

Trabajo Fin de Grado
Grado en Ingeniería de las Tecnologías de
Telecomunicación

SportMeet: Aplicación Web para la gestión de
partidas deportivas usando Spring

Autor: Carlos Javier Martín Escabia

Tutor: Teresa Ariza Gómez

Dpto. de Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2018



Trabajo Fin de Grado
Grado en Ingeniería de las Tecnologías de Telecomunicación

SportMeet: Aplicación Web para la gestión de partidas deportivas usando Spring

Autor:

Carlos Javier Martín Escabia

Tutor:

Teresa Ariza Gómez

Profesor titular

Dpto. de Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2018

Trabajo Fin de Grado: SportMeet: Aplicación Web para la gestión de partidas deportivas usando Spring

Autor: Carlos Javier Martín Escabia

Tutor: Teresa Ariza Gómez

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2018

El Secretario del Tribunal

A mi familia

A mis amigos

A mis maestros

Agradecimientos

Hace unos años tomé una de las decisiones más importantes de mi vida. Tras muchos años de estudio llegaba la hora de decidir realmente qué es lo que me apasionaba y a qué quería dedicar el resto de mi vida. Admito que la decisión no fue fácil, pues me cuesta tomar decisiones en casos como este, pero finalmente, gracias a la ayuda incansable de mis padres decidí optar por la ingeniería y las telecomunicaciones.

Los comienzos no fueron fáciles. Tocaba abandonar mi casa, cambiar de aires y acostumbrarme a una nueva ciudad, a un nuevo sistema de evaluación y conocer a nuevas personas. Pero sin duda, todos estos cambios no trajeron más que alegrías y nuevas experiencias que han servido para hacerme crecer tanto personalmente como profesionalmente.

Es por eso que a día de hoy, a solo un paso de acabar una de las mejores etapas de la vida, es inevitable acordarse de todas aquellas personas que han aportado su granito de arena para que esto sea posible.

En primer lugar me gustaría mencionar a mis compañeros de carrera, uno de los mayores tesoros que la etapa universitaria me ha dado, y sin duda, indispensables desde el primer día que pisamos la Escuela. En nuestra memoria quedarán todos aquellos días de biblioteca, los proyectos interminables y los exámenes que más de una vez nos hicieron tener pesadillas, pero que gracias a la colaboración mutua y al trabajo en equipo hemos conseguido superar.

Otro de los apoyos más necesarios durante estos años ha sido el de mis amigos, esos amigos que llevan contigo una vida entera y que consiguen evadirte de toda clase de agobios cuando no ves la salida, sacándote siempre una sonrisa. Gracias.

Por otro lado, agradecer a cada uno de los profesores a los que he tenido la oportunidad de conocer gracias a esta carrera, ya que de ellos depende en gran parte mi interés por la misma y me han ayudado a conseguir los objetivos esperados. Especialmente, gracias a mi tutora Teresa Ariza por brindarme la oportunidad de realizar este proyecto, ofreciéndome en todo momento su ayuda y asesoramiento y estando disponible cada vez que lo he necesitado.

Por último me gustaría agradecer a mi familia, los auténticos artífices de que haya conseguido mi objetivo. Mi auténtico apoyo fundamental durante todos estos años, siempre confiando en mí y llenándome de motivación cuando lo he necesitado. Gracias a mis abuelos por haberme cuidado desde que llegué al mundo y por haber sido mi principal fuente de inspiración. Y especialmente, gracias a mis padres. Ellos han vivido conmigo el día a día de todos los momentos, buenos y malos, que esta etapa ha dejado y son los responsables de que hoy esté a punto de conseguir uno de mis sueños. Gracias por vuestra educación, vuestra dedicación y por haberme dado la oportunidad de formarme y adentrarme en el maravilloso mundo de las telecomunicaciones.

Carlos Javier Martín Escabia

Sevilla, 2018

Resumen

Hoy en día nos encontramos ante un mundo de constante cambio y evolución. Un mundo en el que los avances tecnológicos se suceden unos tras otros a una velocidad que supera las expectativas.

Entre los avances más significativos de las últimas décadas se encuentra Internet, gracias al cual tenemos la posibilidad de estar en contacto con otras personas en cualquier parte del mundo y de disfrutar de una gran variedad de servicios, cambiando el concepto de “comunicación” y haciendo nuestra vida más fácil.

Para que esto sea posible es necesaria la colaboración de distintas herramientas que se complementan entre sí con el objetivo de obtener los mejores resultados, como es el caso de las aplicaciones web.

Gracias a ellas los usuarios pueden encontrar en la red una amplia variedad de servicios que se ajustan a sus necesidades y que pueden utilizar para mejorar su calidad de vida.

Es por ello que surge SportMeet, una aplicación web destinada a crear relaciones entre personas con los mismos intereses fomentando al mismo tiempo la práctica del deporte. Los usuarios tendrán la posibilidad de crear y organizar sus propios eventos deportivos (partidos de fútbol, tenis, running, etc.), o bien buscar eventos a los que unirse, todo de una manera sencilla e intuitiva. Además, podrán intercambiar mensajes entre ellos y escribir sugerencias para mejorar la aplicación, de forma que ésta se adapte totalmente a sus necesidades. Hacer nuevas amistades y, a la vez, practicar nuestro deporte favorito nunca había sido tan fácil.

Abstract

Nowadays we live in a constant changing and evolving world, where technological advances are appearing one after another in a fast way, exceeding our expectations.

Internet is one of the most important advances of the last decades and thanks to it we have the possibility of being in contact with other people of the world and to enjoy a huge variety of services, changing the 'communication' concept and making our life easier.

To make this possible it is necessary the collaboration of different tools which complement each other in order to obtain the best results. One example of this collaboration are web applications.

Thanks to them users can find a wide variety of services on the network which adjust to their needs and which can be used to improve their quality of life.

SportMeet arises with this goal, a web application dedicated to create relations between people who share the same interests fomenting the sport practise at the same time. Users will have the possibility to create and organize their own sport events (football and tennis matches, running, etc.) or look for events to join, all this in an intuitive and easy way. Also, they will be allowed to interchange messages between them and to write suggestions in order to improve the application, adapting it to their needs. Meeting people and practising our favourite sport never had been easier.

Índice

Agradecimientos	9
Resumen	11
Abstract	13
Índice	15
Índice de Figuras	17
1 Introducción	21
1.1. <i>Motivación y objetivos</i>	21
1.2. <i>Presentación del problema</i>	22
1.3. <i>Antecedentes</i>	22
1.3.1. Plataforma Web para la creación de rutinas de entrenamiento para la App RutinaApp	22
1.3.2. Aplicación móvil Meetup	23
1.3.3. Aplicación móvil Wannay	24
1.4. <i>Descripción de la solución</i>	24
1.4.1. Objetivos principales	24
1.4.2. Funcionalidades de la aplicación	25
1.4.3. Arquitectura de la aplicación	26
1.5. <i>Estructura de la memoria</i>	27
2 Tecnologías Utilizadas	29
2.1. <i>Tecnologías en el servidor</i>	29
2.1.1. Spring Framework	29
2.1.2. Spring Security	34
2.1.3. Spring Boot	34
2.1.4. Java Database Connectivity (JDBC)	35
2.1.5. Java Persistence API (JPA)	36
2.1.6. Maven	37
2.1.7. Thymeleaf	38
2.2. <i>Tecnologías en el cliente</i>	39
2.2.1. Patrón de diseño: MVC	39
2.2.2. Bootstrap	40
2.2.3. HTML5	40
2.2.4. CSS3	41
2.2.5. JQuery	41
2.2.6. JavaScript (JSP)	42
2.3. <i>Persistencia de datos</i>	42
2.3.1. Modelo relacional	43
2.3.2. MySQL	43
3 Herramientas Utilizadas	45
3.1. <i>Spring Tool Suite (STS)</i>	45
3.2. <i>Visual Studio Code</i>	46
3.3. <i>MySQLWorkBench</i>	47
3.4. <i>GitHub</i>	48
3.5. <i>Google Chrome</i>	49
3.6. <i>StarUML</i>	50

4	Servidor de SportMeet	51
4.1	<i>Introducción al servidor</i>	51
4.2	<i>Diagramas de casos de uso</i>	51
4.2.1	Componentes y esquema del diagrama de casos de uso	51
4.2.2	Diagramas de casos de uso de la aplicación	53
4.3	<i>Diagrama de clases</i>	65
4.4	<i>Diagramas de secuencia</i>	67
4.4.1	Creación de un usuario	67
4.4.2	Inicio de sesión	68
4.4.3	Creación de una partida	69
4.4.4	Borrado de una partida	69
4.4.5	Búsqueda y unión una partida	70
4.5	<i>Arquitectura de la base de datos</i>	70
4.5.1	Diagrama de entidad-relación de la base de datos (DER)	70
4.5.2	Elementos del diagrama E-R	71
4.6	<i>Arquitectura de los componentes</i>	73
4.6.1	Componentes de la aplicación	73
4.6.2	Recursos de los controladores	77
5	Interfaz de usuario	81
5.1	<i>Introducción a la interfaz de usuario</i>	81
5.2	<i>Registro de usuario</i>	82
5.3	<i>Inicio de sesión en la aplicación</i>	84
5.4	<i>Página inicial de la aplicación</i>	85
5.5	<i>Mi perfil</i>	86
5.6	<i>Crear partida</i>	86
5.7	<i>Buscar partidas</i>	87
5.8	<i>Mis partidas</i>	88
5.9	<i>Mis mensajes</i>	90
5.10	<i>Sugerencias</i>	91
5.11	<i>Funciones del administrador</i>	92
5.12	<i>Cerrar sesión</i>	93
6	Líneas de mejora y Conclusiones	95
6.1	<i>Líneas de mejora</i>	95
6.1.1	Mejoras de funcionalidad de la aplicación Web	95
6.1.2	Adaptación a otros entornos	96
6.1.3	Implementación con la API de Youtube	96
6.1.4	Implementación con mapas	97
6.1.5	Spring Security	97
6.1.6	Autenticación basada en JWT	98
6.1.7	Implementación de AngularJS	99
6.2	<i>Conclusiones</i>	100
	Referencias	101
	Anexo A: Manual de instalación y Despliegue de la aplicación	103
A.1	<i>Instalación de las herramientas necesarias</i>	103
A.1.1	JAVA SE Development Kit 8	103
A.1.2	Spring Tool Suite (STS)	105
A.1.3	Maven	106
A.1.4	MySQLWorkbench	106
A.2	<i>Despliegue de la aplicación</i>	107
A.2.1	Despliegue de la base de datos	107
A.2.2	Puesta en funcionamiento de la aplicación	109

