Universidad de Valladolid

Escuela de Ingeniería Informática
Grado en Ingeniería del Software

Trabajo de Fin de Grado

JudoCup: herramienta para la gestión de los campeonatos de Judo

Autor: Pablo García Sanz
Tutora: M. Mercedes Martínez González

Agradecimientos

A M.Mercedes por sus conocimientos y consejos que han llevado a cabo la realización de este proyecto.
Al GCC por tantas horas en la facultad y fuera de ella.
Al Trío mágico del aulario por tantas noches de estudio.
A mi familia, por haberme apoyado siempre.
A mi tío Miguel, que una vez me llevó en coche a la facultad.
A Andrea por estar siempre ahí.
Y para finalizar a Emma y a Elías.

Resumen

El objetivo de este proyecto es la realización de una aplicación web que facilite la gestión de los campeonatos de judo, concretamente los de la Federación Castellano Leonesa, que es donde más competiciones he realizado, y he podido observar las deficiencias en cuanto a la gestión manual que realizan.

La aplicación desarrollada podrá ser utilizada en diferentes tipos de dispositivos, tales como tabletas, móviles, ultrabooks, PC, aunque son estos últimos los que mayor preferencia tienen.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

Desde la edad de cinco años, llevo practicando Judo, el cual me ha impuesto valores tales como la cortesía hacia el resto de contrincantes, el coraje, la sinceridad a expresarme libremente, el honor, la modestia, la amistad y la gratitud por la enseñanza recibida, entre otros. Cuando empecé a realizar campeonatos en la Comunidad de Castilla y León, fui observando como iban guardando los resultados de las competiciones y las deficiencias que tenía esta, la gestión etc. Me iba fijando en que no tenían ninguna herramienta software que les ayudara a guardar toda la información, ya que lo hacían y siguen haciendo de forma manual, a la vieja usanza, papel y bolígrafo.

Las tareas generales que se realizan en un campeonato de Judo en la Comunidad de Castilla y León, aunque también sea extensible al resto de comunidades, son las siguientes:

- Inscribir a competidores en un peso.
- Convocar a árbitros a la competición e indicarles el tatami en el que van a ejercer su labor.
- De cada combate, su resultado.
- Al final de la competición, escribir quiénes son los ganadores de cada peso.
- Publicar en su web el listado con los tres primeros de cada peso, es decir, oro, plata y bronce.

Todo ello dio la idea de realizar como Trabajo de Fin de Grado una aplicación que ayudase a la Federación a realizar todas estas tareas de una forma más cómoda.

1.2. Objetivos

El objetivo principal es desarrollar una aplicación para la gestión de los campeonatos de judo, concretamente los que se desarrollen en la federación de Castilla y León de judo. Se pretende que sea fácil de usar para los usuarios de este entorno, es decir, conocedores de las competiciones de judo, ya bien sea los que la organizan (delegado de la competición), o porque colaboran en ella (cronometradores). La funcionalidad cubrirá las tareas que actualmente se vienen realizando de forma manual, a las cuales se les ha hecho referencia en la Motivación.

Como subobjetivos específicos se pueden reseñar:

- Validar la aplicación desarrollada con un conjunto suficiente de usuarios que proporcionen retroalimentación para mejoras futuras.
- Utilizar tecnologías de desarrollo que faciliten, no solo este proyecto, sino también futuras modificaciones con esfuerzos razonables.
- Aplicar los conocimientos aprendidos en los estudios de informática para seleccionar un algoritmo de emparejamiento apropiado, tomando en cuenta las características del entorno donde se aplicará.

- Hacer un desarrollo lo suficientemente seguro para cumplir con los requisitos mínimos de seguridad exigibles actualmente, así como los de privacidad que fuesen oportunos aplicar.

1.3. Entorno

El entorno del proyecto estará formado por todo lo que concierne a las competiciones de Judo. Puesto que todo se hace de forma manual, únicamente disponemos de un fichero excel, donde realizan la inscripción de los competidores en cada una de las competiciones. Este fichero, creado para cada competición, lo usaremos para obtener los campos de entrada que utilizan y pasarlo así a nuestra aplicación. Estos campos son, *nombre*, *apellidos* y *club* en el que va a realizar el campeonato.

1.4. Estructura de la memoria

Esta memoria se estructura en diferentes capítulos. En primer lugar se realiza una breve introducción del trabajo realizado, donde se detallan las motivaciones y objetivos del proyecto. En el segundo capítulo, se hablará sobre la planificación del proyecto y de la metodología utilizada para la realización del mismo.

En el tercero, se detallarán las tecnologías utilizadas, es decir, lenguajes de servidor (PHP), lenguajes del lado del cliente (HTML, CSS, JS), frameworks de apoyo utilizados tales como Materialize[1], Codeigniter[2], Jquery [3], etc. Dedicaremos el cuarto capítulo al análisis: elicitación de requisitos funcionales, no funcionales, de información, casos de uso del sistema..., es decir, los cimientos del proyecto software que se va a realizar.

En el quinto, abarcaremos todo lo relacionado con el diseño, es decir, la arquitectura del sistema. En sexto capítulo, se hablará sobre la implementación y se explicará en profundidad todos los aspectos relevantes relacionados.

En el siguiente capítulo, el séptimo, realizaremos las pruebas necesarias para comprobar que el sistema creado funciona correctamente. La finalidad es tener un sistema fiable, por lo que realizar pruebas finales y que salgan con éxito dan un grado de fiabilidad extra. En caso de que las pruebas saliesen negativas, es decir, que la salida esperada no sea igual a la salida obtenida, se documenta y se pasa a solucionar el problema.

En el octavo capítulo, se presentará la encuesta realizada sobre usuarios potenciales finales para ver el grado de usabilidad y de satisfacción. Se mostrarán los resultados de la encuesta y el análisis hecho a partir de estos resultados. En el último capítulo, se presentará las conclusiones sacadas tras haber realizado este proyecto y trabajo futuro. En los anexos se han incluido el manual de instalación, manual de usuario, diagramas de Gantt, script de la Base de Datos y el contenido del CD.

Capítulo 2

Planificación

A continuación se presenta de forma general la planificación del proyecto. Es necesario antes de empezar a analizar, diseñar e implementar el producto, una planificación exhaustiva para determinar el orden de las tareas y estimar el tiempo de realización de cada una de ellas. Para planificar utilizaremos las ideas que refleja el profesor Pablo de la Fuente en sus diapositivas de la asignatura de Planificación y Gestión de Proyectos [4].

2.1. Alcance, propósitos y objetivos

El objetivo de este proyecto es elaborar una aplicación web que facilite la gestión de campeonatos de judo, cuyos participantes son los competidores, árbitros y jueces-crono.

2.2. Plan de Proceso

El Proceso unificado es un proceso de ingeniería software. Proporciona una guía para asignar tareas y responsabilidades dentro de una organización de desarrollo. El objetivo de toda metodología de desarrollo es asegurar la producción de alta calidad de software. Las actividades que se desarrollan dentro de esta metodología son concretamente la creación y mantenimiento de los modelos.

RUP (Rational Unified Process), es una guía de como usar eficazmente el Lenguaje de Modelado Unificado (UML). UML es un lenguaje de modelado que nos permite comunicar claramente los requisitos, arquitectura y diseños de la aplicación a desarrollar. Originalmente UML fue creado por Rational Software, y ahora la mantiene la OMG(Object Management Gorup). Una de las ventajas de esta metodología es que es adaptable a cada proyecto software, es decir para proyectos grandes se realizará una configuración del RUP y para proyectos de menor envergadura otra diferente.

Seis buenas practicas para el desarrollo efectivo del proyecto son:

- **Desarrollo iterativo e incremental**. A día de hoy los Sistemas Software son bastante sofisticados, y no es posible definir secuencialmente entero el problema, desarrollar el software y probarlo al final. El desarrollo iterativo e incremental permite entender mejor el problema a medida que se van produciendo esas iteraciones. A lo largo de la evolución del proyecto, se han de realizar varias iteraciones hasta que la fase este completamente terminada. En este caso, se intentará cumplir con las iteraciones definidas para cada una de las iteraciones que vienen definidas en el apartado 2.4 de este capítulo.

- **Dirigido por casos de uso**. Describe como elicitar, organizar y documentar los requisitos y restricciones que va a tener la aplicación, facilita la captura y comunicación con el cliente de los requisitos de negocio. El conocimiento de los casos de uso y escenarios que describen el proceso proporcionan una excelente fuente de información de los requisitos funcionales.

- **Arquitectura sólida**. Una de las cosas más importantes, es elaborar una arquitectura sólida que se adapte a nuestro problema en las primeras etapas del desarrollo. RUP describe como

diseñar una arquitectura flexible a los cambios que se puedan producir en las diferentes etapas del desarrollo y que pueda ser reutilizable en los proyectos futuros.

- **Verificar la calidad del software**. A día de hoy uno de los factores más importantes de aceptabilidad de un producto software es su rendimiento. RUP proporciona una planificación, diseño, implementación, ejecución y evaluación de este tipo de test. La calidad está implícita dentro del proceso, de todas las actividades realizadas, involucrando a todos los participantes, usando medidas objetivas.

- **Control de los cambios de software**. El entorno en el que nos movemos los desarrolladores, es cambiante, por lo que el control de los artefactos (modelos, código, documentos, etc) generados tienen que llevarse cuidadosamente. RUP describe como controlar, seguir y monitorizar los cambios para así tener un buen desarrollo iterativo.

- **Gestión de los riesgos**. Enfocada en los riesgos desde la primera fase de desarrollo. Tener un buen plan de actuación para la posible aparición de los riegos disminuye en gran medida que un proyecto software fracase.

2.3. Vista general

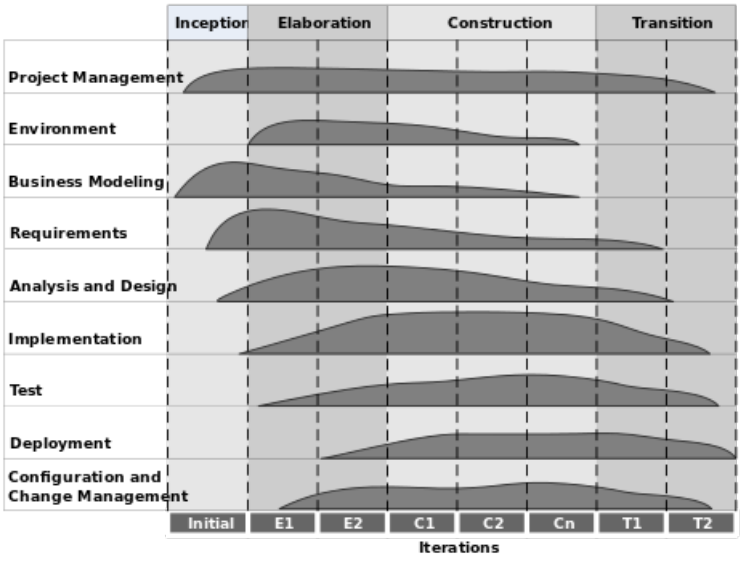


Figura 2.1: Las 4 fases de RUP con sus iteraciones

En la figura 2.1 podemos ver las dos dimensiones, la primera sobre el eje de abscisas que representa el tiempo, expresado en forma de ciclos, iteraciones etc. Y por otro lado tenemos el eje de ordenadas que representa el trabajo realizado en forma de artefactos, actividades, trabajos, etc.

2.4. Fases e iteraciones

En el proceso unificado encontramos las cuatro fases que vemos en la figura 2.1. A grandes rasgos las podríamos describir de la siguiente forma

- **Fase de Inicio** Durante esta fase se establecen las reglas de negocio y el alcance del proyecto. Se definen los requisitos funcionales de la aplicación y los principales actores que van a interactuar con el Sistema. Las reglas de negocio incluyen los principales casos de uso, riesgos, estimación de recursos necesarios, y el plan de proyecto, definiendo la entrega de hitos. Para esta fase realizaremos 1 iteración.

- **Fase de Elaboración** El principal objetivo de esta fase es la construcción de una arquitectura sólida, basada en el análisis del dominio realizado en la fase anterior. La mayor parte de la ingeniería reside en esta fase, por lo que se la considera la más importante. Para esta fase del desarrollo del proyecto, realizaremos 2 iteraciones.

- **Fase de Construcción** Al finalizar esta fase, debe haber versiones funcionales del producto. Se completan los artefactos de análisis, diseño e implementación. En general, de 3 a 4 iteraciones, dependiendo de la envergadura del proyecto.

- **Fase de Transición** Preparación para la puesta en producción del producto y entrega final al cliente, lo que implica la elaboración de los últimos artefactos como manual de usuario, corrección de la memoria final del trabajo realizado, etc.

Cada una de estas fases, corresponde con un hito en el tiempo, donde se tienen que tomar conclusiones si todas las tareas de la fase han sido logradas o no, y si se han completado en el tiempo establecido. En el caso de que se produjeran retrasos, hay que anotarlos en el calendario e intentar reducir el tiempo, si fuese posible, de otras tareas futuras, en caso contrario, se producirán retrasos en la entrega final.

2.5. Gestión del proceso

2.5.1. Plan de puesta en marcha

A continuación se presentarán los conocimientos básicos tanto de herramientas como de lenguajes de programación, etc, para la realización de este proyecto de fin de grado.

- Conocimientos de lenguaje de marcado para el desarrollo de páginas web, **HTML**.
- Conocimientos del lenguaje de scripting **JavaScript**.
- Conocimientos de **CSS** para el diseño gráfico de la página web.
- Conocimientos del framework **Materialize**.
- Conocimiento del framework **JQuery**.
- Conocimientos de **PHP** para la parte de backend.
- Conocimientos del framework **Codeigniter**.
- Conocimientos básicos de **Apache** y **Ubuntu Server**.
- Soltura con el gestor de base de datos **MySQL**.

2.6. Plan de Control

A continuación se muestran las actividades de control.

2.6.1. Control de Requisitos

Si se produjera cambios respecto a los requisitos iniciales, hay que evaluar el posible impacto que puedan producir sobre el normal desarrollo del proyecto, por lo que debe estar contemplado en el plan de riesgos. En caso de que se aceptaran los nuevos cambios, habría que realizar una nueva planificación del calendario de actividades. En el caso de este proyecto, los requisitos están bien definidos desde el principio, los cuales, para este proyecto no se van a cambiar.

2.6.2. Control del calendario

Se utilizará una herramienta que permita gestionar las actividades pendientes de hacer, las que se están realizando y las acabadas, para así tener un control del calendario y ver si se está cumpliendo con la planificación.

2.6.3. Plan de Gestión de Recursos

Los recursos disponibles para la realización del proyecto son de dos tipos: humanos y otros.

- **Recursos humanos**: Una persona realizará el proyecto, por lo que tendrá los roles de analista, programador y *tester*.
- **Otros recursos**: Máquina virtual ofrecida por la Escuela de Ingeniería Informática utilizada para desplegar la aplicación en producción y una máquina para el desarrollo de todo el proyecto, personal.

2.6.4. Plan de trabajo

A continuación se van a mostrar la descripción en detalle de las actividades realizadas.

Id:01	**Comienzo de la fase de inicio**
Predecesoras	-
Duración	0 días hombre.
Descripción	Inicio del grupo de actividades que corresponden a la fase de inicio.

Tabla 2.1: Actividad con id:01, Comienzo de la fase de inicio

Id:02	**Aprendizaje de PHP**
Predecesoras	Id:01
Duración	5 días/ hombre.
Descripción	Aprendizaje del lenguaje de backend PHP. Adquiriendo las aptitudes básicas para el desarrollo del producto. Realizando pequeños programas en este lenguaje

Tabla 2.2: Actividad con id:02, Aprendizaje de PHP

Id:03	**Aprendizaje del framework CodeIgniter**
Predecesoras	Id:02
Duración	5 días/ hombre.
Descripción	Ver las características principales del framework, patrón de diseño que utiliza y funcionalidades que ofrece.

Tabla 2.3: Actividad con id:03, Aprendizaje del framework CodeIgniter

Id:04	**Aprendizaje del framework JQuery**
Predecesoras	Id:01
Duración	5 días/ hombre.
Descripción	Ver las características principales del framework, componentes, funcionalidades que ofrece y buenas prácticas. Realizar ejemplos basados en algún tutorial de internet.

Tabla 2.4: Actividad con id:04, Aprendizaje del framework JQuery

Id:05	**Aprendizaje del framework Materialize**
Predecesoras	Id:04
Duración	5 días/ hombre.
Descripción	Ver las características principales del framework, componentes, funcionalidades que ofrece y buenas prácticas. Realizar ejemplos basados en algún tutorial de internet.

Tabla 2.5: Actividad con id:05, Aprendizaje del framework Materialize

Id:06	**Aprendizaje de Adobe Dreamweaver cs6**
Predecesoras	Id:01
Duración	4 día/ hombre.
Descripción	Aprendizaje de la herramienta de edición de vistas Adobe Dreamweaver.

Tabla 2.6: Actividad con id:06, Aprendizaje de Adobe Dreamweaver CS6

Id:07	**Definición de actividades**
Predecesoras	Id:01
Duración	3 días/ hombre.
Descripción	Definir cada una de las actividades necesarias para la correcta finalización del proyecto. Estimar el tiempo de realización en días hombre, indicando las actividades predecesoras a la realización de la actividad si las hubiera.

Tabla 2.7: Actividad con id:07, Definición de actividades

Id:08	**Planificación de los riesgos**
Predecesoras	Id:07
Duración	1 días/ hombre.
Descripción	Especificación de los riesgos que pueden darse en el desarrollo del producto. Ver proyectos anteriores donde evaluar los riesgos de mayor impacto y mayor frecuencia para tenerlos en primer plano. Definir el plan de actuación de cada riesgo.

Tabla 2.8: Actividad con id:08, Planificación de riesgos

Id:09	**Calendarización**
Predecesoras	Id:08
Duración	2 días/ hombre.
Descripción	En función de la estimación de tiempo de las actividades definidas anteriormente, realizar un calendario con fechas aproximadas de entrega. Realizar el diagrama de Gantt correspondiente a la calendarización.

Tabla 2.9: Actividad con id:09, Calendarización de actividades

Id:10	**Control de versiones**
Predecesoras	Id:01.
Duración	1 día/ hombre.
Descripción	Puesta en marcha del sistema de control de versiones de todos los artefactos, documentos y código, entre otros.

Tabla 2.10: Actividad con id:10, Control de versiones

Id:11	**Fin de la fase de inicio**
Predecesoras	Id:03, Id:05, Id:09 e Id:10.
Duración	0 días / hombre.
Descripción	Fin de la fase de inicio.

Tabla 2.11: Actividad con id:11, Control de versiones

Id:12	**Inicio de la fase de elaboración**
Predecesoras	Id:11.
Duración	0 días / hombre
Descripción	Inicio del grupo de actividades que corresponden a la fase de elaboración.

Tabla 2.12: Actividad con id:12, Inicio de la fase de Elaboración

Id:13	**Elicitar requisitos**
Predecesoras	Id:12.
Duración	7 días/ hombre.
Descripción	Obtener información del sistema, definir escenarios y casos de uso donde poder sacar información de requisitos funcionales, requisitos no funcionales, requisitos de información y reglas de negocio.

Tabla 2.13: Actividad con id:13, Elicitar requisitos

Id:14	**Definir los Casos de uso (CU)**
Predecesoras	Id:13.
Duración	7 días/ hombre.
Descripción	A partir de los datos obtenidos acerca del funcionamiento del Sistema, definir los casos de uso, poniendo mayor atención en los CU más importantes, los críticos.

Tabla 2.14: Actividad con id:14, Casos de uso

Id:15	**Máquinas de estado**
Predecesoras	Id:14.
Duración	4 día/ hombre.
Descripción	Realización máquinas de estado que más complejidad generen.

Tabla 2.15: Actividad con id:15, Máqinas de estado

Id:16	**Definir el modelo de dominio**
Predecesoras	Id:15.
Duración	7 días/ hombre.
Descripción	Tras un primer análisis del problema, definir el modelo de dominio utilizando cualquier herramienta de modelado que facilite la tarea. El lenguaje de modelado será UML.

Tabla 2.16: Actividad con id:16, Modelo de dominio

Id:17	**Diagramas de secuencia**
Predecesoras	Id:16.
Duración	7 día/ hombre.
Descripción	Realización de los diagramas de secuencia de aquellos escenarios críticos de la aplicación.

Tabla 2.17: Actividad con id:17, Diagramas de secuencia

Id:18	**Modelo de despliegue**
Predecesoras	Id:17.
Duración	1 día/ hombre.
Descripción	Realización del modelo de despliegue con notación UML.

Tabla 2.18: Actividad con id:18, Modelo de despliegue

Id:19	**Diseñar la estructura de la Base de Datos**
Predecesoras	Id:16.
Duración	10 días/ hombre.
Descripción	Definir el esquema de la Base de Datos teniendo en cuenta los requisitos de información obtenidos en fases anteriores y el modelo de dominio. Utilizando notación UML.

Tabla 2.19: Actividad con id:19, Diseño de la Base de Datos

Id:20	**Diseño del esquema relacional**
Predecesoras	Id:19.
Duración	3 días/ hombre.
Descripción	Definir el esquema relacional basándose en criterios de diseño de Bases de Datos.

Tabla 2.20: Actividad con id:20, Diseño del esquema Relacional

Id:21	**Definir el script de la Base de Datos**
Predecesoras	Id:20.
Duración	3 días/ hombre.
Descripción	Diseño del script de la base de datos. Poblar con los datos necesarios, acorde a los requisitos de información.

Tabla 2.21: Actividad con id:21, Diseño del script de la Base de Datos

Id:22	**Test de la Base de Datos**
Predecesoras	Id:21.
Duración	2 días/ hombre.
Descripción	Realizar consultas a la base de datos previamente poblada con datos aleatorios para comprobar que se pueden realizar todas las consultas pertinentes para el correcto desarrollo del producto.

Tabla 2.22: Actividad con id:22, Test de la Base de Datos

Id:23	**Diseño de la interfaz de usuario (IU)**
Predecesoras	Id:15.
Duración	4 día/ hombre.
Descripción	Bocetos con las vistas de la aplicación para dar estructura al diseño de la página. Utilizar patrones de diseño de usabilidad para mayor aceptación del usuario final.

Tabla 2.23: Actividad con id:23, Diseño IU

Id:24	**Diseño de la arquitectura del sistema**
Predecesoras	Id:15, Id:17, Id:19.
Duración	10 día/ hombre.
Descripción	Diseño de la arquitectura del sistema.

Tabla 2.24: Actividad con id:24, Diseño de la arquitectura del Sistema

Id:25	**Fin de la fase de Elaboración**
Predecesoras	Id:23, Id:24.
Duración	0 días / hombre.
Descripción	Fin de las actividades del grupo de elaboración.

Tabla 2.25: Actividad con id:25, Fin de la fase de Elaboración

Id:26	**Inicio de la fase de construcción**
Predecesoras	Id:25.
Duración	0 días hombre.
Descripción	Inicio del grupo de actividades que corresponden a la fase de construcción.

Tabla 2.26: Actividad con id:26, Inicio de la fase de Construcción

Id:27	**Instalación de LAMP**
Predecesoras	Id:26.
Duración	1 día/ hombre.
Descripción	Instalación de los siguientes paquetes tanto en el entorno de producción como en el entorno de desarrollo. El entorno de producción será el ofrecido por la Escuela de Ingeniería Informática de Valladolid cuyo Sistema Operativo es : *Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-83-generic*. El entorno de desarrollo tendrá un *Windows 10*. ▪ Instalar Apache Server ▪ Instalar MySQL versión 4.1 o superior ▪ Instalar PHP versión 5.6 o superior

Tabla 2.27: Actividad con id:27, Instalar LAMP

Id:28	**Preparar entorno de desarrollo**
Predecesoras	Id:27.
Duración	2 día/ hombre.
Descripción	▪ Instalar el editor de texto PHPStorm. ▪ Instalar el framework Codeigniter. ▪ Instalar el framework Materialize. ▪ Instalar el framework JQuery.

Tabla 2.28: Actividad con id:28, Preparar entorno de desarrollo

Id:29	**Implementación de las vistas**
Predecesoras	Id:28.
Duración	10 días/ hombre.
Descripción	Implementar las vistas utilizando los lenguajes aprendidos y reforzados en la etapa de inicio. Para ello utilizar el programa de Adobe llamado Dreamweaver.

Tabla 2.29: Actividad con id:29, Implementación de las vistas

Id:30	**Implementación de la capa de modelo y controladores**
Predecesoras	Id:29.
Duración	30 día/ hombre.
Descripción	Realización de la implementación de los controladores y modelos de la aplicación basándose en la arquitectura del Sistema creado en la etapa anterior.

Tabla 2.30: Actividad con id:30, Implementación de la capa de modelo y controladores

Id:31	**Diseño *responsive***
Predecesoras	Id:29.
Duración	2 día/ hombre.
Descripción	Realizar una mejora de la aplicación en las vistas cuando se visualiza sobre un dispositivo de tamaño pequeño, mediano, grande o muy grande. Para ello, usar la funcionalidad que nos ofrece Materialize para el diseño *responsive*.

Tabla 2.31: Actividad con id:31, Diseño *responsive*

Id:32	**Revisar compatibilidad con los navegadores**
Predecesoras	Id:30.
Duración	1 día/ hombre.
Descripción	Revisar que todas las funcionalidades de la aplicación se ejecutan correctamente en cada uno de los navegadores más utilizados (*Chrome, IE, FireFox,etc.*).Corregir aquellos fallos que se puedan producir en algún navegador.

Tabla 2.32: Actividad con id:32, Compatibilidad con los navegadores

Id:33	**Fin de la fase de Construcción**
Predecesoras	Id:33.
Duración	0 días hombre
Descripción	Fin del grupo de actividades que corresponden a la etapa de construcción.

Tabla 2.33: Actividad con id:33, Fin de la fase de Construcción

Id:34	**Inicio de la fase de transición**
Predecesoras	Id:33.
Duración	0 días/ hombre.
Descripción	Inicio del grupo de actividades que corresponden a la etapa de transición.

Tabla 2.34: Actividad con id:34, Inicio de la fase de Transición

Id:35	**Realización de las pruebas finales**
Predecesoras	Id:35.
Duración	7 día/ hombre.
Descripción	Realizar las pruebas finales, utilizando métricas que den fiabilidad. La aplicación debe funcionar correctamente sin ningún tipo de error.

Tabla 2.35: Actividad con id:35, Pruebas finales

Id:36	**Manual de usuario**
Predecesoras	Id:36.
Duración	7 día/ hombre.
Descripción	Crear el manual de usuario indicando las funcionalidades de todo el Sistema, resolviendo las posibles dudas del usuario.

Tabla 2.36: Actividad con id:36, Crear el manual de usuario

Id:37	**Manual de instalación**
Predecesoras	Id:36.
Duración	3 día/ hombre.
Descripción	Crear el manual de instalación de la aplicación.

Tabla 2.37: Actividad con id:37, Crear el manual de instalación

Id:38	**Fin de la fase de Transición**
Predecesoras	Id:37, Id:38.
Duración	0 días/ hombre.
Descripción	Fin del grupo de actividades de la etapa de transición.

Tabla 2.38: Actividad con id:38, Fin de la fase de Transición

El diagrama de Gantt completo resultante lo podemos ver en el Anexo C.1. A continuación se van a exponer los diagramas de Gantt en función de las etapas de desarrollo, es decir, inicio, elaboración, construcción y transición.

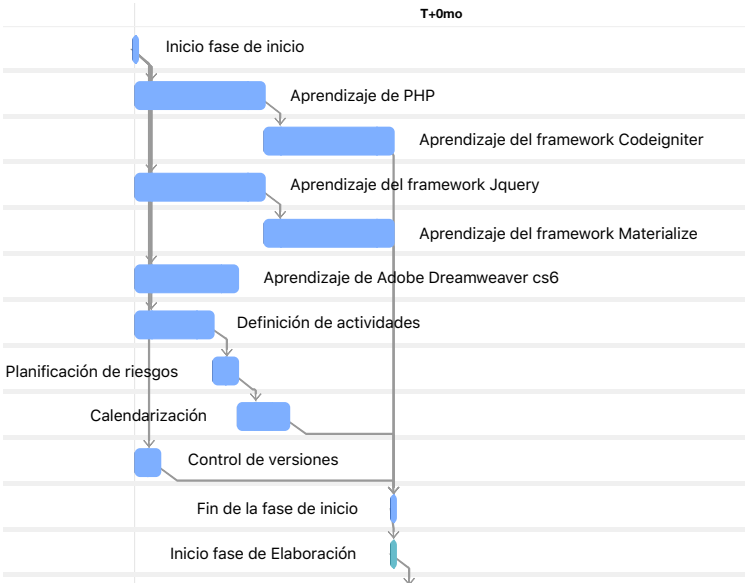


Figura 2.2: Diagrama de Gantt etapa de inicio

En la figura 2.2, tenemos el diagrama de gantt correspondiente a la etapa de inicio. En el se muestran todas las actividades que se han desarrollado para completar esta etapa, la mayoría de las cuales son de aprender nuevas tecnologías, frameworks, lenguajes, etc. Entre otras tareas tenemos la definición de las actividades a desarrollar, calendarización de las actividades, planificación de riesgos y configurar el control de versiones donde almacenaremos y tendremos control de aquellos artefactos que se vayan produciendo a lo largo del proyecto.

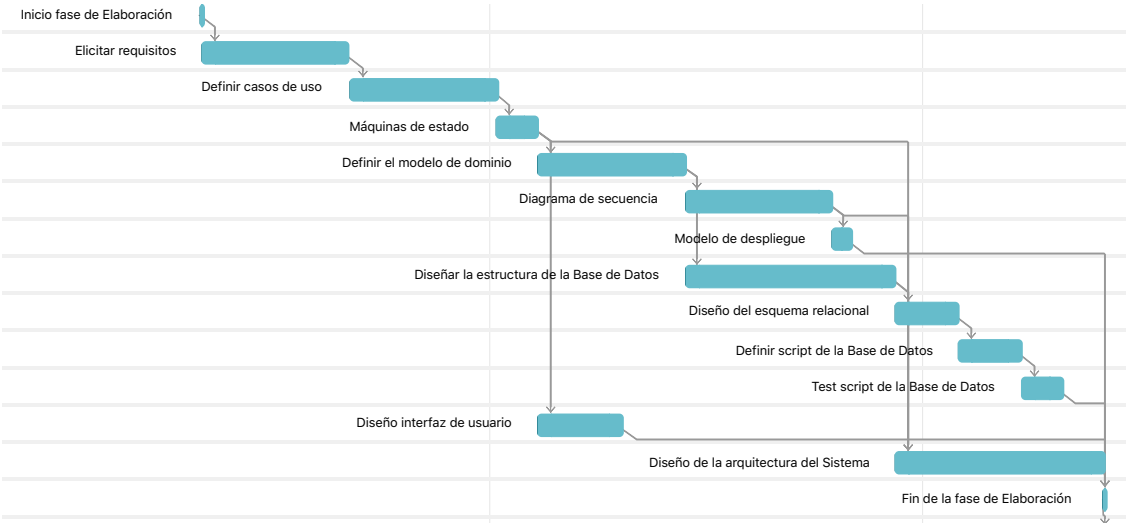


Figura 2.3: Diagrama de Gantt etapa de elaboración

En el diagrama de gantt de la figura 2.3 tenemos las actividades que se han realizado correspondientes a la etapa de elaboración.

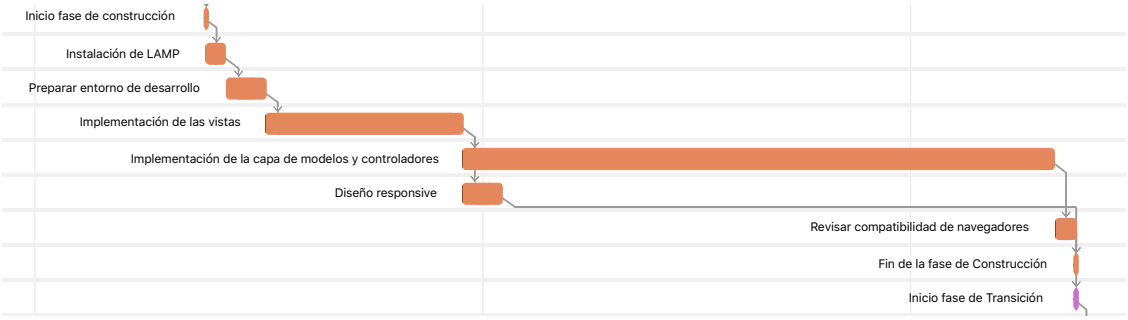


Figura 2.4: Diagrama de Gantt etapa de construcción

En el diagrama de gantt de la figura 2.4 tenemos las actividades que se han realizado correspondientes a la etapa de construcción.

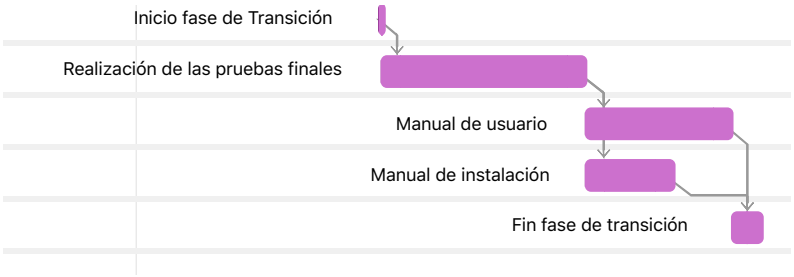


Figura 2.5: Diagrama de Gantt etapa de transición

En el diagrama de gantt de la figura 2.5 tenemos las actividades que se han realizado correspondientes a la etapa de transición.

Debido a que solo se tiene un desarrollador, es decir, un único recurso, la estimación de la duración total del proyecto es de (T0 + 164) días siendo T0 la fecha de inicio del proyecto.

2.6.5. Plan de gestión de Riesgos

La gestión de los riesgos en un proyecto software es uno de los aspectos más importantes y no tenerlos en cuenta provocaría su fracaso. En la siguiente tabla, podemos encontrar los valores que da Pressman[4] para evaluar los riesgos en función del impacto y la probabilidad de que ocurra dicho riesgo.

Impacto/ Probabilidad	**Muy alto**	**Alto**	**Medio**	**Bajo**	**Muy bajo**
Catastrófico	Alto	Alto	Moderado	Moderado	Bajo
Crítico	Alto	Alto	Moderado	Bajo	Ninguno
Marginal	Moderado	Moderado	Bajo	Ninguno	Ninguno
Despreciable	Moderado	Bajo	Bajo	Ninguno	Ninguno

Tabla 2.39: Matriz de identificación/ Probabilidad de riesgos de Pressman

A continuación se describirán los riesgos de software que se han planteado para este proyecto, clasificándolos en tres clases, riesgos de producto, riesgos de proyecto, y riesgos de proceso.

Riesgo 01	Enfermedad del trabajador
Impacto	Crítico
Probabilidad	Baja
Plan de protección	Como el proyecto lo tiene que desarrollar una sola persona, no existe plan de protección
Plan de contingencia	Hay que volver a planificar las actividades de acuerdo a las horas/hombre perdidas por este riesgo

Tabla 2.40: Riesgo 01: Enfermedad del trabajador

Riesgo 02	Máquina de trabajo estropeada
Impacto	Crítico
Probabilidad	Baja
Plan de protección	Realizar copias de seguridad.Utilizar el repositorio creado para guardar todos los artefactos creados
Plan de contingencia	Recuperar el último estado almacenado en la copia de seguridad.

Tabla 2.41: Riesgo 02: Máquina de trabajo estropeada

Riesgo 03	Fallo interfaz externa
Impacto	Crítico
Probabilidad	Medio
Plan de protección	Definir bien que características se usan de esas interfaces para si se produjera un fallo identificarlo fácilmente
Plan de contingencia	Intentar solucionar el problema, si no fuera posible, cambiar el proveedor de la interfaz

Tabla 2.42: Riesgo 03: Fallo interfaces de proveedores

Riesgo 04	Proceso software no documentado
Impacto	Marginal
Probabilidad	Medio
Plan de protección	Ninguno
Plan de contingencia	Preguntar al creador del artefacto correspondiente si fuera posible.

Tabla 2.43: Riesgo 04: Proceso software no documentado

Riesgo 05	Falta de revisiones para validar la aplicación
Impacto	Crítico
Probabilidad	Alta
Plan de protección	Realizar encuestas de aceptación de la aplicación. Reunir a un grupo de usuarios y que utilicen la aplicación
Plan de contingencia	Solucionar los problemas de diseño de interfaz y de aplicación que nos haya comentado el usuario.

Tabla 2.44: Riesgo 05: Falta de revisiones

Riesgo 06	Proceso de diseño pobre
Impacto	Crítico
Probabilidad	Bajo
Plan de protección	Dedicar el tiempo necesario para definir una arquitectura sólida que se ajuste al problema planteado
Plan de contingencia	Si existiera una deficiencia en la arquitectura, que no permitiera solucionar el problema, replanificar el proyecto para redefinirla.

Tabla 2.45: Riesgo 06: Proceso de diseño pobre

Riesgo 07	Fallo entorno de producción
Impacto	Medio
Probabilidad	Bajo
Plan de protección	Comprobar que todo funciona correctamente en cada subida del código al entorno de producción.
Plan de contingencia	Si existiera un problema con el entorno de producción, consultar con los técnicos de la Escuela de Ingeniería Informática para ver la posible solución.

Tabla 2.46: Riesgo 07: Fallo entorno de producción

Riesgo 08	Fallos ortográficos o de otro tipo en la memoria
Impacto	Medio
Probabilidad	Medio
Plan de protección	Revisiones con un corrector ortográfico.
Plan de contingencia	Si se cometiesen fallos ortográficos, hacer revisiones por medio de otras personas ajenas al desarrollo del proyecto.

Tabla 2.47: Riesgo 08: Fallos en la memoria

2.7. Planificación final

Esta segunda planificación surgió debido a la incorporación de una nueva actividad, una encuesta de usabilidad, en la etapa de transición, como resultado se obtuvo el diagrama de la figura 2.6. La introducción de una encuesta de usabilidad surgió de la idea de que el proyecto iba a tener un uso en el futuro y por lo tanto, la aceptación por parte de los usuarios finales de la misma. El diagrama de Gantt resultante tras esta modificación se puede ver en el Anexo C.2.

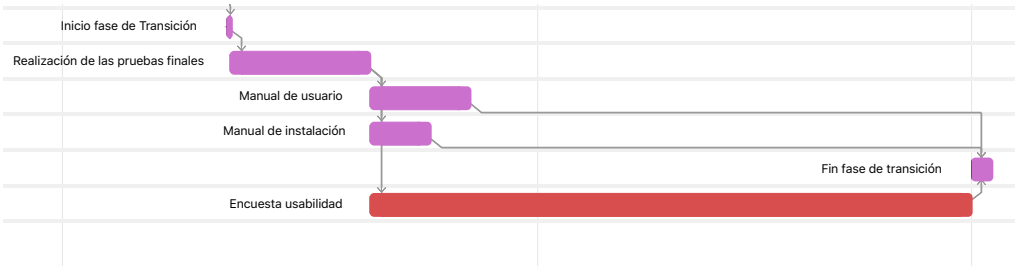


Figura 2.6: Diagrama de Gantt etapa de transición, segunda planificación

Tras este cambio, la estimación de la duración del proyecto ha sido de (T0 + 164 + 30) días, es decir, un total de 194 días de duración. Esto es, la validación con usuarios ha requerido 30 días más de lo que inicialmente se había previsto.

2.8. Informe de seguimiento del proyecto

El proyecto dio comienzo el día 9 Julio de 2017 y finalizó el 29 de Abril de 2018, pero entre medias de este intervalo de fechas, hubo varios periodos vacacionales. A continuación se van a mostrar las tablas con las duraciones reales y estimadas para las cuatro fases del proyecto, inicio, elaboración, construcción y transición (figuras 2.48 a 2.51).

En la tabla 2.48 se encuentran los datos relacionados con la duración de las actividades de la etapa de inicio. Tras analizar estos datos, nos damos cuenta que la estimación realizada es bastante buena, puesto que la diferencia entre la duración estimada y la duración real individualmente y globalmente es aproximadamente cero. Estas estimaciones son buenas, entre otras cosas, porque la mayoría de actividades en esta fase han sido sobre realizar un aprendizaje de diferentes tecnologías, lenguajes, frameworks, etc, en las cuales se han tenido una mayor flexibilidad para ajustarse al plan establecido.

En la tabla 2.49 se muestran los datos recogidos de los tiempos empleados en realizar las actividades de la etapa de elaboración.

Actividad	**Duración estimada (días)**	**Duración real (días)**	**Diferencia en días**
Id:01 Aprendizaje de PHP	5	5	0
Id:02 Aprendizaje del framework de Codeigniter	5	5	0
Id:03 Aprendizaje del framework de JQuery	5	5	0
Id:04 Aprendizaje del framework de Materialize	5	5	0
Id:05 Aprendizaje de Adobe Dream-Weaver cs6	4	5	1
Id:06 Definición de actividades	3	3	0
Id:07 Definición de riesgos	1	1	0
Id:08 Calendarización	2	1	-1
Id:09 Control de versiones	1	1	0
Suma en días	31	31	0

Tabla 2.48: Fase de inicio con tiempos estimados y reales

Actividad	Duración estimada (días)	Duración real(días)	Diferencia (días)
Id:013 Elicitar requisitos	7	10	3
Id:14 Definir casos de uso	7	7	0
Id:15 Máquinas de estado	4	1	-3
Id:16 Modelo de dominio	7	5	-2
Id:17 Diagramas de secuencia	7	5	-2
Id:18 Modelo de despliegue	1	1	0
Id:19 Diseñar estructura de BBDD	10	13	3
Id:20 Diseño del esquema relacional	3	5	2
Id:21 Definir script de la BBDD	2	6	4
Id:22 Test del script de la BBDD	2	1	-1
Id:23 Diseño de la interfaz de usuario	4	6	2
Id:24 Diseño arquitectura del Sistema	10	8	-2
Suma en días	64	68	4

Tabla 2.49: Fase de elaboración con tiempos estimados y reales

Con los tiempos que podemos ver en la tabla 2.49 se llega a la conclusión de que se estimó en la mayoría de las tareas de una forma no adecuada ya bien sea por exceso o por defecto. La duración real de la etapa de elaboración ha sido de 68 días, 4 días más de lo estimado. Se han realizado un total de 2 iteraciones.

Actividad	Duración estimada (días)	Duración real(días)	Diferencia (días)
Id:27 Instalación LAMP	1	1	0
Id:28 Preparación del entorno	2	2	0
Id:29 Implementación de las vistas	10	12	2
Id:30 Implementación de la capa de modelos y controladores	30	40	30
Id:31 Diseño responsive	2	3	1
Id:32 Revisar compatibilidad con los navegadores	1	1	0
Suma en días	46	59	13

Tabla 2.50: Fase de construcción con tiempos estimados y reales

Actividad	Duración estimada (días)	Duración real (días)	Diferencia en días
Id:35 Realización de las pruebas finales	7	7	0
Id:36 Manual de usuario	7	7	0
Id:37 Manual de instalación	3	3	0
Suma en días	17	17	0

Tabla 2.51: Fase de transición con tiempos estimados y reales

La estimación en la fase de construcción (ver figura 2.50) fue algo menos acertada que en las anteriores. El principal factor es la falta de experiencia a la hora de elaborar proyectos en PHP, así como la complejidad encontrada para desarrollar ciertas funcionalidades del sistema. El total de días empleados de más para terminar la fase de construcción ha sido de 18 días. Se han empleado un total de 3 iteraciones en esta fase para poder completarla satisfactoriamente.

Por último, en la fase de transición, la estimación en la segunda planificación del proyecto se ajustó al cien por cien a la duración real de las tareas, por lo que el retraso final, sumando la diferencia de días entre la duración real y estimada sería de un total de 17 días con respecto a lo planeado. Se ha empleado una iteración sobre esta fase.

Como conclusión del seguimiento del proyecto, se ha comprobado que planificar es difícil. Se tenía una holgura muy grande para poder realizar el proyecto, ya que la fecha de presentación límite en la Escuela de Ingeniería Informática es en el mes de julio.

Capítulo 3

Tecnologías utilizadas

En esta sección se detallará las tecnologías utilizadas para el desarrollo del proyecto, explicando en detalle las ventajas que justifican su elección en este proyecto.

3.1. Estado del arte

En la actualidad, no existe una herramienta específica para desempeñar las tareas mostradas en los objetivos de la aplicación, aunque existen aplicaciones en el mercado que utilizan las diferentes Federaciones de Judo a nivel nacional. Una aplicación que se lleva usando, y en cierta medida realiza la funcionalidad de llevar las puntuaciones de los combates sería ***Judo tournament software*** [5], en la figura 3.1. Este software es completamente gratuito y está disponible para cualquier Federación que quiera usarlo para sus torneos.

Figura 3.1: Cronometrador de la Federación Internacional de Judo

Entre otras funcionalidades, esta aplicación lo que permite es:

- Crear combates de manera individual o grupal.
- Tiempos de combate programables en función de la normativa vigente.
- Configuración de las puntuaciones.
- Guardar los resultados de los diferentes combates que se hayan creado en ese cronometrador
- Introducir de forma amigable, por medio de hardware propio las puntuaciones de cada combate, tal y como se muestra en la figura 3.2

Figura 3.2: Botones del cronometrador para introducir puntuaciones

Otra de las aplicaciones, esta vez web, que podría aportar la funcionalidad requerida, no toda, sería JudoInside[6], en la cual podemos buscar los resultados de los competidores de alto nivel, luchadores que han sacado resultados internacionalmente. El aspecto de la página web es el mostrado en la figura 3.3. En este formulario podemos introducir diferentes parámetros para que nos devuelva competidores con resultados.

Figura 3.3: Buscar resultados de competidores en Judo Inside

Una herramienta muy parecida a JudoInside, sería la que tiene la Federación Internacional de Judo [7]. Las funcionalidades que realizan ambas son prácticamente idénticas:

- Búsqueda de competidores por grupo de edad.
- Búsqueda de competidores por peso dentro de un grupo de edad.
- Búsqueda de competidores por país.
- En ciertos casos, hay vídeo de cada combate señalando en que momento de este se realizan las puntuaciones, como se muestra en la figura 3.4.

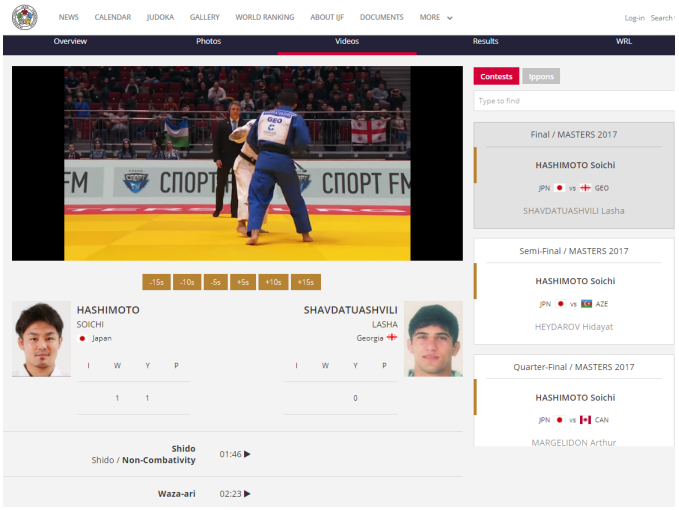


Figura 3.4: Video con las puntuaciones de un competidor de alto nivel

En todas estas aplicaciones, excepto la primera, la gestión de competidores las llevan los administradores de las respectivas páginas, por lo que si se quisiera llevar las mismas funcionalidades a terrenos más pequeños como podría ser la Federación de Castilla y León de Judo, no se podría. Sin embargo, han ayudado en cierta medida a ver funcionalidades que se podrían agrupar en una misma aplicación, y cuales se podrían añadir como funcionalidad nueva.

3.2. Tecnologías web por parte del cliente

3.2.1. Navegador web

Un navegador web, también conocido como buscador, es una aplicación software que sirve para recuperar y presentar información en la WWW (World Wide Web). El recurso que se quiere obtener y mostrar es identificado por medio de la URI (Unified Resource Identifier).

En un primer momento, los navegadores eran una herramienta relativamente sencilla utilizada para la visualización de documentos conocidos como hipertexto. Estos documentos además de presentar la información, pueden contener referencias o hipervínculos a otros documentos. Más adelante, los buscadores fueron creciendo en cuanto a funcionalidad y ya no solo mostraban documentos estáticos, si no que permitía al usuario interactuar con la aplicación y obtener respuestas por parte del servidor. Gracias a que empezaran a ser dinámicos, se extendió a una nueva forma de desarrollar aplicaciones. Hoy en día, la cantidad de dispositivos de diferentes tipos y tamaños donde poder mostrar la información al usuario ha hecho que el desarrollo de aplicaciones web, gracias a su diseño *responsive* que se adapta a todo tipo de pantallas, sea de los más utilizados en este momento, junto con las aplicaciones híbridas. Alguna de las ventajas que ofrece el desarrollo de una aplicación web son las siguiente.

- Un elevado porcentaje de usuarios está familiarizado con el uso del navegador a lo largo del día, ya bien sea en el trabajo o en tiempo de ocio, por lo que reduce el tiempo de aprendizaje del uso de la aplicación ya que tiene una interfaz conocida.

- Otra de las ventajas que la hace fuerte frente a otras alternativas es que sólo hay que desplegarlo una vez en el lado del servidor para poder empezar a usarlo. En el lado del cliente, en la mayoría de ocasiones no es necesario instalar nada (a no ser que la aplicación requiera algún tipo de complemento para poder funcionar correctamente).

- Multiplataforma, en cualquier tipo de Sistema Operativo que se pueda instalar un navegador, se podrá acceder a la información que ofrece la página web

Por último, en la siguiente gráfica, figura 3.5 se puede observar cuáles son los navegadores más utilizados actualmente. Es importante tener constancia de estos datos ya que ciertos navegadores que

están en desuso puede que dejen de implementar ciertas funcionalidades del estándar HTML4 y no funcione la aplicación.

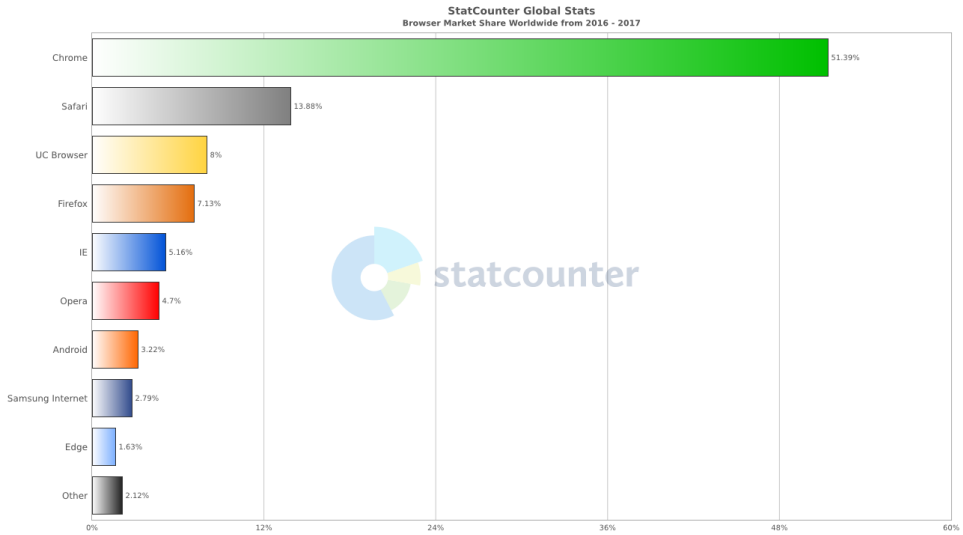


Figura 3.5: Fuente : StatCounter, Global Stats

Según fuentes de StatCounter[8], casi el 52 % del mercado de los navegadores lo ocupa Chrome, por lo que a la hora de realizar la aplicación lo tendremos más en cuenta, aunque siempre comprobando que funcione en el resto de navegadores que cumplen como mínimo el estándar HTML 4.

