecución**: debe introducir un nombre de categoría que no exista previamente.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente.

•**Datos necesarios para poder ejecutar la prueba**: nombre de una categoría.

•**Resultado esperado:** se crea la nueva categoría y aparece creada en una lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse un nombre de categoría válido.

•**Resultado obtenido:** crea correctamente la nueva categoría y agrega ésta a una lista con las categorías que se vayan generando.

•**Valor de salida al ejecutar el caso de prueba:** aparece la categoría creada en una lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la creación de la nueva categoría en la base de datos y su inserción en la lista, y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Crear una nueva categoría

•**Comparación del valor esperado y obtenido para concluir**: no se ha creado la categoría.

•**Posible causa de error:**

- El nombre de la categoría ya existía previamente.
- El campo nombre no está relleno.

•**Posible corrección**: asignar un nuevo nombre de categoría.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido crearse la nueva categoría, es que el nombre de ésta ya existiera previamente. Hay que introducir un nombre válido.

Caso de prueba modificar una categoría

•**Autor**: Bárbara

•**Caso de prueba**: Modificar una categoría

•**Identificador del caso de prueba:** Modificar categoría.

•**Descripción**: el usuario administrador puede modificar una categoría.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de modificar una categoría, asignando un nuevo nombre para ésta.

•**Condiciones de ejecución**: debe introducir un nombre de categoría que no exista previamente.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creada una categoría con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: nombre de una categoría.

•**Resultado esperado:** se modifica categoría y aparece la categoría modificada en la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse un nombre de categoría válido.

•**Resultado obtenido:** modifica correctamente la categoría y aparece la modificación en la lista de categorías.

•**Valor de salida al ejecutar el caso de prueba:** aparece la categoría modificada en la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la modificación de la categoría en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** modificar una categoría

•**Comparación del valor esperado y obtenido para concluir**: no se ha modificado la categoría.

•**Posible causa de error:**

- El nombre de la categoría ya existía.
- El campo nombre no está relleno.

•**Posible corrección**: asignar un nombre válido de categoría.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido modificar una categoría, es que el nombre de ésta ya existiera previamente. Hay que introducir un nombre válido.

Caso de prueba borrar una categoría

•**Autor**: Bárbara

•**Caso de prueba**: Borrar una categoría

•**Identificador del caso de prueba:** Borrar categoría.

•**Descripción**: el usuario administrador puede borrar una categoría.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de borrar una categoría.

•**Condiciones de ejecución**: debe pulsar el botón de eliminar una categoría.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creada una categoría con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: debe haber creada al menos una categoría.

•**Resultado esperado:** se elimina la categoría y desaparece de la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** al pulsar el botón para eliminar la categoría, ésta se elimina de la base de datos y desaparece de la lista de competiciones.

•**Resultado obtenido:** elimina la categoría y desaparece de la lista de categorías.

•**Valor de salida al ejecutar el caso de prueba:** desaparece la categoría eliminada de la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la eliminación de la categoría en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** borrar una categoría

•**Comparación del valor esperado y obtenido para concluir**: no se ha eliminado la categoría.

•**Posible causa de error:**

- Error de conexión con la base de datos.

•**Posible corrección**: comprobar que la base de datos funciona correctamente.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido eliminar una categoría, es que no haya conectado correctamente con la base de datos.

Caso de prueba crear nueva categoría

•**Autor**: Bárbara

•**Caso de prueba**: Crear una nueva especialidad

•**Identificador del caso de prueba:** Crear especialidad.

•**Descripción**: el usuario administrador puede crear una nueva especialidad.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de crear una nueva especialidad, asignando una descripción y número de componentes.

•**Condiciones de ejecución**: debe introducir un nombre de especialidad que no exista previamente.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente.

•**Datos necesarios para poder ejecutar la prueba**: nombre de una especialidad.

•**Resultado esperado:** se crea la nueva especialidad y aparece creada en una lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse un nombre de especialidad válido.

•**Resultado obtenido:** crea correctamente la nueva especialidad y agrega ésta a una lista con las especialidades que se vayan generando.

•**Valor de salida al ejecutar el caso de prueba:** aparece la especialidad creada en una lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la creación de la nueva especialidad en la base de datos y su inserción en la lista, y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Crear una nueva especialidad

•**Comparación del valor esperado y obtenido para concluir**: no se ha creado la especialidad.

•**Posible causa de error:**

- La descripción de la especialidad ya existía previamente.
- El campo descripción no está relleno.
- El número de componentes está vacío

•**Posible corrección**: asignar un nuevo nombre de especialidad.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido crearse la nueva especialidad, es que la descripción de ésta ya existiera previamente. Hay que introducir una descripción válida.

Caso de prueba modificar una especialidad

•**Autor**: Bárbara

•**Caso de prueba**: Modificar una especialidad

•**Identificador del caso de prueba:** Modificar especialidad.

•**Descripción**: el usuario administrador puede modificar una especialidad.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de modificar una especialidad, asignando una nueva descripción o número de componentes para ésta.

•**Condiciones de ejecución**: debe introducir una descripción de especialidad que no exista previamente.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creada una especialidad con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: descripción de una especialidad.