ÍNDICE DE FIGURAS

Figura 1.1: Plataforma Web para la creación de rutinas de entrenamiento para la App RutinaApp	23
Figura 1.2: Aplicación móvil Meetup	23
Figura 1.3: Aplicación móvil Wannay	24
Figura 1.4: Interfaz de usuario de la aplicación Web SportMeet	25
Figura 1.5: Portada de la aplicación web SportMeet	25
Figura 1.6: Arquitectura de la aplicación web SportMeet	27
Figura 2.1: Logo de Spring Framework	29
Figura 2.2: Módulos de Spring	30
Figura 2.3: Inyección de dependencias a través de anotaciones	31
Figura 2.4: Uso del bean inyectado	31
Figura 2.5: Anotación @Service	32
Figura 2.6: Inyección de servicio con anotación @Autowired	32
Figura 2.7: Paquete DAO	33
Figura 2.8: Inyección de objeto DAO	33
Figura 2.9: Uso del objeto DAO	33
Figura 2.10: Extracto IPartidaDao.java	34
Figura 2.11: Logo de Spring Security	34
Figura 2.12: Logo de Spring Boot	35
Figura 2.13: Logo de JDBC	35
Figura 2.14: Logo JPA	36
Figura 2.15: Anotaciones JPA	36
Figura 2.16: Fichero pom.xml	37
Figura 2.17: Logo de Thymeleaf	38
Figura 2.18: Implementación de Thymeleaf en una plantilla	38
Figura 2.19: Ejemplo de uso de Thymeleaf	38
Figura 2.20: Patrón de diseño MVC	39
Figura 2.21: Logo de Bootstrap	40
Figura 2.22: Logo de HTML5	40
Figura 2.23: Logo de CSS3	41
Figura 2.24: Logo de JQuery	41
Figura 2.25: JavaScript en funcionamiento	42
Figura 2.26: Logo de MySQL	43
Figura 3.1: Logo de Spring Tool Suite	45

Figura 3.2: Logo de Visual Studio Code	46
Figura 3.3: Interfaz de usuario de Visual Studio Code	46
Figura 3.4: Logo del plugin Emmet	47
Figura 3.5: Logo de MySQLWorkBench	47
Figura 3.6: Interfaz de usuario de MySQLWorkBench	48
Figura 3.7: Logo de Github	48
Figura 3.8: Interfaz de usuario de Github	49
Figura 3.9: Logo de Google Chrome	49
Figura 3.10: Logo de StarUML	50
Figura 3.11: Interfaz de usuario StarUML	50
Figura 4.1: Diagrama de casos de uso	52
Figura 4.2: Esquema adicional para diagramas de casos de uso	52
Figura 4.3: CU-01: Inicio de sesión	53
Figura 4.4: CU-02: Usuario autenticado (acceso a servicios)	54
Figura 4.5: CU-03: Mi perfil	55
Figura 4.6: Tabla CU-04: Crear partida	56
Figura 4.7: CU-05: Buscar partida	57
Figura 4.8: Tabla CU-06: Unirse a partida	58
Figura 4.9: CU-07: Mis partidas	59
Figura 4.10: CU-08: Mis mensajes	60
Figura 4.11: CU-09: Sugerencias	61
Figura 4.12: CU-10: Gestionar usuarios	62
Figura 4.13: CU-11: Visualizar sugerencias	63
Figura 4.14: CU-12: Cerrar sesión	64
Figura 4.15: Diagrama de clases de la aplicación web	65
Figura 4.16-A: Atributos y métodos de las clases (1)	65
Figura 4.16-B: Atributos y métodos de las clases (2)	66
Figura 4.17: Diagrama de secuencia de creación de usuario	67
Figura 4.18: Diagrama de secuencia del inicio de sesión	68
Figura 4.19: Diagrama de secuencia de creación de una partida	69
Figura 4.20: Diagrama de secuencia de borrado de una partida	69
Figura 4.21: Diagrama de secuencia de la búsqueda y unión a una partida	70
Figura 4.22: Diagrama entidad-relación de la aplicación	71
Figura 4.23: Arquitectura de la aplicación	73
Figura 4.24: Contenido del paquete “com.cjmartin.app”	74
Figura 4.25: Contenido del paquete “com.cjmartin.app.auth.handler”	74
Figura 4.26: Contenido del paquete “com.cjmartin.app.controllers”	74
Figura 4.27: Contenido del paquete “com.cjmartin.app.models.dao”	74

Figura 4.28: Contenido del paquete “com.cjmartin.app.models.entity”	75
Figura 4.29: Contenido del paquete “com.cjmartin.app.models.service”	75
Figura 4.30: Contenido de la carpeta “static”	75
Figura 4.31: Contenido de la carpeta “templates”	76
Figura 4.32: Fichero “application.properties”	76
Figura 5.1: Portada de la aplicación	81
Figura 5.2: Registro de un nuevo usuario en la aplicación	82
Figura 5.3: Aviso de campo sin rellenar en formulario de registro	82
Figura 5.4: Email introducido incorrecto en formulario de registro	83
Figura 5.5: Contraseñas incorrectas en formulario de registro	83
Figura 5.6: Username o email en uso en formulario de registro	83
Figura 5.7: Usuario creado con éxito	83
Figura 5.8: Inicio de sesión en la aplicación	84
Figura 5.9: Nombre de usuario o contraseña incorrectos en inicio de sesión	84
Figura 5.10: Inicio de sesión con éxito en la aplicación	85
Figura 5.11: Página inicial de la aplicación	85
Figura 5.12: Funcionalidad “Mi perfil”	86
Figura 5.13: Funcionalidad “Crear partida”	87
Figura 5.14: Funcionalidad “Buscar partida”	87
Figura 5.15: Unión con éxito a una partida	88
Figura 5.16: Partida completa	88
Figura 5.17: Participante existente en la partida	88
Figura 5.18: Funcionalidad “Mis partidas”	88
Figura 5.19: Borrado de una partida	89
Figura 5.20: Confirmación de borrado de una partida	89
Figura 5.21: Abandono de una partida	89
Figura 5.22: Confirmación de abandono de una partida	89
Figura 5.23: Información de la partida	89
Figura 5.24: Funcionalidad “Mis mensajes”	90
Figura 5.25: Enviar un mensaje	90
Figura 5.26: Confirmación de mensaje enviado	91
Figura 5.27: Borrado de un mensaje	91
Figura 5.28: Confirmación de mensaje eliminado	91
Figura 5.29: Funcionalidad “Sugerencia”	92
Figura 5.30: Confirmación de sugerencia enviada	92
Figura 5.31: Interfaz de usuario del administrador	92
Figura 5.32: Gestión de usuarios (modo administrador)	93
Figura 5.33: Visualización de sugerencias (modo administrador)	93

Figura 5.34: Cierre de sesión de usuario	93
Figura 6.1: Sistemas operativos iOS y Android	96
Figura 6.2: API de Youtube	97
Figura 6.3: Logo de “Here Technologies”	97
Figura 6.4: Logo de Json Web Tokens	98
Figura 6.5: Estructura de un JSON Web Token	98
Figura 6.6: Logo de AngularJS	99
Figura A.1: Descarga de JDK8	103
Figura A.2: Instalación del paquete JDK8	104
Figura A.3: Asistente de instalación JDK8	104
Figura A.4: Comprobación de instalación de JDK8	104
Figura A.5: Descarga de Spring Tool Suite	105
Figura A.6: Instalación de Spring Tool Suite	105
Figura A.7: Ejecución de Spring Tool Suite	105
Figura A.8: Comprobación de versión de Maven	106
Figura A.9: Descarga de MySQLWorkBench	106
Figura A.10: Instalación de MySQLWorkBench	106
Figura A.11: Fichero application.properties	107
Figura A.12: Conexión a la base de datos	107
Figura A.13: Importación del fichero SQL para la creación de la base de datos	108
Figura A.14: Creación del esquema “SportMeet” y de sus correspondientes tablas	108
Figura A.15: Estableciendo el espacio de trabajo en Spring Tool Suite	109
Figura A.16: Importar proyecto a Spring Tool Suite – 1	109
Figura A.17: Importar proyecto a Spring Tool Suite – 2	110
Figura A.18: Aplicación en funcionamiento desde Spring Tool Suite	110
Figura A.19: Aplicación en funcionamiento desde navegador web	111

1 INTRODUCCIÓN

Si tu negocio no está en Internet, tu negocio no existe.

- Bill Gates -

La comunicación ha sido y es una pieza fundamental en la evolución de la humanidad a lo largo de la historia, y como tal, su progreso ha sido uno de los principales objetivos en las últimas décadas. Hace varios años la mayoría de los servicios a los que hoy en día tenemos acceso eran impensables. Sin embargo, ahora son posibles gracias a Internet.