3.2.2. HyperText Markup Language

HTML (HyperText Markup Language) es el lenguaje de marcado empleado para la creación de documentos de hipertexto que puedan ser mostrados por el navegador. A continuación se muestra un pequeño fragmento de código HTML.

```
<body>
<h4> Lenguajes de programacion mas populares </h4>
    <ul>
        <li> PHP </li>
        <li> Ruby </li>
        <li> Java </li>
        <li> Python </li>
    </ul>
</body>
```

En el ejemplo anterior, la primera etiqueta que nos encontramos es la de *body*, ahí es donde debemos de mostrar la información principal de la página. La siguiente etiqueta se utiliza para poner el título de una sección. Por último, tenemos una lista ordenada dónde se muestran cuatro elementos.

En un comienzo, HTML era un lenguaje relativamente sencillo, que fue creciendo en complejidad a medida que los navegadores incorporaban nuevas funcionalidades y etiquetas para satisfacer las necesidades de los desarrolladores. Esto resultó en una incompatibilidad de aplicaciones en ciertos navegadores, en los que la aplicación utilizara una funcionalidad que no hubiera sido implementada en estos.

Para resolver este tipo de incompatibilidades, la organización W3C creo un estándar de las características que tenía que tener el lenguaje HTML. Es responsabilidad de los diseñadores de los navegadores el ceñirse al estándar para que la mayoría que aplicaciones puedan correr en su navegador. Actualmente, existe un nuevo estándar HTML con la versión 5.0, con nuevas etiquetas y funcionalidades.

Para implementar todas las vistas de la aplicación se va a utilizar como mínimo HTML 4 y en ciertos casos, tales como validaciones de campos de ciertos formularios, HTML5, aunque también se va a validar en el lado del servidor.

3.2.3. Cascading Style Sheets (CSS)

Cascading Style Sheets fue desarrollada en 1997 y sirvió para ayudar a los desarrolladores web dar un aspecto visual a su aplicación. La idea principal es separar el contenido de la aplicación de la estructura (aspecto visual) algo que no era posible hasta entonces. La separación de la estructura y estilos permite a HTML añadir mayor rendimiento a sus funciones principales, lenguaje de marcado, sin tener que preocuparse sobre el diseño de la interfaz, más comúnmente conocido como *Look and feel* de la página.

Otra de las ventajas de usar hojas de estilo es que varios ficheros HTML pueden usarlas manteniendo la consistencia y coherencia de la aplicación, sin la incomodidad de tener que definir el estilo de la página (colores, fuentes, posicionamiento) en cada una de las páginas individualmente, lo que provocaba una cantidad elevada de fallos y de inconsistencias. En el proyecto que se va a abordar, se usarán exclusivamente reglas CSS para la presentación de documentos HTML generados por la aplicación.

Existen tres formas de especificar los estilos CSS dentro de un documento:

1. Una hoja externa del documento

```
<link rel="ruta_al_fichero_CSS.CSS">
```

2. Especificando dentro de la sección *head* del documento

```
<style type="text/CSS"> // CSS </style>
```

3. Dentro de cada etiqueta html individual

```
<element style="reglas_CSS"> </element>
```

Estas tres formas de establecer estilos con lenguaje CSS, se pueden combinar para redefinir estilos en ciertas situaciones en que sea conveniente una definición más especifica, estableciéndose una relación de prioridad entre ellas para cuando exista un conflicto sobre qué estilo aplicar. La resolución de conflictos la ha definido el estándar CSS de la siguiente manera: prevalece la definición de estilo más cercano del lugar del documento en el que se quiere aplicar.

Uso de las hojas de estilo

Una regla CSS consiste en un selector, que define el elemento o elementos afectados y una lista de declaraciones de estilo para ese selector, formado por una clave y un valor. La clave es el nombre de la propiedad que se quiere definir y el valor es lo que se le quiere asignar a esa propiedad.

```
selector {
        propiedad : valor;
}
```

El selector puede ser simplemente un elemento HTML, o tomar formas más complicadas como por ejemplo en qué contexto está el elemento el cual queremos dar un diseño específico. Es el caso que se muestra a continuación, donde al elemento **p** solo se le aplicará el estilo cuando esté dentro de un **div**.

```
div p {
        clave:valor;
}
```

Una de las cosas más importantes de CSS es que se pueden heredar estilos y redefinirlos en función de cómo se quiera diseñar la aplicación.

3.2.4. Materialize

Materialize[1] es un framework CSS y JavaScript formado por un conjunto de componentes de interfaz de usuario. Está basado en el principio de *Material Design* de Google por lo que utiliza elementos atractivos, coherentes y funcionales que hacen que la experiencia de usuario con la aplicación sea más cómoda. También se ajusta al modelo responsive que hace que cualquier página que sea desarrollada usando este framework se pueda ver en dispositivos de diferente tamaño. Las características principales de este framework son las siguientes:

- Diseño responsive gracias a su sistema de *grid layout*.
- Incluye nuevas versiones de los controles de interfaz de usuario, como por ejemplo botones, selectores, campos de verificación, etc.
- Uso totalmente libre del framework.

Sistema grid layout

El framework de *Materilize* utiliza un sistema de 12 columnas, cada una de las cuales tiene el mismo tamaño. Las columnas se encuentran contenidas dentro de una fila.

```
1  <div class="row">
2     <div class="col s6">This div is 12-columns wide on all screen sizes</div>
3     <div class="col s6">6-columns (one-half)</div>
4     <div class="col s6">6-columns (one-half)</div>
5   </div>
```

La interpretación del código anterior sería una fila en la cual habría tres columnas del mismo tamaño **s6**, con un total de seis espacios cada una. Frente al sistema de *grid layout*, tenemos *flex*, pero nosotros hemos optado por utilizar la primera forma.

Para los diferentes tamaños de las pantallas de dispositivos tenemos la siguiente tabla de información para diseñar nuestra página.

	Móviles (<= 600px)	**Tablets(>600 px)**	**Escritorio (>992 px)**	**Dispositivos grandes (>1200 px)**
Prefijo de clase	.s	.m	.l	.xl
Tamaño del contenedor	90 %	85 %	70 %	70 %
Número de columnas	12	12	12	12

Tabla 3.1: Clases de pantalla y su correspondencia con el framework

3.2.5. JavaScript

JavaScript es un lenguaje de scripting orientado a objetos que fue creado entre otras cosas para dar dinamismo a documentos HTML. Las acciones que ejecutan código JavaScript se ejecutan en el lado del cliente, por lo que no requiere de interacción con el servidor. Con JavaScript se tiene el control de los eventos que se producen en el documento HTML, como por ejemplo un click de ratón sobre un enlace o arrastrar cierto elemento a otro lado. Todo esto se puede realizar fácilmente con JavaScript.

La gran ventaja que supone que JavaScript sea un lenguaje orientado a objetos es que todos los elementos HTML tienen representación como objetos, con unas propiedades y características. Mediante JavaScript se puede leer el valor de estas propiedades y modificarlo como queramos. En este proyecto, se usa JavaScript junto el framework de JQuery para dar dinamismo a la interfaz y un aspecto visual más elegante a la aplicación. También se utilizará el framework de Ajax para realizar todas las peticiones asíncronas que se nos puedan presentar por lado del cliente hacia el servidor.

3.2.6. JQuery

JQuery[3] es una librería de JavaScript, que facilita en gran medida la programación del lado del cliente. Es de código abierto y de uso libre bajo la licencia del MIT. La sintaxis de JQuery esta diseñada para hacer más fácil navegar por el documento, seleccionar elementos DOM (Document Object Model), crear animaciones, manejar eventos y desarrollar llamadas asíncronas al servidor por medio de la tecnología Ajax. JQuery, en realidad, es una librería de manipulación de elementos DOM. DOM es una estructura de representación de elementos de una página web. La librería facilita las sintaxis para buscar, seleccionar, y manipular todos estos elementos.

Los principios del desarrollo con JQuery son:

- **Eliminar las incompatibilidades entre buscadores**. Muchas funcionalidades de JavaScript que funcionan en unos ciertos buscadores, no funcionan en otros. JQuery lo que hace es manejar este tipo de inconsistencias y proporciona una interfaz para que este problema no ocurra.
- **Extensibilidad**. Poder crear nuevos eventos, elementos, y métodos.
- **Comunidad activa**. Facilita que cualquier bug que pueda aparecer se resuelve rápidamente. También hay una gran cantidad de foros donde poder buscar soluciones a un problema.

Métodos para poder incluír la librería *JQuery* en proyectos software:
Se puede hacer por medio de una copia local del fichero .js

```
<script src="jquery.js"></script>
```

o haciendo una referencia al servidor remoto donde esté alojada la librería.

```
<script
  src="https://code.jquery.com/jquery-3.2.1.min.js"
</script>
```

3.3. Tecnologías web por parte del servidor

A continuación, se mostrará información de las tecnologías utilizadas para la parte del *backend*.

3.3.1. Hypertext Preprocessor (PHP)

PHP[9] es un lenguaje de servidor usado principalmente para el desarrollo web, pero también utilizado para el desarrollo de aplicaciones de propósito general. El código PHP puede ser embebido entre código HTML o HTML5. Es un lenguaje interpretado por un intérprete de PHP implementado como módulo en el servidor web o como un CGI (Common Gateway Interface). Existen múltiples versiones de PHP, las cuales han de ser revisadas cuando se utiliza cualquier tipo de framework que dependa de PHP, como es por ejemplo nuestro caso, ya que hacemos uso de Codeigniter, un framework que necesita como mínimo usar la versión 5.3 de PHP para que funcionen correctamente sus funcionalidades. Las Ventajas del uso de PHP como lenguaje de backend son:

- **Código abierto**. Está desarrollado y mantenido por un grupo de desarrolladores, que dan un soporte a la comunidad de desarrolladores.
- **Estabilidad**. Gracias a que lo mantiene un grupo de desarrolladores, cuando aparece un cierto bug es solucionado rápidamente.
- **Portabilidad**. Puede ejecutarse en múltiples plataformas, incluyendo Windows, Linux y Mac.
- **API potente**. Proporciona una gran cantidad de funcionalidades, como el manejo de acceso a bases de datos, crear PDFs, etc.
- **Frameworks basados en PHP**. Existen muchos frameworks basado en PHP, tales como Simphony, Laravel, Codeigniter que facilitan enormemente las tareas de desarrollo del lado de servidor.

Todas estas ventajas, hacen que PHP sea una buena elección como lenguaje de servidor.

3.3.2. Codeigniter

Codeigniter[2] es un framework de PHP. El uso de este, te permite desarrollar proyectos mucho mas rápidamente que si se escribe código desde cero. Proporciona una rica variedad de librerías que facilitan las tareas que se suelen realizar normalmente. Muchos proyectos de éxito han utilizado Codeigniter, tales como Casio Computers, FeedCump, The mail and Guardian, Buffer, Nissan, Bonfire, entre otros. Sus ventajas son:

- **Fácil de aprender**. Si se tiene cierto conocimiento del lenguaje PHP y del patrón modelo vista controlador que utiliza Codeigniter, la curva de aprendizaje es mucho más baja que en otros frameworks.
- Tiene un excelente **rendimiento**.
- Tiene una **amplia documentación**.
- Es de uso libre, bajo la licencia del MIT.
- Es bastante **menos pesado** que otros frameworks. Una instalación limpia de Codeigniter tiene lo básico para poder empezar, pero se pueden cargar librerías dinámicamente desde los ficheros de configuración del framework.
- **Uso del patrón MVC**, modelo vista controlador, que permite separar la lógica de la presentación.

Modelo-Vista-Controlador (MVC)

Codeigniter está basado en el patrón MVC. Permite separar la lógica de la aplicación de la presentación. En la práctica, esto permite que las páginas web tengan la cantidad mínima de scripts. El Modelo representa la estructura de los datos que se están modelando. Típicamente, estas clases contendrán las funciones que permitirán acceder, insertar, eliminar y actualizar la base de datos de la aplicación. En la Vista estará la información que se le quiere presentar al usuario. Codeigniter permite que una página este formado por fragmentos de dos o más páginas, lo cual aporta una gran extendibilidad y reutilización de las vistas. Un caso típico en el que se evita implementar la misma vista varias veces, son las cabeceras y el pie de página. En la cabecera o *header* es donde suele estar todo lo relacionado con la importación de ficheros CSS e imagenes; y en el pie de página o *footer* es donde se importan los ficheros JavaScript. Tener solamente un fichero para cada uno, y reutilizarlo cuando sea necesario, hace que nuestro proyecto sea mucho más coherente y limpio. Por último, la capa del Controlador hace de intermediaria entre la Vista y el Modelo y otro tipo de recursos necesarios para poder procesar las peticiones HTTP y generar la página web. La figura 3.6 muestra el patrón explicado.

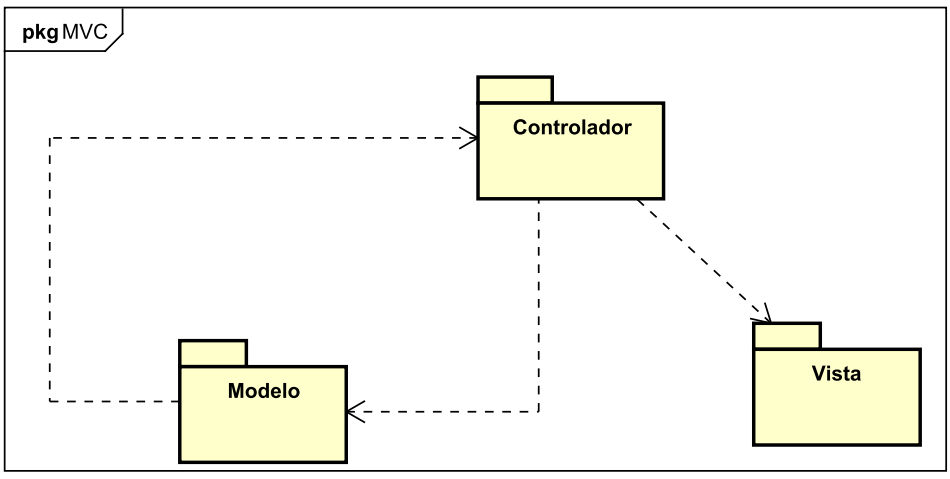


Figura 3.6: Patrón MVC utilizado por Codeigniter

Metas de diseño y arquitectura
Una de las características más importantes de Codeigniter es que busca el máximo rendimiento posible. La incorporación de librerías es dinámica y es posible instanciarlas en caso de que se vayan a utilizar, por lo que ocupa menos memoria e incrementa la velocidad de respuesta del servidor. Los componentes tienen bajo acoplamiento; como los componentes dependen menos con otros, son mucho mas reutilizables y flexibles para los sistemas que se desarrollen en el futuro.

Inconvenientes vistos en Codeigniter
El principal inconveniente tiene relación con la seguridad de los datos, y es que los ficheros de configuración que se alojan en el servidor están en texto plano. Es decir, datos como por ejemplo las credenciales para acceder a la base de datos, APIS externas, etc. Podrían ser accesibles en caso de un ataque desde el exterior. Todos los frameworks investigados tienen las mismas características, por lo que una posible solución a este problema sería encriptar esos ficheros por cuenta propia.

3.3.3. MySQL

Un Sistema Gestor de Base de Datos es el conjunto de programas que permiten el almacenamiento, modificación y extracción de la información de una Base de Datos, además de ofrecer herramientas para realizar las operaciones CRUD(eliminar, actualizar, obtener, añadir) entre otras. Proporciona métodos para garantizar la integridad de los datos, mantener cierto nivel de seguridad de acceso a estos y poder recuperar información que se haya podido corromper. A los datos que se guardan en la Base de Datos, generalmente se acceden mediante lenguajes de consulta tales como SQL. Para el desarrollo de este proyecto, se ha optado por utilizar MySQL[10] como gestor.

MySQL es un gestor de base de datos relacional, originalmente bajo licencia GNU, cuyo código es de libre acceso. Es muy popular en servidores web, debido a que sus funcionalidades están muy optimizadas. También forma parte de la arquitectura de Linux, Apache, MySQL, PHP (LAMP), una combinación de plataformas que son utilizadas para dar soporte a aplicaciones web avanzadas. En concreto, este gestor está optimizado para la producción de aplicaciones web.
Ventajas del uso de MySQL

- **Soporte de transacciones**. Es uno de los más robustos que podemos encontrar. Las transacciones sirven para asegurar la consistencia de la Base de Datos, asegurando que un conjunto de sentencias se ejecuten correctamente o no se ejecuten proporcionando el fallo correspondiente.

- **Escalabilidad**. Propiedad deseable de un SGBD, la cual indica la habilidad para responder y adaptarse a cambios de embergadura del proyecto, como por ejemplo añadiendo mayor número de usuarios, manteniendo la calidad del servicio igual que antes. Muchas de las grandes compañías que usan MySQL, tales como Facebook, Twitter, entre otros, esperan que a medida que vaya creciendo el número de usuarios el gestor responda de la misma manera.

  Para conocer lo bien que escala MySQL se ha obtenido información de pruebas realizados por MySQL[11] de su versión empresarial, la cual utiliza un sistema de hilos que mejoran en ciertos casos hasta 60 veces como en la gráfica 3.7

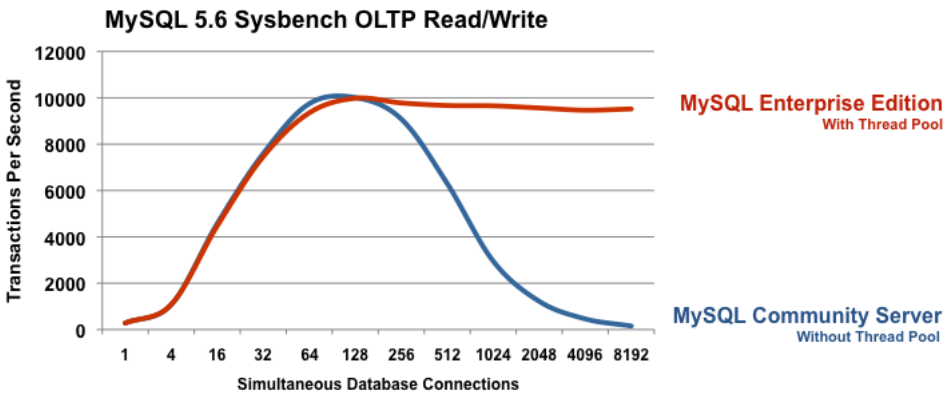


Figura 3.7: MySQL 5.6 Sysbecnh OLTP Escritura/Lectura

El sistema de hilos proporciona una alta escalabilidad, reduciendo la sobrecarga en la gestión de conexiones de clientes y ejecución de hilos. Podemos observar en la gráfica 3.7 que el número de transacciones se mantiene constante a partir de 128 conexiones simultáneas, lo que aporta un gran rendimiento con respecto a MySQL sin el *thread pool*. En cuanto a esta aplicación se refiere (JudoCup), el número de usuarios que van a utilizarla simultáneamente es en un primer momento ínfimo por lo que no sería necesario contratar un servicio que supusiera mayor coste, pero es importante tener estos datos en cuenta por si la aplicación en un futuro tuviera mayor número de usuarios.

- **Facilidad de instalación y uso**. En menos de un día podemos tener instalado y corriendo MySQL en nuestra máquina sin mucho problema, sin importar la plataforma, Windows, Linux, Macintosh.

Capítulo 4

Análisis

4.1. Introducción

Como se dijo en la introducción, el objetivo del presente trabajo es construir una aplicación web que sirva para manejar todo lo que concierne a las competiciones de Judo, desde la gestión de los árbitros colegiados, competidores, equipos de judo con su delegado técnico, etc.

En todo proceso software se necesita la definición de los requisitos, funciones, servicios que va a tener el proyecto, así como las restricciones que va a tener el mismo. En este apartado, se estudiarán a fondo estos aspectos.

En esta etapa de desarrollo del producto es importante generar un modelo de sistema basado en lo que se necesita hacer, definiendo muy bien el producto. Cómo se va a realizar el diseño, se mostrará en el siguiente apartado.

4.2. Entorno

Para entender bien cómo funcionan las competiciones de Judo y todo lo que genera en su entorno se mantuvieron ciertas reuniones con el delegado de Castilla y León. El objetivo era saber cómo actuaban en la Federación castellano-leonesa cuando tenían que gestionar una competición.

4.2.1. Conceptos previos

A continuación, se describirán conceptos que son importantes para entender el entorno en el que se está trabajando:

- **Competición**. Una competición de judo acoge los combates entre los competidores. Está formada por tatamis que es donde se realizan los encuentros y cada tatami tendrá un marcador (cronómetro) donde se visualizan las puntuaciones de los competidores. Las competiciones las crea el delegado de la Federación correspondiente y a la hora de hacerlo indica: nombre de la competición, fecha de realización de la competición, nombre del pabellón, país, provincia donde se va a realizar. Además definirá el numero de tatamis (1..6), elegirá el tipo de competición(regional, liga, nacional, europeo, internacional) y el grupo de edad (benjamín, alevín, infantil, cadete, junior y senior) de los competidores que participan en la competición.
- **Árbitro**. Los árbitros son los que indican a los jueces crono las puntuaciones que se van marcando a lo largo del combate y es el juez-crono quien las anota en la aplicación. Los árbitros tienen diferentes categorías entre las que nos encontramos: juez-crono, autonómico, nacional, continental e internacional. En función de esa categoría podrá o no arbitrar en una competición. Los árbitros pertenecen a un club, Comunidad Autónoma y tienen una nacionalidad. Esto es importante diferenciarlo porque en función de estos campos puede o no arbitrar un combate.

    1. En caso de que sea un campeonato regional, puede arbitrar si y solo si ninguno de los dos competidores es de su club.

2. Si es un campeonato de tipo nacional, puede arbitrar si y solo si ninguno de los competidores es de su misma autonomía.
3. Si es un campeonato internacional, puede arbitrar si y solo si ninguno de los dos competidores es de su misma nacionalidad.

- **Tatami**. Es donde se van a realizar los combates. En un tatami tiene que haber como mínimo un juez crono encargado de introducir las puntuaciones y uno o varios árbitros centrales encargados de indicar las puntuaciones de un combate. En un tatami se realizan combates de varias categorías, de peso, por ejemplo -90 kg y -100 kg, además, se pueden alternar combates de diferentes categorías. Puede existir el caso en el que un combate se traslade a otro tatami si la organización de la competición lo divide, es decir, aunque en un primer momento se fijen los combates en un tatami, se puede cambiar uno o varios de ellos a otro tatami.

- **Combate**. Está formado por dos competidores de la misma categoría de peso y categoría de edad. Uno de los competidores que forman el combate tiene que ir de blanco y el otro de color azul.

- **Cronómetro**. Es el dispositivo físico, en este caso una pantalla, donde muestran las puntuaciones. Está formado por la puntuación del competidor azul, la puntuación del competidor blanco, el tiempo del combate, el tiempo de inmovilización y la fase del combate.

- **Emparejamientos**. En función de los competidores que haya en una cierta categoría, se realiza un tipo de cuadro u otro. Hay tres opciones:

  1. De 1 a 4 competidores, se realiza una liga donde compiten todos contra todos.
  2. De 4 a 6 competidores, se realizan dos ligas, una de N competidores y otra de X - N competidores siendo X el total de competidores inscritos.
  3. Más de 6 competidores se realizan emparejamientos de dos en dos y en función del número de competidores inscritos, se empezará desde una fase u otra.

- **Puntuaciones**. Las normas de judo del año 2017 de judo indican cuatro tipos de puntuaciones en un combate: ippon, waza-ari, shido y hansoku-make.

  - **Ippon**. Equivale a 10 puntos. Cuando un competidor obtiene esta puntuación, finaliza el combate como ganador.
  - **Waza-ari**. Esta puntuación equivale a un punto. Cuando un competidor recibe dos calificaciones de waza-ari equivale a un ippon.
  - **Shido**. Es una falta, con 3 puntuaciones de este tipo el competidor queda eliminado del combate.
  - **Hansoku-make**. Eliminación directa, se produce cuando el competidor realiza alguna acción que va en contra del espíritu del Judo.

- **Juez-crono**. En las competiciones de judo es el encargado de introducir las puntuaciones que le indica el árbitro central del combate.

4.2.2. Tipos de consultas

En esta sección se listarán las consultas que la base de datos debe responder para obtener la funcionalidad deseada en la aplicación. Son consultas que se corresponden con las preguntas que habitualmente se hace en una competición sobre los datos disponibles.

- ¿Qué competidores hay en una cierta categoría?
- ¿Quién ganó una cierta categoría en un campeonato?
- ¿Cuales han sido los resultados de una categoría en un determinado campeonato?

- Dame la lista de combates de un determinado tatami.
- Dame la lista de competidores de una categoría de peso.
- Dame la lista de competidores de una categoría de peso y una categoría de edad.
- Dame los equipos de una competición.
- Dame los competidores de una competición.
- Listado de árbitros que pueden realizar su función en un campeonato de un cierto tipo.
- Dame el top 4 de los equipos que más medallas han sacado en una cierta competición.
- Listado de competidores con sus resultados entre dos fechas.

Estas preguntas nos ayudarán a concretar los requisitos de información y a crear el modelo de dominio. También se utilizarán en el diseño de la Base de Datos, el modelo de Entidad-Relación y posterior diseño del modelo relacional.

4.3. Funcionalidad requerida para la aplicación

Los requisitos que se van detallar en las siguientes secciones van a ayudar a entender cuáles son las necesidades exactas de los usuarios del sistema que se desarrollará.

4.3.1. Requisitos funcionales

Los requisitos funcionales (RF) describen el funcionamiento del sistema en detalle, cómo debe reaccionar ante ciertas entradas de datos y cómo debe comportarse ante situaciones particulares. A continuación se muestran los requisitos funcionales de este proyecto.

- **RF-01**:El Sistema permitirá al usuario identificarse.
- **RF-02**:El Sistema permitirá crear competiciones de judo.
- **RF-03**:El Sistema permitirá definir el número de tatamis asociados a una determinada competición.
- **RF-04**:El Sistema permitirá adjuntar un peso a un competidor.
- **RF-05**:El Sistema permitirá crear competidores.
- **RF-06**:El Sistema permitirá modificar los datos personales de un competidor.
- **RF-07**:El Sistema permitirá cambiar el peso de un competidor a otro de la misma competición.
- **RF-08**:El Sistema permitirá eliminar la o las competiciones que el usuario desee.
- **RF-09**:El Sistema permitirá inscribir a un árbitro en un tatami concreto de una competición.
- **RF-10**:El Sistema permitirá cambiar a un árbitro de un tatami a otro de la misma competición.
- **RF-11**:El Sistema permitirá recuperar la contraseña de acceso a la aplicación en caso de olvido.
- **RF-12**:El Sistema permitirá cambiar la contraseña de acceso a la aplicación a usuarios con el rol de Delegado.
- **RF-13**:El Sistema permitirá la creación de equipos de Judo.
- **RF-14**:El Sistema permitirá actualizar los datos de un equipo.

- **RF-15**:El Sistema permitirá obtener los árbitros registrados por el tipo de categoría arbitral que tengan en ese momento.
- **RF-16**:El Sistema permitirá mostrar los competidores de un determinado peso.
- **RF-17**:El Sistema permitirá mostrar los competidores de un determinado grupo de edad.
- **RF-18**:El Sistema permitirá la creación de árbitros.
- **RF-19**:El Sistema solo permitirá crear una competición del mismo grupo de edad en una fecha indicada.
- **RF-20**:El Sistema permitirá dar de baja de una competición a un determinado competidor inscrito ya.
- **RF-21**:El Sistema permitirá la creación de combates entre competidores del mismo peso dentro de un mismo campeonato.
- **RF-22**:El Sistema permitirá introducir puntuaciones en un determinado combate.
- **RF-23**:El Sistema permitirá cambiar la contraseña de los cronometradores con el rol de Delegado.
- **RF-24**:El Sistema permitirá desinscribir a un determinado árbitro de una competición.

4.3.2. Requisitos no funcionales

En ingeniería del software un requisito no funcional (RNF), es un requisito que especifíca como debe de funcionar el Sistema, sus características. Define las propiedades emergentes, tales como el tiempo de respuesta, necesidades de almacenamiento, fiabilidad, usabilidad, rendimiento. Estos requisitos hay que tenerlos muy en cuenta ya que a veces son más críticos que los requisitos funcionales. Existen tres tipos. En primer lugar, los **requisitos del producto**, que especifican el comportamiento, tales como la velocidad de ejecución, memoria requerida. A continuación, **requisitos de la organización**, políticas y procedimientos de la empresa, procesos estándar utilizados, documentos a entregar. El último tipo de RNF, son los **requistios externos**, que son los que ofrecen factores externos al sistema y a su proceso, como por ejemplo, requisitos legales y éticos. A continuación se muestran los requisitos no funcionales de este proyecto.

- **RNF-01**: El Sistema deberá poder ejecutarse en todos los navegadores que soporten el estándar HTML 4.0 como mínimo.
- **RNF-02**: El Sistema solo ofrecerá los datos sensibles de competidores y árbitros al delegado de la aplicación, que debe estar identificado en la aplicación para poder verlo
- **RNF-03**: El Sistema deberá poder desplegarse en múltiples Sistemas Operativos.
- **RNF-04**: El Sistema deberá estar desarrollado utilizando patrones de diseño de interfaz de usuario para que la experiencia de usuario sea lo más agradable posible
- **RNF-05**: El Sistema deberá permitir a un usuario realizar todas las tareas con un entrenamiento máximo de tres horas.
- **RNF-06**: El Sistema deberá tener copias de seguridad de Base de Datos,
- **RNF-07**: El Sistema deberá tener un tiempo de respuesta aceptable para todas las peticiones.
- **RNF-08**: El Sistema cambiará el idioma de la aplicación en un tiempo razonable.
- **RNF-09**: El Sistema cumplirá con los aspectos de seguridad mínimos que establece la Asociación Española de Protección de Datos.

4.3.3. Requisitos de información

Tras la recopilación de toda la información en base a las entrevistas realizadas y experiencia propia, se han generado los siguientes requisitos de información:

RI-01 : El Sistema deberá almacenar la información de cada competidor

- Nombre
- Primer apellido
- Segundo apellido
- Documento Nacional de Identidad
- País
- Autonomía
- Provincia
- Email
- Peso

RI-02: El Sistema deberá almacenar la información de cada árbitro

- Nombre
- Primer apellido
- Segundo apellido
- Documento Nacional de Identidad
- País
- Autonomía
- Provincia
- Email
- Categoría arbitral

RI-03 : El Sistema deberá almacenar la información de cada competición

- Nombre
- Fecha
- País
- Autonomía
- Provincia
- Pabellón

RI-04: El Sistema deberá almacenar la información de cada tatami

- Número

RI-05: El Sistema deberá almacenar la información acerca de las puntuaciones de cada combate

- Ippon

- Waza-ari
- Shido
- Hansokumake

RI-06: El Sistema deberá almacenar la información acerca de los combates que se dan en un determinada competición

- Tiempo
- Fase
- Numero

RI-07: El Sistema deberá almacenar la información sobre las diferentes categorías arbitrales que se dan a día de hoy en las competiciones de Judo.

- Provincial
- Regional
- Nacional C
- Nacional B
- Nacional A
- Continental
- Internacional

RI-08: El Sistema deberá almacenar la información acerca de los equipos de Judo

- Nombre
- Nacionalidad
- Autonomía
- Provincia
- Nombre del director técnico

RI-09: El Sistema deberá almacenar la información sobre los diferentes tipos de competición que se pueden realizar en un campeonato de Judo

- Benjamin
- Alevín
- Infantil
- Cadete
- Junior
- Senior

RI-10: El Sistema deberá almacenar la información acerca de las inscripciones de competidores y árbitros en un determinado campeonato

- Fecha de inscripción
- Equipo
- Árbitro o competidor

4.4. Objetivos del Sistema

Cuando el sistema esté en producción, los siguientes objetivos del Sistema han de estar completados satisfactoriamente:

- **O-01:** El Sistema deberá gestionar inscripciones de competidores en competiciones.
- **O-02:** El Sistema deberá gestionar los equipos de judo.
- **O-03:** El Sistema deberá gestionar árbitros y competidores.
- **O-04:** El Sistema deberá guardar resultados de combates y puntuaciones en una determinada competición.
- **O-05:** El Sistema deberá gestionar cronometradores encargados de introducir las puntuaciones.
- **O-06:** El Sistema deberá gestionar inscripciones de árbitros en competiciones.
- **O-07:** El Sistema podrá crear combates entre los competidores del mismo peso mediante. diferentes algoritmos de emparejamiento.

4.5. Roles

En el documento de requisitos funcionales, se ha indicado que existen dos tipos de usuario en este momento, el usuario con rol de Delegado y el usuario con rol de Cronometrador.

El usuario con rol de **Delegado**, representa a la persona que actualmente realiza las operaciones de gestión de campeonatos de Judo. Entre otras cosas, será el encargado de registrar competidores, árbitros, crear competiciones, etc. Por otro lado, el usuario con rol de **Cronometrador**, es el encargado de introducir las puntuaciones de los combates del tatami que tenga asignado en una competición.

4.6. Casos de uso

Un caso de uso es un conjunto de escenarios (secuencia de acciones realizadas por el Sistema con la interacción del actor correspondiente), relacionados por el resultado que el actor espera obtener.

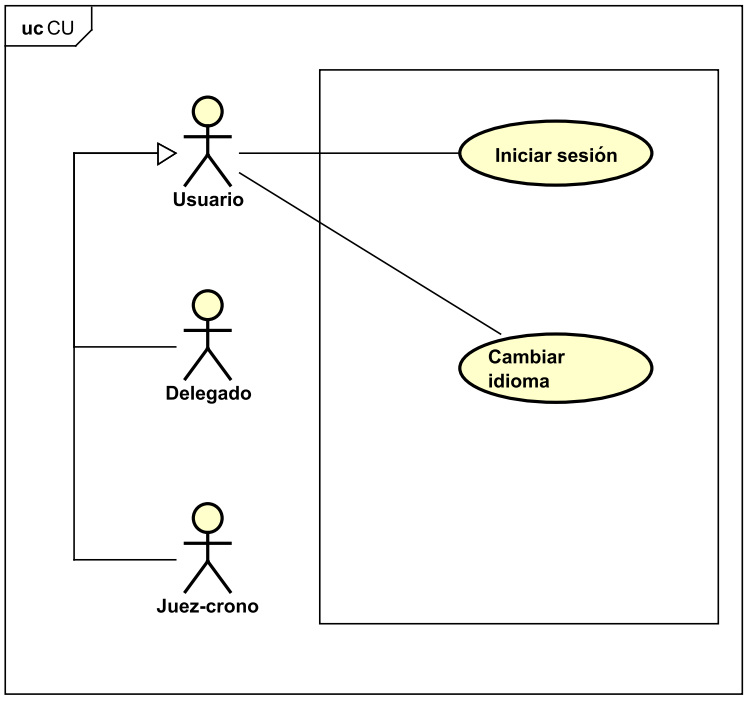


Figura 4.1: Diagrama de casos de uso del actor Usuario

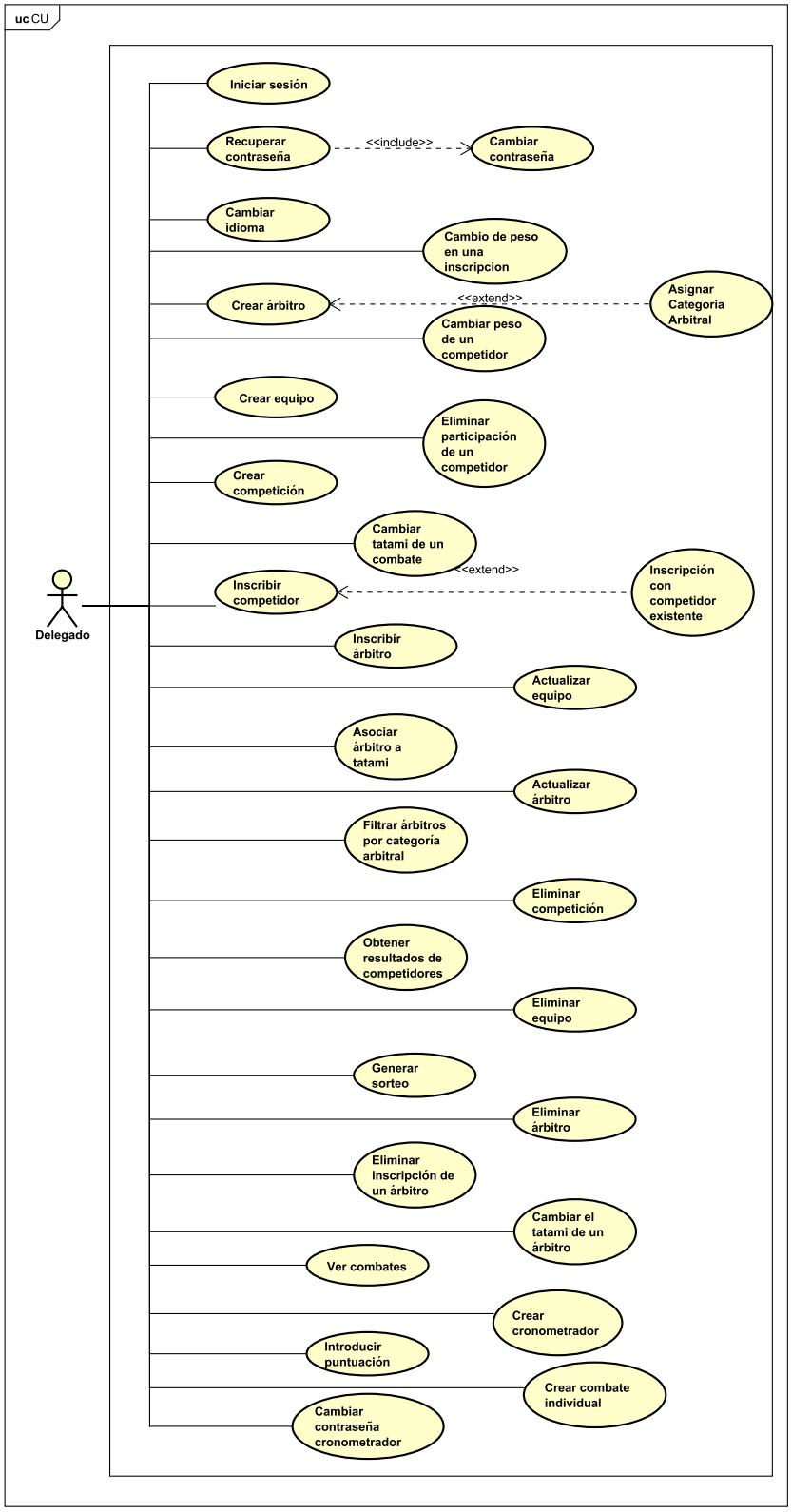


Figura 4.2: Diagrama de casos de uso del actor Delegado I

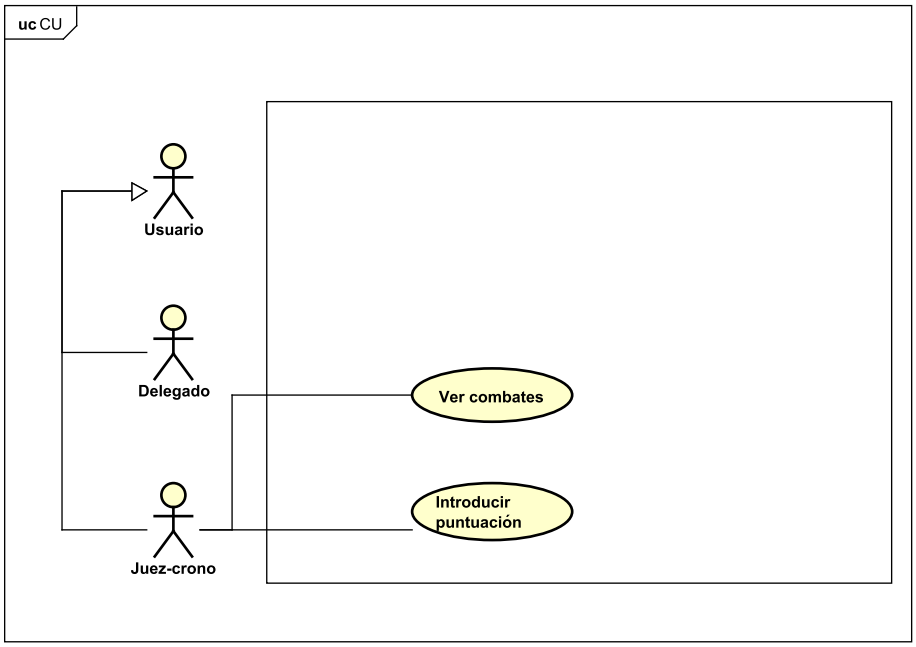


Figura 4.3: Diagrama de casos de uso del juez crono o cronometrador

En la figura 4.1 se muestra el diagrama de caso de uso correspondiente a la figura del actor **Usuario** el cual es la generalización de los dos actores principales que tiene la aplicación, el Delegado y el Cronometrador. En este diagrama se presentan las funcionalidades que realizan ambos roles. En el diagrama de la figura 4.2, se muestran los casos de uso del actor **Delegado** y por último, en la figura 4.3 tenemos los CU del actor Juez-Crono o cronometrador.

Cada uno de estos casos de uso que se observan en los diagramas, tiene una secuencia de acciones que determinan el escenario o los escenarios en los que se puede afrontar dicho caso de uso. Normalmente, cada caso de uso está formado por un escenario que denominamos como secuencia normal, en la que se termina con éxito lo que se quería realizar y uno o varios escenarios secundarios donde por algún motivo, no se termina de realizar el caso de uso. La descripción de cada uno de los CU se muestra a continuación, en las tablas 4.1 a 4.29:

CU-01	Crear árbitro	
Descripción	El sistema deberá permitir al actor Delegado crear un árbitro de Judo.	
Precondición	El actor Delegado deberá estar identificado en el sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona crear un nuevo árbitro de judo
	2	El Sistema muestra los datos a rellenar: Nombre, Primer apellido, Segundo apellido, DNI, Categoría arbitral, País, Comunidad Autónoma, Provincia y correo electrónico
	3	El actor Delegado introduce los datos solicitados anteriormente
	4	El Sistema comprueba los datos introducidos, y crea el nuevo árbitro
Excepciones	**Paso**	**Acción**
	3a	El actor Delegado cancela la operación de crear al árbitro y el caso de uso queda sin efecto.
	4a	Si alguno de los campos requeridos no es introducido por el actor Delegado, el Sistema muestra un mensaje de error y el caso de uso continúa en el paso 2.
	4b	El Sistema comprueba que existe una persona registrada con ese DNI y no tiene asignada una categoría arbitral, se lanza el caso de uso *Asignar categoría arbitral*
	4c	El Sistema comprueba que existe una persona con el DNI introducido y que tiene asignada una categoría arbitral, muestra un mensaje de error y el caso de uso queda si efecto.

Tabla 4.1: Caso de uso: Crear árbitro de Judo

CU-02	Crear equipo	
Descripción	El sistema deberá permitir al actor Delegado crear un equipo de Judo.	
Precondición	El actor Delegado deberá estar identificado en el sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona crear un nuevo equipo de judo
	2	El Sistema muestra los datos a rellenar Nombre del equipo, nombre del director técnico, país, comunidad autónoma y provincia
	3	El actor Delegado introduce los datos solicitados anteriormente
	4	El Sistema comprueba los datos introducidos, y crea el nuevo equipo
Excepciones	**Paso**	**Acción**
	3a	El actor Delegado cancela la operación de crear el equipo y el caso de uso queda sin efecto.
	4a	Si alguno de los campos requeridos no es introducido por el actor Delegado, el Sistema muestra un mensaje de error y el caso de uso continúa en el paso 2.
	4b	El Sistema comprueba que existe un equipo de judo con el mismo nombre, muestra un mensaje de error y caso de uso continúa en el paso 2

Tabla 4.2: Caso de uso: Crear equipo de Judo

CU-03	Crear competición	
Descripción	El sistema deberá permitir al actor Delegado crear una competición de Judo.	
Precondición	El actor Delegado deberá estar identificado en el sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor selecciona crear una nueva competición de judo
	2	El Sistema muestra los datos a rellenar: nombre de la competición, fecha de realización, país, comunidad autónoma, provincia, pabellón o centro deportivo, número de tatamis, categoría y grupo de edad.
	3	El actor introduce los datos solicitados anteriormente
	4	El Sistema comprueba los datos introducidos, y crea la competición
Excepciones	**Paso**	**Acción**
	3a	El actor Delegado cancela la operación de crear la competición y el caso de uso queda sin efecto.
	4a	Si alguno de los campos requeridos no es introducido por el actor Delegado, el Sistema muestra un mensaje de error y el caso de uso continúa en el paso 2.
	4b	El Sistema comprueba que existe una competición en esa fecha y con ese mismo grupo de edad introducido, muestra un mensaje de error y el caso de uso queda sin efecto.

Tabla 4.3: Caso de uso: Crear competición de Judo

CU-04	Inscribir a un competidor	
Descripción	El sistema deberá permitir al actor Delegado inscribir a un competidor en cierta competición de judo.	
Precondición	El actor Delegado deberá estar identificado en el sistema. Deberá existir una competición creada previamente con fecha posterior al día de la inscripción	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona mostrar las competiciones
	2	El Sistema muestra todas las competiciones creadas
	3	El actor Delegado selecciona inscribir competidores en una de las competiciones mostradas en el paso 2
	4	El Sistema muestra los pesos , tanto femeninos como masculinos de la competición seleccionada.
	5	El actor Delegado selecciona un peso donde inscribir al competidor
	6	El Sistema solicita la introducción del DNI del competidor
	7	El actor Delegado introduce el DNI
	8	El Sistema solicita la introducción de nombre, primer apellido, segundo apellido, país, comunidad autónoma, provincia, correo electrónico y nombre del equipo
	9	El actor Delegado introduce los datos solicitados anteriormente
	10	El Sistema comprueba los datos introducidos, e inscribe al competidor en el peso seleccionado
Excepciones	**Paso**	**Acción**
	3a,5a,7a,9a	El actor Delegado cancela la operación y el caso de uso queda sin efecto.
	8a	El Sistema comprueba que existe un competidor con ese DNI, en cuyo caso se realiza el caso de uso CU-05 *Inscripción con competidor existente.*
	10a	El Sistema comprueba que existe una inscripción (como árbitro o como competidor) en la competición seleccionada, muestra un mensaje de error y el caso de uso queda sin efecto.

Tabla 4.4: Caso de uso: Crear competición de Judo

CU-05	Inscripción con competidor existente	
Descripción	El sistema deberá permitir al actor Delegado inscribir a un competidor en un peso determinado.	
Precondición	El actor Delegado deberá estar identificado en el sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El Sistema solicita la introducción del nombre del equipo en el que se quiere inscribir al competidor.
	2	El actor Delegado introduce el nombre del equipo
	3	El Sistema comprueba los datos introducidos, e inscribe al competidor en el peso seleccionado.
Excepciones	**Paso**	**Acción**
	2a	El actor Delegado cancela la operación y el caso de uso queda sin efecto
	3a	El Sistema comprueba que existe una inscripción (como árbitro o como competidor) en la competición seleccionada, muestra un mensaje de error y el caso de uso queda sin efecto.

Tabla 4.5: Caso de uso: Inscripción con competidor existente

CU-06	Inscribir a un árbitro	
Descripción	El sistema deberá permitir al actor Delegado inscribir a un árbitro en una determinada competición.	
Precondición	El actor Delegado deberá estar identificado en el sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona mostrar las competiciones
	2	El Sistema muestra todas las competiciones creadas.
	3	El actor Delegado selecciona inscribir árbitros en una de las competiciones mostradas en el paso 2.
	4	El Sistema muestra los árbitros inscritos en la competición y los asignados a los tatamis.
	5	El actor Delegado selecciona inscribir a un nuevo árbitro.
	6	El Sistema muestra los árbitros disponibles para la competición
	7	El actor Delegado selecciona los árbitros que quiera inscribir, indicando para cada uno de ellos el equipo con el que van a participar.
	8	El Sistema comprueba los datos introducidos, inscribe a los árbitros y el caso de uso finaliza.
Excepciones	**Paso**	**Acción**
	3a,5a,7a	El actor Delegado cancela la operación de inscribir a un árbitro en una determinada competición y el caso de uso queda sin efecto.

Tabla 4.6: Caso de uso: Inscripción de un árbitro de Judo

CU-07	Asociar árbitro a tatami	
Descripción	El sistema deberá permitir al actor Delegado asociar a un árbitro previamente inscrito en un tatami de una competición determinada.	
Precondición	El actor Delegado deberá estar identificado en el sistema. Deberá existir al menos una inscripción de un árbitro en la competición seleccionada.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona mostrar las competiciones
	2	El Sistema muestra todas las competiciones creadas.
	3	El actor Delegado selecciona inscribir árbitros en una de las competiciones mostradas en el paso 2.
	4	El Sistema muestra los árbitros inscritos en la competición y los asignados a los tatamis.
	4	El actor Delegado selecciona asociar un árbitro inscrito a un tatami
	5	El Sistema pregunta en cuál de los tatamis se quiere realizar la asociación del árbitro.
	6	El actor Delegado selecciona uno de los tatamis disponibles
	7	El Sistema comprueba la asociación del árbitro al tatami indicado, y el caso de uso finaliza.
Excepciones	**Paso**	**Acción**
	3a,4a,6a	El actor Delegado cancela la operación de asociar un árbitro a un tatami y el caso de uso queda sin efecto

Tabla 4.7: Caso de uso: Asociar árbitro a tatami

CU-08	Recuperar contraseña	
Descripción	El sistema deberá permitir al actor Delegado recuperar la contraseña de acceso a la aplicación	
Precondición		
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona recuperar la contraseña del correo electrónico para poder acceder a la aplicación
	2	El Sistema solicita el correo electrónico
	3	El actor Delegado introduce el nombre de correo electrónico
	4	El Sistema comprueba que el correo es correcto y manda un email al introducido en el paso 3. Se lanza el caso de uso *Cambiar contraseña*
Excepciones	**Paso**	**Acción**
	3a	El actor Delegado cancela la operación y el caso de uso queda sin efecto.
	4a	El Sistema comprueba que el correo electrónico introducido en el paso 3 no es correcto, se muestra un mensaje de que se ha enviado un link al correo para cambiar la contraseña y el caso de uso continúa sin efecto.