•**Resultado esperado:** se modifica especialidad y aparece la especialidad modificada en la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse una descripción de especialidad válido.

•**Resultado obtenido:** modifica correctamente la especialidad y aparece la modificación en la lista de especialidades.

•**Valor de salida al ejecutar el caso de prueba:** aparece la especialidad modificada en la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la modificación de la especialidad en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** modificar una especialidad

•**Comparación del valor esperado y obtenido para concluir**: no se ha modificado la especialidad.

•**Posible causa de error:**

- La descripción de la especialidad ya existía.
- El campo descripción no está relleno.
- El número de componentes está vacío

•**Posible corrección**: asignar una descripción válida y rellenar el número de componentes para especialidad.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido modificar una especialidad, es que la descripción de ésta ya existiera previamente. Hay que introducir una descripción válida.

Caso de prueba borrar una especialidad

•**Autor**: Bárbara

•**Caso de prueba**: Borrar una especialidad

•**Identificador del caso de prueba:** Borrar especialidad.

•**Descripción**: el usuario administrador puede borrar una especialidad.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de borrar una especialidad.

•**Condiciones de ejecución**: debe pulsar el botón de eliminar una especialidad.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creada una especialidad con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: debe haber creada al menos una especialidad.

•**Resultado esperado:** se elimina la especialidad y desaparece de la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** al pulsar el botón para eliminar la especialidad, ésta se elimina de la base de datos y desaparece de la lista de competiciones.

•**Resultado obtenido:** elimina la especialidad y desaparece de la lista de especialidades.

•**Valor de salida al ejecutar el caso de prueba:** desaparece la especialidad eliminada de la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la eliminación de la especialidad en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** borrar una especialidad

•**Comparación del valor esperado y obtenido para concluir**: no se ha eliminado la especialidad.

•**Posible causa de error:**

- Error de conexión con la base de datos.

•**Posible corrección**: comprobar que la base de datos funciona correctamente.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido eliminar una especialidad, es que no haya conectado correctamente con la base de datos.

Caso de prueba crear nuevo tipo de ejercicio

•**Autor**: Bárbara

•**Caso de prueba**: Crear un nuevo tipo de ejercicio

•**Identificador del caso de prueba:** Crear tipo de ejercicio.

•**Descripción**: el usuario administrador puede crear un nuevo tipo de ejercicio.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de crear un nuevo tipo de ejercicio, asignando una descripción.

•**Condiciones de ejecución**: debe introducir una descripción de tipo de ejercicio que no exista previamente.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente.

•**Datos necesarios para poder ejecutar la prueba**: descripción de un tipo de ejercicio.

•**Resultado esperado:** se crea el nuevo tipo de ejercicio y aparece en una lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse una descripción de tipo de ejercicio.

•**Resultado obtenido:** crea correctamente el nuevo tipo de ejercicio y agrega éste a una lista con los tipos de ejercicios que se vayan generando.

•**Valor de salida al ejecutar el caso de prueba:** aparece el tipo de ejercicio creado en una lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la creación del nuevo tipo de ejercicio en la base de datos y su inserción en la lista, y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Crear un nuevo tipo de ejercicio

•**Comparación del valor esperado y obtenido para concluir**: no se ha creado el tipo de ejercicio.

•**Posible causa de error:**

- La descripción del tipo de ejercicio ya existía previamente.

•**Posible corrección**: asignar una nueva descripción de tipo de ejercicio.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido crearse el nuevo tipo de ejercicio, es que la descripción de éste ya existiera previamente.

Caso de prueba modificar un tipo de ejercicio

•**Autor**: Bárbara

•**Caso de prueba**: Modificar un tipo de ejercicio

•**Identificador del caso de prueba:** Modificar tipo de ejercicio.

•**Descripción**: el usuario administrador puede modificar un tipo de ejercicio.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de modificar un tipo de ejercicio, asignando una nueva descripción.

•**Condiciones de ejecución**: debe introducir una descripción de tipo de ejercicio que no exista previamente.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creado un tipo de ejercicio con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: nombre de un tipo de ejercicio, origen y rol de éste.

•**Resultado esperado:** se modifica tipo de ejercicio y aparece modificado en la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse una descripción de tipo de ejercicio válido.

•**Resultado obtenido:** modifica correctamente un tipo de ejercicio y aparece la modificación en la lista de tipos de ejercicios.

•**Valor de salida al ejecutar el caso de prueba:** aparece el tipo de ejercicio modificado en la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la modificación de un tipo de ejercicio en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** modificar un tipo de ejercicio

•**Comparación del valor esperado y obtenido para concluir**: no se ha modificado un tipo de ejercicio

•**Posible causa de error:**

- La descripción del tipo de ejercicio ya existía.
- Los campos origen o rol no están rellenos

•**Posible corrección**: asignar una descripción válida de tipo de ejercicio,

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido modificar un tipo de ejercicio, es que la descripción de éste ya existiera previamente.

Caso de prueba borrar un tipo de ejercicio

•**Autor**: Bárbara

•**Caso de prueba**: Borrar un tipo de ejercicio

•**Identificador del caso de prueba:** Borrar tipo de ejercicio.