Internet ha revolucionado la forma en la que nos comunicamos. Actualmente vivimos en una sociedad en la que todo está conectado y la comunicación entre las personas es cada vez más sencilla y rápida. Además, los usuarios tienen la posibilidad de acceder a la información que requieren desde prácticamente cualquier parte del mundo. Para ello, hacen uso, entre otras herramientas, de las aplicaciones web. Estas aplicaciones pueden ser accesibles desde cualquier dispositivo (teléfonos móviles, ordenadores, tablets, etc.) a través de un navegador web, y nos permiten el acceso a un sinfín de posibilidades y servicios que son compartidos a través de la red.

Pero la comunicación no es el único ámbito que se pretende mejorar. Es importante tener en cuenta que la mayoría de la sociedad convive con un ritmo de vida el cual no siempre es el deseado y que puede llegar a resultar perjudicial para nuestra salud. En estos casos se aconseja tomar ciertas medidas para mejorar nuestra calidad de vida, como puede ser la práctica deportiva. Es aquí donde las aplicaciones web juegan un papel importante, ya que la “gamificación” que conlleva el uso de una aplicación puede ser un gran incentivo para los usuarios.

La aplicación web SportMeet busca hacer uso de las ventajas que proporciona la red para facilitar la práctica deportiva y la comunicación entre las personas. De esta forma, los usuarios podrán organizar sus propios eventos deportivos y a la vez conocer a nuevos compañeros que compartan los mismos intereses que ellos.

1.1. Motivación y objetivos

Durante el grado, han sido muchos los conceptos estudiados y a los que hemos tenido la posibilidad de tener acceso, pero sin duda, considero que uno de los temas más interesantes y novedosos ha sido la introducción al mundo de las aplicaciones web.

El hecho de desarrollar una aplicación Web desde cero conlleva un gran trabajo y estudio previo de lo que queremos conseguir, pero también es una gran oportunidad para poner en práctica todas las tecnologías que hemos estudiado y conocer cómo funcionan otras nuevas que van apareciendo año tras año y aún no conocíamos, como ha sido el caso de los frameworks Spring, Bootstrap, o también, el motor de plantillas Thymeleaf.

Por otro lado, muchas personas entienden el deporte como algo indispensable y necesario en nuestro día a día, ya que nos ayuda a mantenernos en forma, llevar una vida más saludable y como no, mejorar diferentes aspectos de nuestra vida social. Personalmente, muchos de los grandes recuerdos de mi infancia y adolescencia son gracias al deporte, y es por ello que considero que todo el mundo debería tener acceso a la práctica deportiva de una manera rápida y sencilla.

El objetivo del presente proyecto es la creación de una aplicación Web que facilite a los usuarios la práctica de actividades deportivas en cualquier lugar y en cualquier momento. Gracias a SportMeet, los usuarios podrán organizar sus propios partidos (ya sea de fútbol, tenis, baloncesto, etc.) o eventos deportivos (sesiones de yoga, running, etc.) los cuales serán accesibles por el resto de usuarios. De esta forma, cualquier persona tendrá la posibilidad de unirse a los eventos que desee y podrá disfrutar conjuntamente de un interés común, como es el deporte, al mismo tiempo que se crean nuevas amistades.

En conclusión, gracias a este proyecto he conseguido aunar cada una de mis pasiones, como son el deporte, las redes sociales y la tecnología, con el objetivo de aprender a usar nuevas herramientas y facilitar la práctica deportiva al resto de la sociedad.

1.2. Presentación del problema

El principal problema que se plantea resolver gracias a este proyecto es el hecho de que los usuarios puedan realizar cualquier actividad deportiva de su interés de una forma fácil y cómoda, haciendo uso de las tecnologías que hoy en día tenemos a nuestro alcance, como son las aplicaciones Web e Internet.

Muchas personas, debido al actual frenético ritmo de vida que llevan, no disponen del tiempo necesario para organizar un partido o cualquier evento deportivo, ya que esto conlleva tareas como conocer la disponibilidad del resto de participantes o buscar un lugar y una hora acorde a todos.

Por otro lado, tras una búsqueda a través de la red, se ha comprobado que existen aplicaciones cuyo objetivo es la organización de eventos de cualquier tipo. Pero estas aplicaciones no están disponibles en países como España o no están disponibles para todos los dispositivos (al ser por ejemplo aplicaciones móviles Android).

En este contexto surge la necesidad de desarrollar una aplicación que se centre en el ámbito deportivo, que mejore o solucione los anteriores problemas mencionados que presentan las actuales aplicaciones disponibles en el mercado y que sea accesible por todos desde cualquier dispositivo y cualquier lugar.

1.3. Antecedentes

Este proyecto surge principalmente como solución a satisfacer una necesidad personal. En cada grupo de amigos, suele existir una persona que se encarga de organizar todos o la mayoría de los eventos, como es mi caso. A veces, esta tarea puede volverse algo difícil debido a ciertos factores. Por ejemplo, no siempre tenemos el tiempo suficiente para organizar un partido o, en otros casos, no conocemos a suficientes personas como participantes son necesarios.

Aún así, es importante mencionar que actualmente en el mercado existen una serie de aplicaciones con un objetivo parecido, como son las siguientes:

1.3.1 Plataforma Web para la creación de rutinas de entrenamiento para la App RutinaApp

Esta plataforma es un proyecto realizado por el alumno Francisco José Díaz Romero del Grado en Ingeniería de las Tecnologías de la Telecomunicación en la Escuela Técnica Superior de Ingeniería de la Universidad de Sevilla en el año 2017.

El objetivo principal de este proyecto es el desarrollo de una aplicación Web para la gestión y la creación de rutinas de entrenamiento, proporcionando a su vez las funcionalidades necesarias para ello y que serán usadas por una aplicación móvil con el sistema operativo Android. Podemos observar la interfaz de usuario de la aplicación en la Figura 1.1:

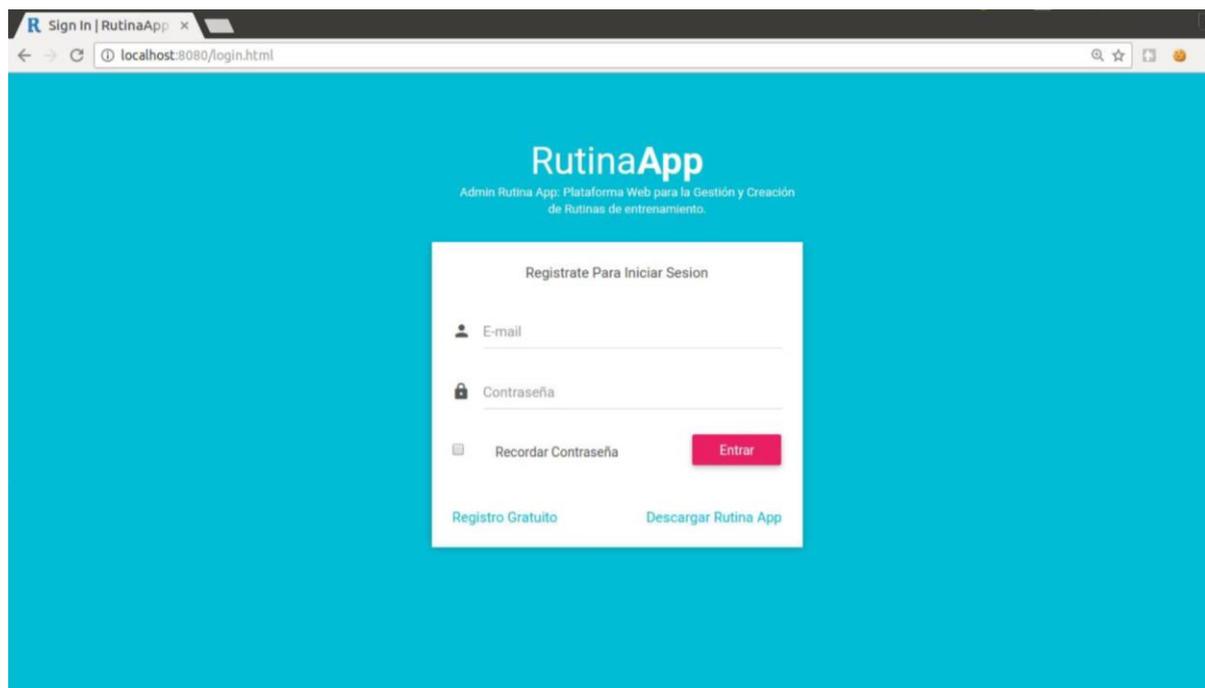


Figura 1.1: Plataforma Web para la creación de rutinas de entrenamiento para la App RutinaApp

El propio usuario es el encargado de crear y elegir las rutinas que desee, basándose en rutinas especializadas o generales, su nivel de forma y otros aspectos descriptivos del ejercicio. A través de la aplicación Web los usuarios podrán descargar sus propias rutinas o las rutinas que el resto de usuarios hayan creado y publicado, pudiendo posteriormente reproducirlas desde su aplicación móvil Rutina App.

Este proyecto ha servido como referencia debido a la similitud en cuanto a tecnologías utilizadas, como es el caso del framework para backend “Spring Framework”.