Tabla 4.8: Caso de uso: Recuperar contraseña

CU-09	Cambiar contraseña	
Descripción	El sistema deberá permitir al actor Delegado recuperar la contraseña de acceso a la aplicación	
Precondición	El actor delegado previamente ha solicitado recuperar la contraseña, es decir, ha recibido un correo electrónico con el link para cambiarla	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona cambiar la contraseña
	2	El Sistema solicita la introducción de una nueva contraseña
	3	El actor Delegado introduce la nueva contraseña
	4	El Sistema solicita que se introduzca otra vez la nueva contraseña
	5	El actor Delegado introduce la nueva contraseña
	6	El Sistema comprueba que las contraseñas introducidas son iguales, y que ambas tienen la longitud mínima, y guarda la nueva contraseña asociado al correo electrónico con lo que el caso de uso finaliza
Excepciones	**Paso**	**Acción**
	3a,5b	El actor Delegado cancela la operación de cambiar la contraseña y el caso de uso queda sin efecto.
	6a	El Sistema comprueba que las contraseñas introducidas no son iguales, muestra un mensaje de error y el caso de uso continúa en el paso 4.
	6b	El Sistema verifica que no tiene la longitud mínima, muestra un mensaje de error y el caso de uso continúa en el paso 2.

Tabla 4.9: Caso de uso: Cambiar contraseña

CU-10	Iniciar sesión	
Descripción	El sistema deberá permitir al actor Usuario iniciar sesión en la aplicación	
Precondición		
Secuencia normal	**Paso**	**Acción**
	1	El Sistema solicita las credenciales de acceso a la aplicación, correo electrónico y contraseña
	2	El actor Usuario introduce el correo electrónico y contraseña
	3	El Sistema comprueba que los datos introducidos son correctos y el caso de uso finaliza
Excepciones	**Paso**	**Acción**
	2a	El actor Usuario cancela la operación y el caso de uso queda sin efecto.
	3a	El Sistema comprueba que el correo electrónico o la contraseña no son correctos, muestra un mensaje de error y el caso de uso continúa en el paso 1.

Tabla 4.10: Caso de uso: Iniciar sesión

CU-11	Obtener resultados de competidores	
Descripción	El sistema deberá permitir al actor Delegado ver los resultados de los competidores	
Precondición		
Secuencia normal	**Paso**	**Acción**
	1	El Sistema solicita el intervalo de fechas en el que se quieren obtener los resultados de los competidores, el grupo de edad y el peso
	2	El actor Delegado introduce las fechas, grupo de edad y peso
	3	El Sistema muestra los datos de los competidores con las especificaciones marcadas en el paso 2.
Excepciones	**Paso**	**Acción**
	2a	El actor Delegado cancela la operación de obtener los resultados de los competidores y el caso de uso queda sin efecto.
	3a	El Sistema comprueba que falta la introducción de alguna de las fechas, muestra un mensaje de error y el caso de uso continúa en el paso 1.
	3b	El Sistema comprueba que no existen competidores con las características especificadas en el paso 2, muestra un mensaje de advertencia y el caso de uso continúa en el paso 1.

Tabla 4.11: Caso de uso: Obtener resultados de competidores

CU-12	Filtrar árbitros por categoría arbitral	
Descripción	El sistema permitirá obtener árbitros por su categoría arbitral	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona ver los árbitros registrados.
	2	El Sistema muestra los árbitros registrados y solicita la entrada de una categoría arbitral
	3	El actor Delegado introduce la categoría arbitral
	4	El Sistema muestra los árbitros que tengan la categoría arbitral seleccionada en el paso 3.
Excepciones	**Paso**	**Acción**
	3a	El Sistema comprueba que no existen árbitros con la categoría arbitral seleccionada, muestra un mensaje de error y el caso de uso continúa en el paso 2.
	3b	El actor Delegado cancela la operación filtrar árbitros por categoría arbitral y el caso de uso queda sin efecto.

Tabla 4.12: Caso de uso: Filtrar árbitros por categoría arbitral

CU-13	Generar sorteo	
Descripción	El sistema generará un sorteo para el peso indicado	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado solicita ver los combates de un determinado peso.
	2	El Sistema muestra todas los combates creados y muestra los diferentes algoritmos de emparejamiento que se pueden aplicar
	3	El actor Delegado selecciona uno de ellos y da a generar sorteo
	4	El Sistema comprueba que hay dos o más competidores registrados en el peso, genera el sorteo y el caso de uso finaliza
Excepciones	**Paso**	**Acción**
	4a	El Sistema comprueba que hay menos de dos competidores inscritos en el peso en esa competición, muestra un mensaje de error y el caso de uso queda sin efecto.
	3a,5a	El actor Delegado cancela la operación de generar sorteo del peso y el caso de uso queda sin efecto.

Tabla 4.13: Caso de uso: Generar sorteo

CU-14	Actualizar equipo	
Descripción	El sistema permitirá al actor Delegado actualizar los valores de un determinado equipo de Judo.	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona ver los equipos de judo registrados hasta ese momento
	2	El Sistema muestra todos los equipos
	3	El actor Delegado selecciona actualizar uno de los equipos mostrados en el paso 2
	4	El Sistema muestra los campos que se quieren actualizar del equipo seleccionado, nombre del director técnico, país, comunidad autónoma y provincia
	5	El actor Delegado introduce los datos solicitados en el paso anterior
	6	El Sistema comprueba los datos introducidos y actualiza al equipo
Excepciones	**Paso**	**Acción**
	3a, 5a	El actor Delegado cancela la operación de actualizar al equipo y el caso de uso queda sin efecto

Tabla 4.14: Caso de uso: Actualizar equipo

CU-15	Actualizar árbitro	
Descripción	El sistema permitirá al actor Delegado actualizar los valores de un determinado árbitro de Judo.	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona ver los árbitros de judo registrados hasta ese momento.
	2	El Sistema muestra todos los árbitros.
	3	El actor Delegado selecciona actualizar uno de los árbitros mostrados en el paso 2.
	4	El Sistema muestra los campos que se quieren actualizar del equipo seleccionado, nombre, primer apellido,segundo apellido, categoría arbitral, país, comunidad autónoma, provincia, correo electrónico.
	5	El actor Delegado introduce los datos solicitados en el paso anterior
	4	El Sistema comprueba los datos introducidos y actualiza al árbitro.
Excepciones	**Paso**	**Acción**
	3a, 5a	El actor Delegado cancela la operación de actualizar al árbitro y el caso de uso queda sin efecto

Tabla 4.15: Caso de uso: Actualizar árbitro

CU-16	Eliminar árbitro	
Descripción	El sistema permitirá al actor Delegado eliminar a un árbitro de Judo previamente registrado.	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona ver los árbitros de judo registrados hasta ese momento.
	2	El Sistema muestra todos los árbitros.
	3	El actor Delegado selecciona eliminar un árbitro de los mostrados en el paso 2.
	4	El Sistema muestra un mensaje de confirmación antes de eliminar al árbitro.
	5	El actor Delegado confirma la eliminación del árbitro
	6	El Sistema elimina al árbitro seleccionado.
Excepciones	**Paso**	**Acción**
	3a,5a	El actor Delegado cancela la operación de eliminar al árbitro y el caso de uso queda sin efecto
	5b	El actor Delegado selecciona que no quiere eliminar al árbitro y el caso de uso queda sin efecto.

Tabla 4.16: Caso de uso: Eliminar árbitro

CU-17	Eliminar competición	
Descripción	El sistema permitirá al actor Delegado eliminar una competición de judo.	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona ver las competiciones creadas hasta ese momento.
	2	El Sistema muestra todas las competiciones.
	3	El actor Delegado selecciona eliminar una competición de las mostradas en el paso 2.
	4	El Sistema muestra un mensaje de confirmación antes de eliminar la competición.
	5	El actor Delegado confirma la eliminación del árbitro
	6	El Sistema elimina la competición, eliminando a su vez los árbitros y competidores inscritos, combates y resultados de la misma.
Excepciones	**Paso**	**Acción**
	3a,5a	El actor Delegado cancela la operación de eliminar la competición y el caso de uso queda sin efecto
	5b	El actor Delegado selecciona que no quiere eliminar la competición y el caso de uso queda sin efecto.

Tabla 4.17: Caso de uso: Eliminar competición

CU-18	Eliminar equipo	
Descripción	El sistema permitirá al actor Delegado eliminar un equipo de Judo.	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona ver equipos creados hasta ese momento
	2	El Sistema muestra todas las competiciones.
	3	El actor Delegado selecciona eliminar un equipo de los mostrados en el paso 2.
	4	El Sistema muestra un mensaje de confirmación antes de eliminar el equipo.
	5	El actor Delegado confirma la eliminación del equipo
	6	El Sistema elimina el equipo.
Excepciones	**Paso**	**Acción**
	3a,5a	El actor Delegado cancela la operación de eliminar el equipo y el caso de uso queda sin efecto
	5b	El actor Delegado selecciona que no quiere eliminar el equipo y el caso de uso queda sin efecto.

Tabla 4.18: Caso de uso: Eliminar equipo

CU-19	Cambiar peso de un competidor	
Descripción	El sistema permitirá al actor Delegado cambiar el peso de un competidor en una determinada competición.	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona ver las competiciones.
	2	El Sistema muestra todas las competiciones.
	3	El actor Delegado selecciona inscripción de competidores en una de las competiciones mostradas en el paso 2.
	4	El Sistema muestra los competidores inscritos por peso
	5	El Sistema solicita el cambio de un competidor a otro peso
	6	El actor Delegado selecciona el competidor y le asigna el nuevo peso
	7	El Sistema guarda los cambios y el caso de uso finaliza.
Excepciones	**Paso**	**Acción**
	3a,6a	El actor Delegado cancela la operación de eliminar el equipo y el caso de uso queda sin efecto
	4a	El Sistema comprueba que ya se ha generado un sorteo para esa competición y no se puede cambiar la inscripción de ningún competidor, muestra un mensaje de error y el caso de uso queda sin efecto
	4b	El Sistema comprueba que la fecha de la competición es anterior al día del cambio de la inscripción, muestra un mensaje de error y el caso de uso queda sin efecto.

Tabla 4.19: Caso de uso: Actualizar peso

CU-20	Cambiar el tatami de un árbitro	
Descripción	El sistema permitirá al actor Delegado cambiar el tatami asignado a un árbitro en una determinada competición.	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona ver las competiciones.
	2	El Sistema muestra todas las competiciones.
	3	El actor Delegado selecciona inscripción de árbitros en una de las competiciones mostradas en el paso 2.
	4	El Sistema muestra los árbitros inscritos y los árbitros asignados a los tatamis que tenga la competición
	5	El Sistema solicita el cambio de un árbitro a otro tatami
	6	El actor Delegado selecciona un árbitro y le asigna el nuevo tatami
	7	El Sistema guarda los cambios y el caso de uso finaliza.
Excepciones	**Paso**	**Acción**
	3a,6a	El actor Delegado cancela la operación de eliminar el equipo y el caso de uso queda sin efecto
	4a	El Sistema comprueba que ya se ha generado un sorteo para esa competición y no se puede cambiar la inscripción de ningún árbitro, muestra un mensaje de error y el caso de uso queda sin efecto
	4b	El Sistema comprueba que la fecha de la competición es anterior al día del cambio de la inscripción, muestra un mensaje de error y el caso de uso queda sin efecto.

Tabla 4.20: Caso de uso: Actualizar tatami

CU-21	Cambiar idioma	
Descripción	El sistema permitirá al actor Usuario cambiar el idioma de la interfaz de la aplicación, para la sesión que vaya a realizar en ese momento.	
Precondición		
Secuencia normal	**Paso**	**Acción**
	1	El actor Usuario selecciona cambiar el idioma de la aplicación.
	2	El Usuario muestra los idiomas disponibles.
	3	El actor Usuario selecciona uno de los idiomas mostrados en el paso 2
	4	El Sistema cambia el idioma para la sesión actual.
Excepciones	**Paso**	**Acción**
	3a	El actor Usuario cancela la operación de cambiar el idioma, y el caso de uso queda sin efecto.

Tabla 4.21: Caso de uso: Cambiar idioma

CU-22	Eliminar participación de un competidor	
Descripción	El sistema permitirá al actor Delegado eliminar la inscripción de un competidor	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona ver las competiciones creadas.
	2	El Sistema muestra todas las competiciones ordenadas cronológicamente.
	3	El actor Delegado selecciona la opción de inscribir competidores en una de las competiciones.
	4	El Sistema muestra las inscripciones de los competidores realizadas hasta ese momento de la competición seleccionada.
	5	El actor Delegado selecciona eliminar la inscripción de uno de los competidores
	6	El Sistema solicita la confirmación antes de eliminar la inscripción
	7	El actor Delegado confirma la eliminación del competidor indicado en el paso 5.
	8	El Sistema elimina la inscripción.
Excepciones	**Paso**	**Acción**
	3a,5a,7a	El actor Delegado cancela la operación y el caso de uso queda sin efecto.
	7b	El actor Delegado selecciona que no quiere eliminar la inscripción y el caso de uso queda sin efecto.
	4a	El Sistema comprueba que la competición seleccionada tiene una fecha anterior a la actual , muestra un mensaje de error y el caso de uso queda sin efecto.

Tabla 4.22: Caso de uso: Eliminar la participación de un competidor

CU-23	Eliminar la inscripción de un árbitro	
Descripción	El sistema permitirá al actor Delegado eliminar la inscripción de un árbitro	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona ver las competiciones creadas.
	2	El Sistema muestra todas las competiciones ordenadas cronológicamente.
	3	El actor Delegado selecciona la opción de inscribir árbitros de una de las competiciones.
	4	El Sistema muestra los árbitros inscritos y los asociados a un tatami
	5	El actor Delegado selecciona eliminar la inscripción de uno de los árbitros
	6	El Sistema solicita la confirmación antes de eliminar la inscripción
	7	El actor Delegado confirma la eliminación del árbitro indicado en el paso 5.
	8	El Sistema elimina la inscripción.
Excepciones	**Paso**	**Acción**
	3a,5a,7a	El actor Delegado cancela la operación y el caso de uso queda sin efecto.
	7b	El actor Delegado selecciona que no quiere eliminar la inscripción y el caso de uso queda sin efecto.
	4a	El Sistema comprueba que la competición seleccionada tiene una fecha anterior a la actual, muestra un mensaje de error y el caso de uso queda sin efecto.

Tabla 4.23: Caso de uso: Eliminar la inscripción de un árbitro

CU-24	Introducir puntuación	
Descripción	El sistema permitirá al actor Cronometrador introducir la puntuación de un determinado combate	
Precondición	El actor Cronometrador deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Cronometrador selecciona ver los combates de un peso determinado
	2	El Sistema muestra todos los combates
	3	El actor Cronometrador selecciona uno de los combates mostrados en el paso 2
	4	El Sistema solicita la introducción de la puntuación del combate
	5	El actor Cronometrador introduce la puntuación del combate
	6	El Sistema guarda los datos de la puntuación.
Excepciones	**Paso**	**Acción**
	3a,5a	El actor Cronometrador cancela la operación y el caso de uso queda sin efecto.

Tabla 4.24: Caso de uso: Introducir la puntuación

CU-25	Cambiar tatami del combate	
Descripción	El sistema permitirá al actor Delegado cambiar el tatami donde se desarrollará el encuentro o combate	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado selecciona ver los combates de un peso determinado
	2	El Sistema muestra todos los combates
	3	El actor Delegado selecciona selecciona uno de los combates mostrados en el paso 2
	4	El Sistema solicita el cambio de tatami y muestra los posibles tatamis donde se puede asginar el combate
	5	El actor Delegado selecciona uno de los tatamis mostrados en el paso anterior
	6	El Sistema guarda el nuevo tatami y el caso de uso finaliza.
Excepciones	**Paso**	**Acción**
	3a,5a	El actor Delegado cancela la operación y el caso de uso queda sin efecto.

Tabla 4.25: Caso de uso: Cambiar tatami del combate

CU-26	Crear cronometrador	
Descripción	El sistema permitirá al actor Delegado crear un nuevo cronometrador en el Sistema	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado solicita crear un nuevo cronometrador
	2	El Sistema solicita los datos necesarios para crear un nuevo cronometrador, que son: email, DNI, nombre, primer apellido, segundo apellido y contraseña de acceso
	3	El actor Delegado introduce los datos solicitados en el paso 2
	4	El Sistema comprueba que los datos introducidos son válidos y guarda el nuevo cronometrador en el Sistema
Excepciones	**Paso**	**Acción**
	3a	El actor Delegado cancela la operación y el caso de uso queda sin efecto.
	4a	El Sistema comprueba que falta algún campo requerido, muestra un mensaje de error y el caso de uso continúa en el paso 2.
	4b	El Sistema comprueba que el email del cronometrador o el DNI coincide con el de otro cronometrados registrado previamente en el Sistema y el caso de uso continúa en el paso 2.

Tabla 4.26: Caso de uso: Crear cronometrador

CU-27	Cambiar contraseña del cronometrador	
Descripción	El sistema permitirá al actor Delegado crear un nuevo cronometrador en el Sistema	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado solicita cambiar la contraseña de un cronometrador
	2	El Sistema solicita introducir la nueva contraseña
	3	El actor Delegado introduce la nueva contraseña
	4	El Sistema cambia la contraseña
Excepciones	**Paso**	**Acción**
	3a	El actor Delegado cancela la operación y el caso de uso queda sin efecto.

Tabla 4.27: Caso de uso: Cambiar contraseña del cronometrador

CU-28	Cambiar inscripción de un competidor	
Descripción	El sistema permitirá al actor Delegado cambiar el peso en el que va a competir un determinado competidor	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado solicita cambiar el peso de un competidor previamente inscrito en la competición
	2	El Sistema muestra los pesos en los que puede realizar el cambio de inscripción
	3	El actor Delegado selecciona el nuevo peso
	4	El Sistema comprueba los cambios realizados en el paso anterior, los guarda y el caso de uso finaliza
Excepciones	**Paso**	**Acción**
	3a	El actor Delegado cancela la operación y el caso de uso queda sin efecto.
	4a	El Sistema comprueba que no se ha seleccionado ningún peso o es el mismo en el cual el competidor estaba inscrito y el caso de uso queda sin efecto

Tabla 4.28: Caso de uso: Cambio de peso de un competidor

CU-29	Asignar categoría arbitral	
Descripción	El sistema permitirá al actor Delegado introducir la categoría arbitral de un árbitro	
Precondición	El actor Delegado deberá estar identificado en el Sistema.	
Secuencia normal	**Paso**	**Acción**
	1	El actor Delegado solicita obtener las diferentes categorías arbitrales
	2	El Sistema muestra las categorías que están registradas en la aplicación
	3	El actor Delegado selecciona una de ellas
	4	El Sistema comprueba los cambios realizados en el paso anterior, los guarda y el caso de uso finaliza.
Excepciones	**Paso**	**Acción**
	3a	El actor Delegado cancela la operación y el caso de uso queda sin efecto.

Tabla 4.29: Caso de uso: Asignar categoría arbitral

4.7. Modelo de dominio inicial

En este apartado se mostrará y explicará el diagrama de modelo de dominio del problema a resolver, el cual nos proporcionará una idea general de cómo se tiene que implementar en posteriores fases las clases del sistema.

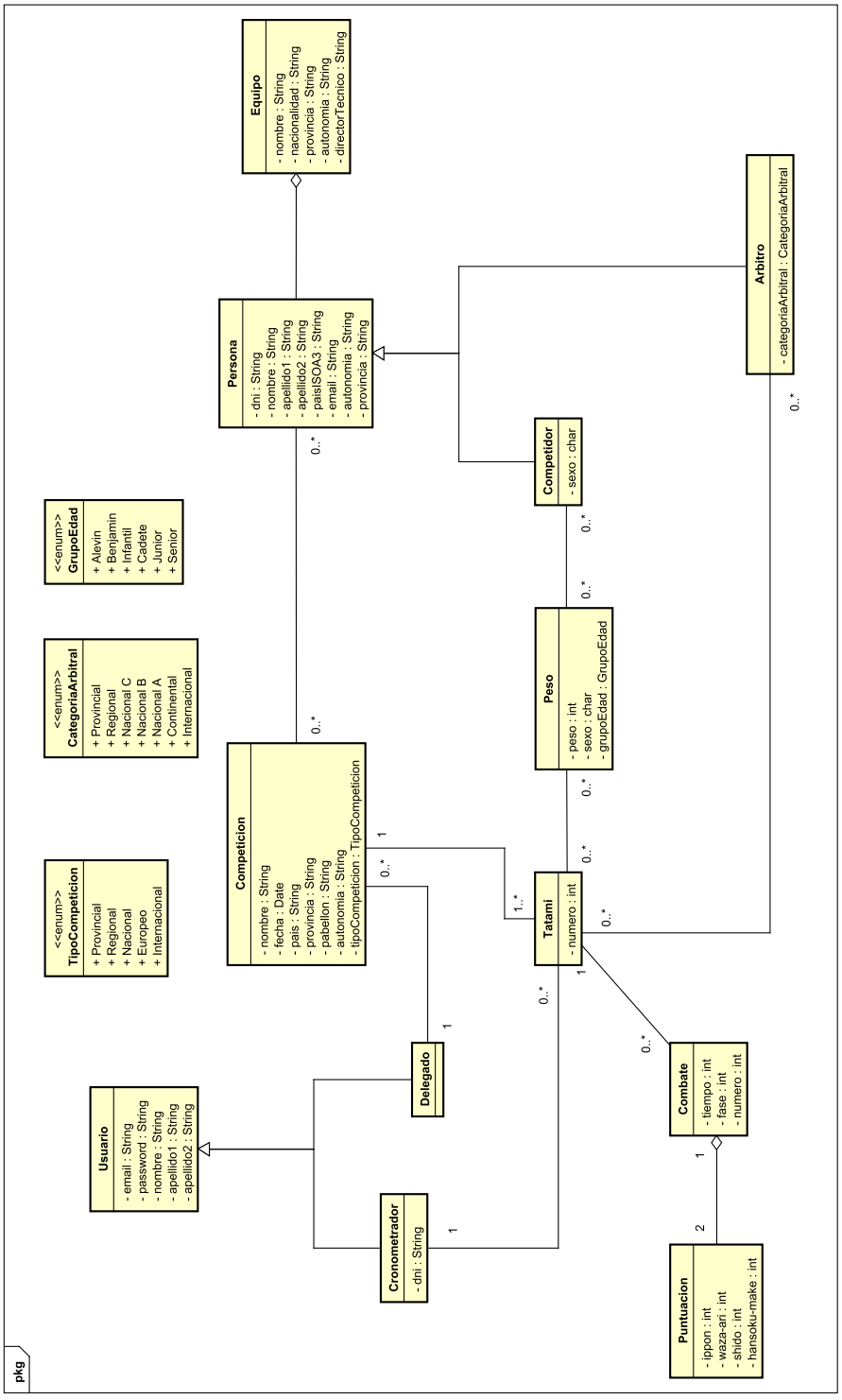


Figura 4.4: Modelo de dominio inicial

Basándose en los documentos iniciales de requisitos funcionales, casos de uso y diferentes reuniones con el delegado de Judo de Castilla y León, se ha obtenido como resultado el diagrama de modelo de dominio inicial. A continuación se describirá cada una de las clases del modelo creado en la figura 4.4.

- **Usuario**. Representa a los usuarios que van a interaccionar con el sistema. En este capítulo se

ha hablado de que existen dos tipos de usuarios, el delegado y el juez-crono o cronometrador. Esta relación de generalización/especialización la encontramos entre la superclase usuario y las clases hijas cronometrador y delegado, ya que ambas tienen características comunes que hacen que se pueda producir esta relación.

- **Delegado**. Clase que representa al delegado de la competición. Encargado, generalmente del manejo de las competiciones. Esta clase hereda de Usuario.
- **Cronometrador**. Clase que representa al cronometrador. Tiene los atributos necesarios para poder identificarlos y diferenciarlos de otros cronometradores.
- **Competición**. Representa las competiciones de judo. Tiene los atributos que se describen este capítulo en la sección (4.3.3) de elicitación de requisitos.
- **Tatami**. Clase que representa a los tatamis de judo, que es donde se realizan los combates entre dos competidores. En cada competición hay un número determinado de tatamis los cuales vienen distinguidos por un número. Normalmente, en las competiciones de Castilla y León se tiene tres tatamis diferenciados por: tatami 1, tatami 2 y tatami 3. La organización decide en que tatami se desarrollan los combates. Normalmente, los pesos están asociados a los tatamis al comienzo de la competición, pero a medida que se va desarrollando esta, se pueden cambiar para que ninguno de los tatamis quede sin ningún combate asignado.
- **Combate**. Representa a los combates de judo. Estos tienen asociados una puntuación para el competidor que viste el kimono [1] de color blanco y otra puntuación para el competidor que lleva el kimono de color azul.
- **Puntuación**. Clase que representa a las puntuaciones de judo. En un combate, se pueden producir diferentes acciones que dan lugar a una puntuación u otra. Actualmente, con las reglas del año 2017, pueden tomar cuatro valores, como he explicado en secciones anteriores (4.3.3) que son: ippon, waza-ari, shido y hansokumake.
- **Peso**. Representa los pesos. En toda competición de judo se tienen diferentes pesos, cada uno de los cuales agrupa un intervalo de kilogramos. Por ejemplo, la categoría de menos de 90 kilogramos abarcaría los pesos de 81.1 kilogramos a 90 kilogramos. En función del grupo de edad del que sea la competición, se tendrán unos pesos u otros. Un pesos también vienen diferenciados por el sexo del competidor; están los pesos masculinos de un grupo de edad y los pesos femeninos dentro de ese mismo grupo de edad.
- **Persona**. Superclase de competidor y árbitro. Como estas dos últimas clases comparten atributos, dan cabida a este tipo de relación de generalización-especialización.
- **Equipo**. Representa a los equipos de judo. Un equipo está formado por un grupo indeterminado de personas, que como he explicado en el párrafo anterior pueden ser tanto competidores como árbitros.

Por último se detallará las clases de tipo enumerado que tenemos en la figura 4.4:

- **GrupoEdad**. Contiene los diferentes tipos de valores que puede tomar la clase, representa los tipos de grupo de edad, estos son: alevín, benjamín, infantil, cadete, junior y senior.
- **TipoCompetición**. Contiene los diferentes valores que puede tomar la clase, representa los tipos de competición: provincial, regional, nacional, europeo e internacional.
- **CategoriaArbitral**. Contiene los diferentes valores que puede tomar la clase. Representa los diferentes tipos de categoría arbitral que hay en el sistema de competición del año 2017. Estos son: provincial, regional, nacional de tipo C, nacional de tipo B, nacional de tipo A, continental e internacional.

[1] Vestimenta para la práctica de judo, compuesto por una chaqueta y un pantalón del mismo color.

4.8. Máquinas de estado

Se van a mostrar a continuación las máquinas de estados de las clases que pueden llegar a tener más problemas a la hora de entender como funcionan

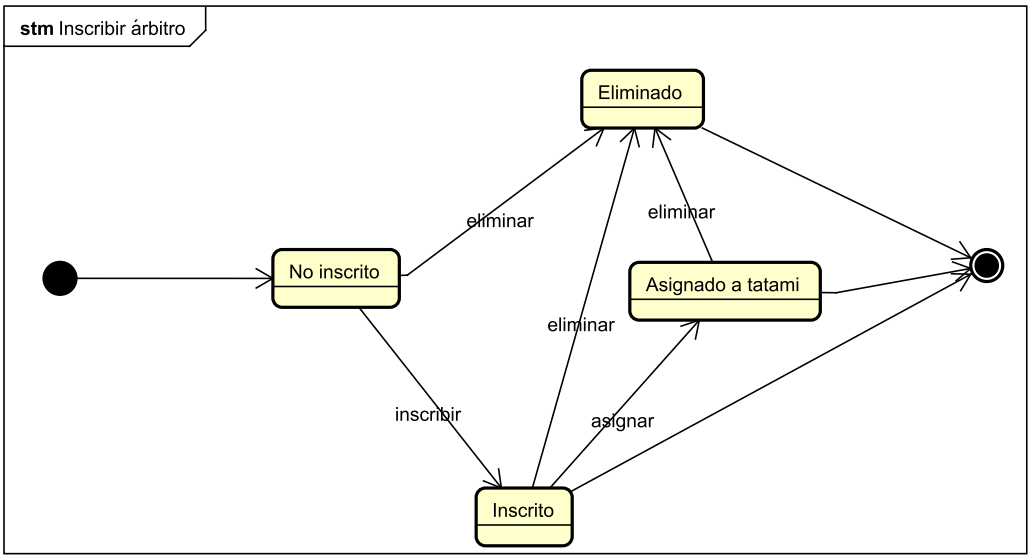


Figura 4.5: Máquina de estado inscribir un árbitro

La figura 4.5 muestra cada uno de los estados por los que puede pasar una inscripción de un árbitro. En un primer momento, cuando se inicia una competición de judo, no existe ninguna inscripción asociada a esta (primer estado en el diagrama, *no inscrito*). Cuando se realiza la inscripción de un árbitro, este pasa a estado de *inscrito*. En este estado puede pasar que sea eliminado del campeonato, entonces pasaría a estado *eliminado* o bien podría ser asignado a un tatami de la competición, pasando a estado de *asignado a tatami*. En cualquier caso, desde cualquier estado, un árbitro puede pasar a estado *eliminado*.

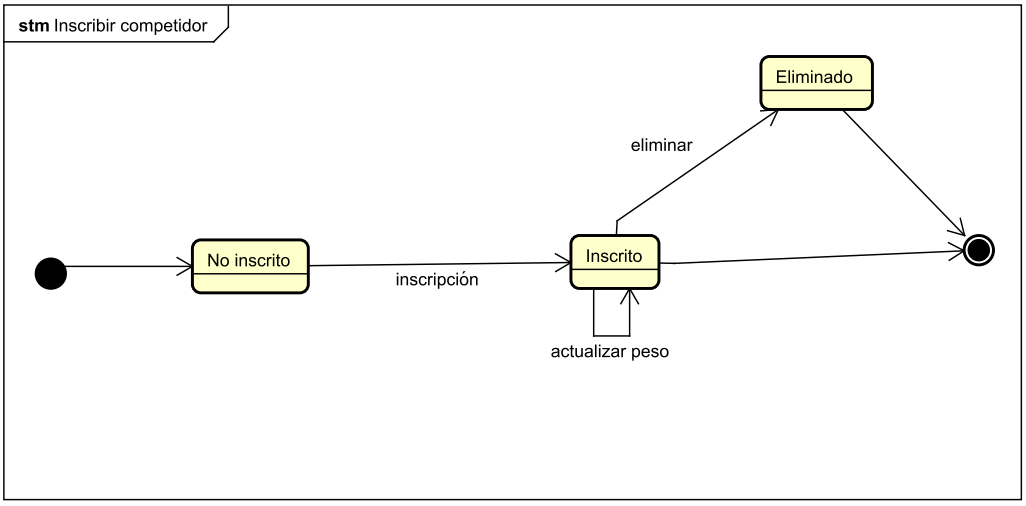


Figura 4.6: Máquina de estado inscribir un competidor

La figura 4.6 muestra la máquina de estados correspondientes a la inscripción de un competidor. Una inscripción puede pasar por tres estados, *no inscrito*, cuando no se ha realizado la inscripción de un competidor en la competición. Por otro lado tenemos el estado de *inscrito*, cuando se ha realizado la inscripción, es decir, se le ha asociado un peso de uno de los posibles del campeonato, y por último, el estado de *eliminado*, es decir cuando se elimina la participación de un competidor en la competición.

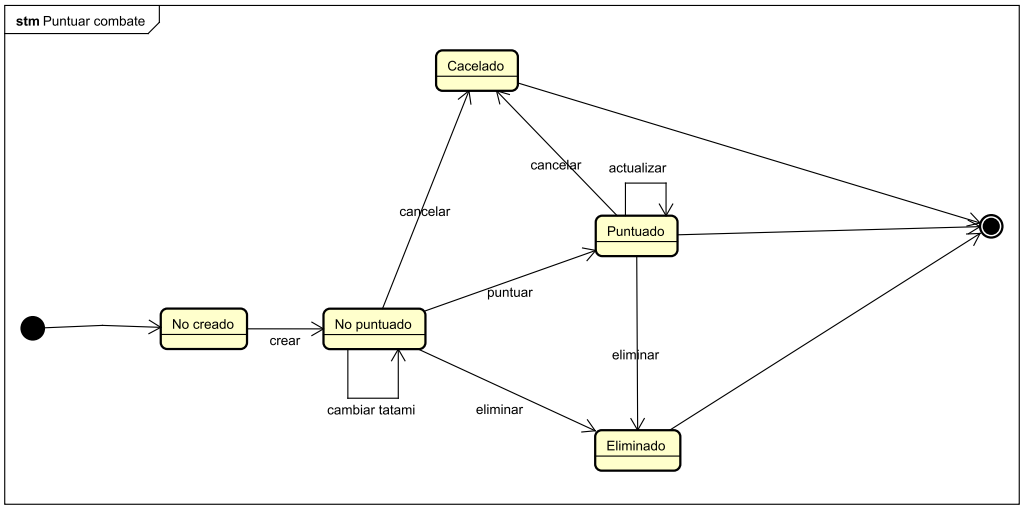


Figura 4.7: Máquina de estado puntuar combate

La figura 4.7 muestra la máquina de estados correspondiente a la puntuación de un combate. El estado inicial de un combate es el de *no creado*. En el momento que se crea un combate, pasa a estado de *no puntuado*. En este último estado puede pasar a estado de *cancelado* si no se quiere introducir ninguna puntuación por el momento, a estado *puntuado* si se realiza algún tipo de puntuación, o a estado de *eliminado* si se elimina el combate.

4.9. Diagramas de secuencia en la etapa de análisis

A continuación se van a mostrar los diagramas de secuencia de las operaciones CRUD, del inglés *Create,Retrieve,Update y Delete*. Estos diagramas son los de la etapa de análisis, por lo tanto lo que se muestra es la interacción del actor o actores con el Sistema a desarrollar.

El primer diagrama de secuencia corresponde a la descripción de caso de uso *Eliminar árbitro*, en la tabla 4.16. En una primera secuencia de mensajes se puede ver como el actor delegado solicita ver los árbitros que están registrados en el sistema y como este se los muestra. En la segunda secuencia, el actor elimina uno de los mostrados en el paso anterior.

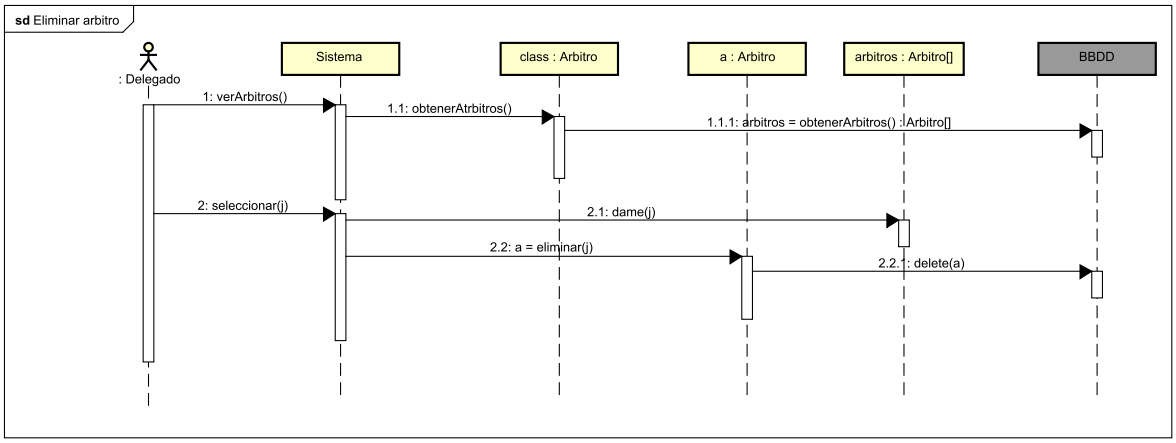


Figura 4.8: Diagrama de secuencia eliminar un árbitro de Judo

El segundo diagrama, figura 4.9 corresponde a la descripción del caso de uso de la tabla 4.1, crear un árbitro de judo. Se puede observar la secuencia normal entre el actor delegado y el sistema y las secuencias alternativas, las excepciones (las que se encuentran dentro del fragmento combinado *alt*).

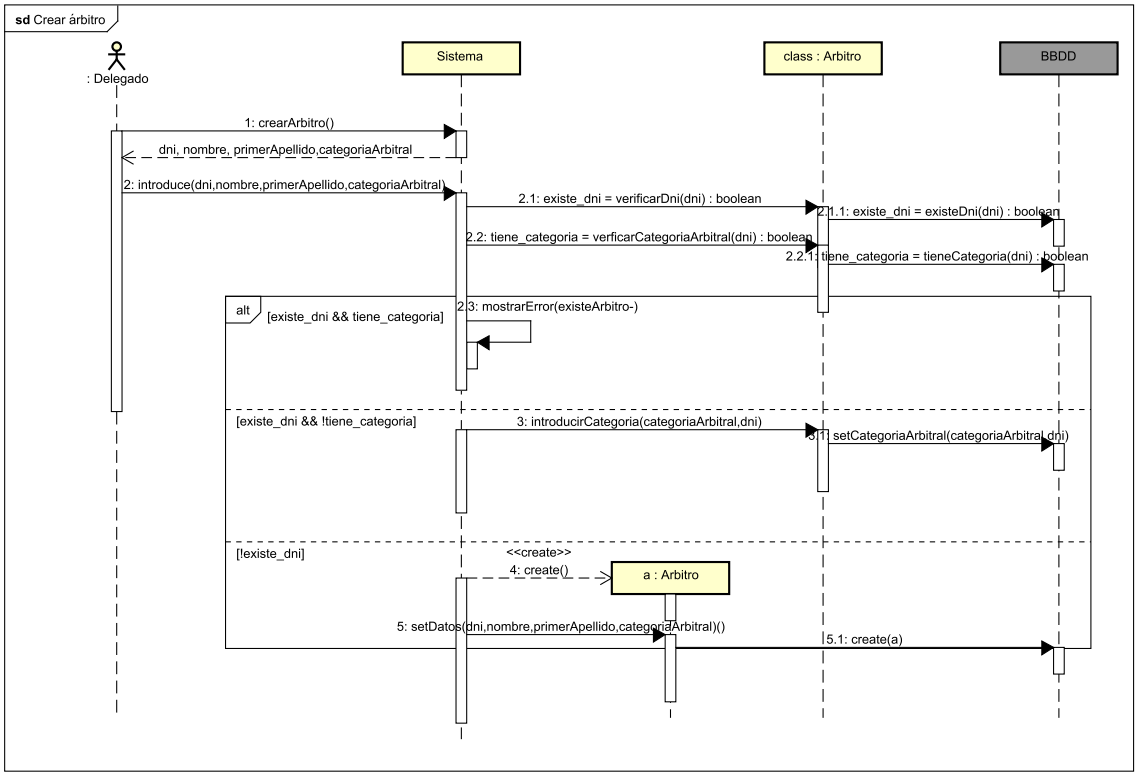


Figura 4.9: Diagrama de secuencia crear un árbitro de Judo

Por último el diagrama de secuencia correspondiente a actualizar la categoría arbitral de un árbitro, cuya descripción de caso de uso la podemos encontrar en la tabla 4.15.

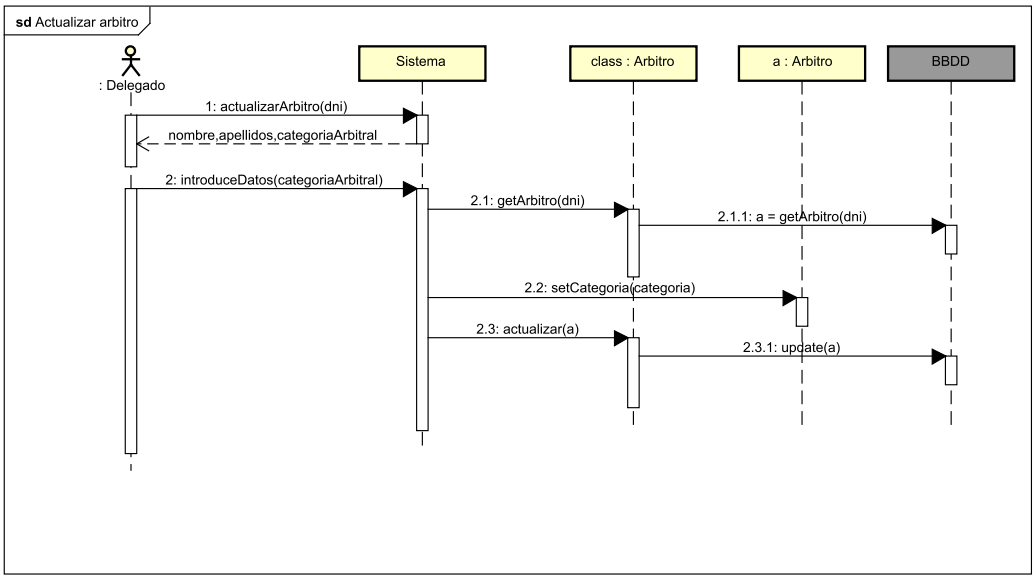


Figura 4.10: Diagrama de secuencia actualizar un árbitro de Judo

4.10. Algoritmos de emparejamiento

En esta sección se van a analizar los diferentes algoritmos de emparejamientos entre competidores que se han valorado para aplicarlo en este sistema.

Algoritmo de emparejamiento de Edmonds

También conocido como algoritmo de Blossom[12], es un algoritmo de la teoría de grafos utilizado para obtener emparejamientos máximos entre vértices cumpliendo ciertas restricciones. La importancia de este algoritmo reside en que probó que es posible encontrar un emparejamiento máximo en tiempo polinomial, $\mathcal{O}(|V|^2)$

En nuestro caso en concreto, este podría ser utilizado para la obtención de todos los posibles emparejamientos de cada uno de los competidores con el resto de competidores. Podría valer para la implementación de la generación del sorteo, pero no se adecúa correctamente a nuestro problema, ya que en este caso lo que necesitamos es un valor aleatorio entre todos los posibles que cumplan la condición de emparejamiento entre dos competidores y el algoritmo de Edmonds nos da todos los posibles emparejamientos. En un futuro, si que podría ser interesante tener la implementación de este algoritmo para así tener todas las posibles combinaciones, pero por el momento, no se va a realizar.

Sistema de todos contra todos o sistema liga

Este algoritmo[13] es utilizado en aquellos casos en los que se quiera que cada uno de los participantes de una determinada competición, en este caso una de Judo, se enfrenten todos contra todos. Es un Sistema llamado liguilla o Round-robin. En el caso de judo, las liguillas sólo se dan en aquellos pesos que tienen un número de competidores pequeño, más concretamente, se da este caso cuando el número de participantes oscila entre tres y seis.

El algoritmo de selección de los emparejamientos sería como se explica a continuación poniendo como caso particular una competición de judo con las reglas actuales de 2017-2018.

Caso en el que el número de competidores n sea igual a 2

En este caso, la aplicación del algoritmo es muy sencilla ya que sólo se pueden enfrentar estos dos.

```
1 Lista de combates: (1-2), (2-1)
```

Caso en el que el número de competidores n oscila entre 3 y 4

En este caso, aplicando las reglas actuales del comité internacional de Judo, sólo se realizan combates una vez entre dos mismos competidores de la misma liga. Por lo tanto, si definimos n como número de competidores, se van a realizar $\frac{n(n-1)}{2}$ encuentros.

```
1 Lista de combates (n = 3): (1-3)(2-3)(1-2)
2 Lista de combates (n = 4): (1-4)(2-3)(1-3)(2-4)
```

Este algoritmo tiene un coste computacional de $\mathcal{O}(n)$, lineal, por lo que es muy bajo. Además, el número de competidores oscila entre 2 y 4 por lo que es todavía más insignificante este coste computacional.

El número de encuentros simultáneos que se pueden ejecutar cuando el número de competidores es par, es $\frac{n}{2}$; mientras que si el número de competidores es impar, quedará un competidor sin competir, quedando compitiendo simultáneamente $\frac{(n-1)}{2}$competidores. Está situación se daría en el caso ideal de que se tuviera un tatami por encuentro disputado, pero normalmente en una competición suele haber del orden de tres a seis tatamis por lo que no suelen competir simultáneamente debido a que existen varios competidores de distintos pesos que necesitan esos tatamis.

Algoritmo de la col, el lobo y la cabra

Algoritmo de la rama de inteligencia artificial[14] que tiene como objetivo resolver el problema de como emparejar elementos dos a dos para cruzar el río en una barca, cumpliendo ciertas restricciones, que son las siguientes

1. La cabra y la col no pueden estar juntas, ni en la barca ni a uno ni a otro lado del río.
2. La cabra y el lobo no pueden estar juntos, ni en la barca ni a uno ni a otro lado del río.

Además, el granjero puede pasar o el solo o con uno de los elementos (cabra, col y lobo) cada vez. La solución de este problema consiste en el siguiente algoritmo.

1. Granjero, cabra, col y lobo en el lado A del río
2. Deja a la cabra en el lado B del río
3. Vuelve
4. Deja al lobo en el lado B del río
5. Trae a la cabra al lado A del río
6. Deja la col en el lado B del río
7. Regresa al lado A el solo
8. Deja a la cabra en el lado B del río
9. Granjero, cabra, col y lobo en el lado B del río

Algoritmo de la Federación Castellano Leonesa de Judo

Este algoritmo es el empleado actualmente de forma manual, con un sistema de papeles y bolsas, en los papeles se ponen los nombres de los competidores de un determinado peso, separados por cabezas de serie en diferentes bolsas. La elección de los emparejamientos se realiza de forma aleatoria eligiendo de cada uno de los dos montones un papel. La condición de que se puedan emparejar es que no pertenezcan al mismo club.

```
EMPEZAR
    MIENTRAS ∃ COMPETIDOR LIBRE

        HACER
            Coger competidor bolsa A
            Coger competidor bolsa B

        MIENTRAS Club competidor A = Club competidor B

            Emparejar(A,B)

        FIN HACER

    FIN MIENTRAS

FIN
```

El coste algortimico de este sería de $\mathcal{O}(n^2)$ siendo n el número de competidores. Que sea cuadrático el algoritmo tampoco implica un gran problema ya que n es insignificante computacionalmente hablando.

Algoritmo utilizado

El coste computacional para cualquier algoritmo que escojamos va a ser ínfimo, debido a que el número de competidores que se quiere emparejar es pequeño. Estamos hablando de un tamaño de n que como máximo podría alcanzar los 60-100 competidores por peso. Como el coste computacional no es un problema mayor, la decisión de qué algoritmo implementar la basaremos en tiempo de implementación del mismo.

De los algoritmos citados en los párrafos anteriores, utilizaremos dos. El primero es el de *Liguilla o Round-robin*, utilizado en el caso que se tenga que realizar una liga entre los competidores de un mismo peso, cumpliendo el número máximo de competidores que puede haber para poder realizarla, que son cuatro. El segundo algoritmo es el que utiliza la federación de Castilla y León de judo para generar los sorteos, que usaremos cuando no tengamos que aplicar el algoritmo de round-robin. Quién decide qué algoritmo de emparejamiento de todos utilizar es el delegado de la competición, es decir, que si para diez competidores quiere hacer el formato de liguilla, lo podría hacer sin ningún problema.

Capítulo 5

Diseño

5.1. Introducción

En este capítulo se mostrará el diseño realizado para la aplicación, cumpliendo los requisitos planteados en el capítulo anterior.

En un primer lugar, se expone una visión general de la arquitectura de la aplicación. Después, se habla sobre el diseño basado en la seguridad, teniendo en cuenta la nueva normativa de la Ley de Protección de Datos. En la sección 5.5 se detallará el procedimiento seguido para diseñar la base de datos (definición del modelo entidad relación, modelo relacional, etc). En la siguiente sección, se hablará sobre los diferentes patrones de diseño utilizados (Modelo-Vista-Controlador y el patrón Comando-Estrategia). Por último, en las secciones 5.7 a 5.9, se hablará sobre los diferentes paquetes que forman la arquitectura lógica y la comunicación entre estos paquetes.

5.2. Arquitectura

La arquitectura de software, también conocida como arquitectura lógica, está formada por un conjunto de patrones y paquetes que conforman una estructura sólida para poder desarrollar el código de la aplicación. En el diseño de la arquitectura lógica de esta aplicación, utilizaremos en gran medida la que proporciona el framework de codeigniter la cual la explicaremos en la sección 5.2.1.

5.2.1. Arquitectura de *Codeigniter*

La arquitectura de la aplicación que usa el framework de *Codeigniter* es la que se muestra en la figura 5.2. Cada vez que llega una nueva petición, lo primero que hace es pasar por la página **index.php**. El framework da la posibilidad de evitar poner en cada llamada al controlador esa página en los ficheros de configuración, para la cual simplemente habría que poner en el fichero config.php la siguiente línea. Así se evita insertar la cadena en cada una de las URL que queramos llamar.

```
1    $config['index_page'] = '';
```

El siguiente paso, la capa de enrutamiento se encargará de decidir si la petición es aceptada o rechazada. En caso de aceptarla, la petición irá al paso 3, guardar la página en caché; en cualquier caso, el usuario indicará una repuesta en caso de que no pase la fase de seguridad. Una vez realizadas se pasa las comprobaciones de seguridad, pasamos a la capa de controladores de la aplicación, donde se cargan todos los recursos necesarios para resolver la petición, tales como librerías, modelos que acceden a la base de datos, fichero útiles, etc. Por último, el controlador pasará todos los datos a la vista o vistas para que muestren el correspondiente resultado al usuario y después guardar todo en caché, para procesar más rápidamente solicitudes posteriores a la misma página.

La estructura del directorio de trabajo donde se encontrarán todos los ficheros que generará esa arquitectura es como se muestra en la figura 5.1.

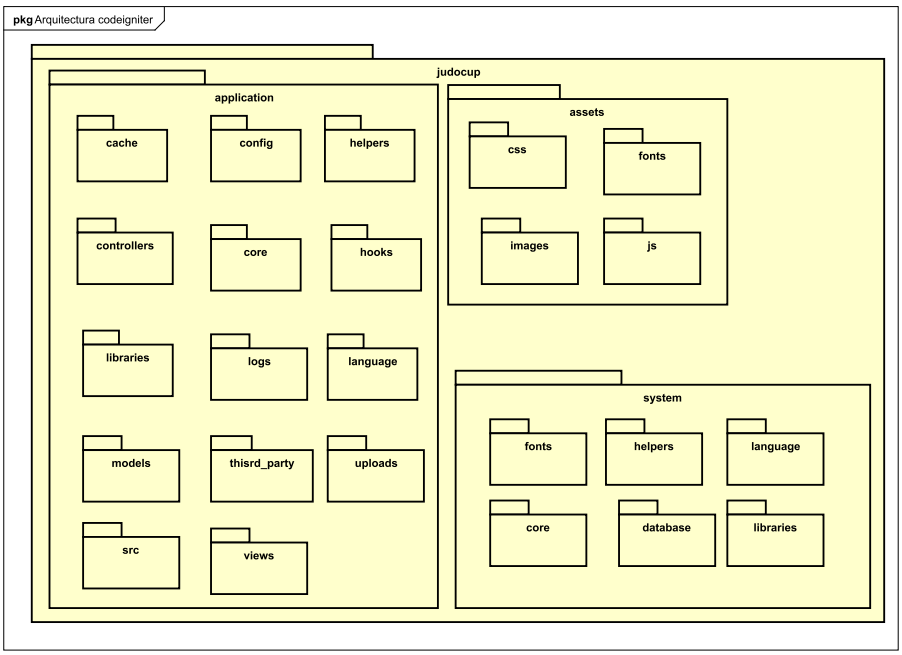


Figura 5.1: Directorio de trabajo usando el framework de Codeigniter

A continuación se explicarán alguno de los directorios más significativos.