•**Descripción**: el usuario administrador puede borrar un tipo de ejercicio.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de borrar un tipo de ejercicio.

•**Condiciones de ejecución**: debe pulsar el botón de eliminar un tipo de ejercicio.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creado un tipo de ejercicio con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: debe haber creada al menos un tipo de ejercicio**.**

•**Resultado esperado:** se elimina el tipo de ejercicio y desaparece de la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** al pulsar el botón para eliminar el tipo de ejercicio, éste se elimina de la base de datos y desaparece de la lista de tipos de ejercicios.

•**Resultado obtenido:** elimina el tipo de ejercicio y desaparece de la lista de tipos de ejercicios.

•**Valor de salida al ejecutar el caso de prueba:** desaparece el tipo de ejercicio eliminado de la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la eliminación del tipo de ejercicio en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** borrar un tipo de ejercicio

•**Comparación del valor esperado y obtenido para concluir**: no se ha eliminado el tipo de ejercicio**.**

•**Posible causa de error:**

- Error de conexión con la base de datos.

•**Posible corrección**: comprobar que la base de datos funciona correctamente.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido eliminar el tipo de ejercicio, es que no haya conectado correctamente con la base de datos.

Caso de prueba crear nuevo usuario

•**Autor**: Bárbara

•**Caso de prueba**: Crear un nuevo usuario

•**Identificador del caso de prueba:** Crear usuario.

•**Descripción**: el usuario administrador puede crear un nuevo usuario.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de crear un nuevo usuario, asignando un login, origen y rol.

•**Condiciones de ejecución**: debe introducir un login de usuario que no exista previamente y rellenar todos los campos.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente.

•**Datos necesarios para poder ejecutar la prueba**: login de un usuario, password y rol.

•**Resultado esperado:** se crea el nuevo usuario y aparece en una lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse un login de usuario, password y rol válidos.

•**Resultado obtenido:** crea correctamente el nuevo usuario y agrega éste a una lista con los usuarios que se vayan generando.

•**Valor de salida al ejecutar el caso de prueba:** aparece el usuario creado en una lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la creación del nuevo usuario en la base de datos y su inserción en la lista, y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Crear un nuevo usuario

•**Comparación del valor esperado y obtenido para concluir**: no se ha creado el usuario.

•**Posible causa de error:**

- El login del usuario ya existía previamente.
- No se han rellenado todos los datos

•**Posible corrección**: asignar un nuevo login de usuario, rellenar todos los campos.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido crearse el nuevo usuario, es que el login de éste ya existiera previamente o que no se hayan rellenado los campos de password ó rol. Hay que introducir un login válido y rellenar todos los campos.

Caso de prueba modificar un usuario

•**Autor**: Bárbara

•**Caso de prueba**: Modificar un usuario

•**Identificador del caso de prueba:** Modificar usuario.

•**Descripción**: el usuario administrador puede modificar un usuario.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de modificar un usuario, asignando un nuevo login, password o rol para éste.

•**Condiciones de ejecución**: debe introducir un login de usuario que no exista previamente, password y rol válidos.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creado un usuario con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: login de un usuario, password y rol de éste.

•**Resultado esperado:** se modifica usuario y aparece modificado en la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** debe introducirse un login de usuario válido. Los campos password y rol deben estar rellenos.

•**Resultado obtenido:** modifica correctamente un usuario y aparece la modificación en la lista de usuarios.

•**Valor de salida al ejecutar el caso de prueba:** aparece un usuario modificado en la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la modificación de un usuario en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** modificar un usuario

•**Comparación del valor esperado y obtenido para concluir**: no se ha modificado un usuario

•**Posible causa de error:**

- El login del usuario ya existía.
- Los campos password o rol no están rellenos

•**Posible corrección**: asignar un login válido de usuario, rellenar los campos vacíos.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido modificar un usuario, es que el login de éste ya existiera previamente o que campos password o rol no están rellenos.

Caso de prueba borrar un usuario

•**Autor**: Bárbara

•**Caso de prueba**: Borrar un usuario

•**Identificador del caso de prueba:** Borrar usuario.

•**Descripción**: el usuario administrador puede borrar un usuario.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol administrador, será capaz de borrar un usuario.

•**Condiciones de ejecución**: debe pulsar el botón de eliminar un usuario.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de administrador, debe haberse logueado correctamente. Debe haber creado un usuario con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: debe haber creada al menos un usuario**.**

•**Resultado esperado:** se elimina el usuario y desaparece de la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** al pulsar el botón para eliminar el usuario, éste se elimina de la base de datos y desaparece de la lista de usuarios.

•**Resultado obtenido:** elimina el usuario de la base de datos y desaparece de la lista de usuarios.

•**Valor de salida al ejecutar el caso de prueba:** desaparece el usuario eliminado de la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la eliminación del usuario en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** borrar un usuario

•**Comparación del valor esperado y obtenido para concluir**: no se ha eliminado el usuario**.**

•**Posible causa de error:**

- Error de conexión con la base de datos.

•**Posible corrección**: comprobar que la base de datos funciona correctamente.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido eliminar el usuario, es que no haya conectado correctamente con la base de datos.