1.3.2 Aplicación móvil Meetup

Meetup es una aplicación móvil desarrollada por Meetup Inc., cuyo objetivo es poner en contacto a personas de una ciudad con intereses comunes para que hagan juntos “lo que les gusta hacer”. En la Figura 1.2 podemos observar un resumen de las funcionalidades que ofrece:

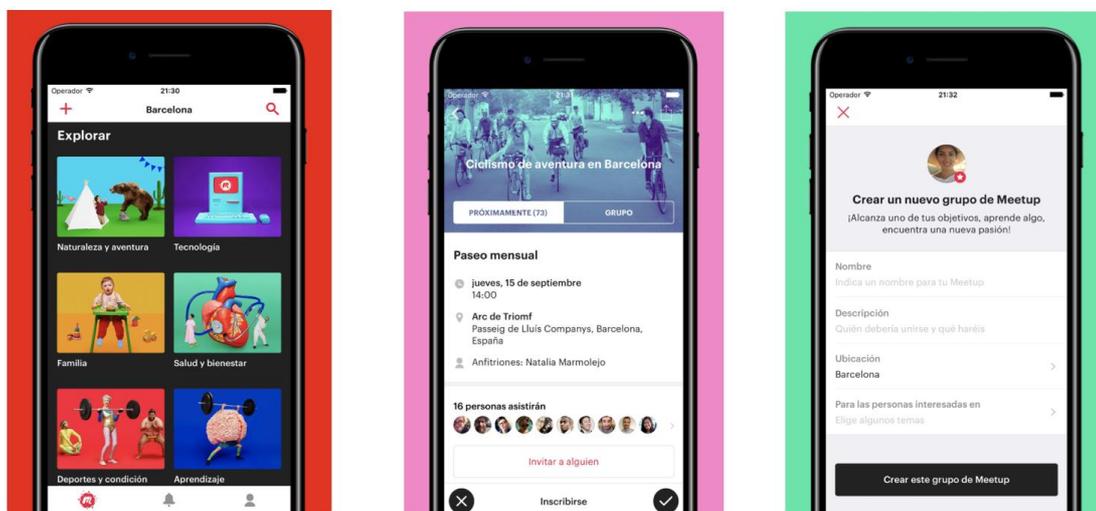


Figura 1.2: Aplicación móvil Meetup

La aplicación ofrece la posibilidad de crear “meetups” de cualquier tipo, ya sean tecnológicos, deportivos o artísticos, para los que es necesaria una suscripción mensual o semestral, dependiendo del lugar donde se encuentre el “Meetup”.

1.3.3 Aplicación móvil Wannay

Wannay es una aplicación móvil cuyo objetivo es crear una aplicación que funcione en todo el mundo, en cualquier idioma y donde los usuarios puedan escribir lo que les gusta hacer y compartir. Ofrece al usuario la posibilidad de contactar con cualquier persona de cualquier lugar del mundo y todo tipo de características definiendo un “nivel de afinidad” y según el cual podrán crear grupos para compartir sus hobbies, como se puede observar en la Figura 1.3:

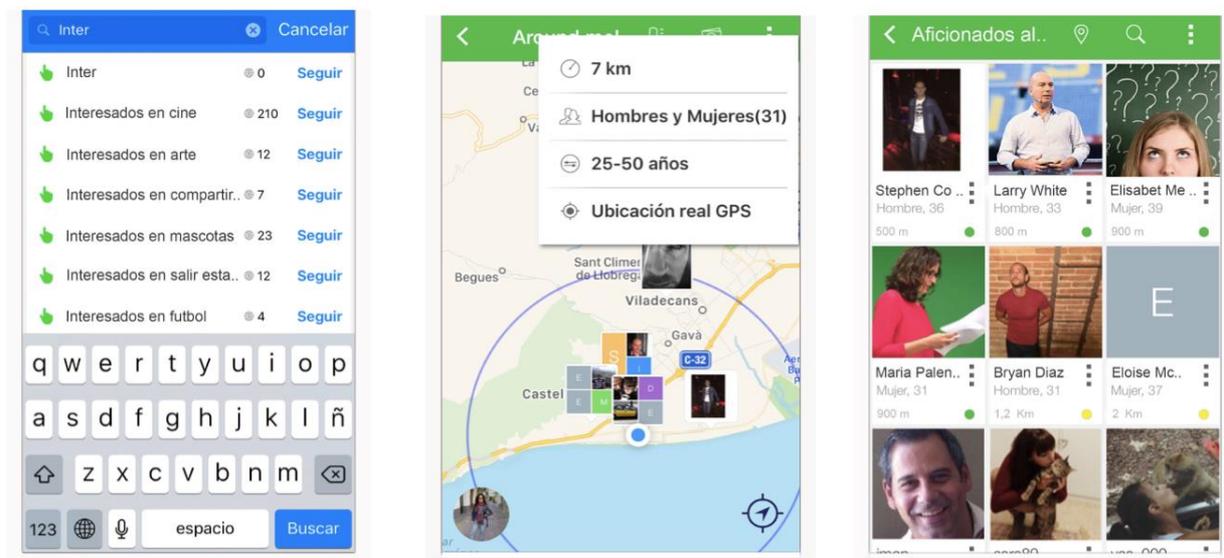


Figura 1.3: Aplicación móvil Wannay

1.4. Descripción de la solución

En esta sección describiremos los principales objetivos que persigue nuestro proyecto así como las funcionalidades a las que el usuario tendrá acceso utilizando la aplicación y la arquitectura que se ha implementado para el funcionamiento de la misma.

1.4.1 Objetivos principales

El objetivo principal de la aplicación consiste en facilitar a los usuarios la tarea de crear y organizar sus propios eventos deportivos, de forma que puedan crear cualquier tipo de partida o evento deportivo según sus preferencias. El resto de usuarios que tengan los mismos intereses y que se puedan adaptar a los requisitos de la partida, podrán unirse a la misma. Esto será posible gracias a la sencilla e intuitiva interfaz Web que ofrece la aplicación, como podemos observar en la Figura 1.4:

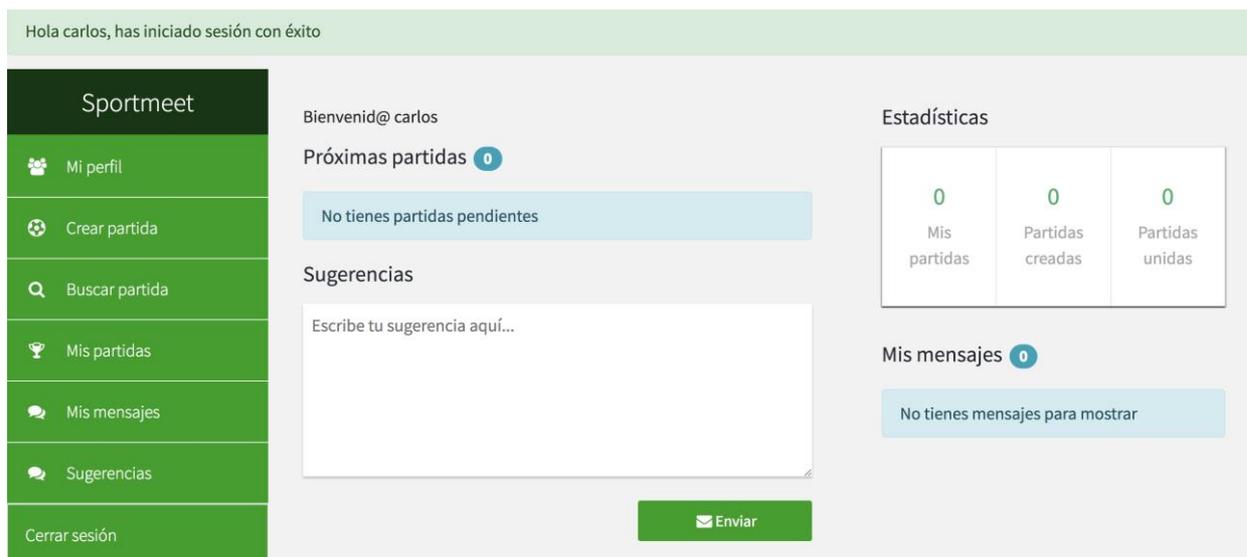


Figura 1.4: Interfaz de usuario de la aplicación Web SportMeet

1.4.2 Funcionalidades de la aplicación

A continuación se describen las funcionalidades a las que podrá acceder el usuario a través del uso de la aplicación Web SportMeet.

Lo primero que el usuario puede observar una vez accede a la aplicación a través de un navegador web será la página de inicio en la que se muestra un resumen de las funcionalidades que aporta la aplicación dividido en distintas secciones, como se observa en la Figura 1.5.

- Nosotros: contiene un “carousel” con distintas imágenes que describen para qué sirve la aplicación.
- Servicios: breve resumen de los servicios que ofrece la aplicación.
- Descubre: imágenes de ejemplo de las actividades que se llevan a cabo.
- Contacto: enlaces a los perfiles de SportMeet en distintas redes sociales

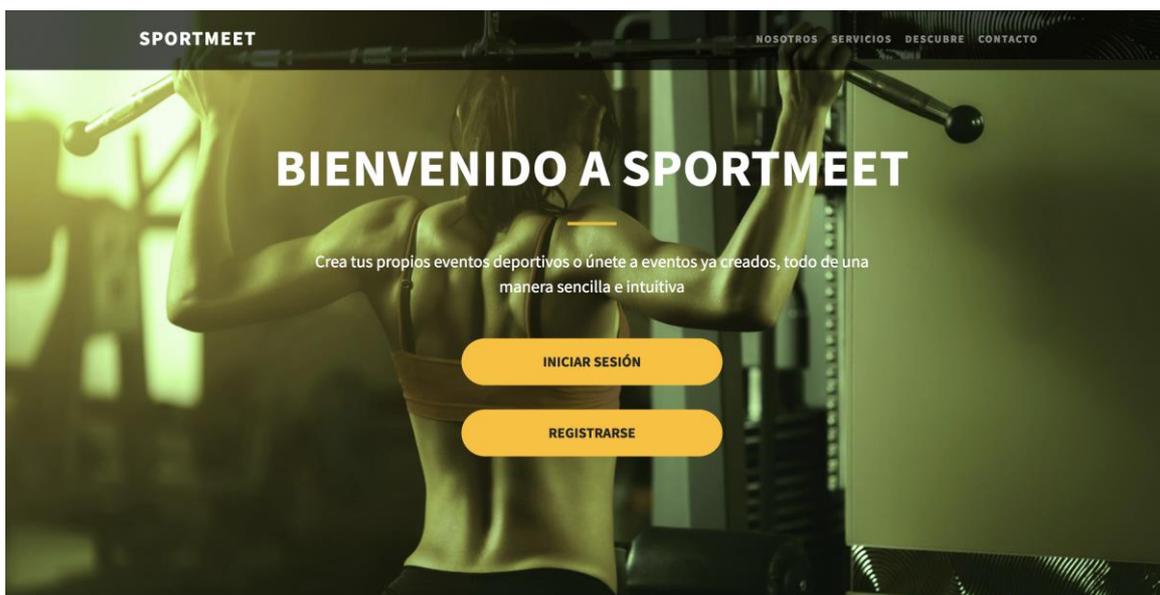


Figura 1.5: Portada de la aplicación web SportMeet

A su vez, tendrá la opción de registrarse en la aplicación, o bien iniciar sesión si ya dispone de una cuenta de usuario. Una vez el usuario disponga de una cuenta y se dirija a la opción de iniciar sesión, existen dos posibilidades:

- Iniciar sesión como un usuario normal (a través de su username y contraseña), que es el caso de cualquier nuevo usuario que se registre en la aplicación.
- Existe un usuario administrador “*admin*”, que tiene acceso a las mismas funciones que el resto de usuarios y además a varias funciones especiales como son la gestión de usuarios, gestión de partidas o la visualización y gestión de sugerencias.

Cuando el usuario inicia sesión en la aplicación, puede acceder a las funcionalidades que se apreciaban anteriormente en la Figura 1.4:

- Mi perfil: permite al usuario gestionar y modificar sus datos personales.
- Crear partida: permite al usuario crear una partida y establecer las características de la misma (hora, lugar, deporte, participantes, etc.).
- Buscar partida: permite al usuario buscar una partida según determinados filtros (deporte, ciudad, provincia).
- Mis partidas: permite al usuario visualizar todas las partidas en las que se encuentra inmerso, a la vez que puede eliminar partidas creadas por él y abandonar u obtener información de partidas a las que se ha unido.
- Mis mensajes: permite al usuario observar los mensajes recibidos por otros usuarios, responder a los mismos de una forma sencilla y rápida y escribir mensajes a cualquier usuario a través de su nombre de usuario.
- Sugerencias: permite al usuario sugerir nuevas mejoras al administrador respecto a la aplicación.
- Cerrar sesión: permite al usuario cerrar su sesión y salir de la aplicación de una forma segura.

Además de estas funcionalidades, desde la página inicial de la aplicación (una vez se ha iniciado sesión) el usuario podrá:

- Observar sus estadísticas: número de partidas totales de las que es participante, número de partidas totales que él mismo ha creado y número de partidas a las que se ha unido.
- Escribir una sugerencia de forma rápida.
- Observar sus próximas partidas.
- Observar y responder a los últimos mensajes recibidos.

1.4.3 Arquitectura de la aplicación

Para el diseño y desarrollo de este proyecto se ha seguido una arquitectura basada en el patrón MVC (modelo-vista-controlador). Este patrón de arquitectura software es el usado en la mayoría de los servicios Web REST ya que separa los componentes de una aplicación dividiéndolos en:

- Datos de la aplicación
- Interfaz de usuario
- Lógica de control

Se trata de un patrón muy contrastado y con una gran solidez y validez en una gran variedad de lenguajes y plataformas de desarrollo, siendo cada vez más utilizado. Esto trae grandes ventajas a nuestro proyecto, ya que al hacer esta división de componentes según su responsabilidad aseguramos que cuando queramos hacer algún cambio en cualquier parte de nuestro código, esto no afecte a otra parte del mismo, respetando así el principio de “responsabilidad única”, el cual defiende que una parte concreta de nuestro código debe tener una sola responsabilidad, sin saber que es lo que hace el resto de la aplicación. Además, la implementación de este patrón mejora considerablemente la reutilización de nuestro código y su adaptabilidad a futuras mejoras.

Dicho esto, a continuación (en la Figura 1.6) mostramos un esquema general que resume la arquitectura de nuestra aplicación y las tecnologías utilizadas:

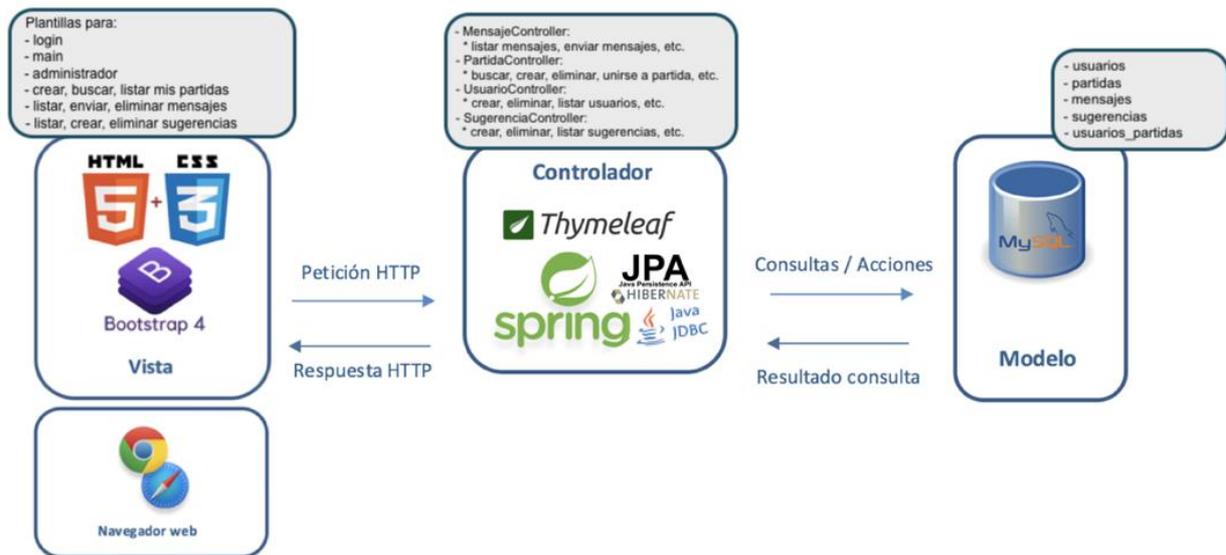


Figura 1.6: Arquitectura de la aplicación web SportMeet

En la figura anterior podemos observar como el usuario interactúa con la aplicación a través de un navegador web (Safari, Chrome, Firefox, etc.), que realiza una petición al servidor mediante el servidor gracias al framework de back-end Java Spring. El controlador se encarga de gestionar la autenticación usando JDBC (Java Database Connectivity) y la consulta o acción con la base de datos haciendo uso de la herramienta JPA (Java Persistence API), mientras el modelo (base de datos MySQL) se encarga de retornar el resultado de la consulta de nuevo al controlador. Finalmente, el resultado de la consulta se reenvía a la vista, que haciendo uso de las tecnologías HTML5 y CSS3 junto al framework de frontend Bootstrap 4, se encarga de mostrar el resultado visual al usuario para que pueda observarlo de una forma más entendible y amigable y seguir interactuando con la aplicación Web.

1.5. Estructura de la memoria

A continuación describiremos brevemente los puntos en los que está dividida la memoria para tener un primer acercamiento al contenido de la misma:

- 1. Introducción:** en esta sección se detallan los conceptos fundamentales y que servirán para tener una idea inicial del objetivo y funcionamiento del proyecto.
- 2. Tecnologías utilizadas:** en este punto se describen cada una de las tecnologías utilizadas y se explica el por qué de su elección.
- 3. Herramientas utilizadas:** se definen las herramientas utilizadas a lo largo del desarrollo y diseño, dejando claros los motivos de su elección.
- 4. Servidor de SportMeet:** se realizará un estudio del proceso de diseño de la aplicación donde se mostrará la estructura del mismo y sus componentes.
- 5. Interfaz de usuario:** se muestra la interfaz de usuario y se explica en qué consistirá la interacción del usuario con la aplicación a través de ejemplos de uso.
- 6. Líneas de mejora y conclusiones:** para concluir, se detallan posibles aspectos a mejorar para futuras versiones de la aplicación y las conclusiones obtenidas durante el desarrollo de la misma.
- 7. Referencias:** muestra un listado con las referencias utilizadas durante la elaboración de la memoria.

Finalmente se añade un anexo que contienen información adicional a tener en cuenta sobre el proyecto:

- **Anexo A:** Manual de instalación y despliegue de la aplicación.

2 TECNOLOGÍAS UTILIZADAS

*La tecnología hizo posible las grandes civilizaciones;
ahora las grandes poblaciones hacen que la tecnología
sea indispensable.*

- José Krutch-

En esta sección se abordarán las tecnologías utilizadas para la realización del proyecto, llevando a cabo una breve descripción de cada una de ellas y distinguiendo las distintas partes de la aplicación en las que afectan.

2.1 Tecnologías en el servidor

A continuación se detallan las tecnologías utilizadas para el desarrollo de la aplicación en el lado servidor.

2.1.1 Spring Framework

Spring es un framework de código abierto que facilita el desarrollo y diseño de aplicaciones a través del lenguaje Java (Figura 2.1).

Su primera versión fue desarrollada por Rod Johnson en octubre de 2002 y lanzada bajo la licencia Apache 2.0 en junio de 2003. Desde ese momento, numerosos desarrolladores han colaborado en la mejora del framework para alcanzar la versión más actual, la 5.0, con una innumerable cantidad de funcionalidades.