- **Caché**. En este directorio, almacenaremos aquellas páginas que se quieren guardar en almacenamiento caché, es decir, aquellas cuyas peticiones son recurrentes y en las cuales los datos no cambian. Esto incrementará notablemente la velocidad de respuesta de acceso a los recursos y por lo tanto el grado de aceptabilidad de la aplicación por parte del usuario.

- **Config**. Este es uno de los directorios más importantes y en los que tendremos que tener más cuidado a la hora de realizar cambios. Aquí se encuentran todos los ficheros de configuración del framework, tales como la inicialización de la base de datos, fichero *database.php*, y el fichero *autoload.php* que se encargará de cargar automáticamente aquellos módulos que le indiquemos cada vez qué llamamos a un cierto controlador de la aplicación, entre otros.

- **Controladores**. Contiene los controladores de la aplicación, en nuestro caso albergaremos aquí los controladores de caso de uso.

- **Modelos**. En esta carpeta se situarán las clases que van a acceder a la base de datos para realizar las operaciones CRUD(eliminar, actualizar, obtener e introducir datos).

- **Vistas**. Colocaremos aquí los ficheros que darán la forma de la aplicación, es decir todos los ficheros PHP que contienen código HTML en su interior.

- **Lenguaje**. Una buena forma de hacer que una aplicación web se pueda traducir fácilmente en varios idiomas es con la utilización de un diccionario. En este directorio, colocaremos dicho diccionario, en los dos idiomas que nos marcan los requisitos del capítulo de análisis, en español e inglés. En el siguiente fragmento de código se muestra un ejemplo del diccionario que se va a utilizar. Para cada uno de los idiomas que tengamos tenemos que tener un homólogo idéntico, con la única diferencia de la traducción del texto.

  La gran ventaja de utilizar un diccionario, es que si se quiere realizar la implantación de un nuevo lenguaje, simplemente tendremos que copiar todos los ficheros en una nueva carpeta con el nombre del idioma y traducir cada una de las claves.

- **Assets**. En esta carpeta se localizan todos los ficheros relacionados con la apariencia de la aplicación y el dinamismo, es decir, ficheros .css, .js, imágenes, las fuentes utilizadas, etc.

En el siguiente diagrama (figura 5.2) se muestra el funcionamiento básico de una petición al framework. Cada petición que realiza el usuario pasa por los estados mostrados en el diagrama. Como hemos comentado en el apartado uno de este capítulo, la petición pasa por el fichero *index.php*, después pasa por el sistema de rutas que es el encargado de mostrar la vista correspondiente si está almacenada en caché o envía la petición al sistema de seguridad que acepta o rechaza la petición. En caso de que sea aceptada, carga la vista con todos los datos necesarios sacados de diferentes ficheros, modelos, bases de datos, etc.

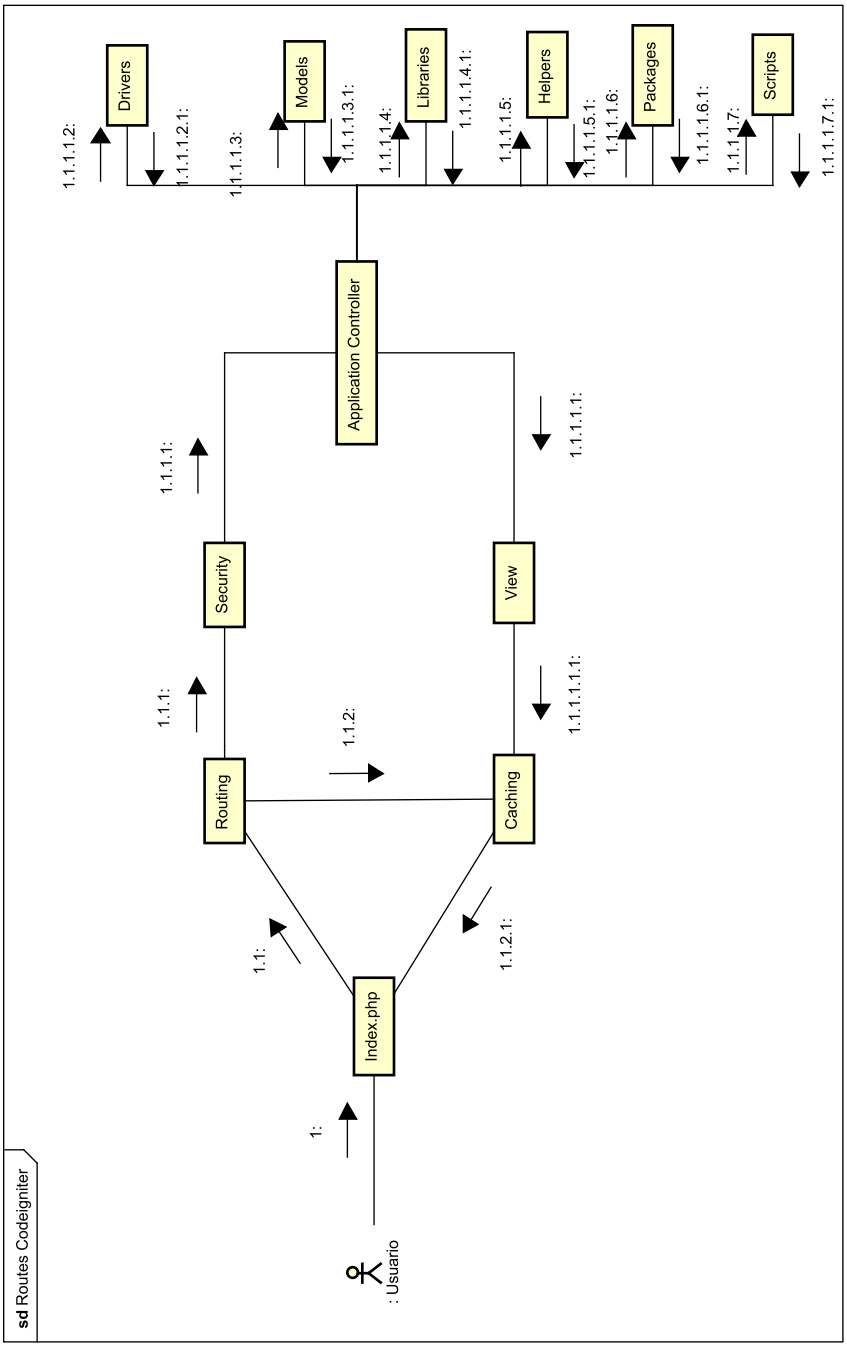


Figura 5.2: Arquitectura de Codeigniter

En la figura 5.3 se muestra el diagrama de despliegue del Sistema. A continuación, se describen brevemente los componentes principales.

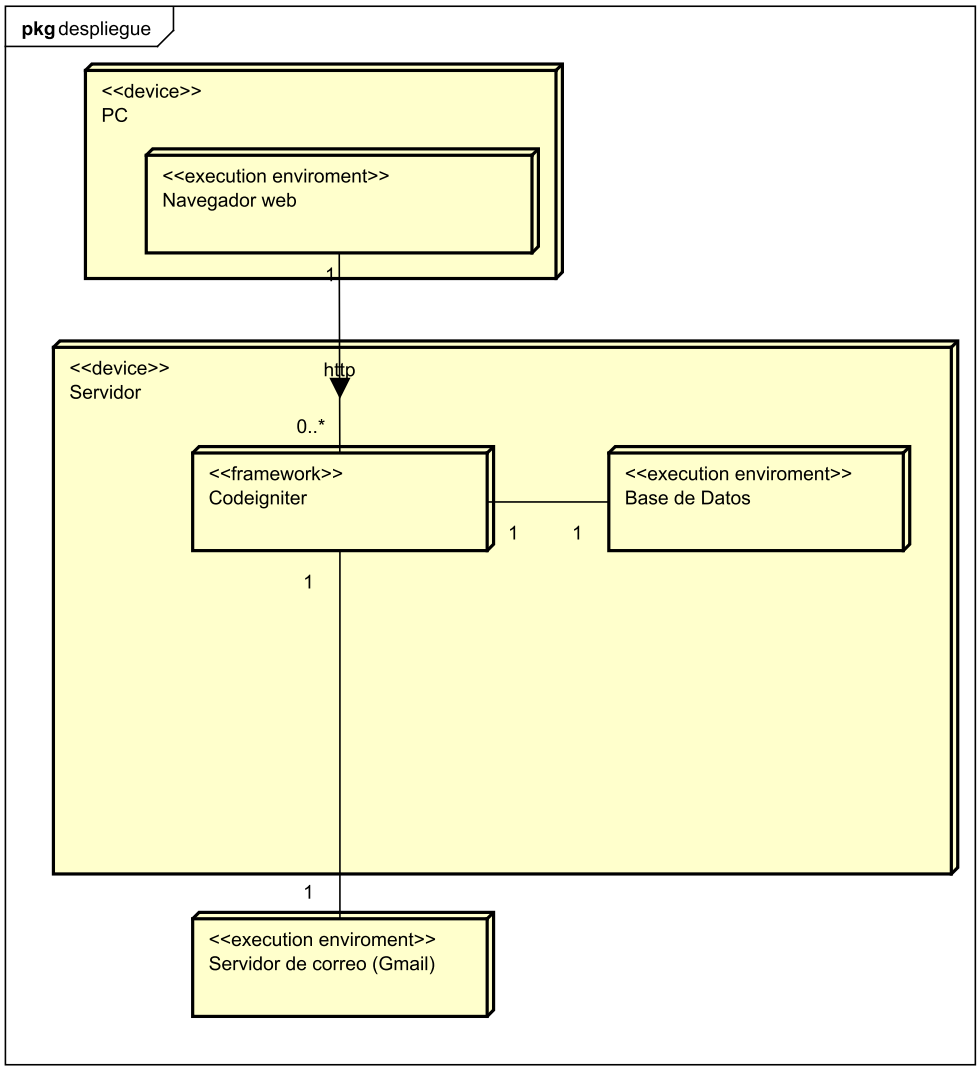


Figura 5.3: Diagrama de despliegue del Sistema

- «***device***»**Navegador web**. Es la parte del cliente donde se van a realizar las peticiones, HTTP en este caso aunque se tiene contemplado en el futuro la incorporación de un certificado de seguridad SSL para que las peticiones corran sobre HTTPs.

- «***device***»**Servidor apache**, donde se alojará el código de la aplicación para que sea interpretado por PHP.

- «***framework***» **Codeigniter**, utilizado para facilitar el desarrollo de la aplicación y que ha servido de esqueleto para realizar todas las funcionalidades requeridas.

- «***execution enviroment***» **Gestor de Base de Datos**. En este caso, el gestor se sitúa en el mismo servidor donde vamos a tener alojado el código de la aplicación. Si en algún momento se decidiese cambiar la localización de la Base de Datos, tendríamos que realizar cambios en el fichero de configuración de Codeigniter, más concretamente en el fichero *database.php*.

- «***execution enviroment***» **Servidor de correo**. Para ciertas funcionalidades, se necesita de un servicio de correo electrónico que envíe correos por medio de la aplicación. Para ello hemos utilizado el servicio que proporciona SMTP de Gmail.

5.3. Diseño en base a la seguridad

Actualmente, a la hora de diseñar cualquier aplicación, hay que tener en mente la seguridad de esta. A continuación se listan aspectos que se han tomado en cuenta a la hora de elaborar el sistema.

- **Balanceo de riesgo y seguridad**. Cuanta más complejidad tenga la aplicación, tendrá mas vulnerabilidades. Hay que considerar las medidas de seguridad necesarias que ayuden a mitigar cualquier característica que la haga vulnerable. Estos son aspectos como la validación de los campos de entrada, y evitar inyecciones SQL. De esto último, en gran medida se ocupa el framework utilizado. Una funcionalidad obvia, pero que necesita de un cuidado especial es el control de acceso de los usuarios. Hay que evitar que un usuario de tipo A acceda a la funcionalidad de un usario de tipo B.

- Tener especial cuidado con los ataques URL de tipo semántico, estos son los que por medio de la manipulación de URLs poder conocer ciertos datos de importancia como podrían ser las credenciales de acceso a la aplicación. En el fragmento de código de arriba, vemos que si se utilizara una implementación del formulario mediante el método GET, quedarían a la vista las credenciales de acceso usuario y contraseña.

```
1 https://virtual.lab.inf.uva.es/judocup/login?user=pepe&password=1234
2
```

  Para evitar esto, utilizaremos POST hacia el servidor en aquellos casos que la información sea confidencial y GET en aquellos casos en los que los datos de entrada no sean tan importantes. Estos últimos podrían ser por ejemplo la filtración de competidores por fecha y peso. En este caso, el usuario con rol de delegado podría guardar la URL completa en favoritos para poder acceder más rápidamente en el futuro.

- **Exposición de las credenciales de acceso**. Es un problema común a todos los frameworks. Los datos de usuario y contraseña son datos sensibles, y en los ficheros de configuración de estos frameworks se exponen en texto simple, es decir, sin cifrar. Para tener protegido este fichero, vamos a configurar el archivo *htaccess* para evitar que cualquier usuario no autorizado pueda acceder desde el exterior a éste y otros archivos de configuración.

- **Inyección SQL**. Evitaremos que un atacante intente introducir consultas que no sean legítimas por medio del filtrado y escapado de los datos en el lado del servidor.

- **Ataques por fuerza bruta**. Método que utiliza la prueba y error para poder acceder a ciertos datos dentro de la aplicación. En el diseño del control de acceso de los usuarios, si estos introducen mal cualquiera de los dos datos que se piden para acceder (correo electrónico y contraseña), se mostrará siempre el mismo mensaje de error, para así evitar que un usuario que este intentando acceder por fuerza bruta sepa si está introduciendo bien o mal el correo electrónico o sólo la contraseña.

- **Variables de sesión**. Sirven para que un usuario que ha iniciado sesión no tenga que volver a hacerlo dentro de un tiempo razonable. Tenemos que evitar que las *cookies* y variables de sesión permanezcan activas en un tiempo prolongado. Aunque no existe un criterio claro sobre la duración que tiene que tener una sesión, se ha decidido que tenga una duración máxima de treinta minutos por sesión. En el caso que la aplicación manejase transacciones bancarias, habría que reducir el tiempo de esta a unos 15 minutos por sesión.

- ***Password Sniffing***. El espionaje de contraseñas por medio de programas de terceros es posible en aquellas aplicaciones en las que los datos que se envían al servidor navegan por el protocolo HTTP y no por el protocolo HTTPS. La configuración de nuestro servidor web se realizará de tal manera que soporte HTTPS para así poder cifrar el canal de comunicación por donde se enviarán los datos.

Todos estos aspectos se tendrán en cuenta para que la aplicación que se está desarrollando sea lo más segura posible.

5.4. Diseño de la Base de Datos

En esta sección se va a tratar el proceso realizado a la hora de diseñar la Base de Datos de la aplicación, siguiendo la metodología propuesta por Thomas M.Conollt y Carolyn E.Begg[15]. El diagrama conceptual de datos y esquema relacional, figuras 5.4 y 5.8 respectivamente respetan el último informe de la Asociación Española de Protección de Datos[16] (AEPD), en cuanto a la publicación de datos sensibles ya que en ésta BBDD no se tienen este tipo de datos. El artículo 5 de este documento cita textualmente lo siguiente:

- **Licitud, lealtad y transparencia**. Los datos personales deben ser tratados de manera lícita, leal y transparente. En nuestro modelo conceptual de datos, no tenemos datos personales tales como tarjetas de crédito, cuentas bancarias, etc. Es verdad que tenemos un campo identificativo para competidores y árbitros que es el DNI, el cual hay que tener especial cuidado de no mostrar a usuarios que no tengan el consentimiento de visualizarlo. Actualmente, en la Federación castellano-leonesa de judo, al principio de la temporada se firma una hoja de consentimiento la cual a grandes rasgos indica que proporcionas tus datos personales a la Federación, entre los cuales se encuentra el DNI.

- **Limitación de la finalidad**. Los datos que se recojan deben tener fines justificados. Cada uno de los campos que recogen cada una de las entidades del diagrama de la figura 5.4 tienen su justificación, es decir, van a ser utilizados de manera lícita y leal para el cometido indicado en el capítulo de análisis.

- **Minimalidad de los datos**. Datos limitados a lo necesario. En el modelo creado se recogen los datos mínimos para poder cumplir con los requisitos y resolver todas las preguntas que se formularon en el capítulo cuatro (4.2.1) de esta memoria.

- **Exactitud**. Datos exactos y si fuera posible actualizados. En este caso, de la actualización y validación de los datos se ocupa el delegado de la Federación.

- **Integridad y confidencialidad**. Los datos confidenciales que se guarden en la base de datos tienes que estar almacenados de forma segura. Datos tales como las claves de acceso de usuarios van a estar encriptados en la base de datos con una encriptación fuerte (*bycrypt*).

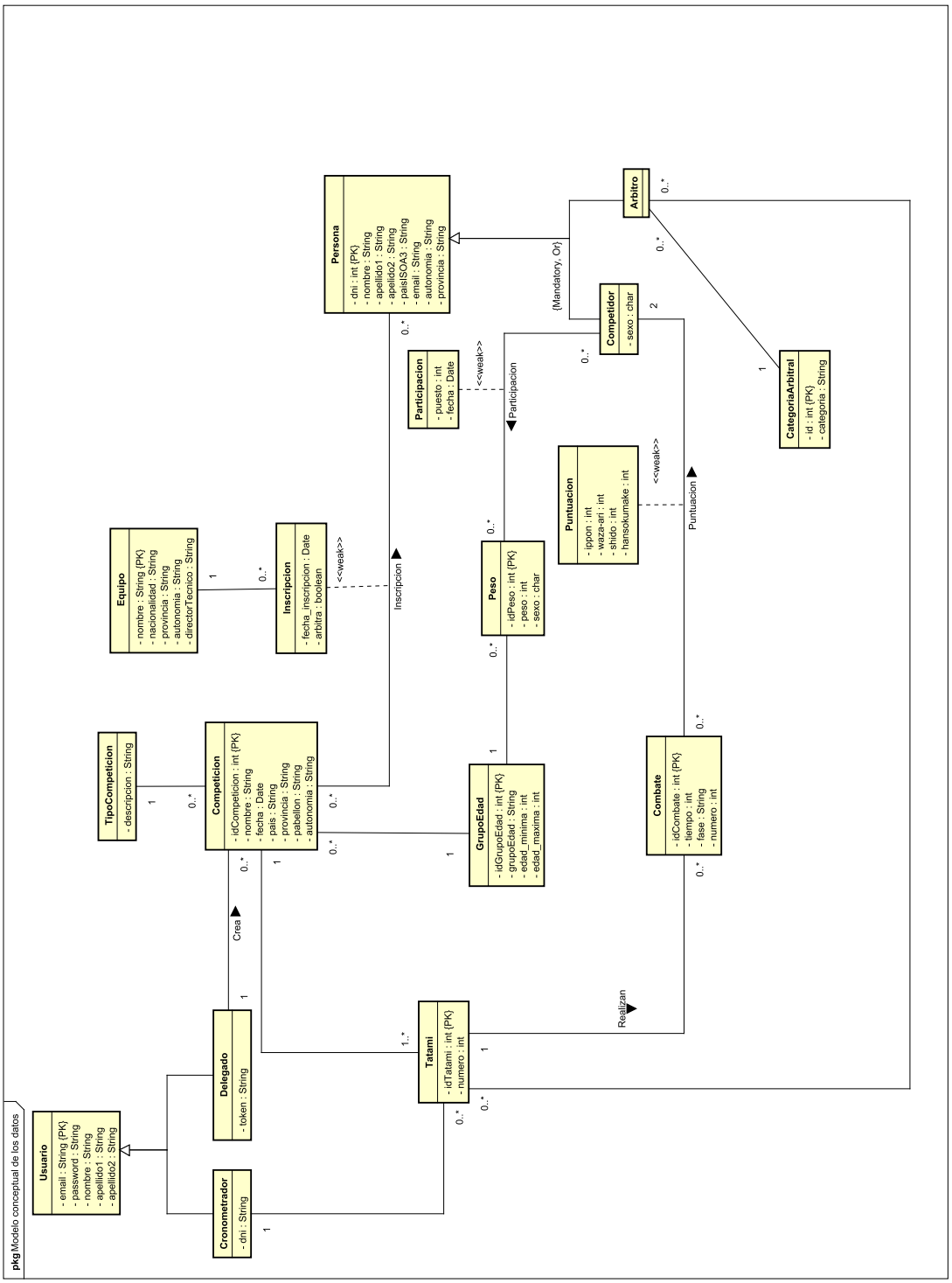


Figura 5.4: Modelo conceptual de los datos

Como se puede ver en el modelo conceptual de datos creado, los datos de los usuarios que van a acceder en la aplicación se guarda en el esquema con las entidades que forman la generalización-especialización Usuario, Cronometrador y Delegado. Se ha optado por almacenarlo en la misma base de datos donde se guardarán el resto de elementos tales como las competiciones, combates, etc, siendo así más fácil el mantenimiento de ésta. Es verdad que la opción de almacenar los datos con las credenciales de acceso de los usuarios en una Base de Datos diferente aumenta los niveles de seguridad, pero como se ha comentado en el apartado 5.5 de esta memoria, se ha configurado el servidor de tal manera que sólo el administrador pueda acceder a la Base de Datos. Por otro lado, muchas de las entidades que se muestran en el modelo de la figura 5.4 tienen como referencia las clases del modelo de dominio inicial, aunque existen diferencias entre ambos modelos. Por ejemplo, aparecen las entidades inscripción y

participación.

A continuación se van a detallar las entidades que se considera que necesitan alguna aclaración adicional a lo ya visto en el análisis.

- **Delegado**. El atributo *token*, que tiene esta entidad es utilizado para poder recuperar la contraseña del delegado. Este token es una variable de tipo *varchar* de al menos 60 caracteres que es descifrable en un solo sentido, lo que incrementa su seguridad.

- **Usuario**. Entre los atributos destacables de esta entidad se encuentra el *password*. Como en el caso anterior también es un *varchar* de al menos 64 caracteres y tiene la misma explicación. Para evitar que si algún atacante accede a la Base de Datos pueda conocer las contraseñas de los usuarios, se encripta la contraseña con una función *hash*.

Las entidades que aparecen nuevas con respecto al modelo de dominio son: Inscripción y Participación. Ambas relaciones que tienen un muchos a muchos en ambas direcciones de la relación. Concretamente, la entidad Inscripción es la relación intermedia entre la entidad Competición y la entidad Persona; y la entidad Participación es la relación intermedia entre la entidad Peso y la entidad Competidor. En el modelo relacional de la figura 5.4 se pueden observar diferentes tipos de relaciones. A continuación se explicará como se van a resolver para pasar el modelo de entidad relación al modelo relacional.

Relaciones binarias uno a muchos (1:*)

En este tipo de situaciones, denominaremos como entidad *padre* a la entidad situada en el lado 1 de la relación, y la entidad situada en el lado mucho la denominaremos entidad *hija*[15]. A la hora de pasar este tipo de relación al modelo relacional realizaremos lo siguiente: Definiremos como entidad padre a la entidad situada en el lado *uno* de la relación, mientras que la entidad situada en el *lado muchos* se denomina entidad hija. Todos lo atributos de la clave principal de la entidad padre se copiaran a la entidad hija, así actuará como clave foránea.

Un ejemplo de este tipo de relación sería entre las tablas *Competición* y *TipoCompetición*. La entidad padre en este caso es *TipoCompetición* y la entidad hija es *Competición*, por lo que esta última tendrá además de los atributos que tiene ahora mismo, la clave primaria de la entidad padre. El objetivo principal de realizar esta operación es el de que la clave primaria copiada actúe como clave foránea. El resultado de esta conversión lo podemos ver en la figura 5.5.

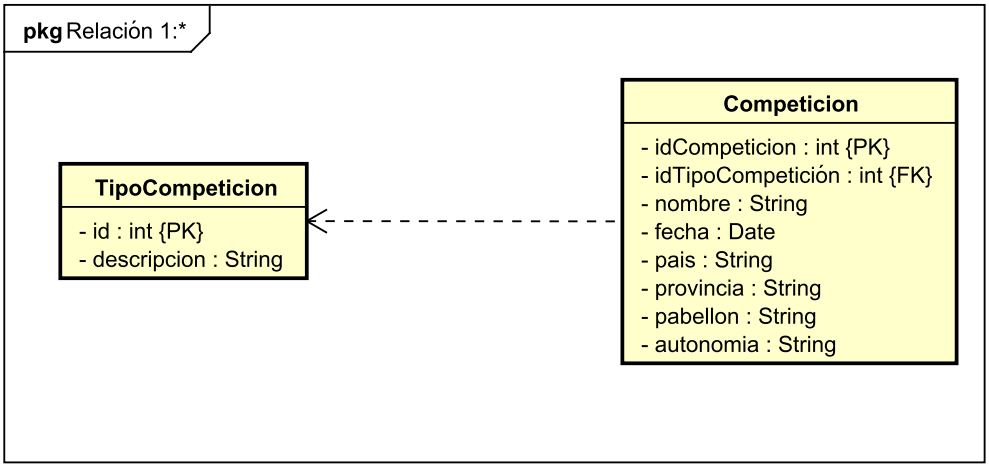


Figura 5.5: Relación binaria 1:*

Relación de superclase/subclase

Para cada relación superclase/subclase del modelo conceptual de datos, identificaremos como entidad padre a la superclase, y como entidad hijas a cada una de las subclases. Existen varias opciones de pasar al modelo relacional, pero en es este caso, por facilidad de implementación y pensando en la

mantenibilidad de las tablas, se ha optado por utilizar en la herencia de nuestro modelo conceptual la opción de *mandatory non-disjoint*, es decir, una única tabla con uno o más campos discriminatorios para distinguir el tipo de entidad. Otra de las razones de la elección de este tipo de relación es porque no existe una cantidad abultada de atributos entre las clases hijas.

El resultado de la aplicación de la relación anterior es el que se muestra en la figura 5.6.

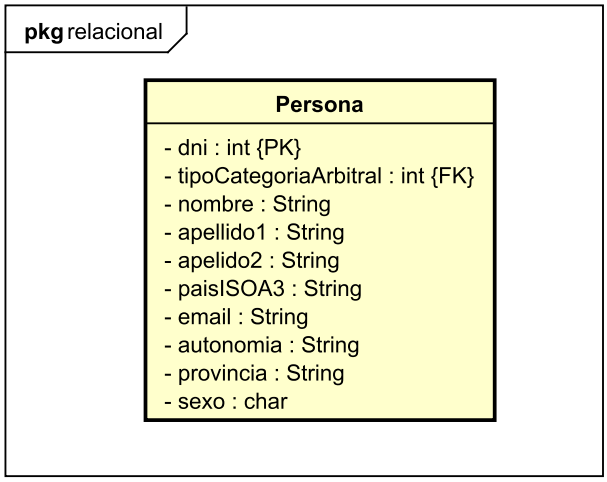


Figura 5.6: Relación de superclase/subclase

El atributo discriminatorio en este caso coincide con la clave foránea a la categoría arbitral que puede o no tener una cierta persona involucrada en el mundo del Judo.

Relación binaria de muchos a muchos (..* : ..*)

Para cada relación binaria de muchos a muchos crearemos una tabla intermedia también llamada tabla asociativa. En ésta añadiremos los atributos que formen parte de la relación y los atributos que formen la clave primera de ambas clases que actuarán de claves foráneas y a su vez como clave principal. Un ejemplo del resultado de aplicar ésta relación en nuestro modelo conceptual es el que se muestra en la figura 5.7.

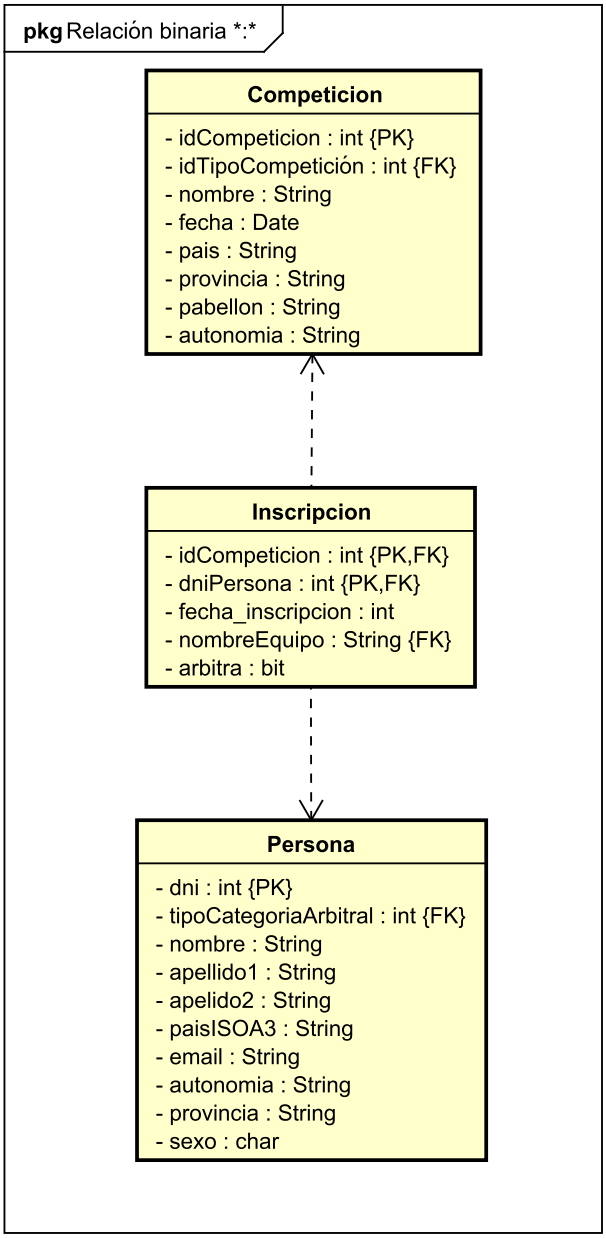


Figura 5.7: Relación binaria muchos a muchos (..* : ..*)

El resultado de realizar estas transformaciones es el siguiente modelo relacional de la figura 5.8.

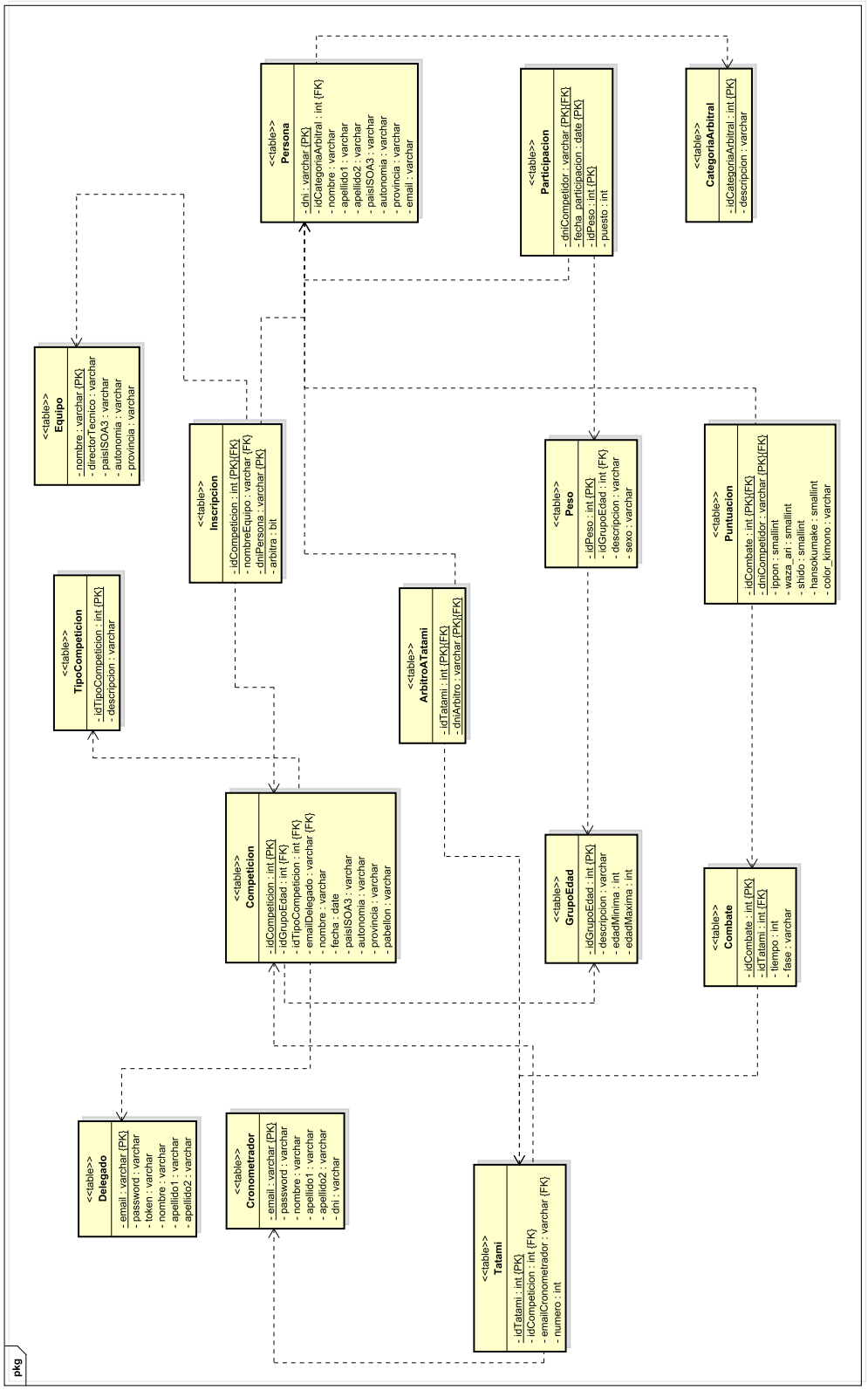


Figura 5.8: Modelo relacional

5.5. Script base de la BBDD

En esta sección se presenta el script generado para crear las tablas, en función del modelo relacional (figura 5.8). De las quince tablas necesarias para el correcto funcionamiento de la aplicación, las que tienen valores insertados por defecto son:
La tabla *Peso*, la cual contiene todos los valores de la actual temporada 2017-2018, cuyos datos han

sido sacados de la Federación Española de Judo y Deportes Asociados[17]. La tabla *CategoriaArbitral* también tiene datos por defecto. En ésta se encuentran las diferentes categorías arbitrales[17]. En cualquier momento en el que se quiera añadir una nueva categoría arbitral, simplemente habría que añadirlo a esta tabla. Otra de las tablas que tiene tuplas por defecto es *TipoCompeticion*, también han sido sacadas de la Federación Española, y además se ha añadido alguna categoría específica para la Federación Castellano Leonesa de Judo y Deportes Asociados. Por último, tenemos la tabla *GrupoEdad*, la cual representa a los diferentes grupos de edad que hay actualmente en las competiciones de Judo.

El apéndice D de esta memoria, donde se encuentra el script completo y funcional de la Base de Datos de esta aplicación, JudoCup.

5.6. Patrones utilizados

5.6.1. Patrón Modelo-Vista-Controlador

Es un patrón cuya función principal es la separación de la lógica de negocio de los datos. Tenemos tres componentes en este patrón, el modelo, el cual representa la información con la que el Sistema trata. El controlador el cual responde a los eventos que se producen en la página web. Este se comunica con la capa de vistas y la capa de modelo. Por último, el paquete con las vistas cuya función es presentar al modelo(información) en el formato correspondiente, tablas, formularios etc.

En la figura 5.12 podemos ver cómo se comunican los diferentes paquetes.

5.6.2. Patrón comando estrategia

El patrón estrategia o patrón comando estrategia[18] es un patrón de diseño que permite mantener un conjunto de algoritmos, en este caso en particular de emparejamientos, en los cuales el objeto cliente puede seleccionar dinámicamente(en ejecución) cuál de todos ellos instanciar.

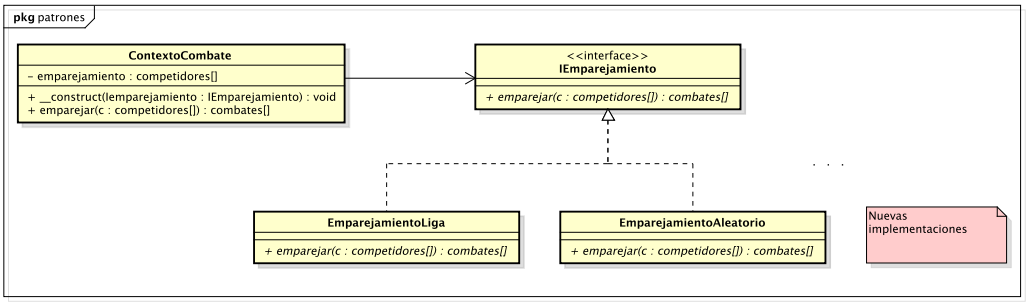


Figura 5.9: Patrón comando estrategia

En la figura 5.9, tenemos el modelo de diseño del patrón comando utilizado. A continuación se explicará en detalle la función de cada una de ellas.

- **ContextoCombate**. Clase encargada de utilizar los algoritmos, delegando en la jerarquía de estrategias.
- **IEmparejamiento**. Interfaz común a todos los algoritmos de emparejamiento.
- **EmparejamientoLiga**. Implementa un algoritmo concreto, en este caso el sistema round-robin mencionado en el capítulo de análisis.
- **EmparejamientoAleatorio**. Implementa un algoritmo concreto, en este caso el sistema aleatorio.

5.7. Diagramas de clases de diseño

Aquí se mostrarán los diversos diagramas de clases en diseño.

5.7.1. Paquete de Controladores (Controllers)

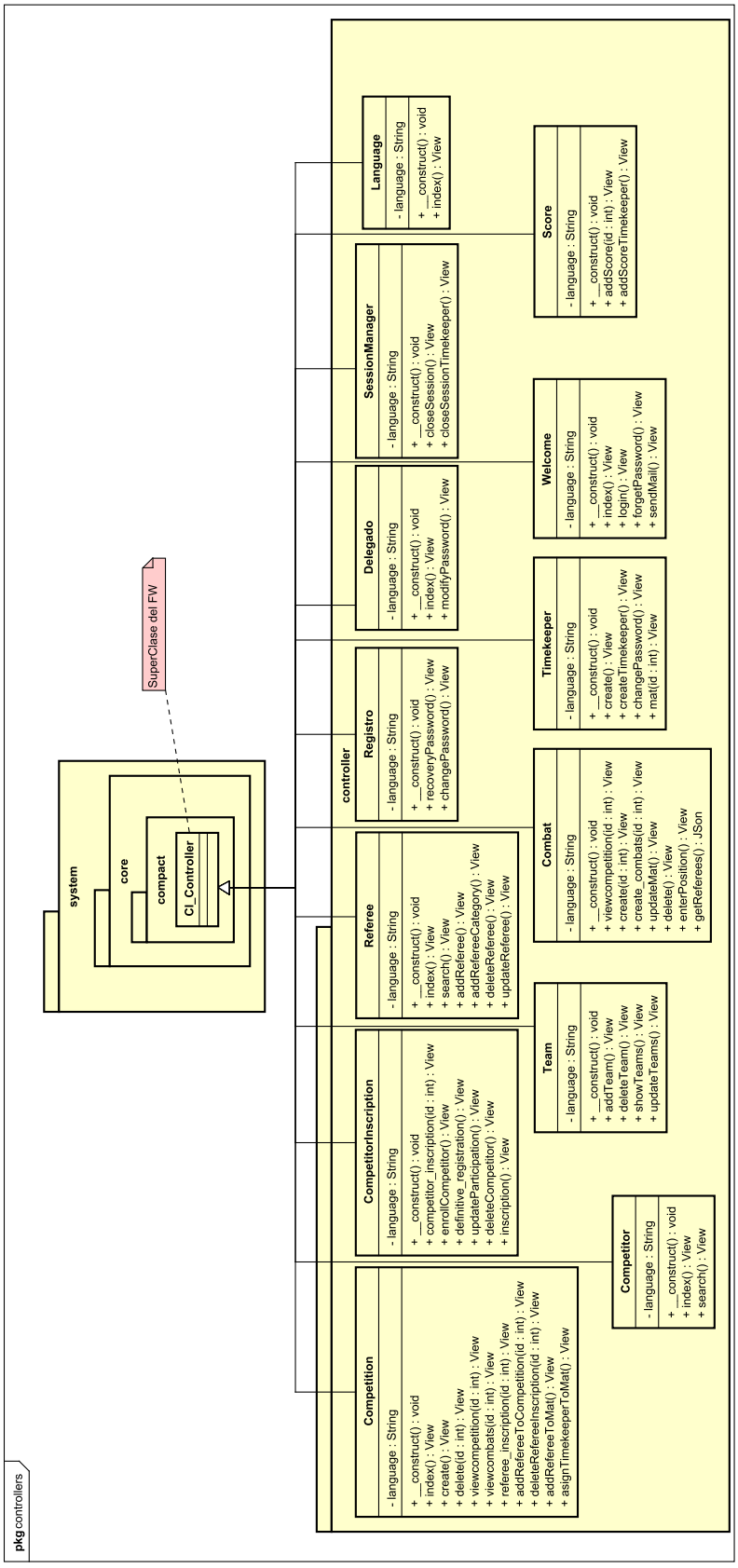


Figura 5.10: Paquete con los controladores

En este paquete encontraremos aquellas clases encargadas de gestionar las peticiones URL que se hacen a la página web. Cualquier petición que llega a la aplicación la tiene que gestionar el controlador

correspondiente

- **Competition.** Encargado de gestionar la creación y eliminación de competiciones. Realiza a mayores las inscripciones de árbitros.

- **Referee.** Encargado de controlar la creación, eliminación, y actualización de árbitros.

- **Combat:** Gestiona todo lo relacionado con los combates y las puntuaciones de los competidores.

- **Timekeeper**. Gestiona todo lo relacionado con los cronometradores.

- **Team.** Gestiona lo relacionado con los equipos de Judo.

- **Score**. Creación de puntuaciones, introducción de las puntuaciones obtenidas en los combates y eliminación.

- **Delegado**. Gestión de la recuperación de contraseñas del delegado, edición etc.

- **SessionManager.** Gestiona las sesiones de los usuarios que inician sesión.

- **Competitor**. Gestiona todo en relación con los competidores.

- **Welcome**. Inicio, cambios de contraseña.

- **Registro**. Gestiona las recuperaciones de contraseña.

- **CompetitorInscription.** Todo lo relacionado con las inscripciones de los diferentes competidores lo gestiona este controlador.

- **Language**. Gestiona el cambio de idioma de la aplicación.

5.7.2. Paquete de modelos

En este caso, los modelos heredan de la clase *CI_Model*. Por cuestión de espacio no se muestra el paquete donde se encuentra esta clase, pero estaría situado en el mismo que el de los controladores, mostrado en la figura 5.10.

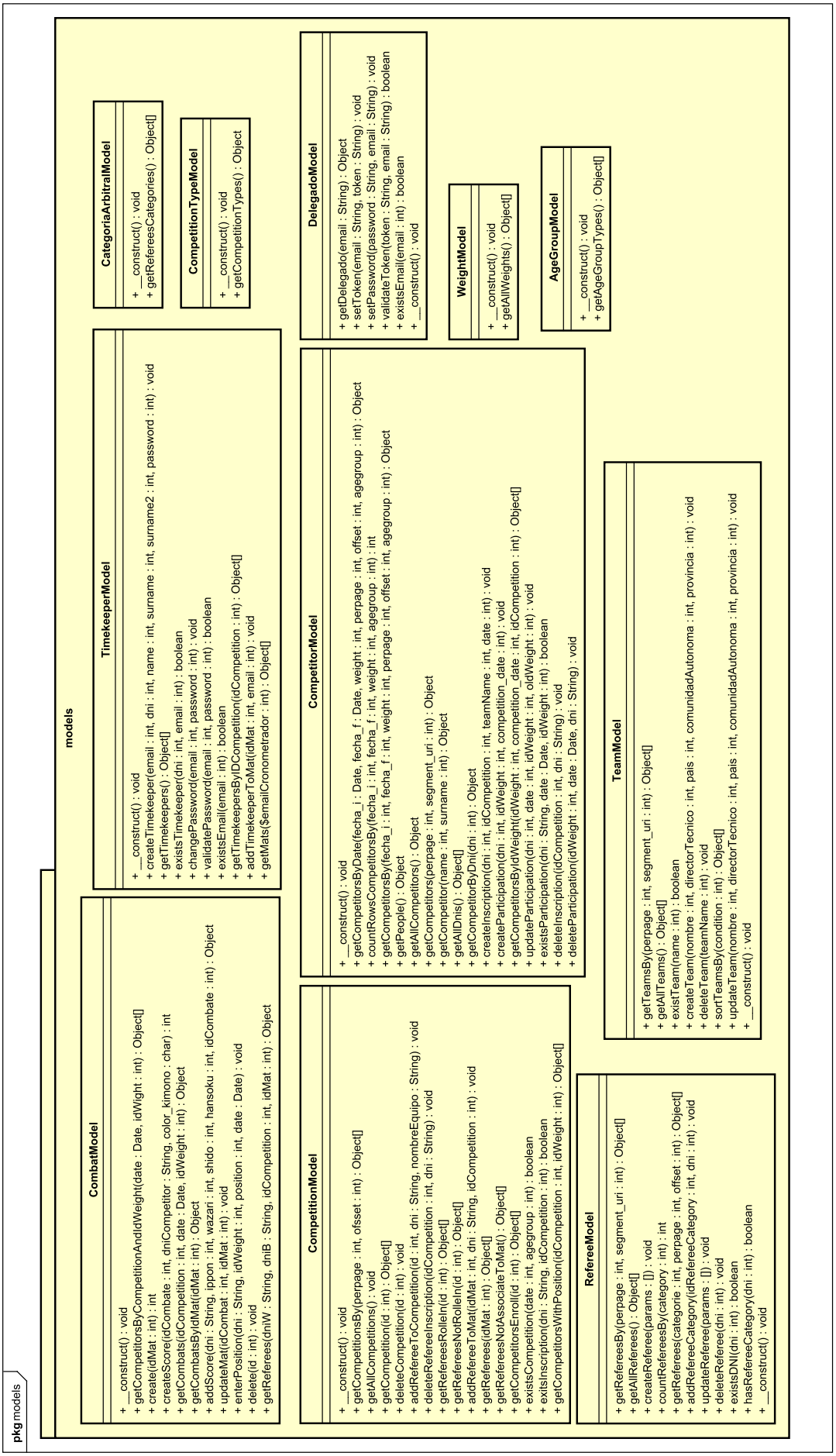


Figura 5.11: Paquete con los modelos

5.8. Comunicación entre paquetes

A continuación se va a explicar como se produce la comunicación entre paquetes de la aplicación. El ejemplo de la figura 5.12 detalla la arquitectura utilizada para el caso de uso de crear un combate (CU-13)

- ***assets***
  - ***javascript*** Ficheros javascript importados de diferentes librerías y propios.
  - ***css*** Ficheros css importados de diferentes librerias y frameworks, una de ellas Materialize.
  - ***fonts*** Fuentes de diferentes tipos, todas ellas de Materialize.
- ***views***
  - ***Estructura*** Este directorio contiene las vistas comunes a cada una de las vistas. Estas son dos, header_principal.php y footer.php.
  - ***Combat*** Contiene las vistas relacionadas con los combates.
- ***controllers*** La creación de combates se realiza mediante el controlador Combat.
- ***models*** Modelos, dos este caso, CompetitionModel y CombatModel.
- ***database*** Base de datos de la aplicación de la cual se obtienen las tuplas, se realizan inserciones, eliminaciones, actualizaciones, etc.
- ***src***
  - ***AlgoritmoEmparejamientos*** Contiene la implementación de los algoritmos de emparejamiento usando el patrón comando-estrategia.

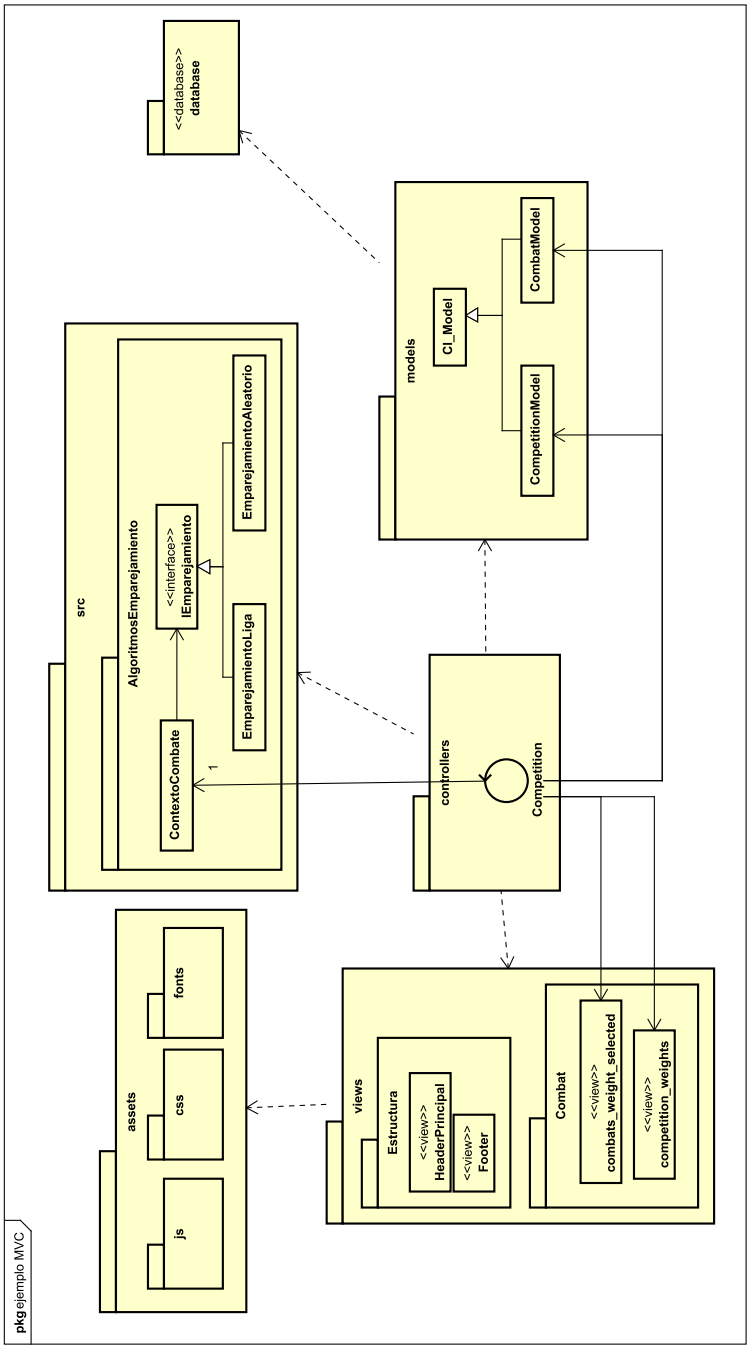


Figura 5.12: Comunicación entre paquetes y clases

5.9. Diagrama de casos de uso de la etapa de Diseño

Se ha optado por desarrollar el caso de uso que más complejidad puede llegar a tener a la hora de comprender el funcionamiento de este. El caso de uso que vamos a detallar es el de Generar sorteo (CU-13).