Caso de prueba crear una nueva inscripción

•**Autor**: Bárbara

•**Caso de prueba**: Crear una nueva inscripción

•**Identificador del caso de prueba:** Crear inscripción

•**Descripción**: el usuario juez puede crear una nueva inscripción.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol juez, será capaz de crear una nueva inscripción.

•**Condiciones de ejecución**: un usuario con rol juez, será capaz de crear una nueva inscripción, seleccionando un club, deportistas, competición, especialidad y categoría.

•**Condiciones de ejecución**: se deben introducir datos válidos.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de juez, debe haberse logueado correctamente. Tienen que estar creados previamente club, deportistas, competición, especialidad y categoría.

•**Datos necesarios para poder ejecutar la prueba**: deben que estar creados previamente club, deportistas, competición, especialidad y categoría.

•**Resultado esperado:** se crea la nueva inscripción y aparece en una lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** deben seleccionarse un club, deportistas, competición, especialidad y categoría.

•**Resultado obtenido:** crea correctamente la nueva inscripción y agrega ésta a una lista con las inscripciones que se vayan generando.

•**Valor de salida al ejecutar el caso de prueba:** aparece la inscripción creada en una lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la creación de la nueva inscripción en la base de datos y su inserción en la lista, y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Crear una nueva inscripción

•**Comparación del valor esperado y obtenido para concluir**: no se ha creado la inscripción

•**Posible causa de error:**

- Falta alguna opción por seleccionar
- Error en la conexión con la base de datos

•**Posible corrección**: seleccionar todos los campos necesarios para crear una inscripción.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido crear la nueva inscripción, es que no se hayan seleccionado todos los campos necesarios para la creación.

Caso de prueba borrar una inscripción

•**Autor**: Bárbara

•**Caso de prueba**: Borrar una inscripción

•**Identificador del caso de prueba:** Borrar inscripción.

•**Descripción**: el usuario juez puede borrar una inscripción.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol juez, será capaz de borrar una inscripción.

•**Condiciones de ejecución**: debe pulsar el botón de eliminar una inscripción.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de juez, debe haberse logueado correctamente. Debe haber creado una inscripción con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: debe haber creado al menos una inscripción.

•**Resultado esperado:** se elimina la inscripción y desaparece de la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** al pulsar el botón para eliminar la inscripción, ésta se elimina de la base de datos y desaparece de la lista de inscripciones.

•**Resultado obtenido:** elimina la inscripción y desaparece de la lista de inscripciones.

•**Valor de salida al ejecutar el caso de prueba:** desaparece la inscripción eliminado de la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la eliminación de una inscripción en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** borrar una inscripción.

•**Comparación del valor esperado y obtenido para concluir**: no se ha eliminado la inscripción

•**Posible causa de error:**

- Error de conexión con la base de datos.

•**Posible corrección**: comprobar que la base de datos funciona correctamente.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido eliminar la inscripción, es que no haya conectado correctamente con la base de datos.

Caso de prueba crear una nueva rotación

•**Autor**: Bárbara

•**Caso de prueba**: Crear una nueva rotación

•**Identificador del caso de prueba:** Crear rotación

•**Descripción**: el usuario juez puede crear una nueva rotación.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol juez, será capaz de crear una nueva rotación.

•**Condiciones de ejecución**: un usuario con rol juez, será capaz de crear una nueva rotación, seleccionando una inscripción.

•**Condiciones de ejecución**: seleccionar una inscripción.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de juez, debe haberse logueado correctamente. Debe seleccionarse una inscripción.

•**Datos necesarios para poder ejecutar la prueba**: debe estar creada previamente al menos una inscripción.

•**Resultado esperado:** se crea la nueva rotación y aparece en una lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** se crea la rotación.

•**Resultado obtenido:** crea correctamente la nueva rotación y agrega ésta a una lista con las rotaciones que se vayan generando.

•**Valor de salida al ejecutar el caso de prueba:** aparece la rotación creada en una lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la creación de la nueva rotación en la base de datos y su inserción en la lista, y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Crear una nueva rotación

•**Comparación del valor esperado y obtenido para concluir**: no se ha creado la rotación

•**Posible causa de error:**

- No se ha seleccionado ninguna inscripción
- Error en la conexión con la base de datos

•**Posible corrección**: seleccionar al menos una inscripción para crear una rotación.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido crear la nueva rotación, es que no se haya seleccionado una inscripción.

Caso de prueba borrar una rotación

•**Autor**: Bárbara

•**Caso de prueba**: Borrar una rotación

•**Identificador del caso de prueba:** Borrar rotación.

•**Descripción**: el usuario juez puede borrar una rotación.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol juez, será capaz de borrar una rotación.

•**Condiciones de ejecución**: debe pulsar el botón de eliminar una rotación.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de juez, debe haberse logueado correctamente. Debe haber creado una rotación con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: debe haber creado al menos una rotación.

•**Resultado esperado:** se elimina la rotación y desaparece de la lista.