Figura 2.1: Logo de Spring Framework

Gracias a Spring los desarrolladores podrán estandarizar su trabajo, además de resolver, agilizar y manejar los problemas y complejidades que vayan apareciendo en su proyecto conforme aumenta la complejidad y la dimensión. Nos permite desarrollar aplicaciones de una manera más rápida, eficaz y corta, ahorrando numerosas líneas de código y tareas repetitivas.

Por otro lado, posee una serie de módulos que proveen al desarrollador de una gran cantidad de funcionalidades, ofreciendo así la posibilidad de crear una aplicación más profesional y completa. En la Figura 2.2 se muestran los principales módulos que contiene Spring:

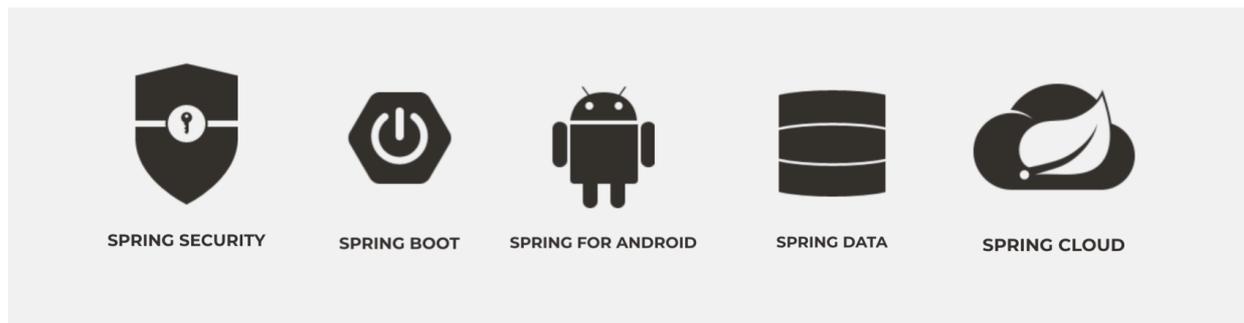


Figura 2.2: Módulos de Spring

Para nuestra aplicación se ha hecho uso de los módulos Spring Security y Spring Boot. Gracias a estos y al resto de módulos el usuario puede disfrutar de una amplia variedad de servicios, como son los siguientes:

- **Contenedor de inversión de control:** permite al usuario configurar los componentes de la aplicación y administrar el ciclo de vida de los objetos Java principalmente a través de la inyección de dependencias.
- **Programación orientada a aspectos:** habilita la implementación de rutinas transversales.
- **Acceso a datos:** se trabaja con RDBMS, usando Java Database Connectivity (JDBC) y herramientas de mapeo.
- **Gestión de transacciones:** unifica distintas APIs de gestión y coordina transacciones para los objetos Java.
- **Modelo vista controlador:** proporciona un framework basado en HTTP y servlets, con herramientas para la personalización y extensión de aplicaciones web y servicios web REST.
- **Framework de acceso remoto:** permite importar y exportar objetos Java a través de redes que soporten RMI, CORBA y protocolos basados en HTTP incluyendo servicios web (SOAP).
- **Autenticación y autorización:** ofrece procesos de seguridad configurables y totalmente adaptables que soportan un rango amplio de protocolos, estándares, prácticas y herramientas.
- **Administración remota:** ofrece configuración de la visibilidad y gestión de objetos Java para la configuración local o remota vía JMX.

2.1.1.1 Inversión de control

La inversión de control (o IoC) es considerada como el concepto central de Spring Framework, una de las razones que lo diferencia de otros frameworks. Este término se refiere a los cambios que se producen en el flujo de ejecución y vida de los objetos respecto a la programación tradicional.

En otras palabras, el principal objetivo es tratar de invertir la forma en la que se controla la aplicación. Por ejemplo, lo que antes dependía del programador, como el orden en el que llamamos a los métodos para darle unos comportamientos u otros dentro de la aplicación, ahora depende de otro ente como es el framework. En este caso, el desarrollador define las respuestas deseadas a sucesos o solicitudes de datos concretas, mientras que otra entidad se encarga de llevar a cabo las acciones de control necesarias. Esto nos ofrece la posibilidad de crear aplicaciones más profesionales, con un comportamiento más encapsulado y automático.

La inversión de control está basada en el concepto de “idea intuitiva”, también conocido como “principio de Hollywood”, el cual era característico en los principales directores de Hollywood al recibir llamadas de sus aspirantes y cuya base es que los directores no necesitan ser llamados, sino que son ellos los que se encargarán de llamar.

Por último, es importante destacar que la inversión de control está directamente relacionada con el “Contenedor de inversión de control” o “IoC Container”, el cual detallaremos a continuación.

2.1.1.2 Inyección de dependencias

La inyección de dependencias en Spring es un tipo concreto de inversión de control en la que el manejo de las propiedades de un objeto son inyectadas a través de un constructor, un servicio, un setter, etc. De esta manera conseguimos crear un control que se adapte al comportamiento más conveniente para nuestra aplicación.

Es un patrón que nos permite conseguir un código más desacoplado, facilitándonos la tarea de hacer cualquier tipo de cambios en la aplicación o bien de testarla. Por otro lado, nos permite cambiar las dependencias en tiempo de ejecución basándonos en los factores que nos interesen para cada situación concreta. Para llevar a cabo esta tarea es necesario un contenedor o inyector que inyecte las dependencias necesarias en el momento que lo necesitemos.

Durante el proyecto hemos llevado a cabo principalmente la inyección de dependencias mediante los métodos Set (o setters). Podríamos indicar a Spring de qué forma queremos que sean inyectadas las dependencias a través de XML o anotaciones. En nuestro caso hemos hecho uso de las anotaciones debido a su sencillez. En la Figura 2.3 podemos ver un ejemplo de su implementación:

```
public class MensajeController {

    protected final Log logger = LoggerFactory.getLog(this.getClass());

    @Autowired
    private IMessageService mensajeService;
```

Figura 2.3: Inyección de dependencias a través de anotaciones

Vemos como a través de la anotación `@Autowired` le estamos indicando a Spring que se debe encargarse de buscar un Bean que cumpla los requisitos especificados para ser inyectado (en este caso que sea del tipo “`IMensajeService`”). A partir de ahora podrá ser usado en el código con todas sus características y métodos, como podemos observar en la Figura 2.4 donde se lleva a cabo el envío de un mensaje y su almacenado posterior en la base de datos:

```
// Método que guarda el mensaje en la BBDD
@RequestMapping(value = "/mis_mensajes", method = RequestMethod.POST)
public String enviar_mensaje(@Valid Mensaje mensaje, BindingResult result, Model model,
    RedirectAttributes flash, SessionStatus status) {

    if (result.hasErrors()) { // si el resultado de la validación tiene error...
        model.addAttribute("titulo", "Sportmeet · Dashboard");
        return "main"; // ...volvemos al registro
    }

    mensajeService.save(mensaje);
    status.setComplete(); // eliminamos el objeto de la sesión y acabamos el proceso
    flash.addFlashAttribute("success", "Mensaje enviado con éxito");
    return "redirect:main";
}
```

Figura 2.4: Uso del bean inyectado

2.1.1.3 Contenedor de inversión de control

El contenedor de inversión de control de Spring supone un elemento fundamental para el correcto funcionamiento de la inyección de dependencias.

Si seguimos el patrón de inyección de dependencias veremos como los componentes se encargarán de declarar y definir sus dependencias, pero no de conseguirlas. Es aquí donde juega un papel clave el contenedor de inversión de control o “IoC Container”, el cual se encargará de conseguir e inyectar las dependencias correspondientes a cada uno de los objetos, sin la necesidad de una definición previa en Java o en el lenguaje de programación orientada a objetos que estemos utilizando.

El contenedor llevará a cabo la creación de los objetos, la relación de ellos entre sí, su configuración y el manejo de su ciclo de vida desde que se crean hasta que se destruyen.

2.1.1.4 Spring Beans

La definición del término “bean” reside en un objeto que es configurado e instanciado en el contenedor de Spring. Los beans, una vez creados, se almacenarán y permanecerán en el contenedor durante todo el tiempo de vida de la aplicación hasta que ésta se detenga o nosotros los destruyamos.

El correcto almacenamiento de los beans en el contenedor es fundamental para poder realizar la inyección de dependencias, ya que esto nos permite poder inyectarlos en otros beans, tener acceso a ellos en cualquier momento y desde cualquier parte de la aplicación y llevar a cabo su reutilización.

En nuestro proyecto, marcamos las clases que queremos que sean instanciadas en el contenedor de Spring a través de la anotación `@Service`, ya que estas clases son especiales porque ofrecen un servicio concreto. Por ejemplo, en la Figura 2.5 podemos ver como hacemos uso de la anotación `@Service` para la clase “`UsuarioServiceImpl`”, la cual ofrece una serie de servicios para gestionar los usuarios y las partidas relacionadas con los mismos:

```
@Service
public class UsuarioServiceImpl implements IUserarioService{

    @Autowired
    private IUserarioDao usuarioDao;

    @Autowired
    private IPartidaDao partidaDao;

    @Override
    @Transactional(readOnly=true)
    public List<Usuario> findAll() {
        return (List<Usuario>) usuarioDao.findAll();
    }

    @Override
    public void save(Usuario usuario) {
        usuarioDao.save(usuario);
    }
}
```

Figura 2.5: Anotación `@Service`

Para inyectar este servicio en cualquier otro bean o controlador de Spring basta con usar la anotación `@Autowired` como se puede observar en la Figura 2.6:

```
@Controller
@RequestMapping("/partida")
@SessionAttributes("partida")
public class PartidaController {

    @Autowired
    private IUserarioService usuarioService;
}
```

Figura 2.6: Inyección de servicio con anotación `@Autowired`

2.1.1.5 Patrón de diseño DAO

El patrón de diseño DAO (Data Access Object) no es un patrón que pertenezca en propiedad o haya sido inventado por Spring, sino que pertenece al catálogo de Core J2EE Patterns y se utiliza en la capa de acceso a datos de cualquier aplicación que lo implemente.