Podemos ver aspectos explicados anteriormente como la comunicación entre los diferentes paquetes de la arquitectura (vistas, modelos, controladores) en la figura 5.13. En este caso en concreto, el usuario envía por medio de un POST hacia el servidor tres parámetros, el identificador de la competición, tipo de sorteo y el identificador del peso. A continuación hay que validar los parámetros introducidos y en caso de error, mostrarlo(*redirect(error)*). Una vez validado todo, el controlador realiza todas las funciones necesarias en comunicación con los modelos para llevar a cabo el caso de uso de Generar el sorteo.

La implementación del patrón comando se muestra en la figura 5.14. En este caso se instancia la

clase *EmparejamientoLiga*, pero se realizaría de la misma manera si se quisiera instanciar cualquiera de las otras subclases de *IEmparejamiento*.

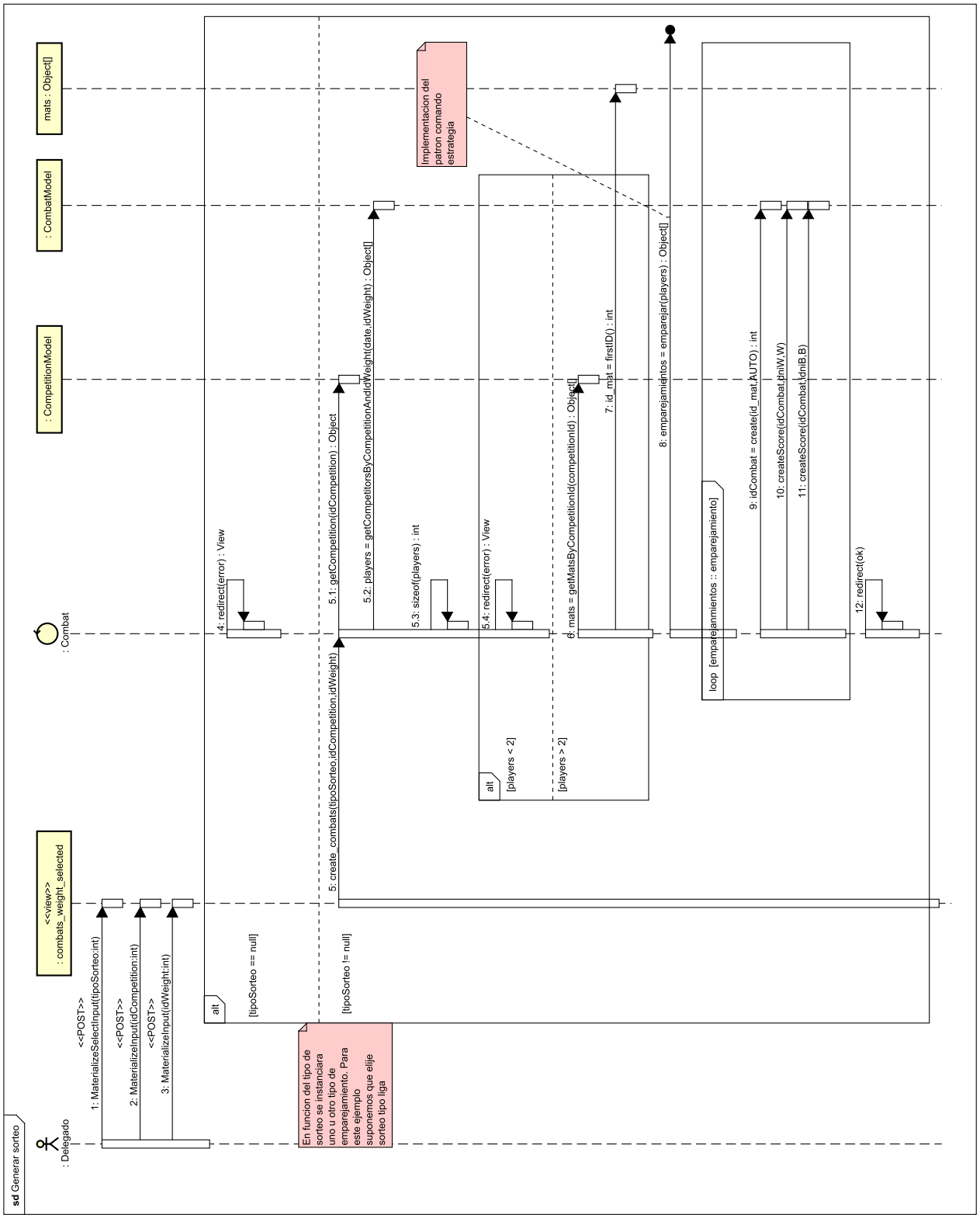


Figura 5.13: Diagrama de secuencia general del caso de uso *Generar sorteo*

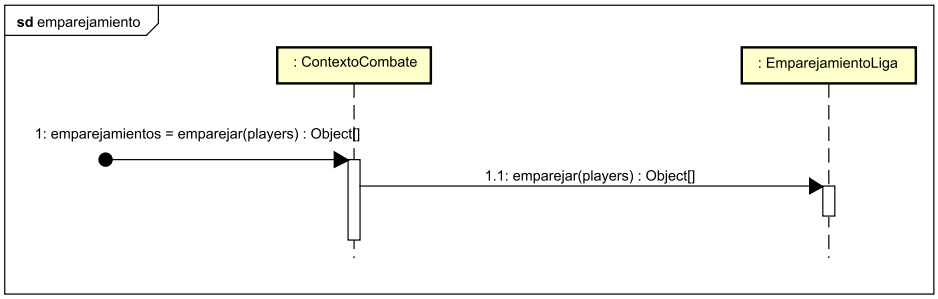


Figura 5.14: Implementación patrón comando-estrategia

Capítulo 6

Implementación

6.1. Introducción

En este capítulo se detallará en profundidad el proceso de implementación de la aplicación, basándose en la arquitectura planteada en los capítulos anteriores de diseño y análisis. Se describirá como se han implementado ciertas funcionalidades de la aplicación, tales como los algoritmos de emparejamiento, cambio de idioma en la aplicación u otros.

Se van a proporcionar ejemplos de porciones de código de cada una de las diferentes clases del patrón MCV que utiliza el framework de Codeigniter, modelos controladores y vistas. En cuanto a la implementación de las animaciones y efectos visuales (como por ejemplo arrastrar componentes de un lado a otro) también se realizará una breve explicación de cómo se ha hecho en este capítulo (sección 6.4).

Los lenguajes y librerías utilizadas para realizar la implementación de la aplicación han sido:

- PHP como lenguaje de servidor o *backend* junto con el framework de Codeigniter.
- HTML5 para la generación de las diferentes vistas.
- JavaScript y JQuery para la inicialización de componentes de la interfaz de usuario, validación de formularios, animaciones, etc.
- Materialize (CSS + JavaScript) para el diseño de la interfaz de usuario.

6.2. Clase Controlador

Como se ha dicho en capítulos anteriores, la clase controlador es la encargada de gestionar las peticiones que se hacen a la web.Toda llamada tiene que pasar obligatoriamente por un controlador. Una de las características principales que tienen los controladores de este framework es que heredan de *CI_Controller*, para así gestionar las peticiones que se hagan a dicho controlador. En la figura 6.1 vemos un ejemplo de una clase controlador.

```php
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Team extends CI_Controller
{
    const PERPAGE = 7;
    const NUM_LINKS = 10;
    private $language;
    function __construct()
    {
        parent::__construct();
        if (!$this->session->has_userdata('user'))
            redirect(base_url() . '?msg=' . SESSION_EXPIRED, 'refresh');

        $this->load->model('teamModel');
        $this -> language = $this->session->userdata('language');
        include "application/src/Pagination.php";
    }
    public function addTeam()
    {
        $nombre = $this->input->post('nombreEquipo');
        $directorTecnico = $this->input->post('directorTecnico');
        $pais = $this->input->post('pais');
        $comunidadAutonoma = $this->input->post('comunidadAutonoma');
        $provincia = $this->input->post('provincia');

        $this->load->model('teamModel');
        $this->teamModel->existTeam($nombre) ?
            redirect(base_url() . 'team?msg=' . ERROR_TEAM_EXISTS, 'refresh') :
            $this->createTeam($nombre, $directorTecnico, $pais, $comunidadAutonoma,
    $provincia);

    }
```

Figura 6.1: Fragmento de código de un controlador

Normalmente, en cada controlador se define un constructor que se va a encargar de inicializar todos aquellas variables, ficheros; APIs que necesite el propio controlador. En nuestro caso, también se ha utilizado este método para comprobar que el usuario ha iniciado sesión antes de llamar a alguno de los métodos de la clase.

Por otro lado tenemos los métodos de clase. Si queremos que se pueda llamar a un determinado método desde una petición URL, se tiene que definir como un método público, en caso contrario, no funcionaría. En el ejemplo mostrado en la figura 6.1, la función *addTeam* recibe cinco parámetros por medio de un POST utilizados para crear un nuevo equipo. Para llamar a la función de la figura anterior, hay que realizar un POST con las variables que indica el método (*addTeam*) a la siguiente dirección URL https://dominio/team/addTeam.

Al final de cada método de la clase Controlador, normalmente tiene que redireccionar a una vista de error o de éxito.

6.3. Clase Modelo

Estas clases heredan de *CI_Model* y se utiliza para realizar todo tipo de operaciones con la Base de Datos. Es muy flexible puesto que se pueden realizar consultas con SQL estándar y con funciones propias del framework. A continuación se van a mostrar ejemplos de cada una de ellas. La elección pasó por lo fácil o difícil que fuera la consulta, si se trata de una consulta sencilla, es decir, obtener una fila completa, o un cierto atributo, utilizamos los métodos que proporciona el framework. En el caso de que la consulta sea algo más compleja, donde haya varias tablas involucradas, usaremos SQL estándar. Un ejemplo de SQL estándar es el que se muestra en la figura 6.2.

```php
/**
     * @param $date Fecha en la que se realiza la competicion
     * @param $idWeight Id unico del peso del competidor
     * @return Persona [] . Que estan registradas en el peso indicado por parametro en la fecha indicada tambien por parametro
     */
    public function getCompetitorsByCompetitionAndIdWeight($date,$idWeight) {
        $sql = 'SELECT P.*
                FROM Persona P , Participacion PA
                WHERE P.dni = PA.dniCompetidor AND PA.fecha_participacion = ? AND PA.idPeso = ?';
    $this->db->trans_start();
        $query = $this->db->query($sql, array($date, $idWeight));
    $this->db->trans_complete();
    if (!$this->db->trans_status()) throw new Exception('Error, ');
        return $query -> result();


    }
```

Figura 6.2: Fragmento de código realizando una consulta SQL estándar

La función devuelve los competidores de una determinada competición y peso.

La figura 6.3 es un ejemplo de como utilizar las funciones que proporciona el framework para actualizar de una forma más cómoda los valores de una determinada tabla.

```php
public function addScore($dni,$ippon,$wazari,$shido,$hansoku, $idCombat) {

        $score = array(
            'ippon' => $ippon,
            'waza_ari' => $wazari,
            'shido' => $shido,
            'hansokumake' => $hansoku
        );

        $this->db->trans_start();
        $this -> db -> where ('idCombate',$idCombat);
        $this -> db -> where ('dniCompetidor',$dni);
        $this -> db -> update('Puntuacion',$score);
        $this->db->trans_complete();
        if (!$this->db->trans_status()) throw new Exception('Error, no se pudo añadir la puntuacion');

    }
```

Figura 6.3: Fragmento de código con funciones del framework

6.4. Vistas de la aplicación y animaciones

Las vistas son ficheros php en cuyo interior se encuentra lenguaje HTML y en la mayoría de los casos también lenguaje PHP y JavaScript.

Animaciones con JQuery y JavaScript

JQuery y JavaScript se han utilizado para infinitud de cosas, una de las principales es para la inicialización de componentes del framework *frontend* de Materialize[1], otra, es la validación en el lado del cliente de los campos introducidos. También se ha utilizado JQuery y JavaScript para implementar funcionalidades más complejas como arrastrar componentes de la interfaz, tales como listas de un lado a otro (figura 6.4).

Este es el fragmento de código utilizado para actualizar inscripciones por medio de animaciones JS y JQuery, basadas en el tutorial[19]. Podemos ver las tres funciones principales, *drag, drop y allowdrop*.

```
<script>
    function allowDrop(ev) {
        ev.preventDefault();
    }

    function drag(ev) {

        ev.dataTransfer.setData("text", ev.target.id);
    }

    function drop(ev) {

        var data = ev.dataTransfer.getData("text");
        ev.target.appendChild(document.getElementById(data));
        data['dni'] = document.getElementById(data).elements.item(0).value;
        data['idCompetition'] = document.getElementById(data).elements.item(1).value;

        console.log('Id peso actualizacion -> ' + ev.target.id);
        $.ajax({
            type: "POST",
            url: "<?php echo base_url()?>competitorInscription/updateParticipation",
            data: {
                dni: document.getElementById(data).elements.item(0).value,
                date: document.getElementById(data).elements.item(1).value,
                oldWeight: document.getElementById(data).elements.item(2).value,
                idWeight: ev.target.id,
                competitionId:<?php echo $competition->idCompeticion?>},
            success: function (result) {
                location.reload();
                console.log(result);


            },
            error: function (result) {
                console.log(result);
            }
        })
    }
</script>
```

Figura 6.4: Fragmento de código con animaciones usando JQuery

Cabe destacar que en la función *drop*, se realiza una llamada Ajax[20] al servidor para guardar todos los datos realizados, este tipo de llamadas son asíncronas las cuales permiten que la operación retorne el control al programa antes de que hayan terminado mientras siguen operando en segundo plano.

6.5. Implementación de la internacionalización

Uno de los aspectos a destacar de la implementación de esta aplicación es que está en diferentes idiomas. De momento tiene que soportar dos lenguajes, inglés y español. Esto se va a realizar gracias a la ayuda del framework, ya que tiene ciertos ficheros que facilitan la internacionalización de la app.

Dentro del directorio principal, *application*, tenemos una carpeta llamada *language*, en la cual se colocan los diferentes ficheros de idioma. En este directorio, por cada idioma nuevo que se implemente, se tiene que crear una nueva carpeta con el nombre del lenguaje.

```
1  -language
2    -english
3        -files_lang.php
4    -spanish
5        -files_lang.php
```

Figura 6.5: Directorio con los ficheros que forman el diccionario

Cada una de las carpetas tiene que tener el mismo número de ficheros php y dentro de cada fichero php el mismo número de claves. El diccionario está implementado por medio de clave y valor. La clave es común a ambos ficheros y es el valor (la traducción) lo que tiene que cambiar. A continuación se muestra un ejemplo del diccionario

Fichero competition_lang en la carpeta de inglés

```
1 $lang['competition_create_name'] = 'Name';
2 $lang['competition_create_date'] = 'Date';
3 $lang['competition_create_country'] = 'Country';
```

Figura 6.6: Fragmento de código que contiene el diccionario de datos en inglés

Fichero competition_lang en la carpeta de español

```
1 $lang['competition_create_name'] = 'Nombre';
2 $lang['competition_create_date'] = 'Fecha';
3 $lang['competition_create_country'] = 'Pais';
4
5
```

Figura 6.7: Fragmento de código que contiene el diccionario de datos en castellano

Podemos observar la estructura mencionada en el párrafo anterior, mismas claves, pero diferentes valores.

Una vez tenemos el diccionario completado, la implementación para intercambiar entre uno u otro lenguaje cuando el usuario cambie el idioma corre por nuestra cuenta. Por defecto, el idioma por defecto es el español, pero el usuario puede cambiar cuando lo desee al inglés. Si cambia, el valor durará lo que dura la sesión activa, es decir 30 minutos desde que el usuario no ha realizado ninguna interacción más con el Sistema.

```
class Language extends CI_Controller {


    public function index() {
            $language = $this -> input -> get('language');
            $request_from = $this -> input ->get ('url');
            $this -> session -> set_userdata('language',$language);
            redirect($request_from,'refresh');
    }
}
```

Figura 6.8: Fragmento de código que muestra el cambio de idioma de la aplicación

La implementación de intercambio de lenguaje se hace en la clase controladora *Language* la cual recibe por medio de un POST el lenguaje que se quiere cambiar y se guarda en sesión. A partir de entonces, cada vez que se carga el diccionario de datos, se hará el que indique el idioma de la sesión, si es que está activada.

6.6. Implementación del algoritmo de emparejamiento

En el capítulo de diseño se comentó que se utiliza el patrón de diseño comando-estrategía para implementar los diferentes tipos de emparejamiento actuales y futuros. Se va a mostrar en detalle la implementación del algoritmo de todos contra todos o sistema de liga[13].

```
class EmparejamientoLiga implements IEmparejamiento
{

    public function emparejar($competidores)
    {
        $combates = array();
        foreach ($competidores as $i) {
            foreach ($competidores as $j) {
                if($i == $j) continue;
                $emparejamiento = array($i,$j);
                sort($emparejamiento);
                if(!in_array($emparejamiento,$combates))
                    $combates[] = $emparejamiento;
            }
        }
        return $combates;
    }
}
```

Figura 6.9: Fragmento de código con la implementación del algoritmo de todos contra todos

El método emparejar recibe un array de competidores y lo que hace es emparejar si y solo si son diferentes competidores. Lo que devuelve es otro array con los emparejamientos entre los competidores.

Si en un futuro se quisieran implementar nuevos algoritmos de emparejamiento, se tendría que crear una nueva clase que implementase la interfaz *IEmparejamiento*.

Capítulo 7

Pruebas

7.1. Introducción

Una vez desarrollada la aplicación, es obligatorio realizar las pruebas necesarias para dar cierto grado de fiabilidad a las tareas que realiza. El conjunto de pruebas que se van a realizar están orientadas a la detección de fallos con respecto a los requisitos planteados en capítulos anteriores de la memoria.

Utilizaremos las pruebas de caja negra, que sirven para realizar pruebas funcionales, basadas en funciones o características del Sistema y su interacción con otros Sistemas o Componentes. Los casos de prueba se van a basar en los Casos de uso definidos en apartados anteriores.

7.2. Pruebas de funcionamiento general

Como requisito previo a la realización de los siguientes casos de prueba, el usuario con rol de Delegado está identificado en el Sistema. A continuación se muestran algunas de las pruebas de caja negra con valores límite realizadas, sobre los dos entornos de trabajo, el de producción (máquina virtual de la Escuela de Ingeniería Informática) y sobre el entorno de desarrollo.

CP-01	Crear un árbitro
Descripción	Crear un árbitro con el rol *Administrador* con un DNI que no corresponda a ningún árbitro ni competidor
Entrada	DNI: 71156516J, nombre: Pablo, primer apellido: García, segundo apellido: Sanz, categoría arbitral: 1
Salida esperada	Creado un nuevo árbitro en el Sistema con los datos introducidos de la entrada
Salida obtenida	Salida esperada.

Tabla 7.1: CP-01: Crear árbitro

CP-02	Crear un árbitro existente
Descripción	Crear un árbitro con el rol *Administrador* con un DNI existente
Entrada	DNI: 71156516J, nombre: Pablo, primer apellido: García, segundo apellido: Sanz, categoría arbitral: 1
Salida esperada	Mensaje de error
Salida obtenida	Salida esperada.

Tabla 7.2: CP-02: Crear árbitro existente

CP-03	Inscribir un árbitro en una competición
Descripción	Realizar la inscripción de un árbitro en una determinada competición
Entrada	DNI:79925270V, nombre del equipo: C.D Doryoku e identificador de la competición: 11
Salida esperada	Inscripción del árbitro con DNI 79925270V en la competición con id 11
Salida obtenida	Salida esperada

Tabla 7.3: CP-03: Inscribir árbitro en una competición

CP-04	Inscripción de un competidor en una competición
Descripción	Inscripción en una competición válida, de un competidor en un determinado peso y club
Entrada	DNI: 71156516J, nombre del equipo: C.D Lourdes Tuvasa, identificador del peso: 5 e **identificador de la competición:** 11
Salida esperada	Inscripción del competidor en la competición indicada en el peso y club indicados en la entrada anterior.
Salida obtenida	Salida esperada

Tabla 7.4: CP-04: Inscripción válida de un competidor

CP-05	Inscripción de un competidor existente en una competición
Descripción	Inscripción en una competición con DNI del competidor existente
Entrada	DNI: 71156516J, nombre del equipo: C.D Lourdes Tuvasa, identificador del peso: 5 e identificador de la competición: 11
Salida esperada	Mensaje de error
Salida obtenida	Salida esperada

Tabla 7.5: CP-05: Inscripción no válida de un competidor

CP-06	Crear combate individual válido
Descripción	Crear un combate de judo entre dos competidores del mismo peso inscritos en la misma competición
Entrada	DNI competidor azul: 58363598V, DNI competidor blanco: 71156516J, identificador del peso: 5 e **identificador de la competición:** 11
Salida esperada	Combate y puntuaciones del combate creadas. Cada una de las puntuaciones asociadas a un competidor (competidor de kimono azul y competidor de kimono blanco)
Salida obtenida	Salida esperada

Tabla 7.6: CP-06: Crear combate individual válido.

CP-07	Crear combates con algoritmo de emparejamiento
Descripción	Crear combates utilizando el algoritmo de emparejamiento de *Todos contra todos*
Entrada	Algoritmo: Todos contra todos, identificador del peso: 5 e identificador de la competición: 11,DNIs de competidores: 71156516J, 04963284E y 75484447B
Salida esperada	Combate 1 entre 71156516J y 04963284E. Combate 2 entre 04963284E y 75484447B combate 3 entre 71156516J y 75484447B
Salida obtenida	Salida esperada

Tabla 7.7: CP-07: Crear combates con algoritmo de emparejamiento

En el siguiente caso de prueba, lo que se busca es validar que al tener menos dos competidores inscritos en el peso, no se puede crear los combates y en consecuencia mostrar el correspondiente mensaje de error.

CP-08	Crear combates con algoritmo de emparejamiento
Descripción	Crear combates con el algoritmo de emparejamiento de *todos contra todos* o sistema de liga
Entrada	Algoritmo: Todos contra todos, identificador del peso: 5 e identificador de la competición: 11, DNIs de competidores: 58363598V
Salida esperada	Mensaje de error
Salida obtenida	Salida esperada

Tabla 7.8: CP-08: Crear combates con algoritmo

CP-09	Validar DNI introducido
Descripción	A la hora de crear un árbitro, validar que el DNI introducido es válido, es decir, el número del DNI módulo 23 tiene que dar como resultado la letra del DNI en ascii introducida.
Entrada	DNI válido, 09252983Z
Salida esperada	DNI introducido válido, y en consecuencia árbitro creado correctamente
Salida obtenida	Salida esperada

Tabla 7.9: CP-09: Validación DNI correcto

CP-10	Validar DNI introducido
Descripción	A la hora de crear un árbitro, validar que el DNI introducido es válido, es decir, el número del DNI módulo 23 tiene que dar como resultado la letra del DNI en ascii introducida.
Entrada	DNI no válido, 11111111F
Salida esperada	Mensaje de error
Salida obtenida	Salida esperada

Tabla 7.10: CP-10: Validación DNI no correcto

En el siguiente caso de prueba, se comprueba que el servidor de Gmail[21] utilizado para enviar correos electrónicos funciona correctamente.

CP-11	Conexión con el servicio SMTP de Gmail
Descripción	Enviar un correo para recuperar la contraseña al usuario con rol de *Delegado*
Entrada	**Protocolo**: SMTP, **Host**: ssl://SMTP.gmail.com,**Puerto**: 465, **Usuario** : judocupapp@gmail.com, **Destinatario**: judocupapp@gmail.com
Salida esperada	Mensaje de correo al destinatario.
Salida obtenida	Válida

Tabla 7.11: CP-11: Verificar Sistema de correo SMTP de Gmail

CP-12	Volver hacia la vista de competiciones del cronometrador
Descripción	Al pulsar en el botón de volver a atrás del navegador o en el botón de la página tiene que mostrar las competiciones en las que tiene asignado tatamis
Entrada	
Salida esperada	Vista de las competiciones asignadas
Salida obtenida	ERR_CACHE_MISS.

Tabla 7.12: CP-12: Volver hacia atrás

El error producido (*ERR_CACHE_MISS*) reflejado en la tabla 7.12, es producido porque la aplicación solicita el reenvío del formulario con los datos POST que se habían realizado en el paso anterior. Tenemos dos opciones para tratar este problema: la primera opción consiste en volver a enviar los datos de forma manual, está opción resultaría incómoda al usuario puesto que estaría obligado a hacer esto cada vez que se vuelve a la ventana anterior y envió datos en un formulario. La segunda opción es meter cabeceras (ver figura 7.1) al iniciar sesión que eviten que se produzca este error. Hemos optado por implementar la segunda opción para que la experiencia de usuario sea más agradable.

```
1        header('Last-Modified:' . gmdate('D,d M Y H:i:s') . 'GMT');
2        header('Cache-Control: no-cache, must-revalidate,max-age=0');
3        header('Cache-Control: post-check=0, pre-check=0', false);
4        header('Pragma: no-cache');
5
```

Figura 7.1: Cabeceras que eviten el error ERR_CACHE_MISS

CP-13	Validar login de usuario
Descripción	Validar el acceso de un usuario con rol de delegado en la aplicación
Entrada	Email de acceso: judocupapp@gmail.com, contraseña: 1234
Salida esperada	Correo y contraseña válidos
Salida obtenida	Salida esperada

Tabla 7.13: CP-13: Validar login de usuario

CP-14	Validar login de usuario, con email no válido
Descripción	Validar el acceso de un usuario con rol de delegado en la aplicación cuyo email no es válido
Entrada	Email de acceso: novalido@gmail.com, contraseña: 1234
Salida esperada	Mensaje de error indicando que el correo electrónico o la contraseña no son válidos
Salida obtenida	Salida esperada

Tabla 7.14: CP-14: Validar login de usuario, con email no válido

CP-15	Validar login de usuario, con contraseña no válido
Descripción	Validar el acceso de un usuario con rol de delegado en la aplicación cuya contraseña no es válida
Entrada	Email de acceso: judocupapp@gmail.com, contraseña: 1234
Salida esperada	Mensaje de error indicando que el correo electrónico o la contraseña no son válidos
Salida obtenida	Salida esperada

Tabla 7.15: CP-15: Validar login de usuario, con contraseña no válido

CP-16	Crear un cronometrador válido
Descripción	Crear un usuario con rol de cronometrador con una entrada de datos válidos
Entrada	correo electrónico: cronometrador1@gmail.com, contraseña: crono1
Salida esperada	Se crea un usuario con rol de cronometrador
Salida obtenida	Salida esperada

Tabla 7.16: CP-16: Crear un cronometrador válido

CP-17	Crear un cronometrador no válido
Descripción	Crear un usuario con rol de cronometrador con una entrada de datos válidos
Entrada	correo electrónico: cronometrador1@gmail.com, contraseña: crono1
Salida esperada	Se crea un usuario con rol de cronometrador
Salida obtenida	Salida esperada

Tabla 7.17: CP-17: Crear un cronometrador no válido

En el caso de prueba CP-17 (ver figura 7.17) se muestra el caso en el que se crea un usuario con rol de delegado y el email introducido ya existe con algún usuario con rol de delegado o cronometrador.

CP-18	Crear un equipo
Descripción	Crear un equipo con todos los datos de entrada válidos
Entrada	nombre: Lourdes Tuvasa, director técnico: Pedro Riaguas, país: ESP, comunidad autónoma: Castilla y León, provincia: Valladolid
Salida esperada	Un nuevo equipo creado
Salida obtenida	Salida esperada

Tabla 7.18: CP-18: Crear un equipo

CP-19	Crear un equipo no válido
Descripción	Crear un equipo con todos los datos de entrada válidos
Entrada	nombre: Lourdes Tuvasa, director técnico: Pedro Riaguas, país: ESP, comunidad autónoma: Castilla y León, provincia: Valladolid
Salida esperada	Mensaje de error indicando que el nombre del equipo ya existe
Salida obtenida	Salida esperada

Tabla 7.19: CP-19: Crear un equipo no válido

Los casos de prueba de las figuras 7.20 a 7.23 corresponden a las pruebas de introducir los diferentes tipos de puntuaciones (con el rol de delegado) para el competidor que tiene el kimono blanco.

CP-20	Introducir puntuación de *ippon* como delegado
Descripción	Introducir la puntuación de un combate con el rol de delegado
Entrada	Puntuación de *ippon* para el competidor que tiene el kimono blanco. Identificador del combate: 23, dni del competidor de kimono blanco: 71156516J
Salida esperada	El competidor de kimono blanco tiene la puntuación de *ippon*
Salida obtenida	Salida esperada

Tabla 7.20: CP-20: Introducir puntuación de *ippon* como delegado (competidor con kimono blanco)

CP-21	Introducir puntuación de *waza-ari* como delegado
Descripción	Introducir la puntuación de un combate con el rol de delegado
Entrada	Puntuación de *waza-ari* para el competidor que tiene el kimono blanco. Identificador del combate: 23, dni del competidor de kimono blanco: 71156516J
Salida esperada	El competidor de kimono blanco tiene la puntuación de *waza-ari*
Salida obtenida	Salida esperada

Tabla 7.21: CP-21: Introducir puntuación de *waza-ari* como delegado (competidor con kimono blanco)

CP-22	Introducir puntuación de *shido* como delegado
Descripción	Introducir la puntuación de un combate con el rol de delegado
Entrada	Puntuación de *shido* para el competidor que tiene el kimono blanco. Identificador del combate: 23, dni del competidor de kimono blanco: 71156516J
Salida esperada	El competidor de kimono blanco tiene la puntuación de *shido*
Salida obtenida	Salida esperada

Tabla 7.22: CP-22: Introducir puntuación de *shido* como delegado (competidor con kimono blanco)

CP-23	Introducir puntuación de *hansoku* como delegado
Descripción	Introducir la puntuación de un combate con el rol de delegado
Entrada	Puntuación de *hansoku* para el competidor que tiene el kimono blanco. Identificador del combate: 23, dni del competidor de kimono blanco: 71156516J
Salida esperada	El competidor de kimono blanco tiene la puntuación de *hansoku*
Salida obtenida	Salida esperada

Tabla 7.23: CP-23: Introducir puntuación de *hansoku* como delegado (competidor con kimono blanco)

Los casos de prueba de las figuras 7.24 a 7.27 corresponden a las pruebas de introducir los diferentes tipos de puntuaciones (con el rol de delegado) para el competidor que tiene el kimono azul.

CP-24	Introducir puntuación de *ippon* como delegado
Descripción	Introducir la puntuación de un combate con el rol de delegado
Entrada	Puntuación de *ippon* para el competidor que tiene el kimono azul. Identificador del combate: 23, dni del competidor de kimono blanco: 71156516J
Salida esperada	El competidor de kimono azul tiene la puntuación de *ippon*
Salida obtenida	Salida esperada

Tabla 7.24: CP-24: Introducir puntuación de *ippon* como delegado (competidor con kimono azul)

CP-25	Introducir puntuación de *waza-ari* como delegado
Descripción	Introducir la puntuación de un combate con el rol de delegado
Entrada	Puntuación de *waza-ari* para el competidor que tiene el kimono azul. Identificador del combate: 23, dni del competidor de kimono blanco: 71156516J
Salida esperada	El competidor de kimono azul tiene la puntuación de *waza-ari*
Salida obtenida	Salida esperada

Tabla 7.25: CP-25: Introducir puntuación de *waza-ari* como delegado (competidor con kimono azul)

CP-26	Introducir puntuación de *shido* como delegado
Descripción	Introducir la puntuación de un combate con el rol de delegado
Entrada	Puntuación de *shido* para el competidor que tiene el kimono azul. Identificador del combate: 23, dni del competidor de kimono blanco: 71156516J
Salida esperada	El competidor de kimono azul tiene la puntuación de *shido*
Salida obtenida	Salida esperada

Tabla 7.26: CP-26: Introducir puntuación de *shido* como delegado (competidor con kimono azul)

CP-27	Introducir puntuación de *hansoku* como delegado
Descripción	Introducir la puntuación de un combate con el rol de delegado
Entrada	Puntuación de *hansoku* para el competidor que tiene el kimono azul. Identificador del combate: 23, dni del competidor de kimono blanco: 71156516J
Salida esperada	El competidor de kimono azul tiene la puntuación de *hansoku*
Salida obtenida	Salida esperada

Tabla 7.27: CP-27: Introducir puntuación de *hansoku* como delegado (competidor con kimono azul)

Los casos de prueba de las figuras 7.28 a 7.31 corresponden a las pruebas de introducir los diferentes tipos de puntuaciones (con el rol de cronometrador) para el competidor que tiene el kimono blanco.

CP-28	Introducir puntuación de *ippon* como cronometrador
Descripción	Introducir la puntuación de un combate con el rol de cronometrador
Entrada	Puntuación de *ippon* para el competidor que tiene el kimono blanco. Identificador del combate: 23, dni del competidor de kimono blanco: 75484447B
Salida esperada	El competidor de kimono blanco tiene la puntuación de *ippon*
Salida obtenida	Salida esperada

Tabla 7.28: CP-28: Introducir puntuación de *ippon* como cronometrador (competidor con kimono blanco)

CP-29	Introducir puntuación de *waza-ari* como cronometrador
Descripción	Introducir la puntuación de un combate con el rol de cronometrador
Entrada	Puntuación de *waza-ari* para el competidor que tiene el kimono blanco. Identificador del combate: 23, dni del competidor de kimono blanco: 75484447B
Salida esperada	El competidor de kimono blanco tiene la puntuación de *waza-ari*
Salida obtenida	Salida esperada

Tabla 7.29: CP-29: Introducir puntuación de *waza-ari* como cronometrador (competidor con kimono blanco)

CP-30	Introducir puntuación de *shido* como cronometrador
Descripción	Introducir la puntuación de un combate con el rol de cronometrador
Entrada	Puntuación de *shido* para el competidor que tiene el kimono blanco. Identificador del combate: 23, dni del competidor de kimono blanco: 75484447B
Salida esperada	El competidor de kimono blanco tiene la puntuación de *shido*
Salida obtenida	Salida esperada

Tabla 7.30: CP-30: Introducir puntuación de *shido* como cronometrador (competidor con kimono blanco)

CP-31	Introducir puntuación de *hansoku* como cronometrador
Descripción	Introducir la puntuación de un combate con el rol de cronometrador
Entrada	Puntuación de *hansoku* para el competidor que tiene el kimono blanco. Identificador del combate: 23, dni del competidor de kimono blanco: 75484447B
Salida esperada	El competidor de kimono blanco tiene la puntuación de *hansoku*
Salida obtenida	Salida esperada

Tabla 7.31: CP-31: Introducir puntuación de *hansoku* como cronometrador (competidor con kimono blanco)

Los casos de prueba de las figuras 7.32 a 7.35 corresponden a las pruebas de introducir los diferentes tipos de puntuaciones (con el rol de delegado) para el competidor que tiene el kimono azul.

CP-32	Introducir puntuación de *ippon* como cronometrador
Descripción	Introducir la puntuación de un combate con el rol de cronometrador
Entrada	Puntuación de *ippon* para el competidor que tiene el kimono azul. Identificador del combate: 23, dni del competidor de kimono blanco: 75484447B
Salida esperada	El competidor de kimono azul tiene la puntuación de *ippon*
Salida obtenida	Salida esperada

Tabla 7.32: CP-32: Introducir puntuación de *ippon* como cronometrador (competidor con kimono azul)

CP-33	Introducir puntuación de *waza-ari* como cronometrador
Descripción	Introducir la puntuación de un combate con el rol de cronometrador
Entrada	Puntuación de *waza-ari* para el competidor que tiene el kimono azul. Identificador del combate: 23, dni del competidor de kimono blanco: 75484447B
Salida esperada	El competidor de kimono azul tiene la puntuación de *waza-ari*
Salida obtenida	Salida esperada

Tabla 7.33: CP-33: Introducir puntuación de *waza-ari* como cronometrador (competidor con kimono azul)

CP-34	Introducir puntuación de *shido* como cronometrador
Descripción	Introducir la puntuación de un combate con el rol de cronometrador
Entrada	Puntuación de *shido* para el competidor que tiene el kimono azul. Identificador del combate: 23, dni del competidor de kimono blanco: 75484447B
Salida esperada	El competidor de kimono azul tiene la puntuación de *shido*
Salida obtenida	Salida esperada

Tabla 7.34: CP-34: Introducir puntuación de *shido* como cronometrador (competidor con kimono azul)

CP-35	Introducir puntuación de *hansoku* como cronometrador
Descripción	Introducir la puntuación de un combate con el rol de cronometrador
Entrada	Puntuación de *hansoku* para el competidor que tiene el kimono azul. Identificador del combate: 23, dni del competidor de kimono blanco: 75484447B
Salida esperada	El competidor de kimono azul tiene la puntuación de *hansoku*
Salida obtenida	Salida esperada

Tabla 7.35: CP-35: Introducir puntuación de *hansoku* como cronometrador (competidor con kimono azul)

CP-36	Asignar un combate a un tatami
Descripción	Asignar un combate a un determinado tatami en una competición concreta.
Entrada	Identificador de tatami: 29, Identificador de combate: 23
Salida esperada	El combate con identificador 23 está asignado al tatami con identificador 29
Salida obtenida	Salida esperada

Tabla 7.36: CP-36: Asignar un combate a un tatami

CP-37	Cambiar el tatami de un combate
Descripción	El combate con un determinado identificador se le va a cambiar el tatami dónde esté asignado
Entrada	Identificador de combate: 23, Identificador de tatami (antiguo): 29, identificador de tatami (nuevo): 30
Salida esperada	El combate con identificador 23 está asignado al tatami con identificador 30
Salida obtenida	Salida esperada

Tabla 7.37: CP-37: Cambiar el tatami de un combate

CP-38	Eliminar un combate
Descripción	Eliminar un combate con un identificador de combate determinado
Entrada	Identificador de combate: 23
Salida esperada	El combate con identificador 23 es eliminado, junto con su puntuaciones
Salida obtenida	Salida esperada

Tabla 7.38: CP-38: Eliminar un combate

CP-39	Actualizar la puntuación de un combate
Descripción	Con el rol de delegado, cambiar la puntuación de un combate que ya había sido puntuado.
Entrada	Identificador del combate: 24, *waza-ari* para el competidor de color blanco
Salida esperada	El combate con identificador 24 cambia la puntuación a waza-ari
Salida obtenida	Salida esperada

Tabla 7.39: CP-39: Actualizar la puntuación de un combate

CP-40	Actualizar los campos de un árbitro
Descripción	Actualizar el campo de la categoría arbitral de un determinado árbitro
Entrada	DNI del árbitro: 09252983Z, identificador de la nueva categoría arbitral: 7
Salida esperada	El árbitro tiene asignada una nueva categoría arbitral
Salida obtenida	Salida esperada

Tabla 7.40: CP-40: Actualizar los campos de un árbitro

CP-41	Actualizar los campos de un equipo
Descripción	Actualizar el campo director técnico de un determinado equipo
Entrada	Nombre del equipo: Lourdes Tuvasa, nuevo nombre del director técnico: Paco de la Calle
Salida esperada	El equipo con nombre Lourdes Tuvasa tiene un nuevo nombre en el campo de director técnico

Tabla 7.41: CP-41: Actualizar los campos de un equipo

CP-42	Asociar cronometrador a un tatami
Descripción	Asignar un cronometrador concreto en un tatami determinado de una competición
Entrada	Identificador de tatami: 29, correo electrónico del cronometrador: cronometrador1@gmail.com
Salida esperada	El cronometrador con el correo electrónico introducido como entrada tiene asociado el tatami con identificador 29
Salida obtenida	Salida esperada

Tabla 7.42: CP-42: Asociar cronometrador a un tatami

CP-43	Eliminar cronometrador de un tatami
Descripción	Eliminar la asignación que tiene un determinado cronometrador en un tatami
Entrada	Identificador de tatami: 29, correo electrónico del cronometrador: cronometrador1@gmail.com
Salida esperada	El tatami con identificador 29 no tiene asignado ningún cronometrador
Salida obtenida	Salida esperada

Tabla 7.43: CP-43: Eliminar cronometrador de un tatami

CP-44	Cambiar árbitro de un tatami
Descripción	Cambiar el tatami donde está inscrito un determinado árbitro
Entrada	DNI del árbitro: 70341324X , identificador del tatami (antiguo): 29, identificador de la competición: 11, identificador del tatami (nuevo): 30
Salida esperada	Cambio de tatami del árbitro indicado en la entrada
Salida obtenida	Salida esperada

Tabla 7.44: CP-44: Cambiar árbitro de un tatami

CP-45	Cambiar el peso de un competidor
Descripción	Reasignar peso de un determinado competidor en una competición concreta
Entrada	DNI del competidor: 71156516J, identificador del peso (antiguo): 5, identificador del peso (nuevo): 4, identificador de la competición: 11
Salida esperada	El competidor es cambiado de peso en la competición indicada
Salida obtenida	

Tabla 7.45: CP-45: Cambiar el peso de un competidor

7.3. Pruebas de seguridad de la aplicación

CP-46	Funcionamiento página web por HTTPS
Descripción	Introducir en el navegador la URL de acceso a la aplicación utilizando el protocolo HTTPS. El navegador nos tiene que indicar que la conexión es segura y que la comunicación entre ambos extremos está cifrada
Entrada	https://virtual.lab.inf.uva.es:20062/tfg/judocup
Salida esperada	La conexión con el servidor está cifrada con un conjunto de cifrado moderno. La conexión utiliza TLS 1.2. La conexión se ha encriptado y autenticado con AES_256_GMC, y utiliza ECDHE_RSA como mecanísmo de intercambio clave. El navegador nos indica que el certificado de servidor no es de confianza
Salida obtenida	Esperada

Tabla 7.46: CP-46: Funcionamiento de la página web por HTTPS

El mensaje que proporciona el servidor sobre que el certificado de servidor utilizado no es de confianza es porque dicho certificado no ha sido emitido por una entidad certificadora. Para que no se muestre este mensaje cada vez que se accede a la aplicación, habría que sustituir el certificado generado de forma gratuita por Ubuntu, por uno de pago de una certificadora oficial.

CP-47	Funcionamiento POST del formulario
Descripción	Verificar que cuando el usuario intorduce las credenciales de acceso, estas no se muestran en la URL
Entrada	Email de acceso : **judocupapp@gmail.com**, contraseña: **1234**
Salida esperada	En la URL no tiene que aparecer ninguno de los dos campos introducidos. Esto significa que funciona correctamente el POST hacia el servidor y no se utiliza el método GET, donde si que se mostrarían el email y la contraseña
Salida obtenida	Esperada

Tabla 7.47: CP-47: Funcionamiento del POST del formulario

CP-48	Tempo máximo de sesión
Descripción	Verificar que el tiempo máximo de inactividad durante una sesión de usuario es de 30 minutos
Entrada	Iniciar sesión con las credenciales de acceso con el rol de Delegado. A continuación no realizar ninguna actividad durante 30 minutos.
Salida esperada	Al pasar los 30 minutos de inactividad, la aplicación tiene que mostrar un mensaje de error indicando que tenemos que volver a iniciar sesión para poder continuar
Salida obtenida	Esperada

Tabla 7.48: CP-48: Tiempo máximo de sesión

CP-49	Usuario y/o contraseña no válidos
Descripción	Verificar que al introducir el correo electrónico que no es válido muestra el mismo mensaje de error que cuando la contraseña no coincide cuando el correo electrónico es correcto
Entrada	Correo electrónico: **correonovlaido@gmail.com**, contraseña: **1234**
Salida esperada	El sistema muestra un mensaje de error indicando que el usuario o la contraseña son incorrectos
Salida obtenida	Esperada

Tabla 7.49: CP-49: Usuario y/o contraseña no válidos

CP-50	Inyección SQL (I)
Descripción	En el login de usuario para acceder a la aplicación, probar que si se intenta hacer una inyección SQL, no accede a la aplicación
Entrada	email: 'asdf' OR 'a'='a'y contraseña = 'asdf' OR 'a'='a'
Salida esperada	Mensaje de error indicando que el email introducido no tiene la estructura de un correo electrónico válido
Salida obtenida	Salida esperada

Tabla 7.50: CP-50: Inyección SQL (I)

CP-51	Inyección SQL (II)
Descripción	En el login de usuario para acceder a la aplicación, probar que si se intenta hacer una inyección SQL, no accede a la aplicación
Entrada	email: judocupapp@gmail.com y contraseña = 'asdf' OR 'a'='a'
Salida esperada	Mensaje de error indicando que el email o la contraseña son incorrectos
Salida obtenida	Salida esperada

Tabla 7.51: CP-51: Inyección SQL (II)

Capítulo 8

Encuesta

La validación por parte de un usuario final de la aplicación, es uno de los aspectos importantes a considerar, ya que son estos los que van a usarla. Para llevar a cabo esta evaluación se decidió realizar un cuestionario de usabilidad y aceptabilidad, que involucraba a diferentes tipos de usuarios, desde gente experimentada en Judo, entrenadores nacionales, competidores, hasta gente que no entendida de la materia a tratar. En este capítulo de la memoria, se describe como se planteó el procedimiento para evaluar la aceptabilidad de la aplicación por parte del usuario, se comenta los resultados obtenidos y por último se hace un análisis de estos.

8.1. Planteamiento del experimento

En el desarrollo del experimento, primeramente se les hacía una demostración general de como funcionaba la aplicación, las funcionalidades principales que tenia y para que podría ser usado. El tiempo dedicado a esta tarea no excedió los diez minutos. Después, se pedía al usuario que realizase la tarea que encontraremos en la sección 8.1.1, para posteriormente reflejar su opinión en la encuesta que podemos encontrar en esta dirección URL [1]. Antes de realizarla, se le explican los bloques que hay y si tiene alguna duda en alguna de las preguntas que se le formulan. El tiempo máximo para la realización de la encuesta es de diez minutos.

El procedimiento de la tarea que tiene que desarrollar el usuario se detalla a continuación:

8.1.1. Descripción de la tarea: Crear una competición de judo

1. Seleccionar crear una nueva competición de judo rellenando los campos obligatorios.
2. Inscribir un árbitro en la competición y asignarle un tatami.
3. Inscribir dos competidores de judo, uno en un peso y otro en otro diferente. Uno de ellos tiene que ser un competidor que haya realizado alguna competición, es decir, inscripción por dni existente y el otro se tiene que crear, es decir, que no haya realizado ninguna competición todavía. El dni del competidor existente se les ofrecía al encuestado.
4. Se cambia la participación de uno de los dos competidores al mismo peso del otro.
5. Crear un combate entre los dos competidores inscritos
6. Puntuar el combate.

8.1.2. Descripción del cuestionario

El cuestionario esta formado por diferentes bloques en los que se evalúan los aspectos más importantes a tener en cuenta y poder mejorar, en el futuro la aplicación, para que sea más fácil de utilizar por el usuario final. Se utilizan las preguntas formuladas por *Chin, John P. , Diehl, Virginia*

[1] https://goo.gl/forms/wE5XlZ5jHVga3B382

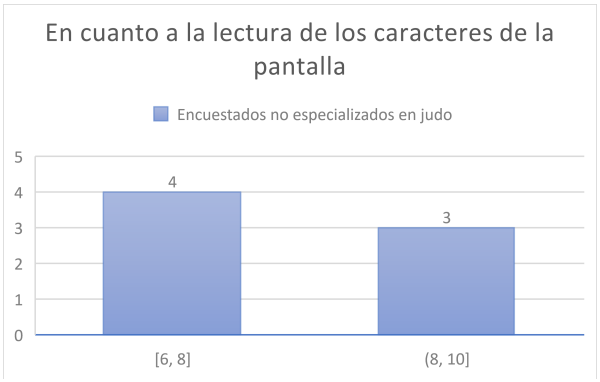
A. y Norman, Kent L en [22]. En el Anexo F tenemos el formato de la encuesta con las preguntas formuladas a los encuestados. Las preguntas que el usuario veía aparecen también en las figuras de resultados (figura 8.1 a 8.16).

- En el primer bloque se pregunta sobre aspectos generales de la pantalla. La primera pregunta es sobre los caracteres de la pantalla, si su lectura es o no cómoda. La segunda, sobre la organización del contenido, cómo está expuesto, si es agradable o no a la vista. La tercera pregunta de este bloque aborda la secuencia de pantallas, es decir si el usuario es capaz de ir navegando por la aplicación con facilidad o en cambio es confusa.
- En el segundo bloque de preguntas se realizan cuestiones en cuanto a la terminología usada en la aplicación. Se realizan preguntas tales como, uso de la terminología en la aplicación, terminología usada en cada tarea, posición de los mensajes en la pantalla, entrada de datos, es decir, si es cómoda cuando se tienen que, por ejemplo introducir valores para crear un árbitro. Otra de las preguntas es sobre la información acerca del progreso de las tareas y los mensajes de error cuando se está realizando una tarea.
- En el tercer bloque, se realizan preguntas sobre el aprendizaje que tiene el usuario a medida que usa la aplicación y de las capacidades del sistema. Se pregunta sobre si es fácil o no aprender la funcionalidad del sistema, explorar las funcionalidades por prueba error (es decir si cometes algún tipo de fallo esta te lo indica). Otra de las cuestiones es sobre si es fácil recordar el uso de comandos o nombres para realizar un tipo de tarea, materiales de referencia secundarios tales como mensajes de información sobre los elementos, o el uso de una *FAQ (Frecuently Asked Questions), preguntas más frecuentes*. Otro tipo de preguntas son sobre la velocidad de la aplicación, si va fluida o no, la confiabilidad que tiene el usuario hacia la aplicación, la corrección de errores, y por último, si está diseñada para todo tipo de usuarios.
- Y por último, en el cuarto bloque, se le permite al usuario que está realizando la encuesta indicar los aspectos negativos y positivos que ha visto cuando ha estado utilizando la aplicación.

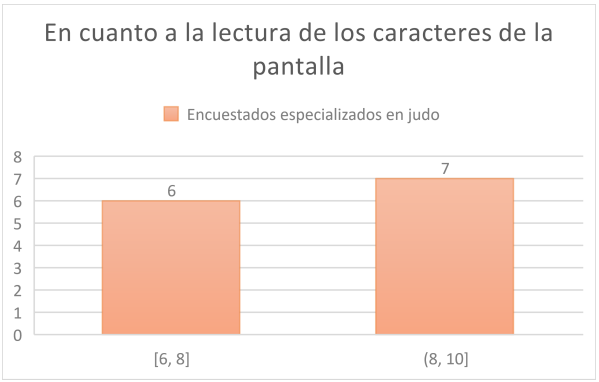
8.2. Resultados obtenidos de la encuesta

En esta sección se van a mostrar los resultados obtenidos tras realizar los cuestionarios. En el eje de ordenadas se representan el número de encuestados y en el eje de abscisas la puntuación recibida en la pregunta.