- **Valor esperado para el correcto funcionamiento del proyecto:** al pulsar el botón para eliminar la rotación, ésta se elimina de la base de datos y desaparece de la lista de rotaciones.
- **Resultado obtenido:** elimina la rotación y desaparece de la lista de rotaciones.
- **Valor de salida al ejecutar el caso de prueba:** desaparece la rotación eliminado de la lista.
- **Comparación del valor esperado y obtenido para concluir:** se esperaba la eliminación de una rotación en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** borrar una rotación.

•**Comparación del valor esperado y obtenido para concluir**: no se ha eliminado la rotación

•**Posible causa de error:**

- Error de conexión con la base de datos.

•**Posible corrección**: comprobar que la base de datos funciona correctamente.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido eliminar la rotación, es que no haya conectado correctamente con la base de datos.

Caso de prueba crear nueva puntuación

•**Autor**: Bárbara

•**Caso de prueba**: Crear un nueva puntuación

•**Identificador del caso de prueba:** Crear puntuación.

•**Descripción**: el usuario juez puede crear una nueva puntuación.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol juez, será capaz de crear una nueva puntuación, asignando dificultad, ejecución, artístico penalización, total.

•**Condiciones de ejecución**: se deben rellenar todos los campos.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de juez, debe haberse logueado correctamente. Deben estar creadas las rotaciones previamente.

•**Datos necesarios para poder ejecutar la prueba**: rotaciones creadas con anterioridad, rellenar todos los campos.

•**Resultado esperado:** se crea la nueva puntuación y se manda al podium.

•**Valor esperado para el correcto funcionamiento del proyecto:** deben rellenarse todos los campos.

•**Resultado obtenido:** crea correctamente la nueva puntuación.

•**Valor de salida al ejecutar el caso de prueba:** aparece el usuario creado en una lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la creación de la nueva puntuación en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Crear un nueva puntuación

•**Comparación del valor esperado y obtenido para concluir**: no se ha creado la puntuación

•**Posible causa de error:**

- No se han rellenado todos los datos

•**Posible corrección**: rellenar todos los campos.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido crearse el nueva puntuación, es que no se hayan rellenado todos los campos.

Caso de prueba modificar una puntuación

•**Autor**: Bárbara

•**Caso de prueba**: Modificar una puntuación

•**Identificador del caso de prueba:** Modificar puntuación.

•**Descripción**: el usuario juez puede modificar una puntuación.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol juez, será capaz de modificar una puntuación, asignando nuevos valores a los campos que sea necesario.

•**Condiciones de ejecución**: debe haber sido creada una puntuación previamente.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de juez, debe haberse logueado correctamente. Debe haber creado una puntuación con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: previa creación de al menos una puntuación.

•**Resultado esperado:** se modifica puntuación y aparece modificado en la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** todos los campos deben estar rellenos.

•**Resultado obtenido:** modifica correctamente una puntuación.

•**Valor de salida al ejecutar el caso de prueba:** se modifica la puntuación en la base de datos.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la modificación de una puntuación en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** modificar una puntuación

•**Comparación del valor esperado y obtenido para concluir**: no se ha modificado una puntuación

•**Posible causa de error:**

- No se han rellenado todos los campos

•**Posible corrección**: rellenar los campos vacíos.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido modificar una puntuación, es que no estén rellenos todos los campos.

Caso de prueba crear nuevo podium

•**Autor**: Bárbara

•**Caso de prueba**: Crear un nuevo podium

•**Identificador del caso de prueba:** Crear podium.

•**Descripción**: el usuario juez puede crear una nuevo podium.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol juez, será capaz de crear una nuevo podium, seleccionando la categoría y la especialidad.

•**Condiciones de ejecución**: se deben seleccionar todas las opciones.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de juez, debe haberse logueado correctamente. Deben estar creadas las puntuaciones previamente.

•**Datos necesarios para poder ejecutar la prueba**: puntuaciones creadas con anterioridad, rellenar todos los campos.

•**Resultado esperado:** se crea el nuevo pódium.

•**Valor esperado para el correcto funcionamiento del proyecto:** deben rellenarse todos los campos.

•**Resultado obtenido:** crea correctamente el nuevo podium.

•**Valor de salida al ejecutar el caso de prueba:** aparece el usuario creado en una lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la creación del nuevo podium en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** Crear un nuevo podium

•**Comparación del valor esperado y obtenido para concluir**: no se ha creado la podium

•**Posible causa de error:**

- No se han rellenado todos los datos
- No se conecta con la base de datos

•**Posible corrección**: rellenar todos los campos.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido crearse el nuevo podium, es que no se hayan rellenado todos los campos.

Caso de prueba borrar un podium

•**Autor**: Bárbara

•**Caso de prueba**: Borrar un podium

•**Identificador del caso de prueba:** Borrar podium.

•**Descripción**: el usuario juez puede borrar un podium.

•**Breve explicación sobre el objetivo del caso de prueba**: un usuario con rol juez, será capaz de borrar un podium.

•**Condiciones de ejecución**: debe pulsar el botón de eliminar un podium.

•**Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba**: un usuario con rol de juez, debe haberse logueado correctamente. Debe haber creado un podium con anterioridad.