Cuando diseñamos una aplicación es recomendable que los componentes de la misma sean lo más transparentes posible al sistema de persistencia o base de datos. De esta forma, las migraciones entre distintos tipos de almacenamiento, distintos tipos de fabricantes y diferentes fuentes de datos serán mucho más sencillas.

El patrón DAO nos ayuda a solventar este problema, ya que consiste en un objeto de acceso a datos el cual se encarga de abstraer y encapsular el acceso a los datos. Un objeto DAO se encarga de manejar el establecimiento de la conexión con la fuente de los datos para acceder a ellos y guardarlos como sea debido.

Por otro lado, gracias a los objetos DAO no son necesarias las transacciones, ya que siempre realizan operaciones atómicas contra la base de datos, como por ejemplo la creación, modificación o borrado de un registro, obtener todos los registros, búsquedas por una clave concreta, etc. Es por ello que suele crearse un DAO por cada objeto que tengamos en nuestra aplicación.

En nuestro caso hemos definido un DAO para cada uno de los objetos que usamos en la aplicación web: mensaje, usuario, partida y sugerencia, como podemos ver en la Figura 2.7:



Figura 2.7: Paquete DAO

Finalmente, usar el objeto DAO y acceder a la base de datos desde los controladores, tendremos que inyectarlo gracias a la anotación `@Autowired` como se puede observar en la Figura 2.8:

```

@Controller
@RequestMapping("/partida")
@SessionAttributes("partida")
public class PartidaController {

    @Autowired
    private IPartidaDao partidaDao;
  }
  
```

Figura 2.8: Inyección de objeto DAO

Ahora podremos utilizar el objeto y realizar todas las operaciones necesarias gracias a los métodos que hemos definido en la clase DAO correspondiente. En el extracto de código que se muestra en la Figura 2.9 utilizamos el objeto "partidaDao" para hacer una búsqueda de partidas según determinados parámetros:

```

List<Partida> partidas;

if (partida.getDeporte() != "" && partida.getCiudad() == "" && partida.getProvincia() == "") {
    partidas = partidaDao.findByDeporte(partida.getDeporte());
} else if (partida.getDeporte() == "" && partida.getCiudad() != "" && partida.getProvincia() == "") {
    partidas = partidaDao.findByCiudad(partida.getCiudad());
}
  
```

Figura 2.9: Uso del objeto DAO

Los métodos “findByDeporte” o “findByCiudad” han sido previamente definidos en la interfaz correspondiente al objeto DAO, como podemos observar en la Figura 2.10 donde se muestra un extracto del código de la interfaz “IPartidaDao.java”:

```
public interface IPartidaDao extends CrudRepository<Partida, Long>{

    @Query("select p from Partida p where p.deporte=?1")
    public List<Partida> findByDeporte(String deporte);

    @Query("select p from Partida p where p.ciudad=?1")
    public List<Partida> findByCiudad(String ciudad);
}
```

Figura 2.10: Extracto IPartidaDao.java

2.1.2 Spring Security

Spring Security es un framework de control de acceso bastante potente y totalmente personalizable para cumplir con los requisitos de cualquier usuario. Es el estándar utilizado para aportar seguridad a todas las aplicaciones web basadas en Spring (Figura 2.11).



Figura 2.11: Logo de Spring Security

Este framework se centra sobre todo en proveer los procesos de autenticación y de autorización en aplicaciones Java. Como todos los proyectos de Spring, su principal poder reside en la facilidad de su implementación para obtener resultados importantes.

Entre sus características principales encontramos la integración opcional con Spring Web MVC, la protección que aporta contra ataques como falsificación de solicitudes, “clickjacking” o fijación de sesión.

En el proyecto, Spring Security ha sido utilizado para implementar la autorización y la autenticación de los usuarios a la hora de iniciar y mantener su sesión.

2.1.3 Spring Boot

Spring Boot es un módulo de Spring que nos permite desarrollar y poner en funcionamiento todo tipo de aplicaciones y servicios creados a partir de Spring de una forma fácil y rápida (Figura 2.12).

Gracias a Spring Boot simplificaremos enormemente tareas convencionales como son el despliegue del servidor o la creación de independencias y nos basaremos en la ejecución de la aplicación.

Los principales motivos por los que hacemos uso de este módulo en nuestra aplicación son algunas de las características importantes que aporta, las cuales se describen a continuación:

- **Convención sobre configuración:** ahorra al usuario tareas de configuración y validación, aportando las configuraciones necesarias para distintas situaciones.
- **Gestión de dependencias:** el usuario solo tendrá que indicar las dependencias que necesita sin necesidad de especificar su versión, ya que Spring Boot se encarga de hacer un análisis de las dependencias de proyectos externos para proveerlas correctamente.
- **Auto-configuración:** realiza la configuración de manera autónoma ahorrando esta tediosa tarea al usuario.
- **Servidor embebido:** este módulo ofrece soporte para Tomcat, Jetty, Undertow embebidos.
- **Actuadores:** provee acceso a una serie de endpoints que nos ofrecen información interesante sobre la infraestructura interna del proyecto, como los /beans, /metrics (para visualizar métricas), /health (para conocer el estado de la aplicación), etc.



Figura 2.12: Logo de Spring Boot

2.1.4 Java Database Connectivity (JDBC)

Java Database Connectivity (o JDBC) es una interfaz de programación de aplicaciones del lenguaje de programación Java que nos permite realizar todo tipo de operaciones sobre bases de datos y que utiliza el dialecto SQL (Figura 2.13). Es totalmente independiente del tipo de sistema operativo desde el que se ejecuta o de la base de datos a la que se pretende acceder. Un programa que utilice JDBC puede conectarse con cualquier sistema de gestión de base de datos (DBMS) con el único requisito de que exista un controlador para dicho DBMS.



Figura 2.13: Logo de JDBC

La API en sí consiste en un conjunto de interfaces y clases escritas en Java, que son utilizadas por desarrolladores para la conexión con bases de datos, enviar consultas escritas en el lenguaje SQL y su posterior procesado.

En el proyecto, JDBC es usado para realizar la conexión y autenticación con la base de datos y permitir el posterior acceso a los datos y consultas a la misma.

2.1.5 Java Persistence API (JPA)

Java Persistence API (o JPA) proporciona al usuario un modelo de persistencia basado en POJO's (Plain Old Java Object) el cual es utilizado para el acceso a bases de datos relacionales en Java. Fue desarrollado por el grupo de expertos de EJB 3.0 como parte de JSR 220, y suele utilizarse en aplicaciones web y aplicaciones cliente desde plataformas como Java EE o Java SE (Figura 2.14).



Figura 2.14: Logo JPA

Con este modelo, la relación entre las tablas de la base de datos en las entidades Java se realiza mediante anotaciones en las propias clases de entidad, ahorrando así el uso de ficheros XML. Por otro lado, también podemos definir transacciones como anotaciones JPA. En la Figura 2.15 podemos observar un extracto del código en el que se implementan anotaciones JPA para crear las tablas de la base de datos a partir de la clase entidad:

```
@Entity
@Table(name = "usuarios")
public class Usuario implements Serializable {

    private static final long serialVersionUID = 1L;

    // Variables

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(length = 30, unique = true)
    private String username;

    @Column(length = 60)
    private String password;

    @Column(length = 60)
    private String repPassword;
```

Figura 2.15: Anotaciones JPA

En el proyecto se ha hecho uso de la tecnología JPA para la creación de las tablas de la base de datos y las consultas y acceso a datos a través de la misma.

2.1.6 Maven

Maven es una herramienta *open-source* desarrollada en 2001 y cuyo objetivo es conseguir la simplificación de los procesos de compilación y generación de ejecutables del código fuente. Ofrece una gran integración con Spring Tool Suite.

Anteriormente, era necesario realizar de manera manual todo el proceso del “build” de la aplicación, debiendo hacer un complicado estudio de la aplicación cada vez que era necesario compilar y generar ejecutables. Gracias a Maven, solo es necesario ejecutar un comando (concretamente “mvn install”) para realizar la build de cualquier proyecto, ahorrando pérdidas de tiempo respecto a sus dependencias, módulos, librerías, etc. Además, es una herramienta capaz de gestionar proyectos software completos, desde la comprobación del código hasta el despliegue de la aplicación.

Durante el build del software se distinguen distintas etapas las cuales se enumeran a continuación y que podrán realizarse ejecutando el comando “mvn” seguida de la palabra que vemos entre paréntesis en cada proceso:

- Validación (validate)
- Compilación (compile)
- Test (test)
- Empaquetado (package)
- Pruebas de integración (integration-test)
- Verificación (verify)
- Instalación del código (install)
- Despliegue (deploy)

Estas distintas etapas se realizan en cadena, es decir, si ejecutamos “mvn test”, automáticamente Maven ejecuta la etapa de validación y compilación, y posteriormente la de test.