8.2.1. Bloque I: Aspectos generales de la pantalla



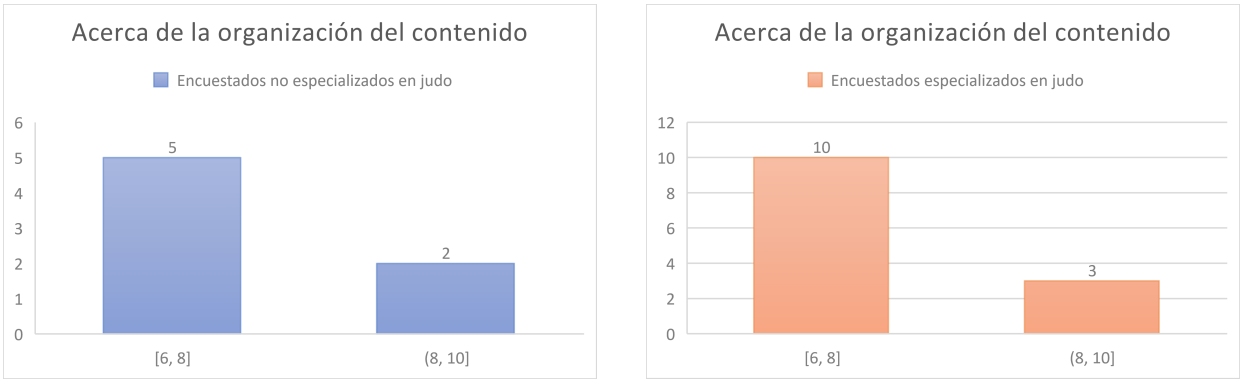
(a) No especializados en judo



(b) Especializados en judo

Figura 8.1: En cuanto a la lectura de los caracteres

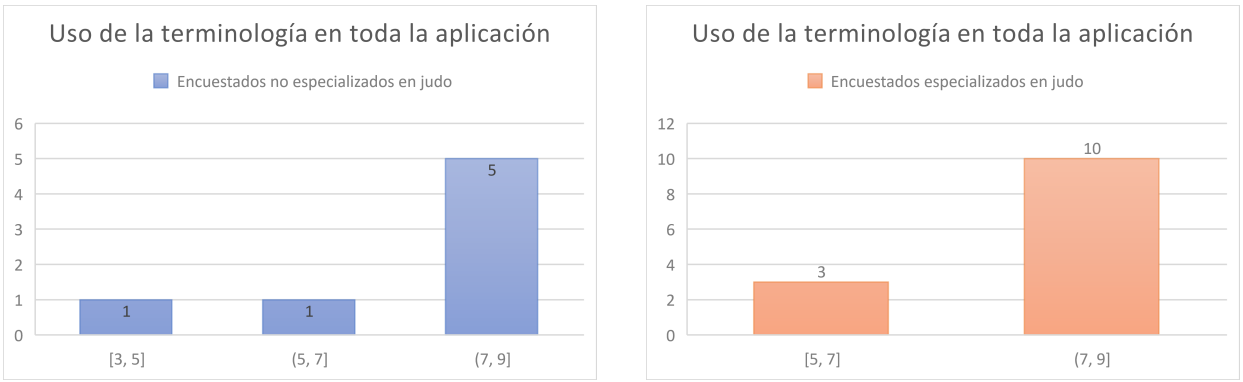
El 57.1428 % de los encuestados que no están especializados en judo consideran que la lectura de los caracteres de la pantalla es buena, mientras que el 42.8571 % la considera muy fácil/ agradable (resultados obtenidos de la figura 8.1a). Por otro lado, el 46.1538 % de los encuestados con conocimientos en judo y la gestión de campeonatos piensa que el aspecto de la pantalla es bueno y el 53.8461 % restante muy buena (resultados obtenidos de la figura 8.1b).



(a) No especializados en judo (b) Especializados en judo

Figura 8.2: Acerca de la organización del contenido

Analizando el histograma de la figura 8.2b, el 71.4285 % de los encuestados no especializados piensa que la organización de los contenidos es la adecuada y el 28.5714 % opina que es muy buena. Por otro lado, tras analizar el histograma de la figura 8.2b tenemos que el 76.9230 % de los encuestados experimentados en judo piensa que es buena y el tanto por ciento restante opina que es muy buena.

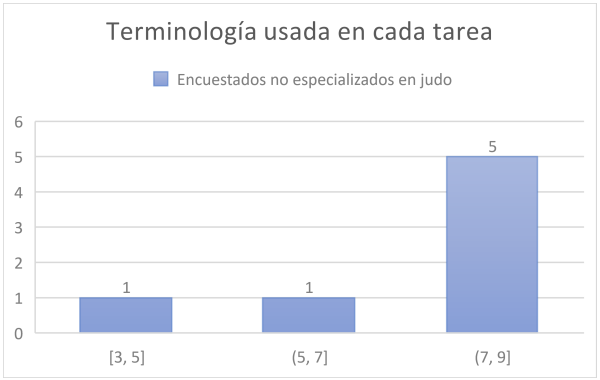


(a) No especializados en judo (b) Especializados en judo

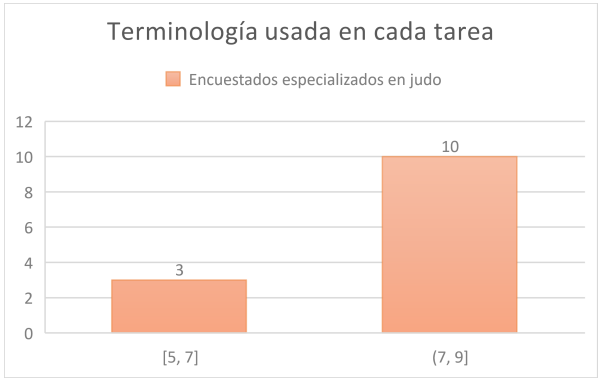
Figura 8.3: Uso de la terminología en toda la aplicación

En cuanto al uso de la terminología en la aplicación, hay un porcentaje de usuarios que no está demasiado de acuerdo, un 19.0476 % entre especializados y no especializados en judo (figuras 8.3a y 8.3b). El resto de usuarios opina que el uso de la terminología es bueno y muy bueno, el 80.9523 %.

8.2.2. Bloque II: Terminología usada en la aplicación



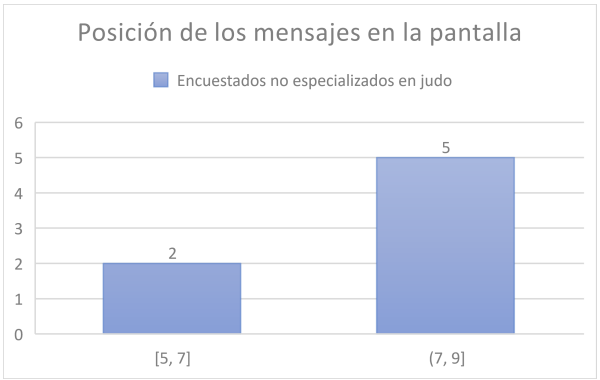
(a) No especializados en judo



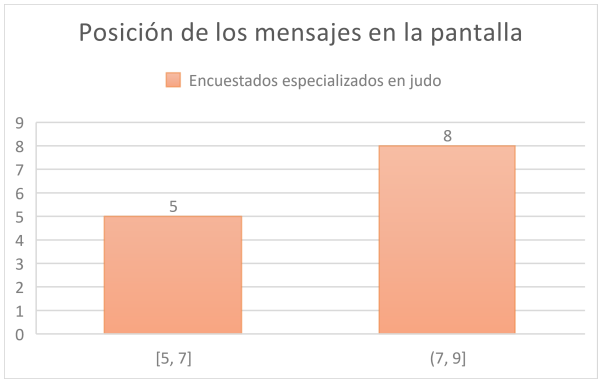
(b) Especializados en judo

Figura 8.4: Terminología usada en cada tarea

A la hora de puntuar esta característica hubo un 14.2857 % de los encuestados que no practican judo a los que no les parecía adecuada la terminología utilizada, poniendo comentarios como que se repetían las mismas palabras entre diferentes vistas de la aplicación. En cambio, al 85.7142 % restante le parecía bastante adecuada la terminología utilizada. Los especialistas en judo (figura 8.4b) están bastante conformes, la nota media ronda entre 7 y 9 sobre 10 del 76.9230 % de los encuestados especializados en judo.



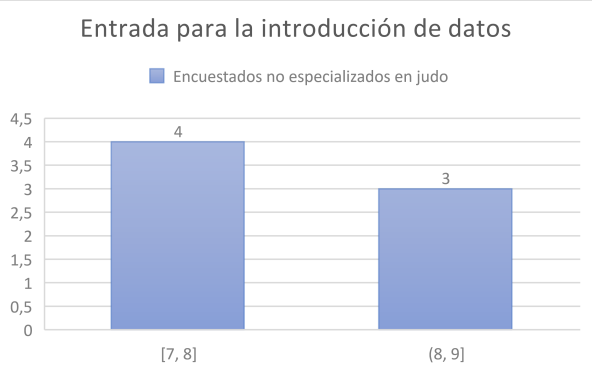
(a) No especializados en judo



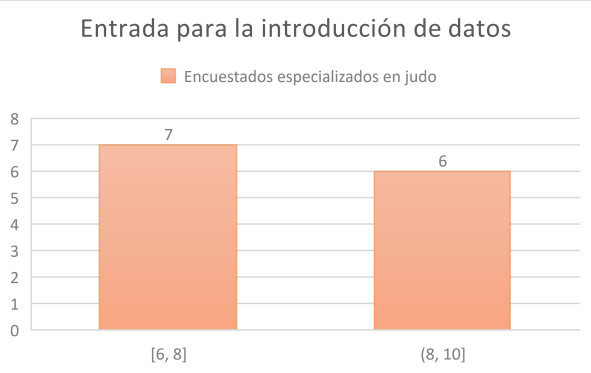
(b) Especializados en judo

Figura 8.5: Posición de los mensajes en la pantalla

En la puntuación de esta característica, ambos grupos de encuestados están bastante conformes (figuras 8.5a y 8.5b), con una nota superior a 5 puntos sobre 10. Hay que señalar, que hubo un par de personas que indicaron que en pantallas 4K o de un tamaño superior a las 27 pulgadas hay demasiados espacios en blanco entre los diferentes componentes web, aunque también reflejaron que a pesar de ese defecto, se podían realizar las tareas perfectamente.



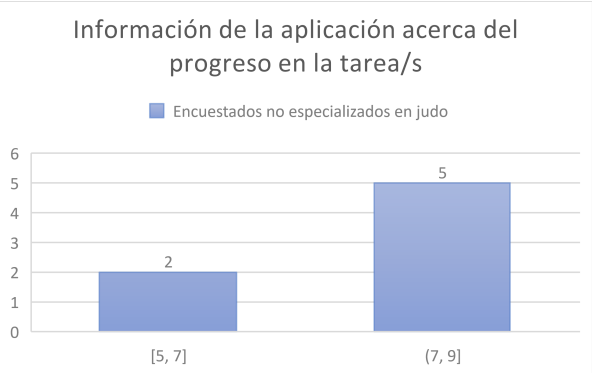
(a) No especializados en judo



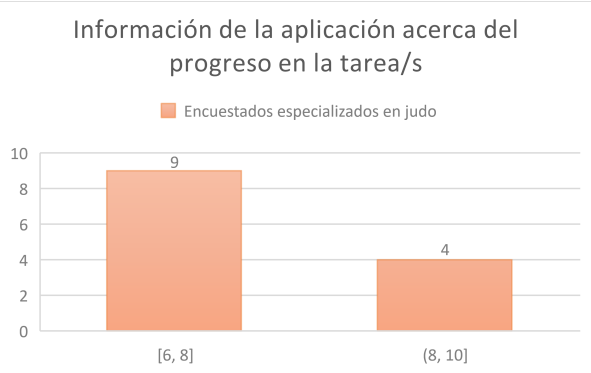
(b) Especializados en judo

Figura 8.6: Entrada para la introducción de datos

Para la introducción de datos, ningún usuario experimentó ningún tipo de problema y así lo reflejan las puntuaciones obtenidas (histogramas 8.6a y 8.6b). Usuarios no experimentados con 57.1482 % lo puntúan como buena y el 42.8571 % restante como muy buena. En el grupo de experimentados en judo, con 53.8461 % lo puntúan como buena y el 46.1538 % restante como muy buena.



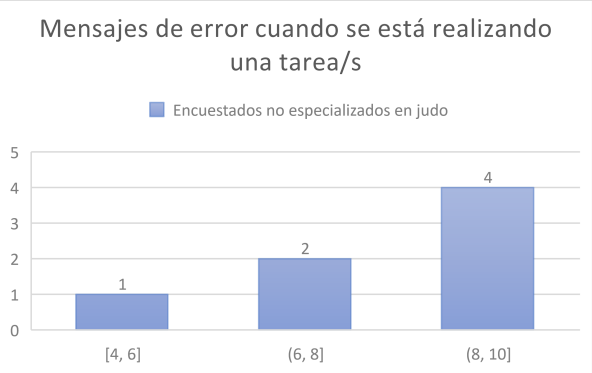
(a) No especializados en judo



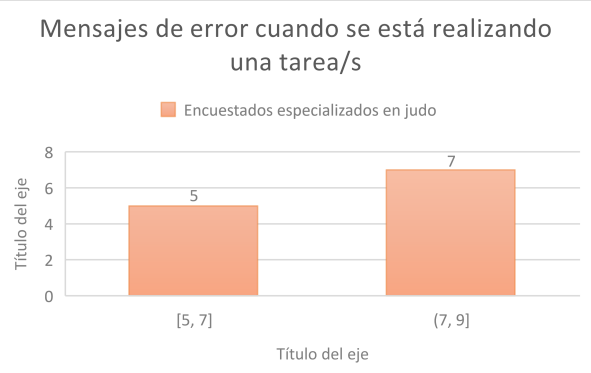
(b) Especializados en judo

Figura 8.7: Información de la aplicación acerca del progreso en la tarea/s

En los histogramas mostrados en las figuras 8.7a y 8.7b vemos que los usuarios también otorgan una puntuación muy positiva cuando se les pregunta por la información sobre el progreso de la tarea o tareas. Se ha de comentar que algunos usuarios experimentaron dudas sobre cómo seguir una vez creada la competición e inscritos a los competidores y árbitros. Tras una breve explicación se solventaron sus dudas.



(a) No especializados en judo

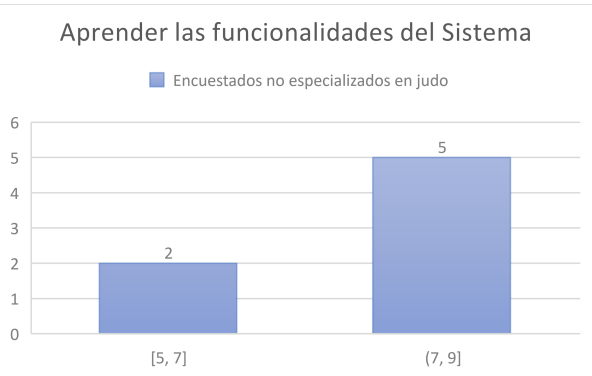


(b) Especializados en judo

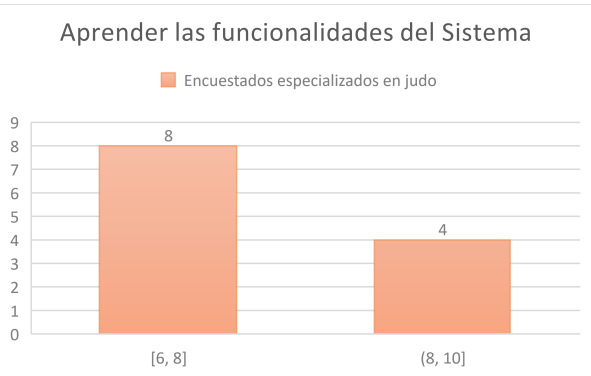
Figura 8.8: Mensajes de error cuando se esta realizando una tarea/s

En el histograma mostrado en la figura 8.8a se puede ver que un 14.2857 % indican una cierta disconformidad con los mensajes de error que se dan desde la aplicación. Un caso muy claro que provocaba al usuario cierto desagrado con el mensaje de error era cuando creaba un árbitro con todos los datos correctos menos el documento nacional de identidad (DNI) y recibía un mensaje de error. Al limpiar todos los campos, el usuario podría llegar a frustrarse, ya que tenía que volver a introducirlos de nuevo. Una posible solución a este problema sería validar el DNI antes de pedir que rellene el valor del resto de campos. El 85.7142 % restante opina que los mensajes de error son los correctos.

Los usuarios experimentados, no vieron los mismos problemas mencionados en el párrafo anterior (ver figura 8.8a). Obtenemos los siguientes datos: un 38.4615 % consideran buenos los mensajes de error y el 61.5384 % restante los consideran muy buenos.



(a) No especializados en judo

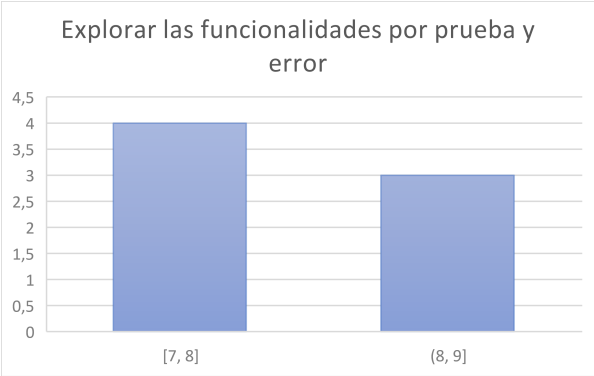


(b) Especializados en judo

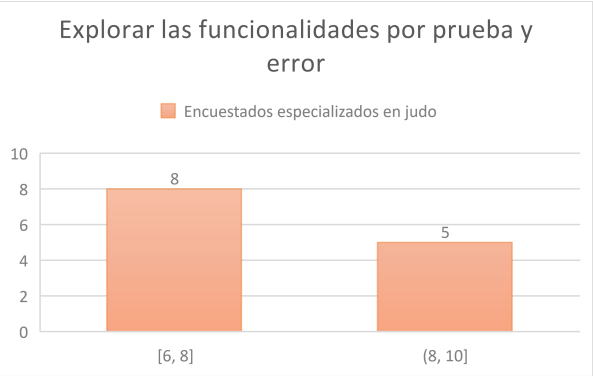
Figura 8.9: Aprender la funcionalidad del sistema

En la última pregunta de este bloque del cuestionario, sobre cómo de fácil es el aprendizaje de las funcionalidades del sistema, ambos grupos lo consideran bastante fácil de usar, con una nota de 7.6488 en promedio sobre 9. Los resultados individuales para cada grupo se pueden ver en la figura 8.9.

8.2.3. Bloque III: Aprendizaje



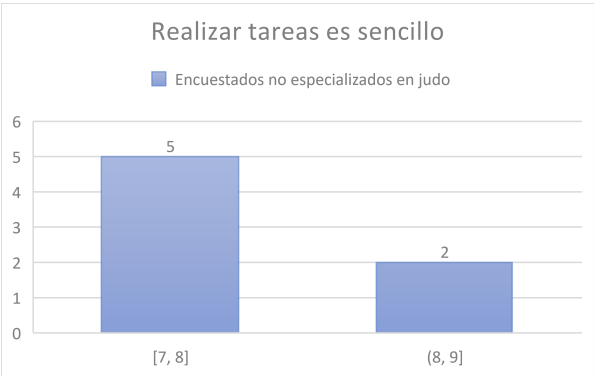
(a) No especializados en judo



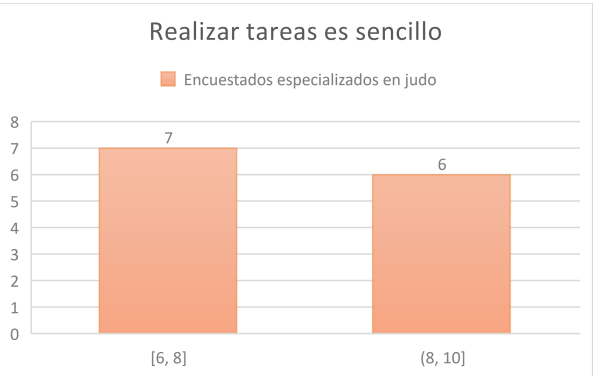
(b) Especializados en judo

Figura 8.10: Explorar las funcionalidades por prueba y error

Con esta pregunta se pretende saber la dificultad que encontrarán los usuarios para aprender a utilizar las funcionalidades que tiene la aplicación por prueba y error, es decir, probando acciones como crear árbitros, competidores, eliminarlos y ver que es lo que va mostrando la aplicación. Analizamos los datos del histograma de la figura 8.10a. Un 57.1482 % considera que es fácil y el 42.8571 % restante muy fácil. En el diagrama homólogo, figura 8.10b, se muestran los mismos resultados para los usuarios experimentados. De estos, el 61.5384 % dice que es fácil explorar las funcionalidades y el porcentaje restante, muy fácil.



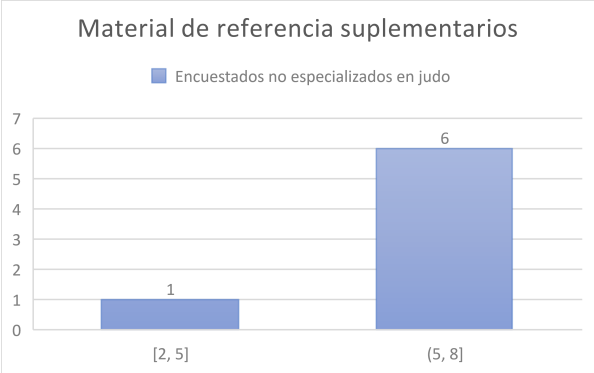
(a) No especializados en judo



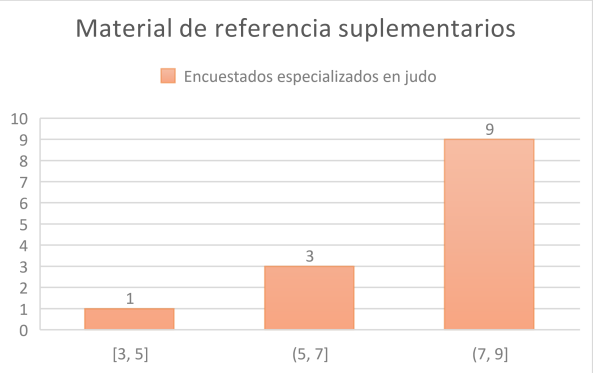
(b) Especializados en judo

Figura 8.11: Realizar tareas es sencillo

En los histogramas de las figuras 8.11a y 8.11b se pregunta sobre la impresión general sobre la realización de tareas. Ambos grupos indican que es fácil. En esta pregunta nos detenemos para comentar una observación hecha por uno de los encuestados. Un usuario experimentado dijo que usuarios que no conociesen cómo se gestionan las competiciones de judo, podría resultarles difícil. Pero conviene tener en cuenta que el usuario tipo de esta aplicación es un delegado o responsable de la federación o federaciones, es decir, gente con conocimientos en judo.



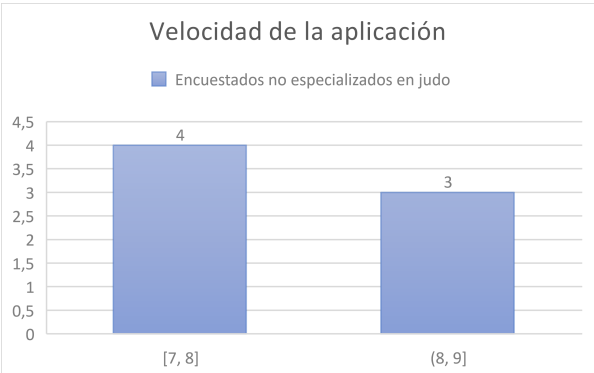
(a) No especializados en judo



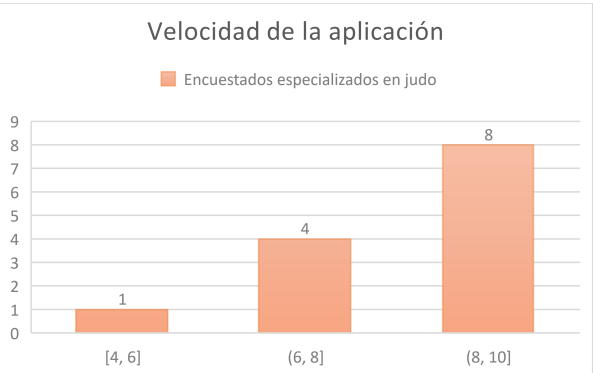
(b) Especializados en judo

Figura 8.12: Material de referencia suplementarios

Esta cuestión es una de las que peores resultados obtiene, y así lo reflejan los histogramas 8.12a y 8.12b, de los grupos no experimentados y experimentados respectivamente. Aunque la aplicación es intuitiva para la mayoría de usuarios, hay alguno que pide una sección con FAQ, para resolver las preguntas tipo, o videos aclarativos explicando cómo ejecutar las funcionalidades más difíciles.



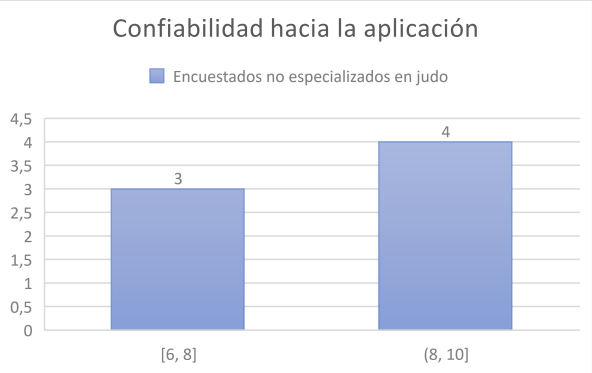
(a) No especializados en judo



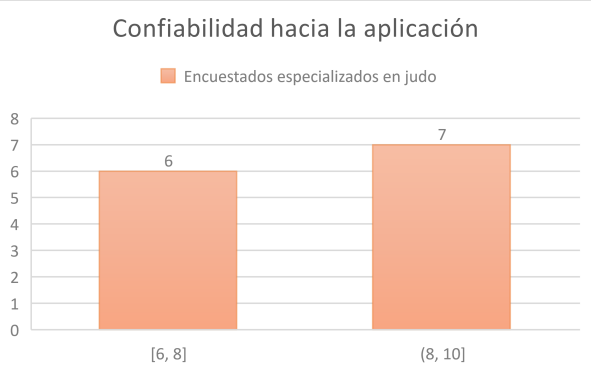
(b) Especializados en judo

Figura 8.13: Velocidad de la aplicación

Uno de los aspectos más importantes que busca un usuario de cualquier aplicación es que ésta responda en un tiempo aceptable, y a todos los usuarios que participaron a esta encuesta les pareció que esto era así. Una de las pruebas que se ejecutó para ver el rendimiento de la aplicación es la auditoría que ofrece la herramienta de Chrome (Audits). En la opción de *performance* se obtiene una puntuación de 75 % en la mayoría de vistas de la aplicación.



(a) No especializados en judo

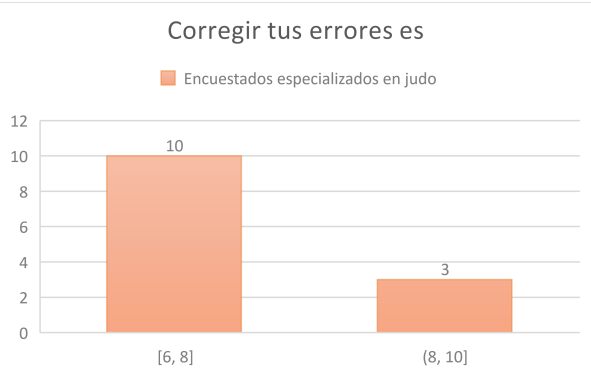


(b) Especializados en judo

Figura 8.14: Confiabilidad hacia la aplicación

En este apartado, lo que más preocupa a los usuarios es el mensaje que ofrecen los navegadores cuando una aplicación navega sobre conexión segura, HTTPS, y el certificado no es reconocido por el navegador, como es en este caso, ya que el certificado es el que se genera automáticamente por el comando de Ubuntu *openssl*. Una vez aclarado que todos los datos que se envíen desde cualquier cliente al servidor, en este caso *virtual.lab.inf.uva.es:20063* viajan encriptados, por lo que se considera un servidor seguro. En ambos casos, como podemos ver en los histogramas de las figuras 8.14a y 8.14b, la califican como bastante fiable.

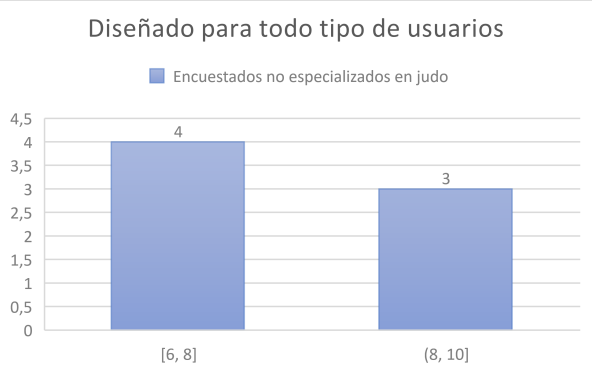
(a) No especializados en judo



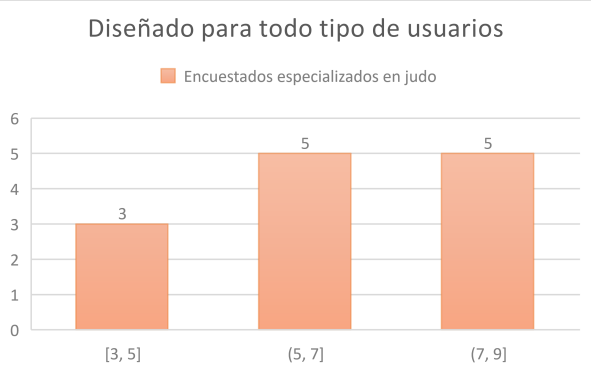
(b) Especializados en judo

Figura 8.15: Cómo de difícil es corregir tus errores

Esta pregunta se interesa por como de fácil resulta para los usuarios corregir una entrada de datos mal introducida. Puesto que está implementada la eliminación de la mayoría de objetos creados, no tienen ningún problema para volver a un paso anterior. Ciertos usuario echaron en falta un botón de deshacer la acción realizada.



(a) No especializados en judo



(b) Especializados en judo

Figura 8.16: Diseñado para todo tipo de usuarios

La última pregunta de este bloque se ocupa de recopilar su opinión sobre el diseño. Deben indicar si lo consideran apto para cualquier tipo de usuario. Tras analizar esta respuesta junto con los aspectos positivos y negativos, podemos responder que si es para todo tipo de usuarios si tienen conocimientos en judo. El que no conozca el judo, le va a costar más aprender a utilizarla, así lo reflejan los histogramas de las figuras 8.16a y 8.16b.

8.2.4. Bloque IV: Aspectos positivos y negativos

En esta sección se van a exponer las respuestas de los encuestados respecto a que aspectos son los más positivos y cuales los más negativos que han observado cuando han realizado el experimento.

Listado de los aspectos más positivos

- "Una aplicación útil y fácil de utilizar."
- "Muy visual."
- "La posibilidad de realizar las tareas en dispositivos móviles la hace muy agradable."
- "La aplicación es eficiente y no me ha costado aprender a utilizarla."
- "Es intuitiva, al poco de jugar con ella te haces, diseño adaptable."
- "Fácil uso y muy intuitiva."
- "Organización, creatividad y novedoso."
- "Está genial para los organizadores de competiciones, así como para personas que participen en las mismas."
- "Buena organización y aspectos claros a la hora de la gestión."

Listado de los aspectos más negativos

- "En pantallas muy grandes se ve mucho espacio en blanco entre los elementos de la página, aunque no hace que se puedan realizar las tareas que se quiere hacer."
- "En algunas interfaces de usuario hay demasiado espacio sin utilizar y los componentes no están bien distribuidos."
- "Muy difícil de encontrar la documentación de la aplicación se echa de menos un faq, y una explicación más detallada de qué se puede hacer/para qué sirve en cada sección en el carrusel inicial."

- “La redacción de cada apartado es un poco repetitiva, pero muy clara.”
- “Veo difícil que sea intuitivo para una persona sin conocimientos en la materia, aunque no es el público objetivo de esta app.”
- “No se puede dar prioridad a los judokas que hayan quedado primeros en un determinado ranking o sean campeones de una competición importante, a la hora de realizar los sorteos de la competición.”

Conclusiones finales

Una vez revisadas todas las respuestas de la encuesta, y la impresión general de los usuarios que realizaron el experimento, se puede decir que ha tenido una gran aceptación por gran parte de estos (en la mayoría de los casos con notas superiores al 60 %), salvando las distancias con alguna pregunta. La idea y propósito de la realización de este experimento es mejorar la aplicación de cara al usuario final, para lo que se tendrán en cuenta todos los aspectos negativos recalcados por éstos.

Capítulo 9

Conclusiones

El desarrollo de este proyecto me ha servido entre otras cosas para saber cómo se gestiona un proyecto de mediana embergadura, y hacerme consciente de la importancia de una buena planificación para poder llevar a cabo todas las tareas dentro del plan establecido. La falta de experiencia en planificación de proyectos de este calibre, ha provocado ciertos retrasos en la entrega de las tareas, pero que no ha penalizado demasiado la planificación inicial. Esto es, la planificación prevista inicialmente era aceptable. El resultado de este proyecto es una aplicación totalmente funcional cuya puesta en producción se podría realizar sin problemas. Aunque no se ha podido mostrar aún a la Federación, la aplicación está funcionando, se prevé hacerlo próximamente. Para ello será necesario organizar una demostración con ellos, en algún momento que estén disponibles. Esto parece viable, puesto que mostraron gran interés al comienzo del proyecto.

Por otro lado, cabe indicar que la seguridad se ha tenido en cuenta en todas las fases del desarrollo del producto. La actualización de la normativa sobre Protección de Datos, el Reglamento General de Protección de Datos (RGPD) que entra en vigor el 25 de mayo de 2018, va a involucrar a todas aquellas personas responsables o encargados del tratamiento de datos que pertenezcan a la Unión Europea (UE), aunque también afecta a todas aquellas personas que no residan en la UE pero que tratan con datos de personas que sí pertenezcan a la UE. Se ha tenido esto en cuenta desde el diseño, y s eha puesto atención en medidas de seguridad que inciden en la protección de datos. Como trabajo futuro en este aspecto, antes de la puesta en producción de la aplicación, se plantea una revisión exhaustiva de los diferentes aspectos regulados por el RGPD.

Una cualidad destacable es la portabilidad obtenida, pues al tratarse de una aplicación web basada en una arquitectura cliente/servidor, para que el usuario pueda interactuar con ella simplemente tiene que tener un navegador web con conexión el servidor. En caso de que el servidor web estuviera alojado en el exterior, además se requeriría de una conexión a internet, en cambio, si el servidor estuviera alojado en local y la Base de Datos también, no sería necesaria esta conexión. La utilización de tecnologías tales como Apache, PHP, MySQL, etc, hace que la aplicación pueda ser desplegada en diferentes plataformas.

En el capítulo donde se realizaron las pruebas del producto, se ha comprobado que la aplicación funciona de acuerdo a los requisitos funcionales y no funcionales planteados en el capítulo de análisis. Posteriormente, con la validación por parte de los usuarios mediante la encuesta realizada, se ha verificado el resto de requisitos tales como la facilidad de uso, que la interfaz fuera adecuada para los usuarios finales, tiempos de respuesta adecuados, etc.

En cuanto a la posibilidad de adaptación de esta aplicación a otras Federaciones de judo, no solo a la de Castilla y León, no supondría un coste elevado de desarrollo ya que muchas de éstas tienen características comunes. Además gracias al diseño realizado con la ayuda del framework de Codeigniter se potencia esta extensibilidad.

Para finalizar, el trabajo de fin de grado me ha supuesto un acercamiento a un proceso de desarrollo software más complejo del que estaba acostumbrado a realizar. He puesto en práctica los conocimientos teóricos adquiridos en las diferentes asignaturas que he cursado. Por otro lado, también he aprendido nuevos lenguajes de programación y frameworks así como diferentes técnicas de desarrollo de software.

9.1. Objetivos alcanzados

Después de terminar el proyecto software planteado, se puede decir que se han cumplido los siguientes objetivos:

- Se ha logrado una aplicación con un rendimiento muy aceptable.
- Se ha creado una aplicación que ayuda en gran medida a gestionar los campeonatos de Judo, uno de los objetivos principales de la aplicación.
- Es una aplicación fácil de mantener, cada uno de los métodos de las diferentes clases están documentados individualmente, indicando la funcionalidad que realiza cada uno de ellos, mediante diagramas UML que facilitan la compresión de la misma. Es extensible a otras federaciones de Judo.
- Se ha conseguido una aplicación validada en cuanto usabilidad y satisfacibilidad por parte de los usuarios encuestados. Entre ellos se encuentran seleccionadores nacionales, entrenadores nacionales, árbitros de diferentes categorías y también competidores de diferentes niveles.

9.2. Trabajo futuro

- Crear una API Rest o Soap que proporcione los resultados de las competiciones y palmarés de competidores hacia el exterior, de modo que cualquier cliente que quiera esos datos los pueda tener de una forma más clara. De esta forma, se podrían crear aplicaciones en dispositivos móviles tanto en IOS como en Android, híbridas, etc.
- Validar campos tales como comunidades autónomas, provincias, países con recursos externos, APIs u otros. Actualmente la aplicación no valida este tipo de campos.
- Sería interesante implementar vistas de cara al administrador de la aplicación para que pueda gestionar la creación de delegados con un panel de control.
- Realizar una refactorización general a la aplicación. Aunque se ha intentado a lo largo de todo el desarrollo del proyecto refactorizar las clases, sería necesario una general para ayudar a mantenerlo en el futuro.
- Poder importar desde la aplicación de la Federación Internacional de Judo y Deportes Asociados los combates creados.
- Poder sacar resúmenes y estadísticas en gráficos por equipos, competidores, en las diferentes competiciones que participen.
- Crear un algoritmo de emparejamiento que tenga en cuenta la cabeza o cabezas de serie de cada peso.
- Cumplir con todos los aspectos de nueva la Ley de Protección de Datos, adaptados al RGPD. Aunque es verdad que se han tenido en cuenta muchos de los aspectos de la nueva normativa, algunos no han podido ser introducidos por falta de recursos. Serían, entre otros, la implementación de notificaciones de violaciones de seguridad de los datos, nombrar un delegado de protección de datos, realizar evaluaciones de impacto sobre la protección de datos y mantener un registro de tratamientos. Esto sería aceptable según circunstancias, es decir, si la aplicación se utilizase en entornos suficientemente amplios, o llegase a manejar datos especialmente protegidos.

Bibliografía

[1] Alex Mark y Kevin Louie Alvin Wang Alang Chang. *Framework Materialize*. URL: http://materializecss.com/. (Última visita: 2017-10-01).

[2] British Columbia Institute of Technology EllisLab. *Framework Codeigniter*. URL: https://codeigniter.com/docs. (Última visita :2017-11-05).

[3] John Resig. *Jquery API documentation*. URL: https://api.jquery.com/. (Última visita: 2017-10-01).

[4] Pablo de la Fuente. *Diapositivas asignatura de Planificación y Gestión de Proyectos*. URL: http://virtual.lab.inf.uva.es:20062/PGP/. (Última visita: 2018-02-03).

[5] IJF. *Judo Tournament Software*. URL: http://www.ippon.org/draw/. (Última visita: 2018-03-05).

[6] Judo Inside. *Herramienta para visualizar los resultados de competidores*. URL: https://www.judoinside.com/. (Última visita: 2018-01-23).

[7] IFJ. *Página web de la Federación Internacional de Judo*. URL: https://www.ijf.org/. (Última visita: 2018-02-23).

[8] StatCounter. *Popular browsers*. URL: http://gs.statcounter.com/browser-market-share. (Última visita: 2017-10-01).

[9] The PHP Group. *PHP documentation*. URL: http://php.net/docs.php. (Última visita: 2017-10-10).

[10] Oracle Corporation and/or its affiliates. *Mysql documentation*. URL: https://dev.mysql.com/doc/. (Última visita: 2017-10-05).

[11] Oracle Corporation and/or its affiliates. *MySQL Bench Read/Write*. URL: https://www.mysql.com/why-mysql/benchmarks/). (Última visita: 2017-11-20).

[12] Jack Edmonds. *Blossom algorithm: Paths, trees, and flowers*. URL: https://cms.math.ca/openaccess/cjm/v17/cjm1965v17.0449-0467.pdf. (Última visita: 2017-10-01).

[13] *Sistema de todos contra todos*. URL: https://es.wikipedia.org/wiki/Sistema_de_todos_contra_todos. (Última visita: 2017-10-01).

[14] Alcuino de York. *Problema de la col, el lobo y la cabra*. URL: https://es.wikipedia.org/wiki/Acertijo_del_lobo,_la_cabra_y_la_col. (Última visita: 2017-10-01).

[15] Thomas M. Connolly y Carolyn E. Begg. *Sistemas de Bases de Datos. Un enfoque práctico*. Pearson, 2003, págs. 255-313.

[16] AEPD. *Asociación Española de Protección de Datos. Guía de análisis de riesgos*. URL: http://www.agpd.es/portalwebAGPD/revista_prensa/revista_prensa/2018/notas_prensa/news/2018_02_28-ides-idphp.php. (Última visita: 2018-03-15).

[17] Federación Española de Judo y Deportes Asociados. *Pesos de Judo*. URL: http://www.rfejudo.com/index.php/normativa-index. (Última visita: 2017-10-01).

[18] Gof design patterns Gamma et al. *Strategy pattern*. URL: http://w3sdesign.com/GoF_Design_Patterns_Reference0100.pdf. (Última visita: 2017-10-01).

[19] Eric Bidelman. *Native HTML5 Drag and Drop*. URL: https://www.html5rocks.com/en/tutorials/dnd/basics/. (Última visita: 2017-10-01).

[20] Jesse James Garrett. *Ajax programming*. URL: http://adaptivepath.org/ideas/ajax-new-approach-web-applications/. (Última visita: 2017-10-20).

[21] G Suite administrator. *Gmail SMTP configuration*. URL: https://support.google.com/a/answer/176600?hl=es. (Última visita: 2017-10-01).

[22] Kent L. Norman John P. Chin Virginia A. Diehl. *Development of a Tool User Satisfaction of the Human-Computer Interface*. URL: http://lap.umd.edu/quis/publications/chin1988.pdf. (Última visita: 2017-11-20).

[23] Brennen Bearnes. *How to install LAMP*. URL: https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-16-04. (Última visita: 2018-03-05).

[24] Justin Ellingwood. *How To Create a SSL Certificate on Apache for Ubuntu 14.04*. URL: https://www.digitalocean.com/community/tutorials/how-to-create-a-ssl-certificate-on-apache-for-ubuntu-14-04. (Última visita: 2018-03-05).

[25] Etel Sverdlov. *Htaccess configuration*. URL: https://www.digitalocean.com/community/tutorials/how-to-use-the-htaccess-file. (Última visita: 2017-10-01).

Apéndice A

Manual de instalación

A.1. Introducción

En este apartado de la memoria se describirán los pasos a seguir para la instalación de la aplicación junto con la base de datos, además del software necesario para su correcto funcionamiento.

A.2. Requisitos previos a la instalación de la aplicación

A continuación se describirá la instalación de LAMP[23] que se llama al conjunto de software formado por Linux + Apache + MySql y PHP. Para instalar las versiones del software se han basado en los requisitos que tiene que tener un servidor para usar el framework backend *Codeigniter*[2].

A.2.1. Servidor Web

Para el funcionamiento de esta aplicación es necesario el uso de un servidor web, en este caso, Apache Server 2.

Como el desarrollo del proyecto se ha realizado en dos entornos de programación, uno de desarrollo o preproducción con un Sistema Operativo Windows 10 y un entorno de producción con un Sistema Operativo Linux, se detallarán las indicaciones de la instalación de Apache en el entorno de producción

Para la instalación en **Linux**, usaremos el repositorio que ofrece Ubuntu. Lo primero que tenemos que hacer es actualizar las librerías y paquetes de dicho centro

```
1 $sudo apt-get update
```

Figura A.1: Actualización del repositorio de Ubuntu

Cuando se hayan actualizado, ya podremos instalar Apache 2 con el siguiente comando

```
1 $sudo apt-get install apache2
```

Figura A.2: Instalación del servidor apache

El siguiente paso una vez que se ha instalado el servidor web es añadir el nombre del servidor en el fichero situado en /etc/apache2/apache2.conf.

```
1     ServerName virtual.lab.inf.uva.es:20063
2
```

Figura A.3: Declarar el nombre del servidor

Una vez hecho esto ya tendríamos disponible nuestro servidor para poder realizar los siguientes pasos, instalación de un gestor de base de datos, en este caso MySql y PHP.

Para la instalación del gestor de base de datos MySql en Linux seguiremos usando el centro de software ejecutando el siguiente comando.

```
$sudo apt-get install mysql-server
```

Figura A.4: Instalación de MySql

Importante: Hay que dejar libre el puerto 3306 para que MySql server pueda iniciarse.Si ese puerto estuviera siendo utilizado por otro proceso, tendríamos dos opciones válidas. La primera sería asignarle un puerto diferente (que no esté siendo utilizado). La segunda opción seria averiguar que proceso esta utilizando el puerto que usa MySql por defecto (3306) y cambiárselo a él.

Una vez instalado, tenemos que asegurarnos que la versión instalada es la 5.1.x o superior como indica en su página web el framework de Codeigniter.

Comprobada la versión de MySql, ya tendríamos listo el gestor de base de datos, por último tenemos que instalar PHP, en nuestro caso instalaremos la versión 7.0, la versión mínima recomendada por el framework es la 5.6.x o superior. También es cierto que debería de funcionar en la versión 5.3.7, pero aspectos de seguridad y rendimiento no están debidamente tratados por lo que no se recomienda su uso.

```
$ sudo apt-get install php libapache2-mod-php php-mcrypt php-mysql
```

Figura A.5: Instalación de las librerias de PHP y MySql

Para la instalación en **Windows** de Apache lo que vamos a hacer es descargarnos el software de XAMPP queya nos proporciona el servidor apache con php y mysql. Para la descarga, tenemos que ir a la página oficial de XAMPP https://www.apachefriends.org/es/download.html y seguir los pasos de instalación.

A.2.2. Configuración del certificado SSL

Una vez que tenemos el servidor listo, para cumplir con el requisito no funcional número , tenemos que instalar el correspondiente certificado SSL. Para ello debemos de seguir los pasos de la guía[24]. Una vez instalado el certificado, nuestra aplicación será más segura, pues toda la información viajara encriptada usando el protocolo TLS 1.2 , ECDHE_RSA con P-256 (una clave fuerte), y AES_256_GCM (un fuerte cifrado). Es cierto que es un certificado que no ha sido emitido por una certificadora oficial, en el momento que se reemplace por uno, al usuario se le mostrará el mensaje de que puede confiar en la página donde esta interactuando.

A.2.3. Configuración de la Base de Datos

En pasos anteriores ya hemos instalado el gestor de Base de Datos (MySQL), ahora tenemos que crear una Base de Datos y ejecutar el script que podemos encontrar en el apéndice C de esta memoria. Lo primero que tenemos que hacer es descargar el script de la base de datos que esta situado en la carpeta sql del repositorio

```
git clone https://0xpabgarc@bitbucket.org/0xpabgarc/judocup.git
```

Figura A.6: Clonación del repositorio dónde se encuentra el código de la aplicación

Una vez clonado el repositorio ya podemos crear la Base de Datos

```
sudo mysql -u root -p

create database judocup;

exit;

sudo mysql -u root -p judocup < ./judocup/sql/judocupv2.sql
```

Figura A.7: Creación de la base de datos

Si ejecutamos el comando *show tables*, tendría que tener el siguiente aspecto:

```
+-------------------------+
| Tables_in_judocupv2     |
+-------------------------+
| arbitroatatami          |
| categoriaarbitral       |
| combate                 |
| competicion             |
| cronometrador           |
| delegado                |
| equipo                  |
| grupoedad               |
| inscripcion             |
| participacion           |
| persona                 |
| peso                    |
| puntuacion              |
| tatami                  |
| tipocompeticion         |
+-------------------------+
```

Figura A.8: Tablas de la base de datos de JudoCup

Ya tendríamos listo la Base de Datos lista para ser enlazada a la aplicación.

Por último, para guardar copias de seguridad diarias de la BBDD tenemos que crear el siguiente script y asociarlo a crontab de Linux.

```
#!/bin/bash

# Credenciales de la base de datos
user='user'
password='password'
host='localhost'
db_name='judocupv2'

# definimos las variables del path y la fecha
backup_path='/backup'
date=$(date +"%d-%b-%Y")

# Establecemos los permisos predeterminados
umask 177

# Dump de la base de datos
mysqldump -user=$user password=$ -password -host=$host $db_name > $backup_path/
    $db_name$date.sql

#Comprimir backup
xz -9 $backup_path/$db_name-$date.sql
```

Figura A.9: Configuración de la copia de seguridad diaria de la base de datos

Para asociarlo a crontab, tenemos que realizar la siguiente operación:

```
crontab -e
// Añadimos al final del fichero la siguiente sentencia
@daily ruta_al_fichero.sh // 0 0 * * * ruta_al_fichero_sh
```

Figura A.10: Asociar el script de la figura A.9 a crontab

A.2.4. Instalación de la aplicación

Nos tenemos que situar en el directorio /var/www/html y una vez allí tenemos dos opciones, o clonarlo del repositorio (ver figura A.6), o si lo tenemos en un archivo comprimido, descomprimirlo aqui.

Nos movemos al directorio de configuración del proyecto descargado o se ha descomprimido

```
cd ./judocup/application/config/
```

Figura A.11: Ruta dónde se encuentra el código de la aplicación

Allí tenemos que editar estos dos ficheros, config.php y database.php. El primero para definir la URL base y el segundo para asociar la Base de Datos creada en pasos anteriores a la aplicación.

```
sudo vim config.php
<?php
...
$config['base_url'] = 'https://virtual.lab.inf.uva.es:20063/judocup/';
...
?>
```

Figura A.12: Configuración de la URL con el dominio del servidor

```
sudo vim database.php
// Editamos el fichero (hostname, username y password del gestor de BBDD)
$active_group = 'default';
$query_builder = TRUE;

$db['default'] = array(
        ...
        'hostname' => 'localhost',
        'username' => 'username',
        'password' => 'password',
        'database' => 'judocupv2',
        ...

);
```

Figura A.13: Configuración del fichero *database.php* para enlazar a la base de datos

Guardamos todos los cambios y ya tendríamos lista la aplicación para poder acceder a ella mediante la url: https://virtual.lab.inf.uva.es:20063/judocup/.

Es importante indicar que dentro de este directorio descargado, en la raiz principal se encuentra un fichero .htaccess, que sirve, entre otras cosas para limitar o bloquear el acceso a determinados directorios. La configuración utilizada es la que viene en [25].

Las credenciales para acceder a la aplicación son las siguientes. Una vez accedido, es recomendable cambiar la clave de acceso.

- **Usuario delegado**: judocupapp@gmail.com con contraseña 1234

Apéndice B

Manual de usuario

En este capítulo se presenta el manual de usuario, donde se explicará en cierta profundidad como realizar las tareas que permite la aplicación, es decir, sus funcionalidades. Se han creado subsecciones dentro de este capítulo para separar las funcionalidades por clases. En un primer apartado hablaremos de como iniciar sesión, recuperar la contraseña y cambio de idioma. En el siguiente explicaremos todo lo que este relacionado con las competiciones, es decir, inscripción de árbitros, competidores, creación de combates etc. En el tercer apartado hablaremos sobre la gestión de árbitros, desde su creación, hasta cambio de categorías arbitrales. En el apartado cuatro, la gestión de equipos, apartado cinco gestión de cronometradores, apartado seis, resultados de competidores, apartado siete, rol de cronometrador.