•**Datos necesarios para poder ejecutar la prueba**: debe haber creado al menos un podium.

•**Resultado esperado:** se elimina el podium y desaparece de la lista.

•**Valor esperado para el correcto funcionamiento del proyecto:** al pulsar el botón para eliminar el podium, ésta se elimina de la base de datos y desaparece de la lista de podiums.

•**Resultado obtenido:** elimina el podium y desaparece de la lista de podiums.

•**Valor de salida al ejecutar el caso de prueba:** desaparece el podium eliminándolo de la lista.

•**Comparación del valor esperado y obtenido para concluir:** se esperaba la eliminación de un podium en la base de datos y así ha sido.

Errores

•**Autor**: Bárbara

•**Caso de prueba:** borrar un podium.

•**Comparación del valor esperado y obtenido para concluir**: no se ha eliminado el podium

•**Posible causa de error:**

- Error de conexión con la base de datos.

•**Posible corrección**: comprobar que la base de datos funciona correctamente.

•**Detallar la posible forma de corregir el problema**: los posibles problemas para que no haya podido eliminar un podium, es que no haya conectado correctamente con la base de datos.

8.- EXPLOTACIÓN

La implantación es la fase más crítica del proyecto ya que el sistema entra en producción, es decir opera en un entorno real, con usuarios reales.

8.1Planificación

La aplicación se porta en medios extraíbles o se descarga comprimido, El único requisito previo es tener un servidor apache y una base de datos mysql disponibles.

- Copiar el contenido de la aplicación en la raíz de htdocs
- Cambiar, si fuera necesario la configuración de la aplicación para el acceso a la base de datos
- Levantar el servicio del servidor y de la base de datos
- Iniciar sesión y empezar a utilizar la aplicación

8.2 Preparación para el cambio

Para favorecer el proceso de adaptación a la aplicación se ofrecen documentos de ayuda para el inicio del uso y una explicación detallada de cómo realizar todos los procesos desde su despliegue hasta su administración

8.3 Manual de usuario

Se encuentra dentro de la propia aplicación, en la carpeta de documentación en la raíz de la aplicación. Decidimos dejarla en formato digital para evitar impresiones innecesarias de en papel y así al ir con el producto no se pierde.

8.4 Implantación propiamente dicha

Se puede ver el producto final o bien en el repositorio del proyecto en git:

https://github.com/alvarinchin/ProyectoFinal

8.5 Pruebas de implantación

Realizamos pruebas manuales sobre la aplicación para comprobar que todos los casos de uso que determinamos se cumplen correctamente. Decidimos no implementar baterías de test para no añadir más complejidad al desarrollo además el tiempo es limitado y dedicar horas a prepararlos no nos beneficiaría.

9.- DEFINICIÓN DE PROCEDIMIENTOS DE CONTROL Y EVALUACIÓN

A lo largo del ciclo de vida del proyecto se producirán cambios e incidencias que deberán controlarse y registrarse.

- Para controlar las incidencias, dentro del proyecto de Taiga.io que creamos (https://tree.taiga.io/project/alvarinchin-gestor-competiciones/) se pueden poner indecencias para que se puedan solucionar lo antes posible
- Para el control de versiones nos decantamos por Git sobre un servidor público en internet en github (https://github.com/alvarinchin/ProyectoFinal)

10.- CONCLUSIONES

Se han cumplido casi todas las tareas previstas dentro del tiempo. Pese a la falta de disponibilidad con las prácticas y el problema que supone no disponer de el resto del equipo cerca para poder consultar dudas o ayudar con ciertas partes, hemos llevado a buen puerto todas las ideas que marcamos como “imprescindibles” cuando creábamos las bases del proyecto. Siempre hay margen de mejora y mejores tecnologías pero al trabajar sobre una base sólida e independiente , como es la que ofrece un servicio REST, poder realizar cambios o actualizaciones es un trabajo sencillo.

11.- FUENTES

11.1 Legislación

ASIR

Enseñanzas mínimas: Real Decreto 1629/2009 (B.O.E. 18/11/2009)

http://www.madrid.org/fp/ense_fp/catalogo_LOE/pdf/IFCS01/titulo/RD20091629_TS_Admon_Sistemas_Informaticos_en_Red.pdf

Currículo: Decreto 12/2010 (B.O.C.M. 15/04/2010)

http://www.madrid.org/fp/ense_fp/catalogo_LOE/pdf/IFCS01/curriculo/D20100012_Administracion_SistemasInformaticos.pdf DAM

DAM

DAM

Enseñanzas mínimas: Real Decreto 450/2010, de 16 de abril (BOE 20/05/2010)

http://pdf/IFCS02/titulo/RD20100450_TS_Desarrollo_Aplicaciones_Multiplataforma.pdf

Currículo: D. 3/2011, de 13 de enero (BOCM 31/01/2011)

http://pdf/IFCS02/curriculo/D20110003_TS_Desarrollo_Aplicaciones_Multiplataforma.pdf DAW

DAW

DAW

Enseñanzas mínimas: Real Decreto 686/2010, de 20 de mayo (BOE 12/06/2010)

http://pdf/IFCS03/titulo/RD20100686_TS_Desarrollo_Aplicaciones_Web.pdf

Currículo: Decreto 1/2011, de 13 de enero (BOCM 31/01/2011)

http://pdf/IFCS03/curriculo/D20110001_TS_Desarrollo_Aplicaciones_Web.pdf Definición de procedimientos de control y evaluación:

Definición de procedimientos de control y evaluación:

Definición de procedimientos de control y evaluación:

- http://www.xperta.es/es/descripcion.asp
- http://www.xperta.es/es/aquienvadirigido.asp
- http://churriwifi.wordpress.com/2010/04/10/gestion-de-incidencias/
- http://es.wikipedia.org/wiki/Control_de_versiones

11.1Bibliografía

https://angularjs.org/

https://www.w3schools.com/angular/

https://jwt.io/

12.- ANEXOS

12.1 Autenticación con Token en PHP (J.W.T)

La autenticación por Token, es una forma de "securizar" una aplicación manteniéndola en todo momento sin información de estado (o stateless): Dicho "token" será almacenado en el lado del cliente, no habrá tal información de estado.

Las siglas se traducirían como "Json Web Token", y hacen referencia a la utilización de dichos objetos Json para almacenar toda la información necesaria dentro de un contenedor para poder realizar la comprobación de datos, siempre en el lado cliente.

Su funcionamiento se basa principalmente en el siguiente esquema:

- El usuario se autentica en el Gestor de Competiciones con un par usuario/password.
- Una vez autenticado, cada petición que haga irá acompañado de un "token" recién generado en las cabeceras. Dicho token tendrá una composición determinada como se explicará más tarde. Este "token" no se almacenará en el lado servidor, si no en el lado cliente de tres maneras posibles: a través de almacenamiento con cookie; Almacenamiento con sesiones; O "localstorage". Para el proyecto hemos elegido el primero.
- Una vez realizada la petición el "token" es descifrado y atendiendo a la veracidad o existencia de su composición, se redirigirá la aplicación hacia un determinado punto.

12.1.1 EL TOKEN

El "token" será el encargado de transportar consigo la información relativa al usuario que permita su navegación por la aplicación o por determinados puntos de ella. Este "token" tendrá una estructura definida, marcada por la existencia de tres partes diferenciadas:

1. *Cabecera o Header*: La cabecera contendrá información sobre el algoritmo de codificación ("alg") del token y el tipo de token ("typ").
2. *Carga útil o Payload*: Llevará consigo toda la información relativa a la creación, contenido, etc. del "token". Se puede parametrizar a través de un gran número de parámetros predefinidos o no. Los más importantes para nosotros por ser los que hemos utilizado, son:
    - *"iat"*: Indica la fecha de creación del "token".

- *"data"*: Con formato de objeto, añadimos diferentes parámetros identificadores del usuario como su nombre, password o rol, que lo compondrán.

Existen otros también importantes pero que por mantener una estructura sencilla hemos desestimado, como son:

- *"iss"* - Identifica al creador del JWT.
- *"sub"* - Identifica la razón de creación del JWT.
- *"aud"* - Identifica quién va a recibir el JWT. Etc...

3. *Firma o Signature*: Contendrá los dos otros campos anteriores codificados en base64, junto a una clave secreta que estableceremos.

Por tanto, el aspecto visual de un token, visto lo anterior sería el siguiente, estando cada parte de las anteriores separada de la siguiente por un punto:

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE0OTc1NDE0NDMsImRhdGEiOnsibG9naW4iOiJwZXBlIiwicGFzc3dvcmQiOiJwZXBlIiwicm9sIjoiMiJ9fQ.AKcpMSYMhAtg7pvCO70DBt5lxAfUYZ-eCjQl5iy71Ug

El almacenamiento del token, como se ha dicho ya, se realizará a nivel local, para mantener la navegación sin estado, de tal manera que toda la información que se necesite viaje insertada en el almacenamiento elegido. Existen tres formas diferenciadas de almacenarlo:

- Almacenamiento Web HTML 5 (HTML 5 Web Storage): Ambos métodos de almacenamiento incluídos en esta categoría, son susceptibles a ataques de Cross-Side Scripting (XSS), pues cualquier código Javascript ejecutándose tendrá acceso al almacenamiento local. Aunque esta vulnerabilidad se ha ido corrigiendo mediante la utilización de CDNso gestores de dependencias, no deja de ser una posible fuente problemática. En caso de utilizar esta vía, se recomienda enviar el token por protocolo HTTPSen lugar de HTTP.

  1. *Almacenamiento por sesión* (sessionstorage): Utiliza el almacenamiento en sesiones de usuario, de manera que, cuando se cierra el navegador, se eliminan los datos que se requieren.

  2. *Almacenamiento local* (localstorage): Utiliza el almacenamiento Web HTML 5 para almacenar el token. Permite una mayor capacidad de almacenamiento y el acceso por dominio y protocolo.

- Almacenamiento en cliente: Utiliza las cookies para encapsular la información del token y el propio JWT.