Bien es verdad que esta aplicación tiene diseño *responsive*, es decir, para todo tipo de dispositivos (tamaño de pantalla), pero el manual que se muestra a continuación es para dispositivos de pantallas grandes tales como las de un ordenador común o PC.

B.1. Menú de inicio, login de acceso , cambio de contraseña y cambio de idioma

Figura B.1: Pantalla de inicio, login de usuario

En la figura B.1 se muestra la pantalla de inicio de sesión para ambos roles de usuarios, como Cronometrador y como Delegado. Para poder acceder a la aplicación simplemente hay que introducir las credenciales de acceso, e-mail de usuario y la contraseña en los campos correspondientes. Una vez verificadas las credenciales, nos llevará a la vista B.2 si accedemos con el rol de Delegado, o bien si accedemos como un cronometrador nos llevará a la figura B.33.

Cambio de contraseña

En caso de que no recordemos la contraseña si estamos con el rol de delegado, tendríamos que pulsar

el botón de *Olvidaste la contraseña* que nos llevará a la vista de la figura B.3. Para poder cambiar la contraseña simplemente tenemos que introducir el correo con el que queremos acceder y esta nos enviará un e-mail cuyo mensaje se muestra en la figura B.4. Pulsamos en *Cambiar contraseña* y nos redireccionará a la vista que se muestra en la figura B.5. El siguiente paso es introducir los valores de la nueva contraseña y ya podremos acceder a la aplicación de nuevo.

Figura B.2: Pantalla de inicio general, tras iniciar sesión

En la imagen superior, figura B.2, tenemos en la parte superior los diferentes apartados a los que podemos acceder con el rol de Delegado. De izquierda a derecha sería, competiciones, equipos, árbitros, cronometradores,y ajustes de usuario. En la siguiente sección vamos a hablar de como crear una competición completa.

Figura B.3: Recuperar contraseña paso 1

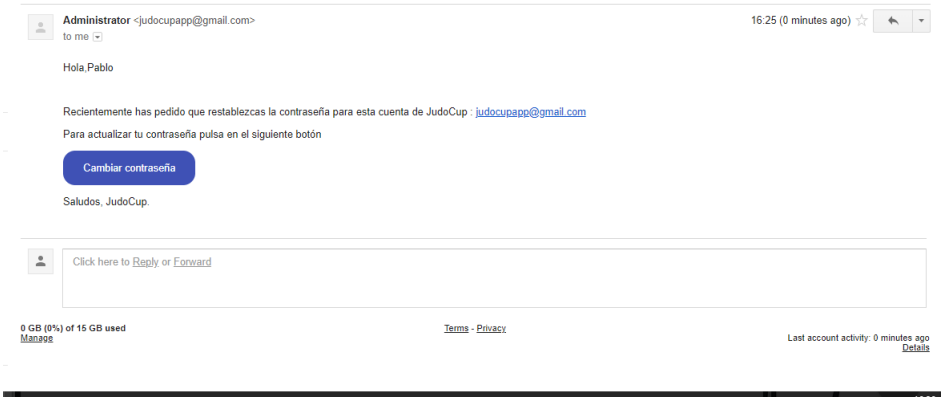


Figura B.4: Recuperar contraseña paso 2

Figura B.5: Recuperar contraseña paso 3

Cambio del idioma de la aplicación

Para cambiar el idioma de la aplicación, lo podemos hacer desde cualquier vista en la que estemos situados en ese momento. Simplemente tendremos que pulsar el botón de la parte superior derecha (figura B.6 en color rojo) y seleccionar el idioma que queramos elegir. El diccionario de datos hasta este momento solo da la posibilidad de inglés y español, por lo que solo podremos elegir entre uno de esos dos lenguajes.

Figura B.6: Cambio de idioma

B.2. Competiciones

Pulsando en la pestaña de *Competiciones*, nos llevará a la pantalla que se muestra en la figura B.7. Si tenemos creada ya alguna tendrá el aspecto que se muestra en la figura. Para cada competición registrada en el Sistema se mostrará una tarjeta del aspecto de la figura B.7. Cada una nos ofrecerá información, siguiendo de arriba hacia abajo,tenemos : nombre de la competición, nos da la opción de poder eliminar por completo esta, fecha de realización, grupo de edad y pabellón. Por último, tenemos, la creación de combates, inscripción de competidores e inscripción de árbitros. Pulsando en cada uno de ellos nos llevará a una vista diferente.

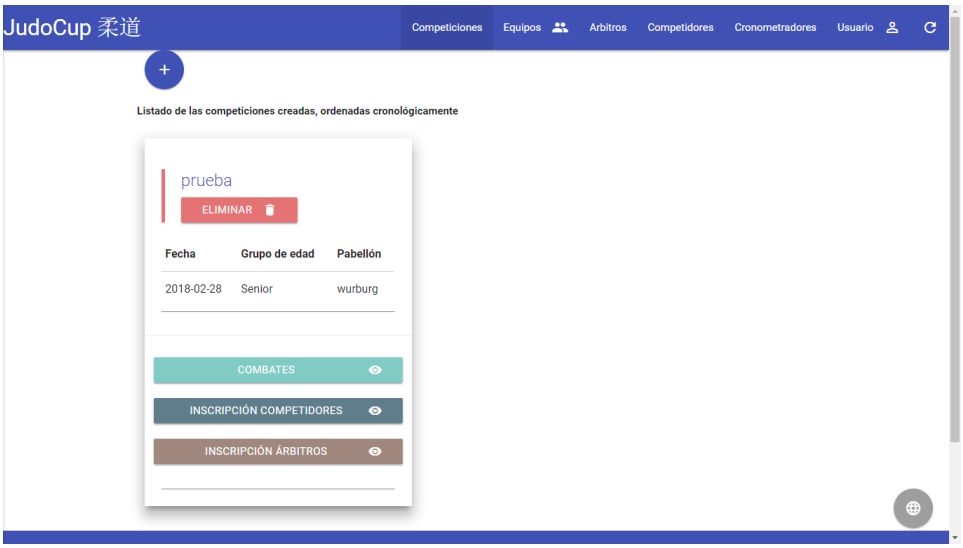


Figura B.7: Pantalla con las competiciones creadas

Por otro lado, si queremos crear una nueva competición, tendremos que pulsar en el botón de + de color azul y nos saldrá un cuadro de dialogo igual que se observa en la figura B.8. Para poder crearla correctamente, tendremos que rellenar los campos obligatorios, que son los que se muestran con el siguiente símbolo *(*)*.

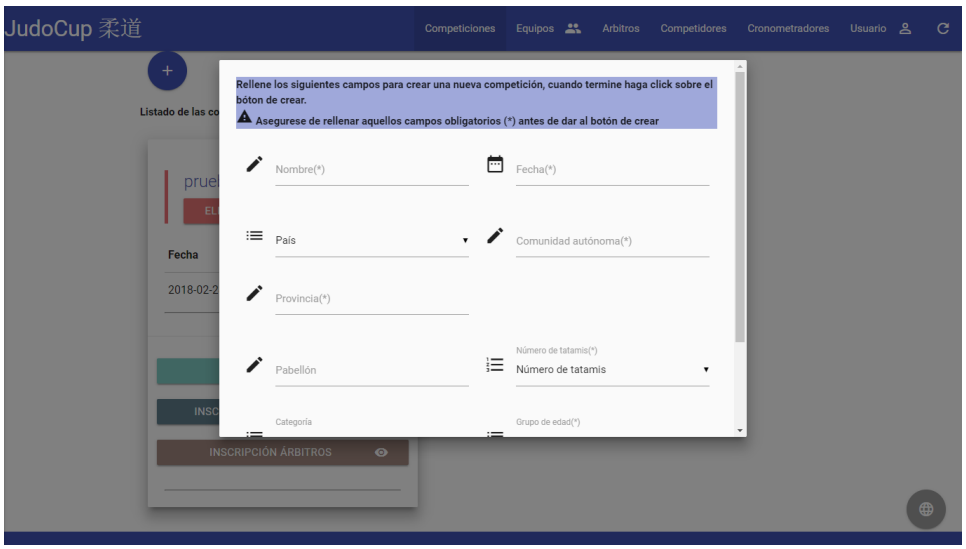


Figura B.8: Crear una nueva competición

B.2.1. Inscripción de competidores

En este apartado se hablará de las diferentes formas que hay de inscribir un competidor en una determinada competición. Pulsado sobre los diferentes pesos que se muestran en la figura B.9, podremos realizar la inscripción de un competidor. Llegados a este punto, tenemos que introducir el DNI del competidor que queremos inscribir. La aplicación validará si existe una persona que este inscrita con ese DNI, da igual si es árbitro o competidor. En caso de que exista, ya no tendremos que volver a rellenar los campos personales de la persona, si no que simplemente tenemos que indicar el equipo con el que va a realizar la competición (figura B.11). Si no existiera el DNI, tendríamos que crear a la persona de cero figura B.8 rellenando como mínimo los campos obligatorios *(*)*.

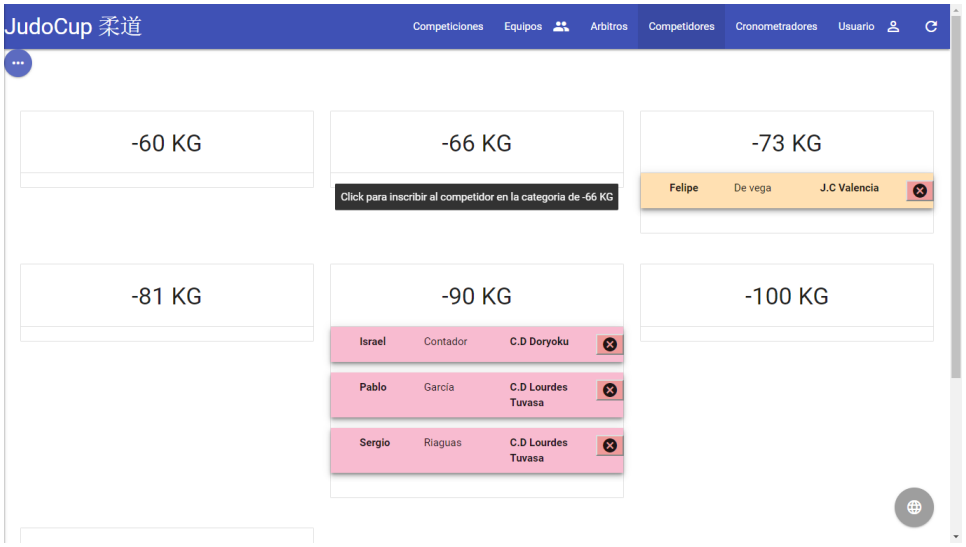


Figura B.9: Inscripción de competidores, pantalla principal

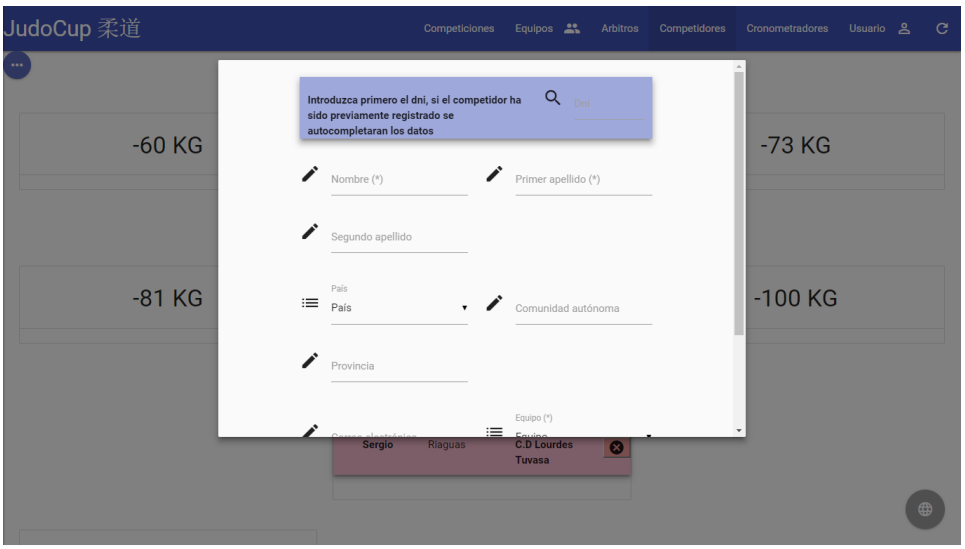


Figura B.10: Inscripción de competidores, si no existe un competidor con el DNI introducido

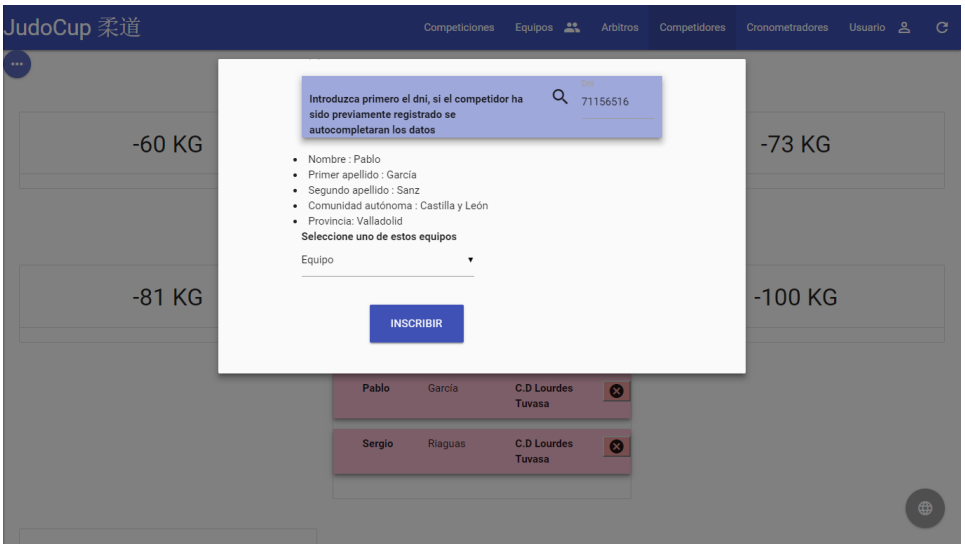


Figura B.11: Inscripción de competidores, si existe un competidor con el DNI introducido

Una vez realizadas inscripciones en el campeonato, tenemos la opción de poder cambiar el peso en el que va a competir la persona. Para poder realizar esta acción lo que tenemos que hacer es seleccionar un competidor y arrastrarlo hacia otro peso. En el ejemplo de la figura B.12 el competidor Felipe De vega del club J.C Valencia está inscrito en la categoría de menos de 73 kilogramos. Una vez realizado el cambio de peso, el competidor está en la categoría de menos de 66 kilogramos (ver figura B.13).

Para realizar todas estas acciones, pero en los pesos femeninos, tendríamos que pulsar el botón de los tres puntos situado en la parte superior izquierda para sacar el navegador lateral, figura B.14. El siguiente paso es pulsar en el *switch* femenino y nos redireccionará a los pesos femeninos de la competición indicada.

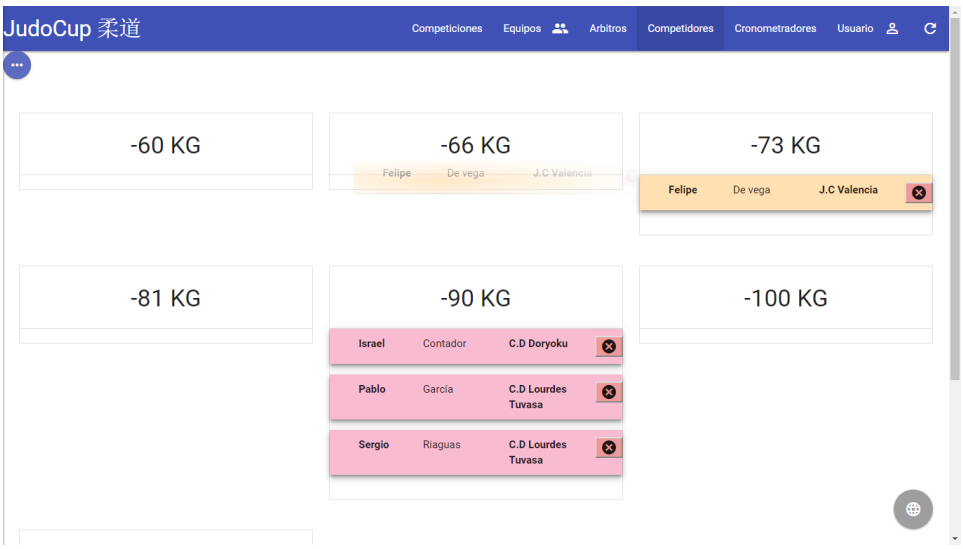


Figura B.12: Inscripción de competidores, cambio de inscripción paso 1

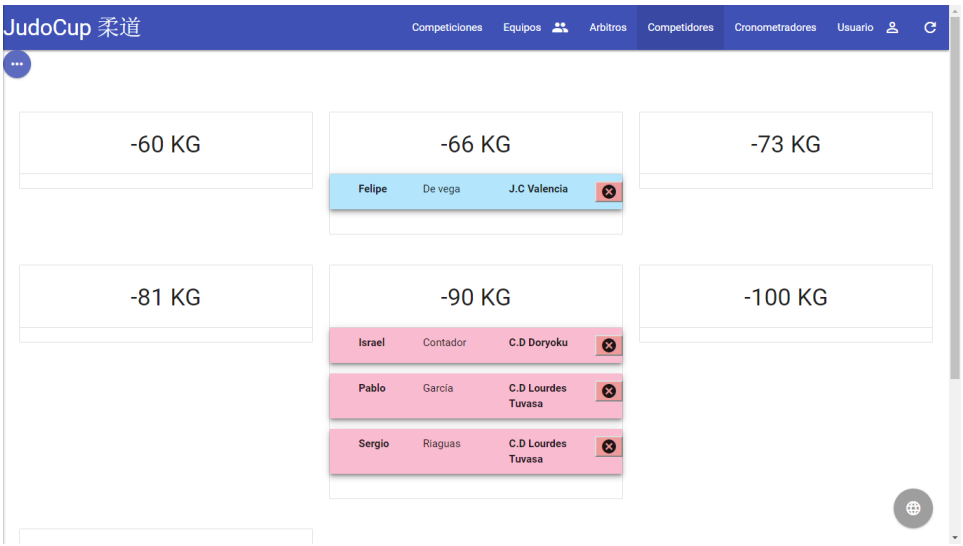


Figura B.13: Inscripción de competidores, cambio de inscripción paso 2

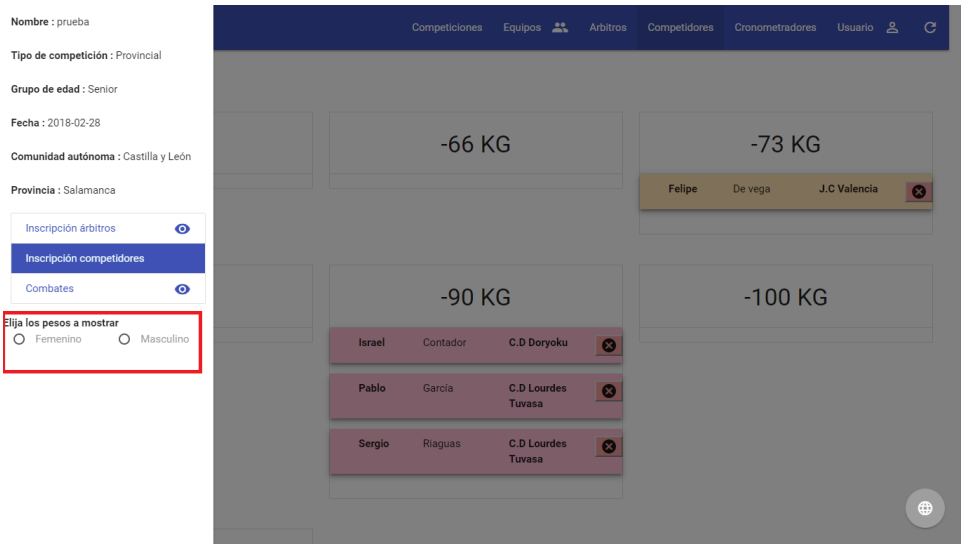


Figura B.14: Inscripción de competidores, cambiar a los pesos femeninos

B.2.2. Inscripción de árbitros y cronometradores

En esta sección se explicará como se deben de hacer las inscripciones de árbitros y cronometradores a una competición que este creada con anterioridad.

En primer lugar, para realizar la inscripción de un árbitro, tendremos que pulsar sobre el botón de + de color azul y nos saldrá un cuadro de diálogo parecido al de la figura B.15. Para seleccionar un árbitro, tenemos que indicar el club con el que esta federado y va a realizar la competición. Esto será importante para el futuro porque determinará que combates puede arbitrar y cuales no. En el ejemplo de la figura anterior citada, inscribimos a Sandra De la Cruz. El siguiente paso que nos quedaría es asociarla a un tatami de los que tenga la competición. En este ejemplo, podríamos asociarla al tatami 1, 2 o 3. Hemos elegido asociarlo al tatami 3 por lo que tenemos que hacer es arrastrar la tarjeta hasta ese tatami (figura B.16) y se completará la inscripción (figura B.17).

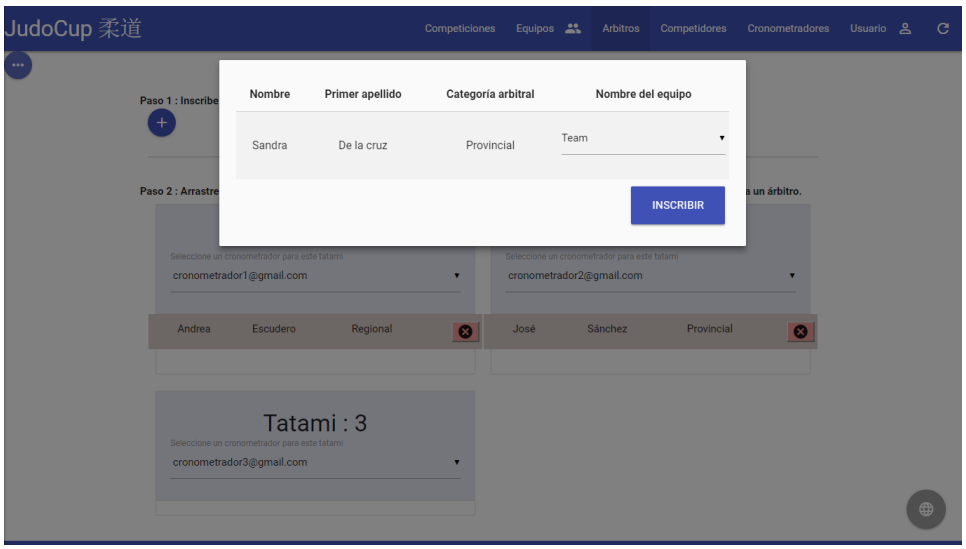


Figura B.15: Inscripción de árbitros, selección entre los disponibles

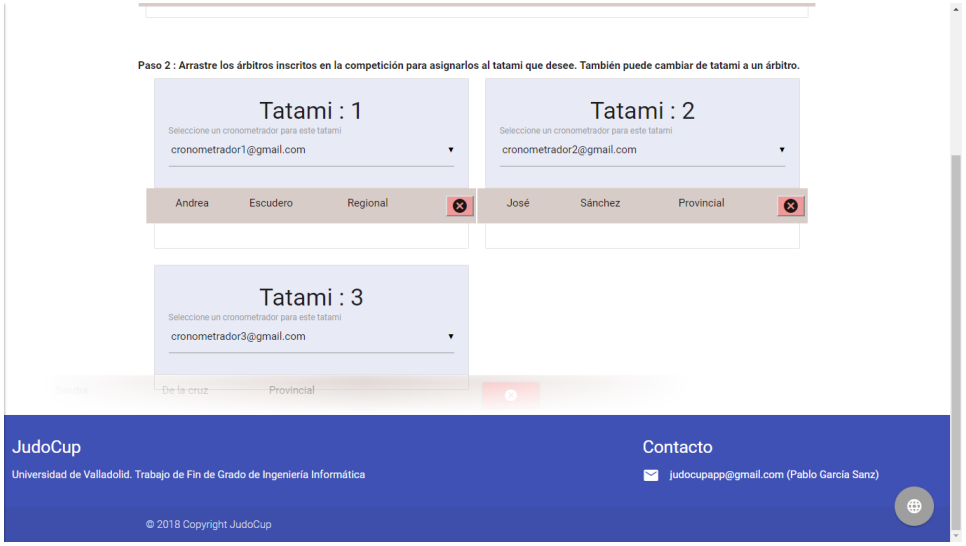


Figura B.16: Inscripción de árbitros, arrastrar hacia el tatami en el que va a arbitrar

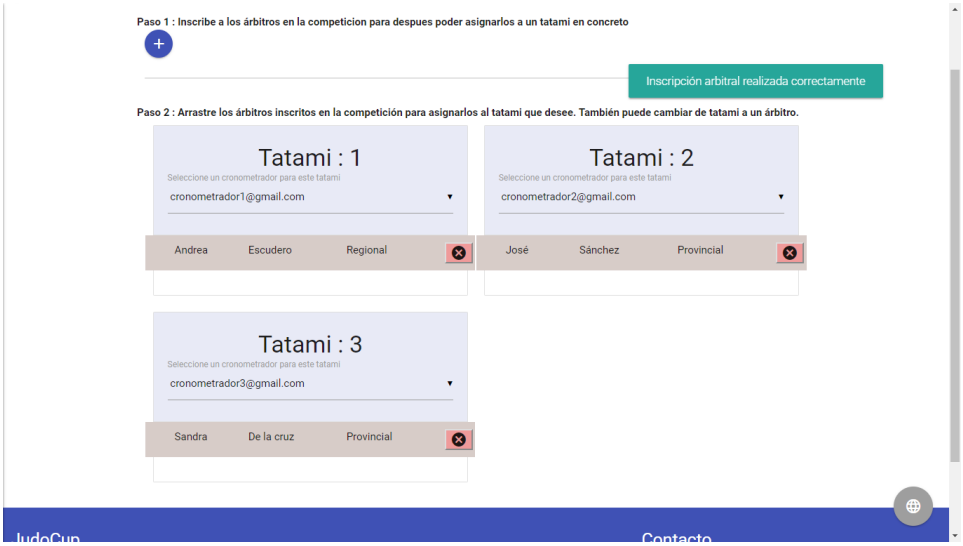


Figura B.17: Inscripción de árbitros, inscripción realizada

Realizar la asociación de un cronometrador a un tatami de la competición es mucho mas sencillo aún. Simplemente tendremos que hacer *click* sobre el desplegable, figura B.18 y seleccionar uno de los disponibles.

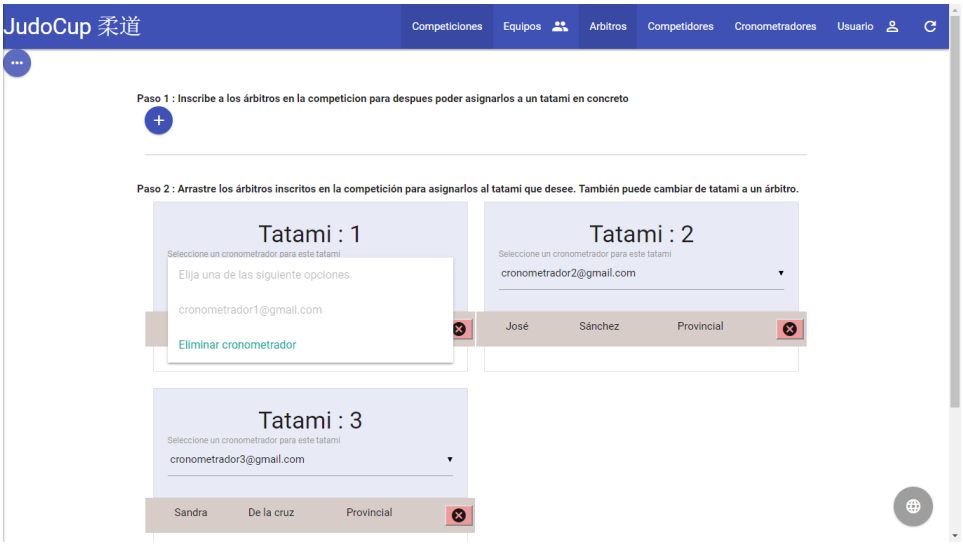


Figura B.18: Inscripción de cronometradores, seleccionar un cronometrador disponible

B.2.3. Creación de combates

El último apartado relacionado con las competiciones, sería la creación de combates entre los competidores inscritos dentro del mismo peso. Para ello, lo primero que tenemos que hacer es seleccionar uno de los pesos que se muestran en la figura B.19. Ahí se muestran tanto los pesos femeninos como los pesos masculinos. En el ejemplo hemos pulsado sobre el peso de menos de 90 kilogramos que nos redireccionará a la pantalla de la figura B.20. En esa figura podemos ver que tenemos, el listado de pesos sin puntuar, listado de pesos puntuados por el delegado o por el cronometrador y las posiciones en las que han quedado los competidores de ese peso.

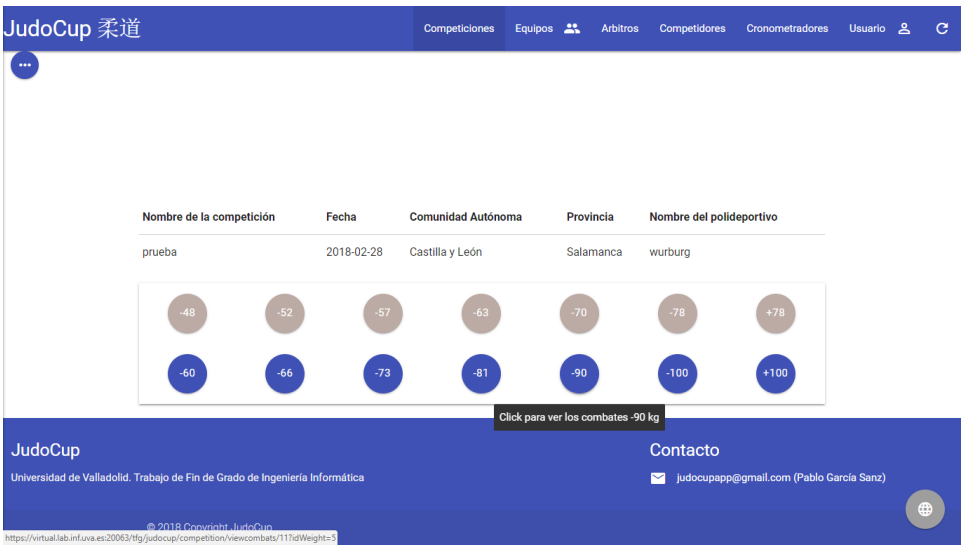


Figura B.19: Combates de la competición, pesos de la competición

Para puntuar uno de los combates que están sin puntuar, simplemente lo que tenemos que hacer es pulsar sobre el y se desplegará como en la imagen de la figura B.21, tendremos que introducir los valores correspondientes a al combate que se esté efectuando y pulsar el botón de *puntuar*. En este momento el combate pasara a la lista de combates puntuados.

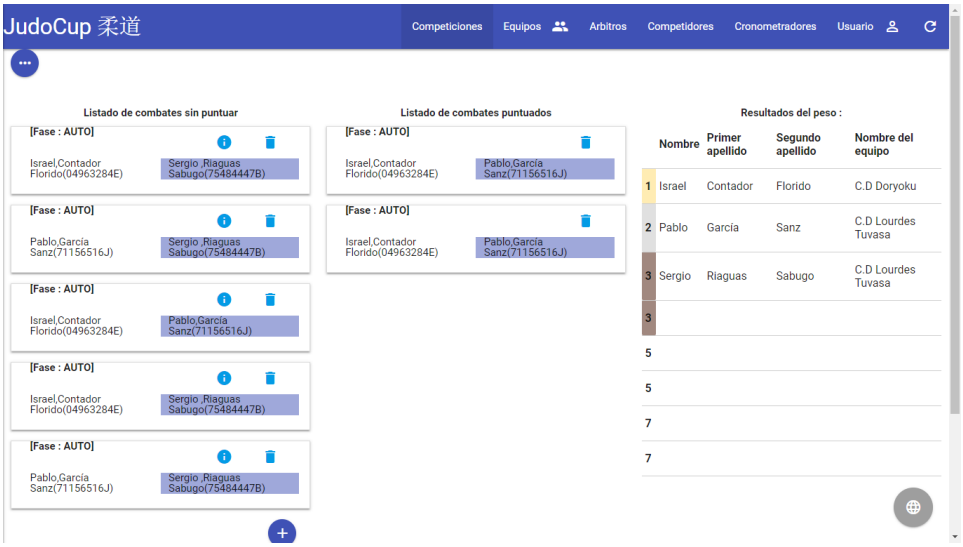


Figura B.20: Combates de la competición, pantalla con los combates creados

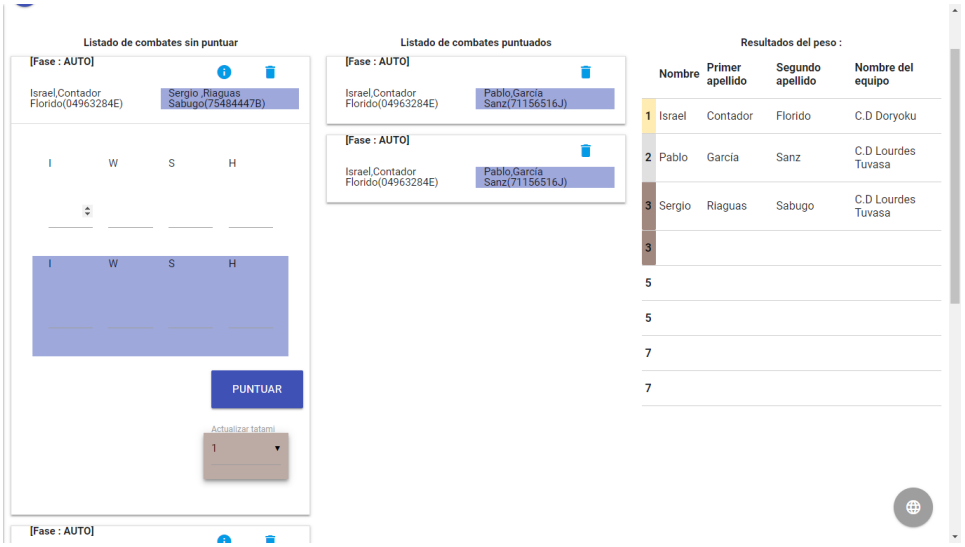


Figura B.21: Combates de la competición, puntuar un combate con rol de delegado

El cambio de tatami de un combate a otro diferente también es fácil de hacer, tendremos que pulsar sobre el combate que queramos cambiar, se abrirá el desplegable mencionado anteriormente y pulsaremos sobre el selector de tatami, figura B.22. Seleccionado uno de ellos y el combate se cambiará.

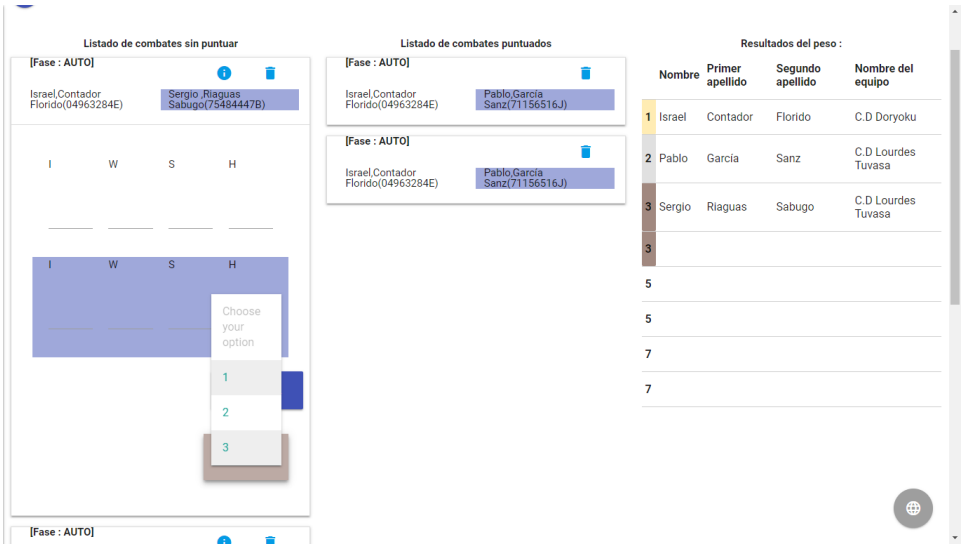


Figura B.22: Combates de la competición, cambio de tatami de un determinado combate

Una funcionalidad interesante, es **emparejar automáticamente**. Pulsando en los tres puntos de la parte superior izquierda de la pantalla de combates y nos saldrá un navegador lateral (figura B.23) Haciendo *click* sobre el desplegable podremos elegir el tipo de emparejamiento.

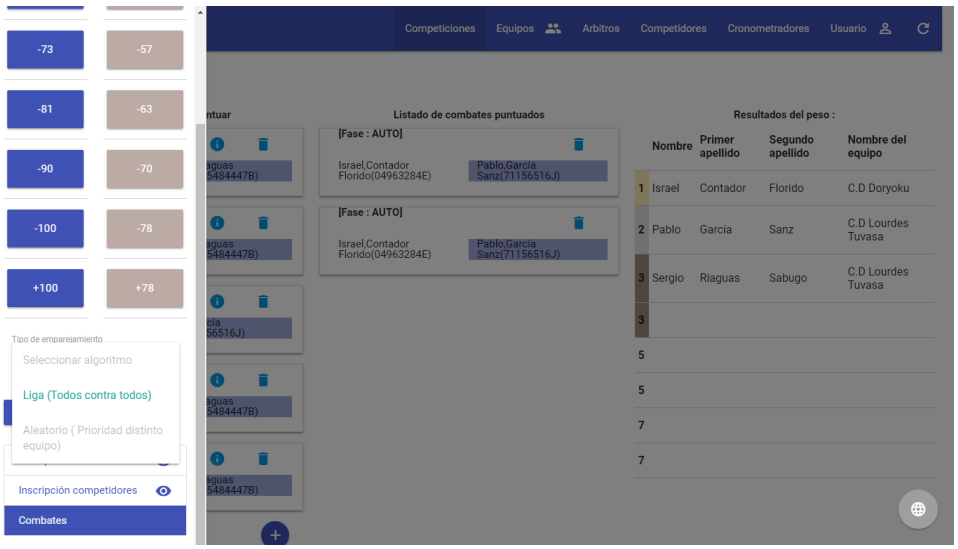


Figura B.23: Combates de la competición, seleccionar algoritmo de emparejamiento

Por último, para posicionar a un competidor en la tabla de resultados (a la derecha de la imagen), tendremos que situarnos sobre la posición que queramos colocar al competidor (ver figura B.24) y nos saldrá un desplegable donde tendremos que seleccionarle.

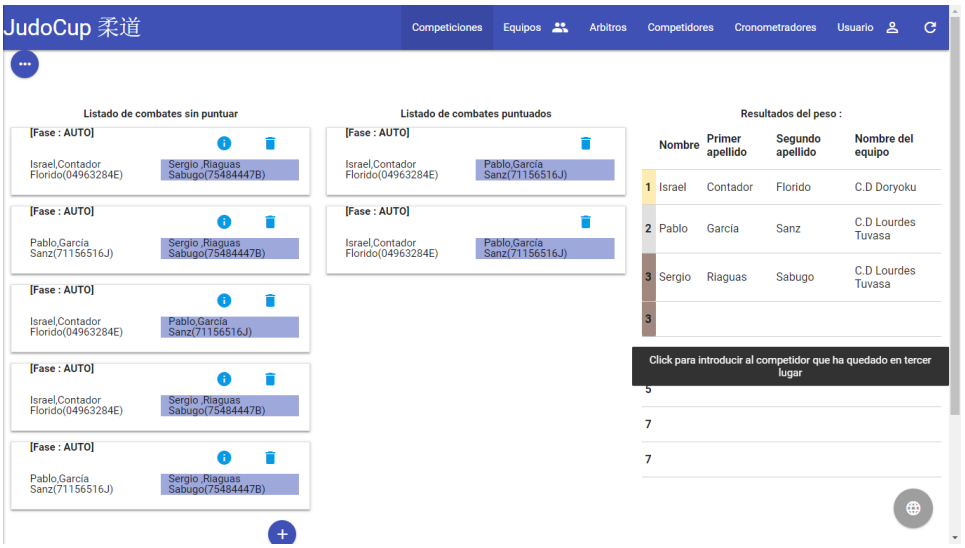


Figura B.24: Combates de la competición, introducir posición de un competidor

B.3. Gestión de árbitros

En este apartado se explicará más detalladamente la creación y gestión de árbitros. En primer lugar, cuando hacemos *click* en la pestaña de árbitros nos encontraremos con la vista que se muestra en la figura B.25. Tenemos a la izquierda de la imagen un formulario utilizado para la creación de un nuevo árbitro, en el centro los árbitros de una cierta categoría arbitral o todas con en este caso en concreto y a la derecha, tenemos un selector que se utiliza para mostrar los árbitros en función de una determinada categoría arbitral (ver figura B.26, el desplegable).

Creación de un árbitro

Para crear un árbitro, tenemos que rellenar los campos obligatorios *(*)* y si queremos los campos opcionales también. Se puede dar la situación que, el DNI coincida con el de un competidor que no tenía asignado una categoría arbitral, y es cuando la aplicación nos ofrece asignársela con el formulario que se ve en la figura B.29.

Actualización de la categoría arbitral

En el caso que queramos cambiar la categoría arbitral de un determinado árbitro, pulsaremos sobre el que queramos cambiar y se nos desplegarán dos botones, como los que se ven en la figura B.27. Elegimos la opción de *actualizar* y se nos abrirá un nuevo cuadro de dialogo, figura B.28. Dentro de este, tendremos que ir al desplegable de categoría arbitral y cambiársela por una de las opciones que nos ofrece la aplicación.

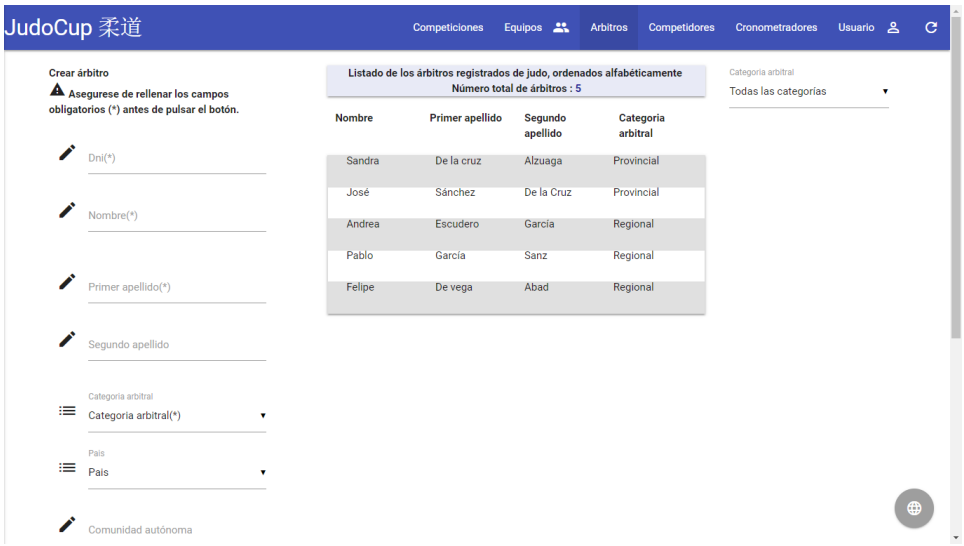


Figura B.25: Pantalla principal de árbitros

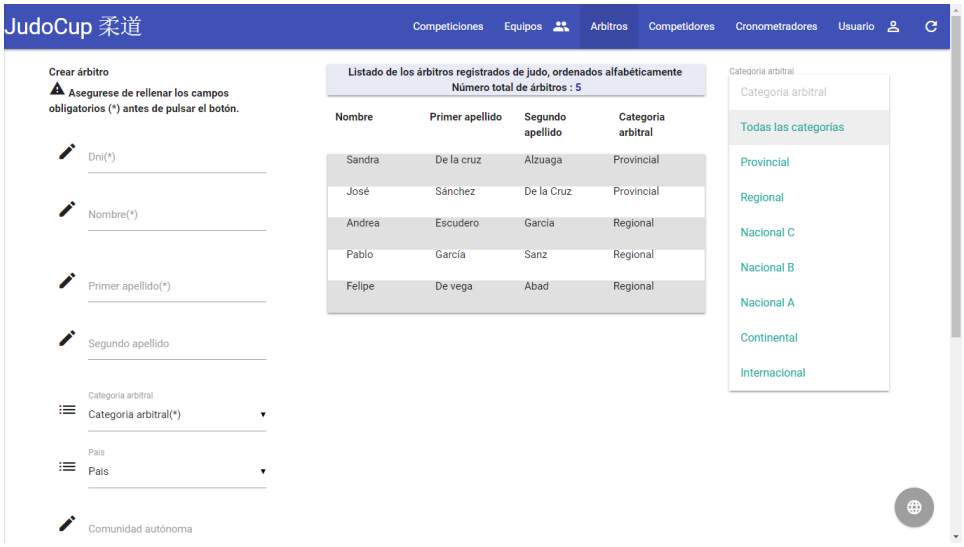


Figura B.26: Seleccionar árbitros por categoría arbitral

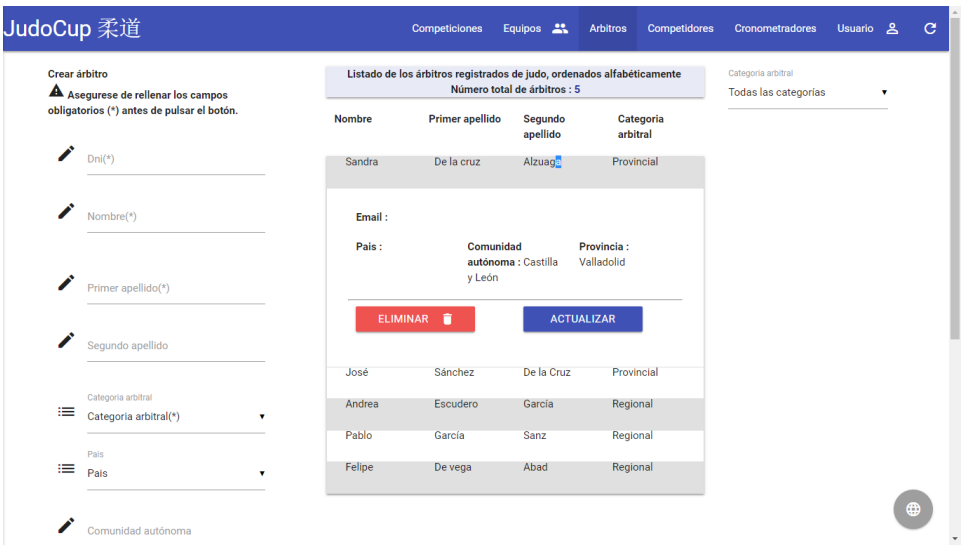


Figura B.27: Edición de un árbitro

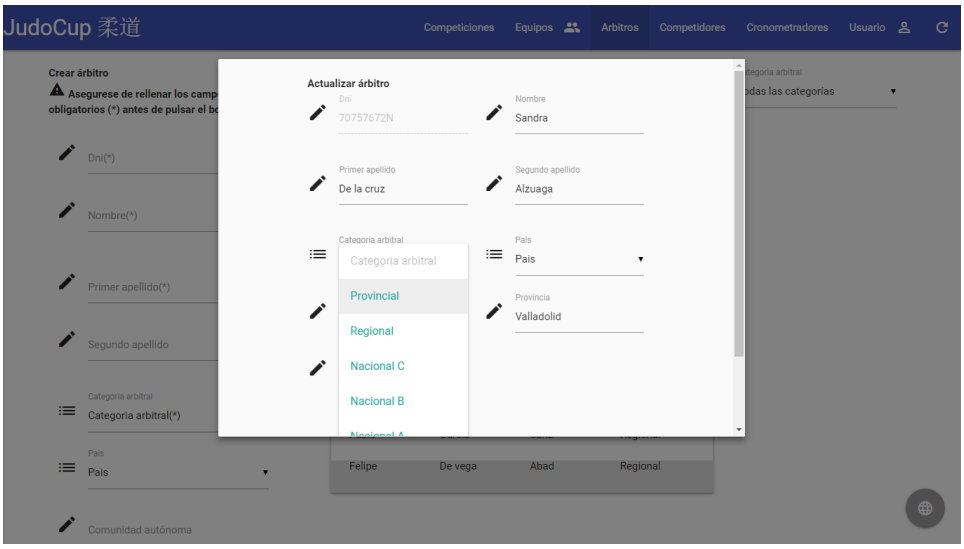


Figura B.28: Cambio de categoría arbitral

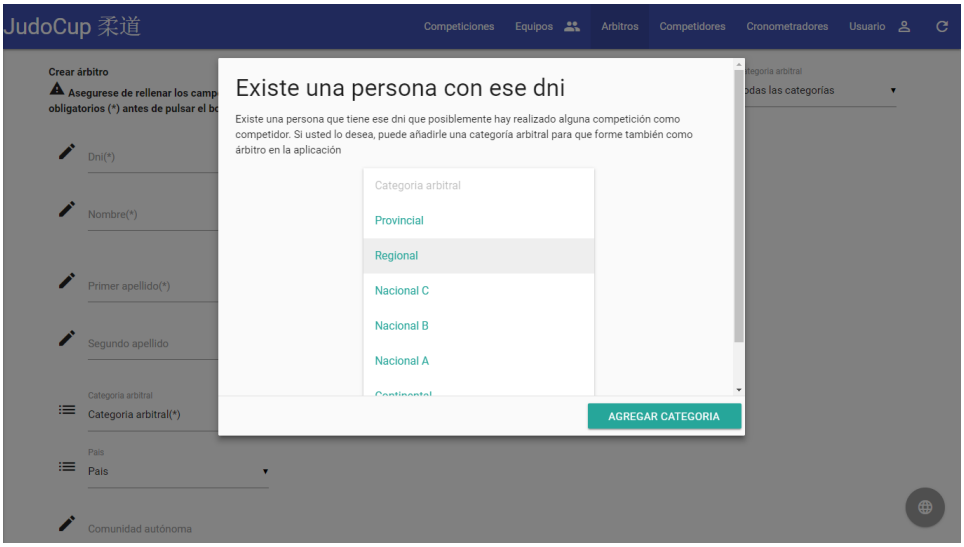


Figura B.29: Crear un árbitro con un DNI existente

B.4. Gestión de equipos

En esta sección se va a explicar como se gestionan los equipos de judo dentro de la aplicación JudoCup. Cuando realizamos *click* sobre la pestaña de árbitros, nos redirecciona a la vista que vemos en la figura B.30. En esta, podemos apreciar que a la izquierda se muestran todos los equipos registrados y a la derecha un formulario para poder crear un nuevo equipo de judo.

Actualizar equipo de judo

Si queremos actualizar cualquiera de los equipos, tenemos que situarnos sobre uno de ellos y se desplegarán dos botones, uno de ellos para actualizarle y otro para eliminarle, figura B.31. En este caso que queremos actualizarle tendríamos que pulsar sobre el botón de *actualizar* y se nos desplegará un formulario figura B.32 donde podremos editar cualquiera de los campos mostrados.

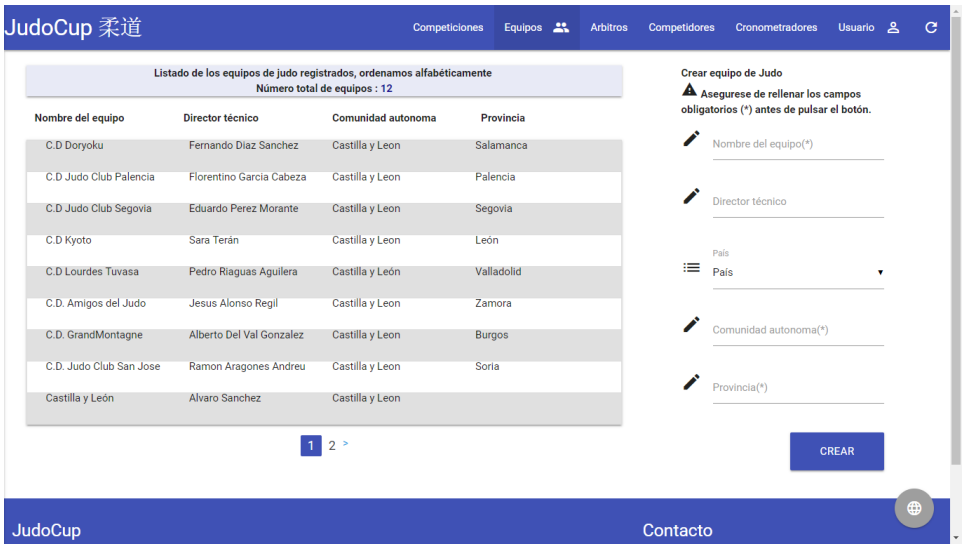


Figura B.30: Pantalla con los equipos registrados

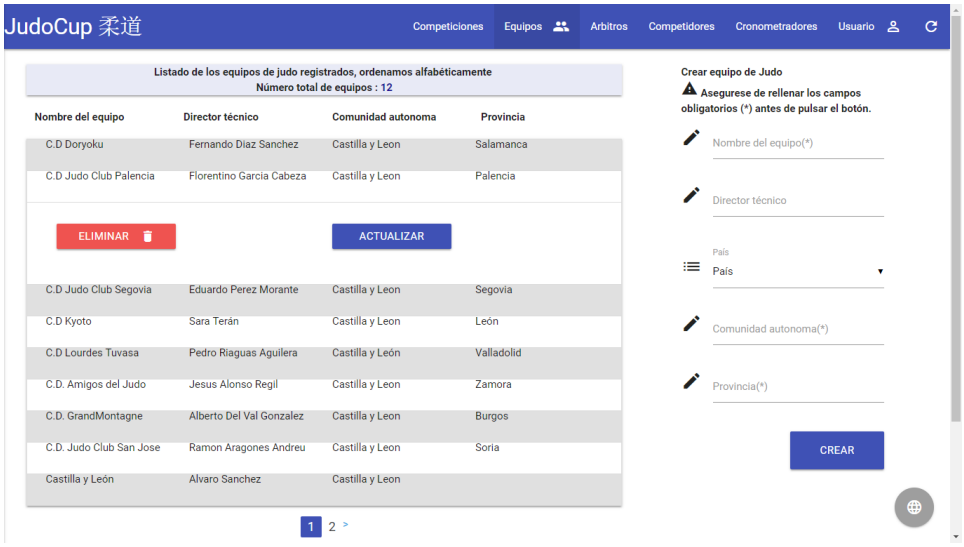


Figura B.31: Pantalla de edición de un determinado equipo

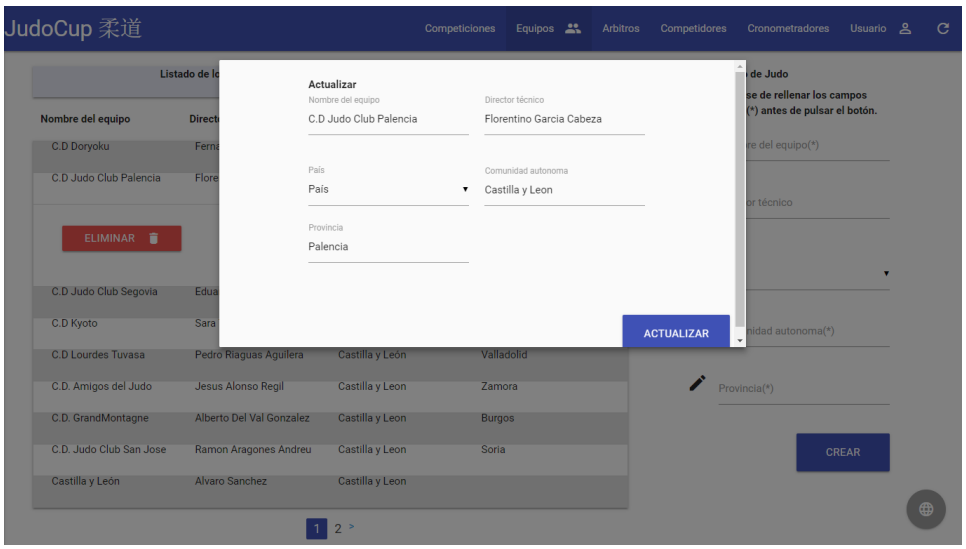


Figura B.32: Pantalla editando un equipo

B.5. Gestión de cronometradores

En esta sección se va a explicar más detalladamente como se gestionan los cronometradores que van a introducir la puntuación de los combates. Para llegar a la pantalla que se muestra en la figura B.33 tendremos que hacer *click* sobre la pestaña de cronometradores.

Crear un nuevo cronometrador

Para crear un nuevo cronometrador tendremos que presionar sobre el botón + de color azul, situado en la esquina superior izquierda y se nos abrirá un cuadro de dialogo como el que se puede ver en la figura B.34. Rellenamos los campos obligatorios y le asignamos una contraseña, damos a crear cronometrador y ya tendríamos uno listo para poder añadirle a las competiciones.

Cambiar la contraseña

En caso de que queramos cambiar la contraseña de un determinado cronometrador, tendremos que situarnos sobre el y se nos desplegará un botón, pulsamos sobre él y nos lanzará el cuadro de dialogo de la figura B.35, introducimos el valor de la nueva contraseña y ya la tendremos cambiada para poder utilizarla cuando queramos. Cabe destacar que es la única forma de cambiar la contraseña de un cronometrador, ya que es el delegado de la federación el que los gestiona.

Figura B.33: Listado de cronometradores creados

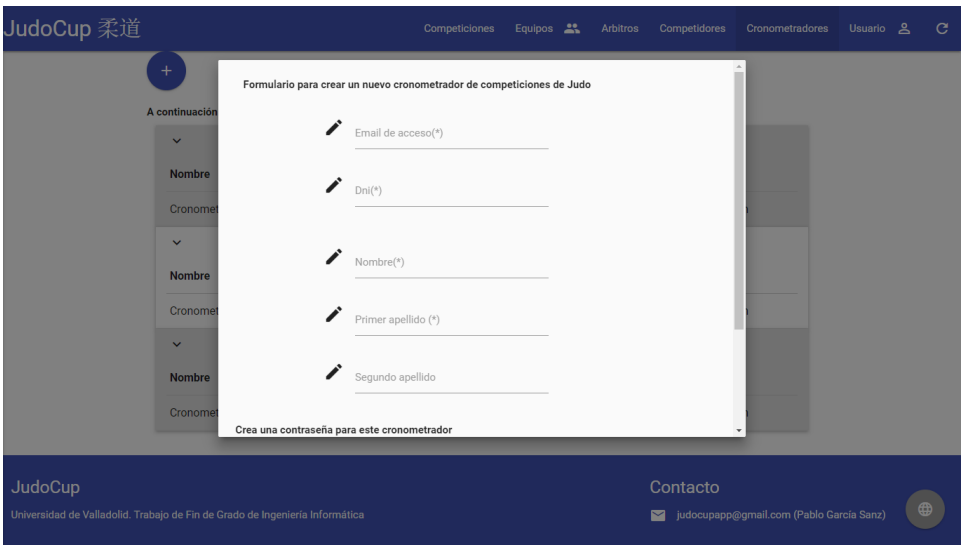


Figura B.34: Crear un nuevo cronometrador

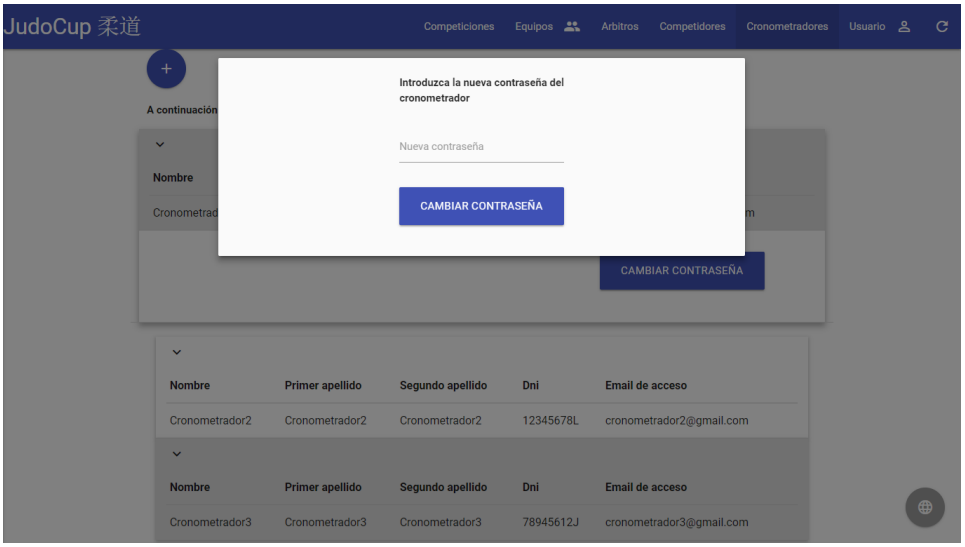


Figura B.35: Cambio de contraseña del cronometrador

B.6. Resultados de competidores

Si queremos ver los resultados de los competidores de los cuales se les ha ido puntuado en las competiciones que han sido inscritos, tendremos que pulsar en la pestaña de competidores. Tras realizar el *click*, se nos mostrará la pantalla de la figura B.36. En esta tenemos en la parte superior, un formulario que sirve para obtener competidores que cumplan las características seleccionadas. Tenemos para elegir un intervalo de fechas(fecha de inicio y fecha de fin de búsqueda), y selector de grupos de edad y un peso o todos los pesos.

Ver los resultados de un competidor

En el caso que queramos ver el medallero de un determinado competidor, tendremos que pulsar sobre uno de ellos y se nos abrirá un desplegable parecido al de la figura B.37, en el que veremos los resultados que ha sacado en el intervalo de fechas seleccionado.

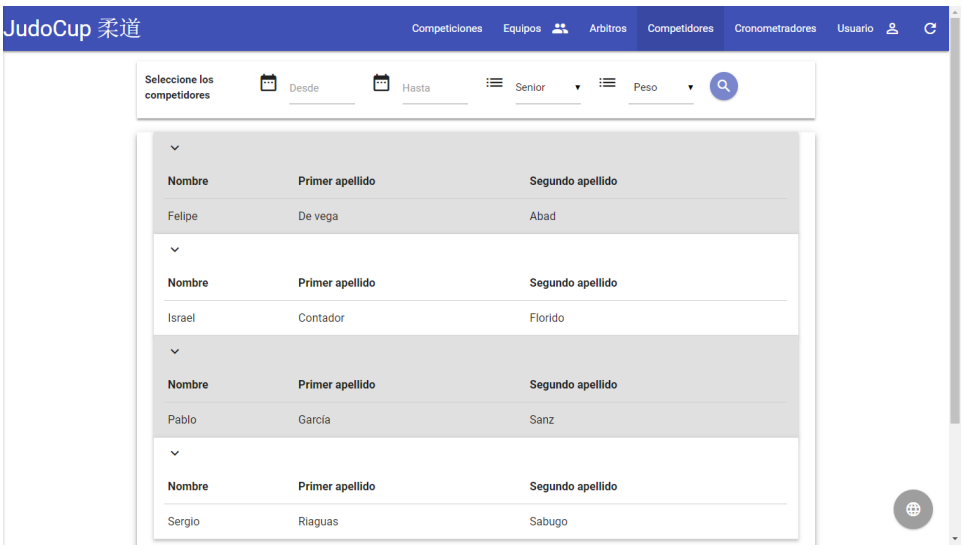


Figura B.36: Pantalla con todos los competidores que han sido inscritos en una competición

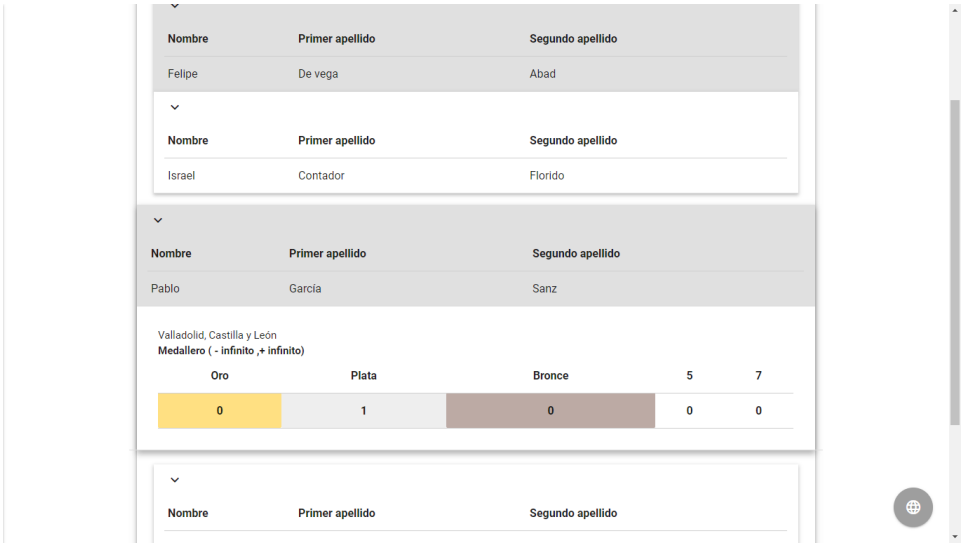


Figura B.37: Resultados de un competidor en concreto

B.7. Rol de cronometrador,puntuaciones

Cuando accedemos a la aplicación con un usuario de tipo cronometrador, se nos mostrará la vista de la figura B.38 donde aparecerán todas las competiciones en las que el cronometrador se le haya registrado. En este caso solo esta asignado en una competición, si se quieren ver los combates de ese cronometrador tendremos que pulsar sobre el botón de *ir*. Una vez pulsado, nos dirigirá a la pantalla de la figura B.39. En esta, tendremos dos listas, la de la izquierda son los combates que están sin puntuar y a la derecha los combates que ya han recibido una puntuación. De los combates no puntuados el cronometrador podrá ver los resultados, lo que no podrá es editar la puntuación, eso solo lo puede hacer el delegado.

Puntuar un combate

El cronometrador tendrá que seleccionar uno de los combates de la lista de no puntuados y se le abrirá un desplegable como el de la imagen B.40. Una vez terminado de introducir todas las puntuaciones,para enviar el resultado tendrá que pulsar el botón de *puntuar*.

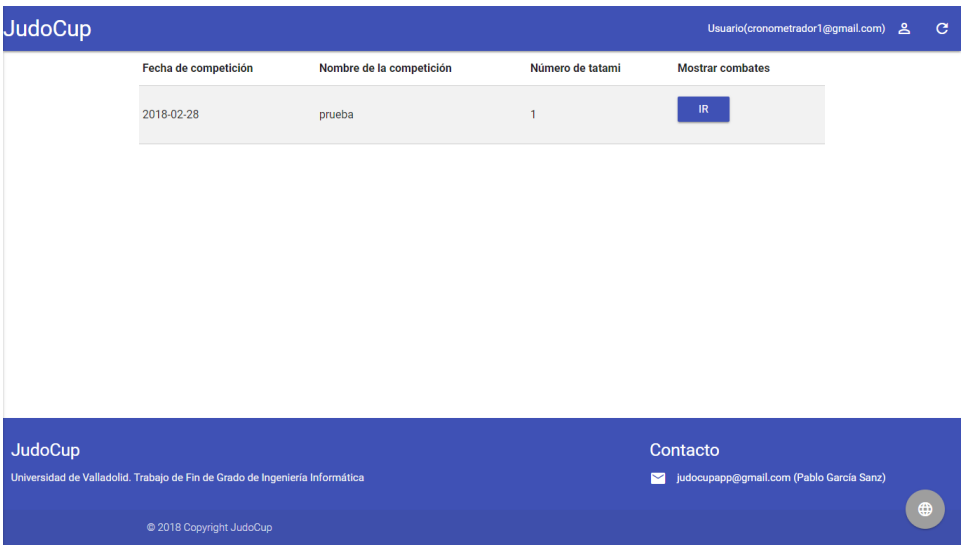


Figura B.38: Competiciones de un cronometrador

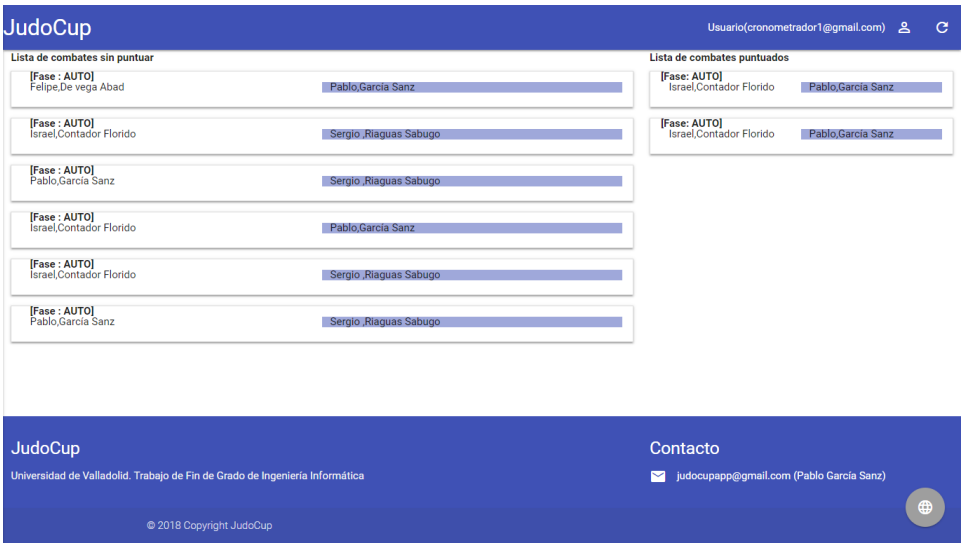


Figura B.39: Combates que tiene que puntuar y no puntuados

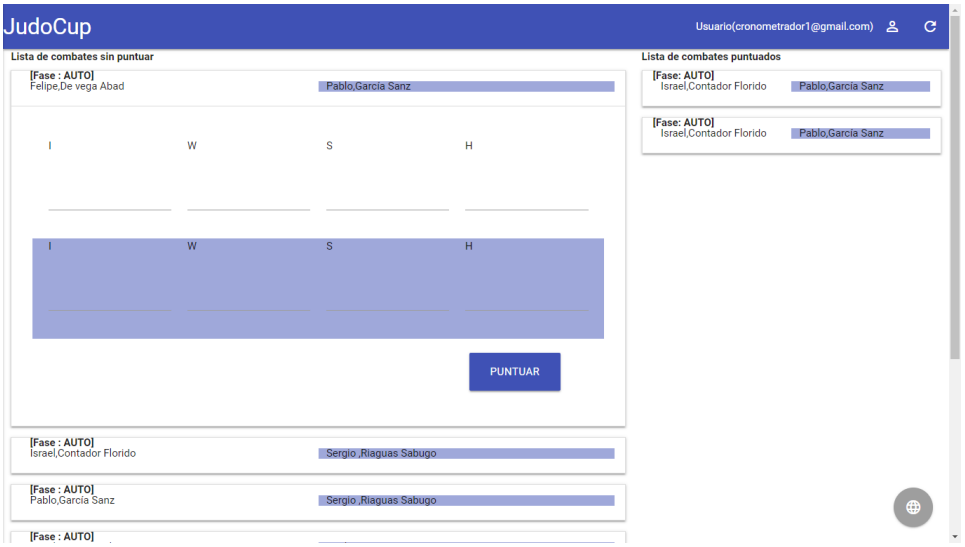
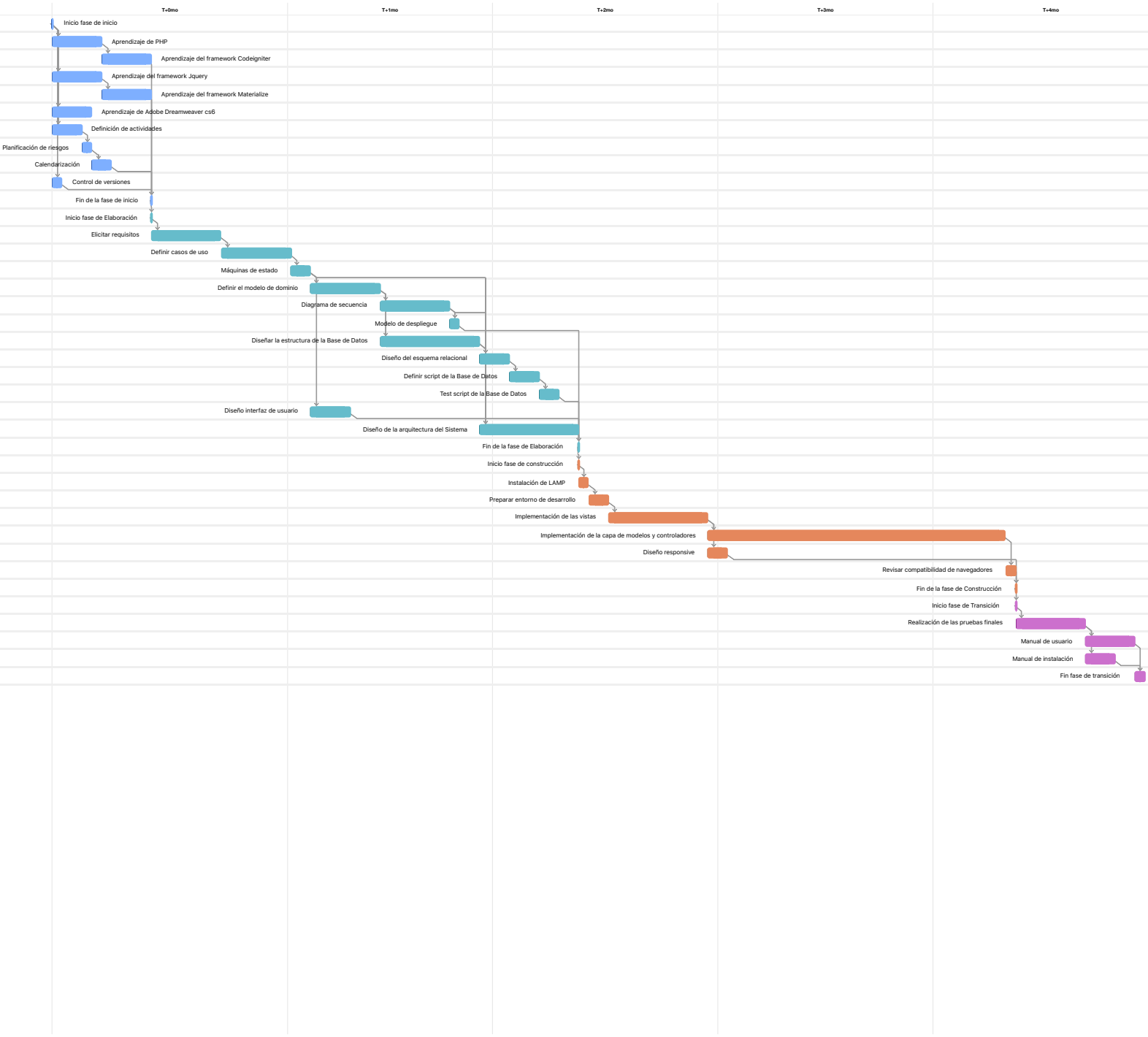


Figura B.40: Puntuación de un determinado combate

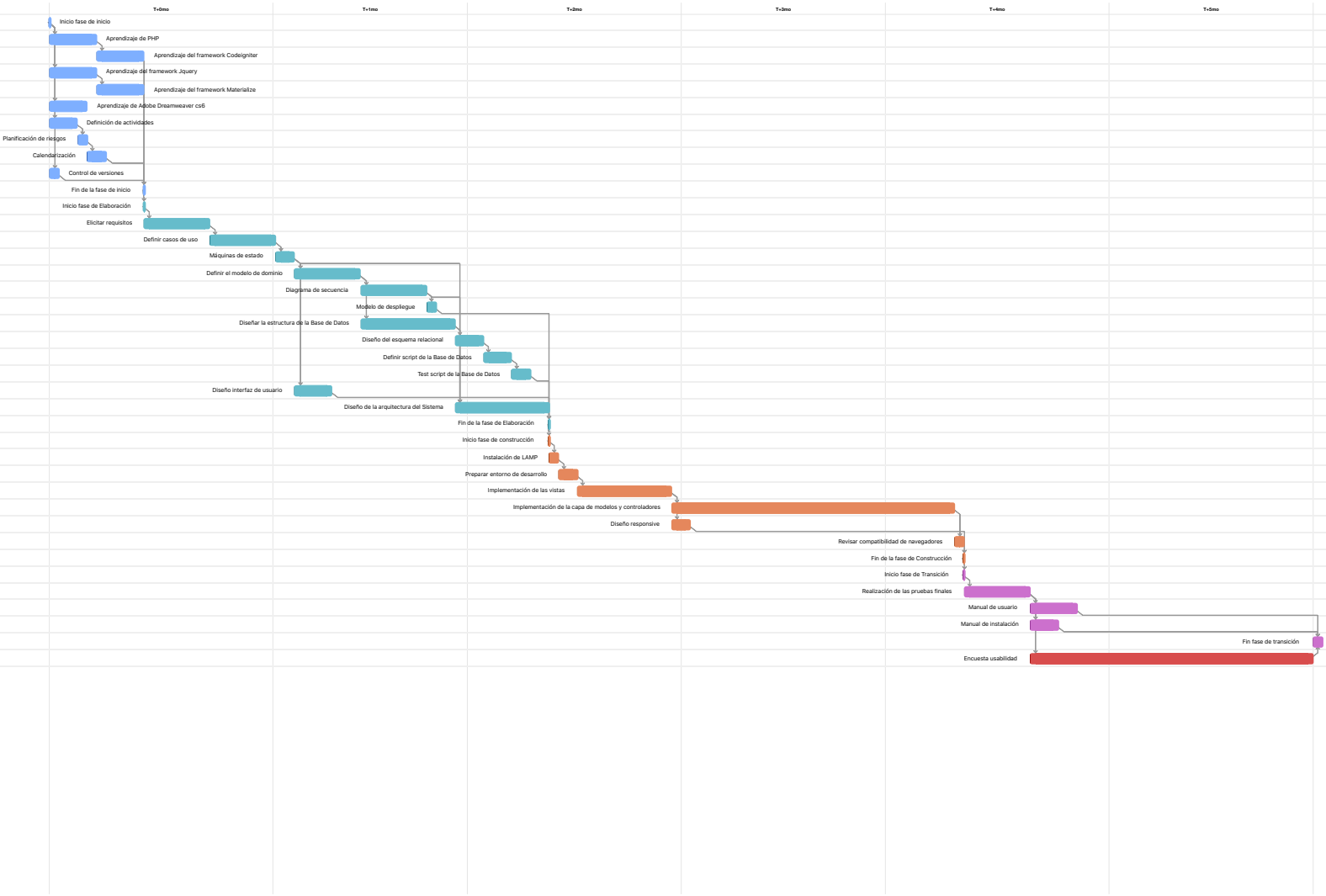
Apéndice C

Diagramas de Gantt

C.1. Planificación inicial


T+0mo
T+1mo
T+2mo
T+3mo
T+4mo
Inicio fase de inicio
Aprendizaje de PHP
Aprendizaje del framework Codeigniter
Aprendizaje del framework Jquery
Aprendizaje del framework Materialize
Aprendizaje de Adobe Dreamweaver cs6
Definición de actividades
Planificación de riesgos
Calendarización
Control de versiones
Fin de la fase de inicio
Inicio fase de Elaboración
Elicitar requisitos
Definir casos de uso
Máquinas de estado
Definir el modelo de dominio
Diagrama de secuencia
Modelo de despliegue
Diseñar la estructura de la Base de Datos
Diseño del esquema relacional
Definir script de la Base de Datos
Test script de la Base de Datos
Diseño interfaz de usuario
Diseño de la arquitectura del Sistema
Fin de la fase de Elaboración
Inicio fase de construcción
Instalación de LAMP
Preparar entorno de desarrollo
Implementación de las vistas
Implementación de la capa de modelos y controladores
Diseño responsive
Revisar compatibilidad de navegadores
Fin de la fase de Construcción
Inicio fase de Transición
Realización de las pruebas finales
Manual de usuario
Manual de instalación
Fin fase de transición

C.2. Planificación final


T+0mo
T+1mo
T+2mo
T+3mo
T+4mo
T+5mo
Inicio fase de inicio
Aprendizaje de PHP
Aprendizaje del framework CodeIgniter
Aprendizaje del framework Jquery
Aprendizaje del framework Materialize
Aprendizaje de Adobe Dreamweaver cs6
Definición de actividades
Planificación de riesgos
Calendarización
Control de versiones
Fin de la fase de inicio
Inicio fase de Elaboración
Elicitar requisitos
Definir casos de uso
Máquinas de estado
Definir el modelo de dominio
Diagrama de secuencia
Modelo de despliegue
Diseñar la estructura de la Base de Datos
Diseño del esquema relacional
Definir script de la Base de Datos
Test script de la Base de Datos
Diseño interfaz de usuario
Diseño de la arquitectura del Sistema
Fin de la fase de Elaboración
Inicio fase de construcción
Instalación de LAMP
Preparar entorno de desarrollo
Implementación de las vistas
Implementación de la capa de modelos y controladores
Diseño responsive
Revisar compatibilidad de navegadores
Fin de la fase de Construcción
Inicio fase de Transición
Realización de las pruebas finales
Manual de usuario
Manual de instalación
Fin fase de transición
Encuesta usabilidad

Apéndice D

Script de Base de Datos

```sql
-- @author Pablo Garcia Sanz
-- Scrypt Base de Datos de Judocup
-- Mysql

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Base de datos: `judocupv2`
--

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `arbitroatatami`
--

CREATE TABLE `arbitroatatami` (
  `idTatami` int(11) NOT NULL,
  `dniArbitro` char(30) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `categoriaarbitral`
--

CREATE TABLE `categoriaarbitral` (
  `idCategoriaArbitral` int(11) NOT NULL,
  `descripcion` char(30) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Volcado de datos para la tabla `categoriaarbitral`
--

INSERT INTO `categoriaarbitral` (`idCategoriaArbitral`, `descripcion`) VALUES
(1, 'Provincial'),
(2, 'Regional'),
(3, 'Nacional C'),
(4, 'Nacional B'),
(5, 'Nacional A'),
(6, 'Continental'),
```

```
(7, 'Internacional');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `combate`
--

CREATE TABLE `combate` (
  `idCombate` int(11) NOT NULL,
  `tiempo` int(11) DEFAULT NULL,
  `fase` char(30) DEFAULT NULL,
  `idTatami` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `competicion`
--

CREATE TABLE `competicion` (
  `idCompeticion` int(11) NOT NULL,
  `nombre` char(30) NOT NULL,
  `fecha` date NOT NULL,
  `paisISOA3` char(3) DEFAULT NULL,
  `autonomia` char(30) DEFAULT NULL,
  `provincia` char(30) DEFAULT NULL,
  `pabellon` char(30) DEFAULT NULL,
  `emailDelegado` char(30) DEFAULT NULL,
  `idGrupoEdad` int(11) DEFAULT NULL,
  `idTipoCompeticion` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;



-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `cronometrador`
--

CREATE TABLE `cronometrador` (
  `email` varchar(30) NOT NULL,
  `password` varchar(60) NOT NULL,
  `nombre` varchar(30) NOT NULL,
  `apellido1` varchar(30) NOT NULL,
  `apellido2` varchar(30) NOT NULL,
  `dni` varchar(30) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `delegado`
--

CREATE TABLE `delegado` (
  `email` varchar(30) NOT NULL,
  `password` varchar(60) NOT NULL,
  `token` varchar(60) DEFAULT NULL,
  `nombre` varchar(30) DEFAULT NULL,
  `apellido1` varchar(30) DEFAULT NULL,
  `apellido2` varchar(30) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

```
--
-- Volcado de datos para la tabla `delegado`
--

INSERT INTO `delegado` (`email`, `password`, `token`, `nombre`, `apellido1`, `apellido2`) VALUES
('judocupapp@gmail.com', '$2y$10$t5mXNatLtgZytmCxu4exKeyfgLiAmAXcnDYD1r/DaHCgI.vesjZGi', '$2y$10$//Wf0ca2Nym2G6TlFRFRu.T8QNhQjufYgJu1zwNogRmEqAuA2vrn2', 'Pablo', 'Garcia', 'Sanz');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `equipo`
--

CREATE TABLE `equipo` (
  `nombre` varchar(30) NOT NULL,
  `directorTecnico` varchar(30) DEFAULT NULL,
  `paisISOA3` char(3) DEFAULT NULL,
  `autonomia` varchar(30) DEFAULT NULL,
  `provincia` varchar(30) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `grupoedad`
--

CREATE TABLE `grupoedad` (
  `idGrupoEdad` int(11) NOT NULL,
  `descripcion` char(30) NOT NULL,
  `edadMinima` int(11) DEFAULT NULL,
  `edadMaxima` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Volcado de datos para la tabla `grupoedad`
--

INSERT INTO `grupoedad` (`idGrupoEdad`, `descripcion`, `edadMinima`, `edadMaxima`) VALUES
(1, 'Benjamin', 9, 10),
(2, 'Alevin', 11, 12),
(3, 'Infantil', 13, 14),
(4, 'Cadete', 15, 17),
(5, 'Junior', 18, 20),
(6, 'Senior', NULL, NULL);

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `inscripcion`
--

CREATE TABLE `inscripcion` (
  `idCompeticion` int(11) NOT NULL,
  `dniPersona` varchar(9) NOT NULL,
  `fecha_inscripcion` date NOT NULL,
  `nombreEquipo` varchar(30) DEFAULT NULL,
  `arbitra` bit(1) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------
```

```
--
-- Estructura de tabla para la tabla `participacion`
--

CREATE TABLE `participacion` (
  `idPeso` int(11) NOT NULL,
  `dniCompetidor` varchar(9) NOT NULL,
  `puesto` int(11) DEFAULT NULL,
  `fecha_participacion` date NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `persona`
--

CREATE TABLE `persona` (
  `dni` varchar(9) NOT NULL,
  `nombre` varchar(30) NOT NULL,
  `apellido1` varchar(30) NOT NULL,
  `apellido2` varchar(30) DEFAULT NULL,
  `paisISOA3` char(3) DEFAULT NULL,
  `autonomia` varchar(30) DEFAULT NULL,
  `provincia` varchar(30) DEFAULT NULL,
  `idCategoriaArbitral` int(11) DEFAULT NULL,
  `email` varchar(30) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `peso`
--

CREATE TABLE `peso` (
  `idPeso` int(11) NOT NULL,
  `descripcion` varchar(11) NOT NULL,
  `sexo` char(1) NOT NULL,
  `idGrupoEdad` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Volcado de datos para la tabla `peso`
--

INSERT INTO `peso` (`idPeso`, `descripcion`, `sexo`, `idGrupoEdad`) VALUES
(1, '-60', 'M', 6),
(2, '-66', 'M', 6),
(3, '-73', 'M', 6),
(4, '-81', 'M', 6),
(5, '-90', 'M', 6),
(6, '-100', 'M', 6),
(7, '+100', 'M', 6),
(8, '-48', 'F', 6),
(9, '-52', 'F', 6),
(10, '-57', 'F', 6),
(11, '-63', 'F', 6),
(12, '-70', 'F', 6),
(13, '-78', 'F', 6),
(14, '+78', 'F', 6),
(15, '-55', 'M', 5),
(16, '-60', 'M', 5),
(17, '-66', 'M', 5),
(18, '-73', 'M', 5),
(19, '-81', 'M', 5),
```

```
(20, '−90', 'M', 5),
(21, '−100', 'M', 5),
(22, '−44', 'F', 5),
(23, '−48', 'F', 5),
(24, '−52', 'F', 5),
(25, '−57', 'F', 5),
(26, '−63', 'F', 5),
(27, '−70', 'F', 5),
(28, '−78', 'F', 5),
(29, '+78', 'F', 5),
(30, '−46', 'M', 4),
(31, '−50', 'M', 4),
(32, '−55', 'M', 4),
(33, '−60', 'M', 4),
(34, '−66', 'M', 4),
(35, '−73', 'M', 4),
(36, '−81', 'M', 4),
(37, '−90', 'M', 4),
(38, '+90', 'M', 4),
(39, '−40', 'F', 4),
(40, '−44', 'F', 4),
(41, '−48', 'F', 4),
(42, '−52', 'F', 4),
(43, '−57', 'F', 4),
(44, '−63', 'F', 4),
(45, '−70', 'F', 4),
(46, '+70', 'F', 4),
(47, '−34', 'M', 3),
(48, '−38', 'M', 3),
(49, '−42', 'M', 3),
(50, '−46', 'M', 3),
(51, '−50', 'M', 3),
(52, '−55', 'M', 3),
(53, '−60', 'M', 3),
(54, '−66', 'M', 3),
(55, '−73', 'M', 3),
(56, '+73', 'M', 3),
(57, '−36', 'F', 3),
(58, '−40', 'F', 3),
(59, '−44', 'F', 3),
(60, '−48', 'F', 3),
(61, '−52', 'F', 3),
(62, '−57', 'F', 3),
(63, '−63', 'F', 3),
(64, '−70', 'F', 3),
(65, '+70', 'F', 3),
(66, '−30', 'M', 2),
(67, '−34', 'M', 2),
(68, '−38', 'M', 2),
(69, '−42', 'M', 2),
(70, '−46', 'M', 2),
(71, '−50', 'M', 2),
(72, '−55', 'M', 2),
(73, '−60', 'M', 2),
(74, '−66', 'M', 2),
(75, '+66', 'M', 2),
(76, '−32', 'F', 2),
(77, '−36', 'F', 2),
(78, '−40', 'F', 2),
(79, '−44', 'F', 2),
(80, '−48', 'F', 2),
(81, '−52', 'F', 2),
(82, '−57', 'F', 2),
(83, '−63', 'F', 2),
(84, '+63', 'F', 2),
(85, '−32', 'M', 1),
```

```
(86, '+100', 'M', 5);

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `puntuacion`
--

CREATE TABLE `puntuacion` (
  `idCombate` int(11) NOT NULL,
  `dniCompetidor` char(30) NOT NULL,
  `ippon` smallint(6) DEFAULT NULL,
  `waza_ari` smallint(6) DEFAULT NULL,
  `shido` smallint(6) DEFAULT NULL,
  `hansokumake` smallint(6) DEFAULT NULL,
  `color_kimono` varchar(1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `tatami`
--

CREATE TABLE `tatami` (
  `idTatami` int(11) NOT NULL,
  `numero` int(11) NOT NULL,
  `idCompeticion` int(11) NOT NULL,
  `emailCronometrador` varchar(30) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `tipocompeticion`
--

CREATE TABLE `tipocompeticion` (
  `idTipoCompeticion` int(11) NOT NULL,
  `descripcion` varchar(30) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Volcado de datos para la tabla `tipocompeticion`
--

INSERT INTO `tipocompeticion` (`idTipoCompeticion`, `descripcion`) VALUES
(1, 'Provincial'),
(2, 'Regional'),
(3, 'Nacional'),
(4, 'Sector');

--
-- Indices para tablas volcadas
--

--
-- Indices de la tabla `arbitroatatami`
--
ALTER TABLE `arbitroatatami`
  ADD PRIMARY KEY (`idTatami`,`dniArbitro`),
  ADD KEY `dniArbitro` (`dniArbitro`);

--
-- Indices de la tabla `categoriaarbitral`
--
```

```
ALTER TABLE `categoriaarbitral`
  ADD PRIMARY KEY (`idCategoriaArbitral`);

--
-- Indices de la tabla `combate`
--
ALTER TABLE `combate`
  ADD PRIMARY KEY (`idCombate`),
  ADD KEY `idTatami` (`idTatami`);

--
-- Indices de la tabla `competicion`
--
ALTER TABLE `competicion`
  ADD PRIMARY KEY (`idCompeticion`),
  ADD KEY `emailDelegado` (`emailDelegado`),
  ADD KEY `grupoEdad` (`idGrupoEdad`),
  ADD KEY `idTipoCompeticion` (`idTipoCompeticion`);

--
-- Indices de la tabla `cronometrador`
--
ALTER TABLE `cronometrador`
  ADD PRIMARY KEY (`email`);

--
-- Indices de la tabla `delegado`
--
ALTER TABLE `delegado`
  ADD PRIMARY KEY (`email`);

--
-- Indices de la tabla `equipo`
--
ALTER TABLE `equipo`
  ADD PRIMARY KEY (`nombre`);

--
-- Indices de la tabla `grupoedad`
--
ALTER TABLE `grupoedad`
  ADD PRIMARY KEY (`idGrupoEdad`);

--
-- Indices de la tabla `inscripcion`
--
ALTER TABLE `inscripcion`
  ADD PRIMARY KEY (`dniPersona`,`idCompeticion`),
  ADD KEY `nombreEquipo` (`nombreEquipo`),
  ADD KEY `idCompeticion` (`idCompeticion`);

--
-- Indices de la tabla `participacion`
--
ALTER TABLE `participacion`
  ADD PRIMARY KEY (`idPeso`,`dniCompetidor`,`fecha_participacion`),
  ADD KEY `dniCompetidor` (`dniCompetidor`);

--
-- Indices de la tabla `persona`
--
ALTER TABLE `persona`
  ADD PRIMARY KEY (`dni`),
  ADD KEY `idCategoriaArbitral` (`idCategoriaArbitral`);

--
```

```
-- Indices de la tabla `peso`
--
ALTER TABLE `peso`
  ADD PRIMARY KEY (`idPeso`),
  ADD KEY `idGrupoEdad` (`idGrupoEdad`);

--
-- Indices de la tabla `puntuacion`
--
ALTER TABLE `puntuacion`
  ADD PRIMARY KEY (`idCombate`,`dniCompetidor`),
  ADD KEY `dniCompetidor` (`dniCompetidor`);

--
-- Indices de la tabla `tatami`
--
ALTER TABLE `tatami`
  ADD PRIMARY KEY (`idTatami`),
  ADD KEY `idCompeticion` (`idCompeticion`),
  ADD KEY `emailCronometrador` (`emailCronometrador`);

--
-- Indices de la tabla `tipocompeticion`
--
ALTER TABLE `tipocompeticion`
  ADD PRIMARY KEY (`idTipoCompeticion`);

--
-- AUTO_INCREMENT de las tablas volcadas
--


--
-- AUTO_INCREMENT de la tabla `categoriaarbitral`
--
ALTER TABLE `categoriaarbitral`
  MODIFY `idCategoriaArbitral` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=8;
--
-- AUTO_INCREMENT de la tabla `combate`
--
ALTER TABLE `combate`
  MODIFY `idCombate` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=7;
--
-- AUTO_INCREMENT de la tabla `competicion`
--
ALTER TABLE `competicion`
  MODIFY `idCompeticion` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;
--
-- AUTO_INCREMENT de la tabla `grupoedad`
--
ALTER TABLE `grupoedad`
  MODIFY `idGrupoEdad` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=7;
--
-- AUTO_INCREMENT de la tabla `peso`
--
ALTER TABLE `peso`
  MODIFY `idPeso` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=87;
--
-- AUTO_INCREMENT de la tabla `tatami`
--
ALTER TABLE `tatami`
  MODIFY `idTatami` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=39;
--
-- AUTO_INCREMENT de la tabla `tipocompeticion`
--
ALTER TABLE `tipocompeticion`
  MODIFY `idTipoCompeticion` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;
```

```
509 --
510 -- Restricciones para tablas volcadas
511 --
512
513 --
514 -- Filtros para la tabla `arbitroatatami`
515 --
516 ALTER TABLE `arbitroatatami`
517   ADD CONSTRAINT `arbitroatatami_ibfk_1` FOREIGN KEY (`idTatami`) REFERENCES `tatami`
      (`idTatami`) ON DELETE CASCADE,
518   ADD CONSTRAINT `arbitroatatami_ibfk_2` FOREIGN KEY (`dniArbitro`) REFERENCES `persona`
      (`dni`) ON DELETE CASCADE;
519
520 --
521 -- Filtros para la tabla `combate`
522 --
523 ALTER TABLE `combate`
524   ADD CONSTRAINT `combate_ibfk_1` FOREIGN KEY (`idTatami`) REFERENCES `tatami` (`
      idTatami`) ON DELETE CASCADE;
525
526 --
527 -- Filtros para la tabla `competicion`
528 --
529 ALTER TABLE `competicion`
530   ADD CONSTRAINT `competicion_ibfk_1` FOREIGN KEY (`emailDelegado`) REFERENCES `
      delegado` (`email`),
531   ADD CONSTRAINT `competicion_ibfk_2` FOREIGN KEY (`idGrupoEdad`) REFERENCES `grupoedad
      ` (`idGrupoEdad`),
532   ADD CONSTRAINT `competicion_ibfk_3` FOREIGN KEY (`idTipoCompeticion`) REFERENCES `
      tipocompeticion` (`idTipoCompeticion`);
533
534 --
535 -- Filtros para la tabla `inscripcion`
536 --
537 ALTER TABLE `inscripcion`
538   ADD CONSTRAINT `inscripcion_ibfk_1` FOREIGN KEY (`dniPersona`) REFERENCES `persona`
      (`dni`) ON DELETE CASCADE ON UPDATE NO ACTION,
539   ADD CONSTRAINT `inscripcion_ibfk_2` FOREIGN KEY (`nombreEquipo`) REFERENCES `equipo`
      (`nombre`) ON DELETE SET NULL ON UPDATE NO ACTION,
540   ADD CONSTRAINT `inscripcion_ibfk_3` FOREIGN KEY (`idCompeticion`) REFERENCES `
      competicion` (`idCompeticion`) ON DELETE CASCADE ON UPDATE NO ACTION;
541
542 --
543 -- Filtros para la tabla `participacion`
544 --
545 ALTER TABLE `participacion`
546   ADD CONSTRAINT `participacion_ibfk_1` FOREIGN KEY (`idPeso`) REFERENCES `peso` (`
      idPeso`) ON DELETE CASCADE ON UPDATE NO ACTION,
547   ADD CONSTRAINT `participacion_ibfk_2` FOREIGN KEY (`dniCompetidor`) REFERENCES `
      persona` (`dni`) ON DELETE CASCADE ON UPDATE NO ACTION;
548
549 --
550 -- Filtros para la tabla `persona`
551 --
552 ALTER TABLE `persona`
553   ADD CONSTRAINT `persona_ibfk_1` FOREIGN KEY (`idCategoriaArbitral`) REFERENCES `
      categoriaarbitral` (`idCategoriaArbitral`) ON DELETE SET NULL ON UPDATE NO ACTION;
554
555 --
556 -- Filtros para la tabla `peso`
557 --
558 ALTER TABLE `peso`
559   ADD CONSTRAINT `peso_ibfk_1` FOREIGN KEY (`idGrupoEdad`) REFERENCES `grupoedad` (`
      idGrupoEdad`) ON DELETE CASCADE ON UPDATE NO ACTION;
560
561 --
```

```
562 -- Filtros para la tabla `puntuacion`
563 --
564 ALTER TABLE `puntuacion`
565   ADD CONSTRAINT `puntuacion_ibfk_1` FOREIGN KEY (`idCombate`) REFERENCES `combate` (`
      idCombate`) ON DELETE CASCADE ON UPDATE CASCADE,
566   ADD CONSTRAINT `puntuacion_ibfk_2` FOREIGN KEY (`dniCompetidor`) REFERENCES `persona`
      (`dni`) ON DELETE CASCADE ON UPDATE NO ACTION;
567
568 /*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
569 /*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
570 /*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

Apéndice E

Contenido del CD

El CD-ROM contine en su interior los siguientes ficheros y documentos :

- **Manual de usuario**. Contiene información de ambos roles, delegado y cronometrador.
- **Manual de instalación**. Contiene información de cómo instalar la aplicación.
- **Memoria**. Versión en PDF de la memoria del TFG.
- **Código fuente:** Contiene todos los ficheros necesarios para el correcto funcionamiento de la aplicación.
- **Script de la BBDD**. Necesario para que funcione la aplicación.

Apéndice F

Formulario de la encuesta

Las conclusiones del estudio van a ser publicados en un Trabajo fin de Grado (u otro tipo de publicación). La publicación de los resultados siempre van a ser anónimos, de modo que no pueda llevarse a cabo una clara asociación entre tu identidad y los resultados.

Datos personales del encuestado

- **Nombre:**
- **Apellidos:**
- **¿Sabe como se gestiona una competición de judo?**

Aspectos relacionadas con la pantalla

		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**		**NC**
1.- Lectura de los caracteres de la pantalla	**Dificil**											**Fácil**	
2.- Organización de la información	**Confusa**											**Clara**	
3.- Secuenciación de pantallas	**Confusa**											**Clara**	

Terminología e información del Sistema

		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**		**NC**
1.-Uso de la terminología en todo el Sistema	**Débil**											**Fuerte**	
2.- Terminología usada en cada tarea	**Nunca**											**Siempre**	
3.- Posición de los mensajes en la pantalla	**Débil**											**Fuerte**	
4.- Entradas para la introducción de datos	**Confusa**											**Clara**	
5.- Información de la aplicación sobre el progreso de la / las tareas	**Nunca**											**Siempre**	
6.- Mensajes de error que la aplicación muestra	**Inútil**											**Útil**	

En cuanto al aprendizaje

		0	1	2	3	4	5	6	7	8	9		NC
1.- Aprender las funcionalidades del Sistema	**Difícil**											**Fácil**	
2.- Explorar las funcionalidades por prueba y error	**Difícil**											**Fácil**	
3.- Recordar nombre y uso de comandos	**Difícil**											**Fácil**	
4.- Realizar tareas es sencillo	**Nunca**											**Siempre**	
5.- Materiales de referencia suplementarios	**Confuso**											**Claro**	
6.- Velocidad de la aplicación	**Lenta**											**Rápida**	
7.- Confiabilidad hacia la aplicación	**Poca**											**Mucha**	
8.- Corregir tus errores en la app	**Difícil**											**Fácil**	
9.- Diseñado para todo tipo de usuarios	**Nunca**											**Siempre**	

Lista de los aspectos más **negativos** vistos en la aplicación JudoCup

1.

2.

3.

Lista de los aspectos más **positivos** vistos en la aplicación JudoCup

1.

2.

3.