3. *Almacenamiento por cookies.*: En principio, elimina la vulnerabilidad anterior, gracias a que imposibilita el acceso Javascriptmediante la utilización de la flag HttpOnly. A pesar de que tradicionalmente, las cookies han sido mecanismos de navegación con estado, en la actualidad esto no es necesario, gracias a Json y a JWTque encapsula todo lo que el servidor necesita en el token, que, a su vez se guarda en la Cookie.

A pesar de todo, este tipo de almacenamiento no está exento de peligro, ya que por sus características es vulnerable a ataques de tipo Cross-Side Request Forgery. Básicamente esto se produce al explotar "la confianza" del sevidor que examina la cookie, independientemente de su origen.

Teniendo en cuenta este hecho, la mayoría de frameworks actuales cuentan con implementaciones que permiten erradicar o impedir este tipo de vulnerabilidades, por ello, resulta más recomendable este método que los anteriores.

12.1.2 IMPLEMENTACIÓN EN EL PROYECTO

En el proyecto hemos implementado la seguridad por tokens, usando como base la librería de Firebase\JWT que contiene los métodos necesarios para codificar y decodificar el token, y basándonos en diferentes clases y sus métodos:

jwtAuth:

- *Descripción breve*: Es la clase principal utilizada para codificar o decodificar un token.
- *Autor*: José Manuel Samper.
- *Version*: 1.0.
- *Atributos*: $key (private)

Esta clase se ha creado con el objetivo de que funcionase a modo de librería para poder ser accedida desde cualquier otra clase que requiriera la codificación o decodificación de datos en un token Jwt. La clase contará con un atributo privado ($key), que será la clave necesaria para decodificar el token, y que podremos cambiar desde aquí. Básicamente cuenta con tres métodos:

1. **Constructor**:
    - *Descripción breve*: Constructor para la clase jwtAuth.

El uso de este constructor es crear una instancia de la clase jwtAuth, y asignarle una clave, a través de su atributo privado $key. Dicha clave, por motivos de seguridad, será codificada en base64, pudiendo admitir otras formas de hash. La clave será modificada a mano, según necesidad y dicha modificación afectará a la seguridad de toda la aplicación, centralizando, por tanto, su modificación.

- *Código*:

```
public function __construct() {

$this->key = base64_encode ( "_clave_" );}
```

2. **CodificarToken**:
   - Descripción breve: Permite codificar un token atendiendo a los parámetros del usuario introducido como parámetro.
   - Params: $usuario (Bean Usuario).
   - Return: $jwt (Datos de usuario encapsulados en token Jwt).

Este método permite codificar los datos encapsulados en un objeto del bean Usuario, tras haber consultado los mismos en la Base de datos. Solo introduce en el token, datos de usuario, password y su rol, así como la fecha de creación del token. La cabecera del mismo, será introducida automáticamente. Devolverá una cadena de caracteres con un token, incluyendo sus tres partes: cabecera, payload y firma.

- Código:

```
public function codificarToken($usuario) {

    $key = $this->key;

$tok = [

 "iat" => time (),

 "data" => [

  "login" => $usuario->login,
```

```
        "password" => $usuario->password,

        "rol" => $usuario->rol]
    ];

    $jwt = JWT::encode ( $tok, $key);

    return $jwt;}
```

3. **DecodificarToken**:
    - Descripción breve: Permite decodificar un token, introducido como parámetro.
    - Params: $jwt (Cadena de caracteres con formato token Jwt).
    - Return: (Un token Jwt con forma de objeto de PHP) o null

    Este método permite decodificar la cadena de caracteres propia de un token Jwt, introducida como parámetro. siempre que, al tomar la clave de la instancia del objeto jwtAuth, esta coincida con la clave de la cabecera de la cadena de caracteres y se trate de un objeto válido. En caso contrario, devolverá null.

    - Código:

```
public function decodificarToken($jwt) {
        $key = $this->key;

        if (is_object ( JWT::decode ( $jwt, $key, array('HS256')) )) {
                return JWT::decode ( $jwt, $key, array('HS256'));
        } else {
                return null;
        }
}
```

12.2 Sincronización de vistas

A la hora de desarrollar la pantalla de puntuaciones, esta debía mantenerse siempre actualizada con los últimos datos subidos a la base de datos, para ello la mejor opción era un Websocket.

Un websocket es una conexión entre el controller y la vista que se mantiene siempre abierta y es bidireccional. Esto quiere decir que a diferencia de un modelo http normal donde el cliente hace una petición y el servidor response, con un websocket el servidor también puede mandar de manera autónoma una respuesta sin necesidad de petición previa. En esencia un websocket es una llamada http que no muere nunca.

Al intentar implementarlo nos vimos con varios problemas: 1-Para tener este websocket es preciso levantar un servidor específico para esta finalidad y no todos los servicios de host permiten acceso a la pantalla de comandos del servidor. 2-La compatibilidad con CodeIgniter es muy mala. CI utiliza un sistema propio de clases y herencias para montar el routing de la aplicación y para levantar la mayor parte de los websockets teníamos que crear clases que heredaban de unas específicas.

viendo estos dos grandes problemas decidimos crear un sistema que emule este servicio, de esta manera mediante Jquery y angular emulamos un servicio de websocket salvando los problemas que encontramos.