Por otro lado, Maven nos ayuda a simplificar la gestión de dependencias entre los distintos módulos y librerías. Para ello lo único necesario es indicar los distintos módulos y librerías que componen el proyecto en el fichero de configuración de Maven, llamado “pom.xml”. En la Figura 2.16 podemos ver un extracto del fichero “pom.xml” de nuestro proyecto:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5     <modelVersion>4.0.0</modelVersion>
6
7     <groupId>com.cjmartin.sportmeet.app</groupId>
8     <artifactId>sportmeet</artifactId>
9     <version>0.0.1-SNAPSHOT</version>
10    <packaging>jar</packaging>
11
12    <name>sportmeet</name>
13    <description>Demo project for Spring Boot</description>
14
15    <parent>
16        <groupId>org.springframework.boot</groupId>
17        <artifactId>spring-boot-starter-parent</artifactId>
18        <version>2.0.2.RELEASE</version>
19        <relativePath /> <!-- lookup parent from repository -->
20    </parent>
21
22    <properties>
23        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
24        <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
25        <java.version>1.8</java.version>
26    </properties>
27
28    <dependencies>
29        <dependency>
30            <groupId>com.google.apis</groupId>
31            <artifactId>google-api-services-youtube</artifactId>
32            <version>v3-rev198-1.23.0</version>
33        </dependency>
```

Figura 2.16: Fichero pom.xml

2.1.7 Thymeleaf

Thymeleaf es una librería del lenguaje de programación Java que implementa un motor de plantillas de XML/XHTML/HTML5 y que puede ser utilizado para entornos web y otros no web. Proporciona un módulo para la integración con Spring MVC, lo cual permite reemplazar a los archivos JavaScript en nuestra aplicación (Figura 2.17).



Figura 2.17: Logo de Thymeleaf

Su objetivo principal es la creación y diseño de plantillas, mejorando su estética y elegancia a la vez que presentamos un código bien estructurado. Además, proporciona flexibilidad y una correcta adecuación a estándares como HTML5.

A continuación se enumeran varias de sus principales características, motivos por los que ha sido escogido para su integración en este proyecto:

- Principalmente está diseñado para XML, XHTML o HTML5, pero es extensible a otros formatos.
- Es multiformato, es decir, puede ser usado en distintos ámbitos, no solo en el web.
- Presenta soporte para la internacionalización de su contenido.
- Se integra perfectamente con el framework Spring.
- Implementa un sistema de caché el cual ofrece un alto rendimiento a la aplicación.
- Posee soporte para la evaluación de expresiones.

Se trata de una herramienta fácil de utilizar e implementar, como podemos ver en la Figura 2.18 donde se muestra un extracto de código de una plantilla HTML5 en la que se integra Thymeleaf:

```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3 <head>
```

Figura 2.18: Implementación de Thymeleaf en una plantilla

Simplemente con esta sentencia ya podremos comenzar a utilizar las funcionalidades que Thymeleaf ofrece. Por ejemplo, en la Figura 2.19 se puede observar un ejemplo de uso de nuestro código en el que accedemos a los valores de un objeto para mostrarlos a través del fichero HTML5 con la sentencia “th:text”:

```
<ul class="list-group">
  <li class="list-group-item bg-success text-white" th:text="'Usuario: ' + ${usuario.username}></li>
  <li class="list-group-item" th:text="${usuario.nombre} + ' ' + ${usuario.apellido}></li>
  <li class="list-group-item" th:text="${usuario.edad}></li>
  <li class="list-group-item" th:text="${usuario.email}></li>
</ul>
```

Figura 2.19: Ejemplo de uso de Thymeleaf

2.2 Tecnologías en el cliente

Una vez descritas las tecnologías usadas en el servidor procederemos a comentar las tecnologías usadas en el cliente de nuestra aplicación web.

2.2.1 Patrón de diseño: MVC

Antes de empezar a nombrar las tecnologías, es importante mencionar el patrón de diseño que se ha seguido a la hora de desarrollar la interfaz de usuario de la aplicación, el patrón modelo-vista-controlador o MVC.

Este patrón se caracteriza por la división que realiza de la aplicación en tres partes bien diferenciadas y separadas en distintos objetos: un modelo de datos, información de control e información de presentación. Es decir, separa los datos y la lógica de negocio de una aplicación de su representación y el módulo que se encarga de la gestión de los eventos y las comunicaciones. Se definen así tres componentes principales (modelo, vista y controlador), los cuales pueden observarse en la Figura 2.20 y cuyos objetivos se describen a continuación de la misma:



Figura 2.20: Patrón de diseño MVC

- **Modelo:** contiene los datos de la aplicación, su lógica de negocio y sus mecanismos de persistencia, implementando los privilegios de acceso que previamente han sido descritos en la aplicación. Recibe las consultas de acceso y manipulación de los datos a través del controlador y envía a la vista la información que se le solicita en cada momento.
- **Vista:** su objetivo principal es mostrar al usuario los datos obtenidos a través del modelo de forma que pueda interactuar con la aplicación, a través de un registro de su controlador asociado. Sabe cómo acceder a los datos pero no entiende su significado ni cómo manipularlos.
- **Controlador:** actúa como intermediario entre los otros dos componentes (entre el modelo y la vista). El controlador escucha los sucesos producidos por la vista y reacciona ante cada uno de ellos de la manera más apropiada (normalmente realiza peticiones al modelo invocando alguno de sus métodos).

Este modelo puede implementarse de varias formas distintas, pero el flujo de funcionamiento más común es el siguiente:

- 1) El usuario interactúa con la interfaz de usuario y realiza acciones.
- 2) El controlador recibe notificaciones de las peticiones que el usuario ha realizado a través de la interfaz de usuario y gestiona los eventos.
- 3) El controlador accede al modelo y realiza la acción solicitada por el usuario (normalmente suele actualizarlo o modificarlo).
- 4) La vista obtiene los datos del modelo y genera la interfaz apropiada para el usuario de modo que puedan observarse los cambios realizados en el modelo.
- 5) La interfaz de usuario espera nuevas interacciones con el usuario para volver a comenzar el ciclo.

2.2.2 Bootstrap

Bootstrap es un framework de código abierto desarrollado en sus inicios por Twitter cuyo objetivo principal es facilitar el diseño web al desarrollador a partir de librerías CSS. Permite crear webs con un diseño “responsive”, adaptable a cualquier tipo de dispositivo y tamaño de pantalla (Figura 2.21)



Figura 2.21: Logo de Bootstrap

Este framework ofrece al usuario numerosas ventajas, las cuales han sido decisivas al tomar la decisión de implementarlo en nuestro proyecto. A continuación se describen las principales características del mismo:

- Permite la utilización de una gran variedad de elementos web (desde iconos e imágenes a desplegables, combinando herramientas como HTML5, CSS y JavaScript).
- Implementa el “grid system” o maquetado por columnas, el cual facilita la implementación y ordenación de los elementos.
- Posee una perfecta integración con las librerías de JavaScript
- Ofrece implementaciones externas con Wordpress, Drupal, etc.
- Su curva de aprendizaje permite un manejo sencillo y asequible, más rápido aún si se tiene experiencia.
- Compatible con la mayoría de navegadores web disponibles en el mercado.

2.2.3 HTML5

HTML5 (o HyperText Markup Language, versión 5) es un estándar que representa el lenguaje básico de la World Wide Web y que se encuentra actualmente en su quinta versión (Figura 2.22). Es un lenguaje de “markup” usado para estructurar y representar contenido web de forma correcta a los usuarios, considerándose un aspecto fundamental en el funcionamiento de los sitios web.



Figura 2.22: Logo de HTML5

Esta última versión presenta novedades como son la aparición de nuevas etiquetas (article, header, footer, nav, section, etc.), un sistema más novedoso de inclusión de multimedia y una dependencia más reducida respecto a los plug-ins.

2.2.4 CSS3

CSS3 (Hojas de Estilo en Cascada) es un lenguaje de diseño gráfico cuyo objetivo principal es definir el estilo y la apariencia de páginas web escritas en HTML (o documentos XML). También permite aplicar estilos no visuales, como las hojas de estilo auditivas (Figura 2.23).



Figura 2.23: Logo de CSS3

Este lenguaje nos permite separar el contenido de la aplicación web de su apariencia, de forma que los desarrolladores puedan controlar de manera más precisa la estética de sus páginas definiéndola en un fichero externo. Por lo tanto, si se desea cambiar la apariencia solo es necesario modificar este fichero externo, aportando flexibilidad y una mejor adaptación sin necesidad de modificar los ficheros JSP o HTML.

En esta última versión implementa novedosas características como son las esquinas redondeadas, las sombras, los gradientes, las transiciones o animaciones y nuevos layouts como las multicolumnas, las cajas flexibles o las maquetas de diseño en cuadrícula.

2.2.5 JQuery

JQuery es una biblioteca multiplataforma *open source* de Javascript, que fue desarrollada principalmente por John Resig y presentada en el BarCamp NYC en 2006. Es la biblioteca Javascript más utilizada y puede ser utilizada en cualquier sistema operativo y en múltiples navegadores web (Figura 2.24).



Figura 2.24: Logo de JQuery

Al ser *open source* cuenta con un soporte de gran calidad, constante y rápido, recibiendo así actualizaciones de manera constante. Además cuenta con un framework especial disponible para dispositivos móviles llamado “jQuery Mobile”.

jQuery es una herramienta muy potente que cuenta con un amplio historial de uso en aplicaciones web, la cual nos permite llevar a cabo los siguientes objetivos:

- Manipular el árbol DOM.
- Simplificar la interacción con los documentos HTML.
- Manejo de eventos.
- Desarrollo de animaciones.
- Agregar interacción con la técnica AJAX a páginas web.
- Simplificar el código Javascript de la aplicación.

En cuanto a su uso, consiste en un único fichero JavaScript el cual contiene las funcionalidades comunes del DOM, de eventos, efectos y AJAX. Esta biblioteca nos permite cambiar y actualizar el contenido de la página web sin recargarla manipulando el árbol DOM y realizando peticiones AJAX, consiguiendo así un efecto más dinámico e interactivo. En nuestra aplicación es usado por el framework Bootstrap para aplicar estilos y conseguir una apariencia más profesional de la aplicación web.

2.2.6 JavaScript (JSP)

JavaScript es un lenguaje de programación que nos permite crear y realizar actividades complejas en una aplicación web. Gracias a esta tecnología, una página web puede pasar de un contenido totalmente estático a una interfaz mucho más dinámica, intuitiva y fácil de usar.

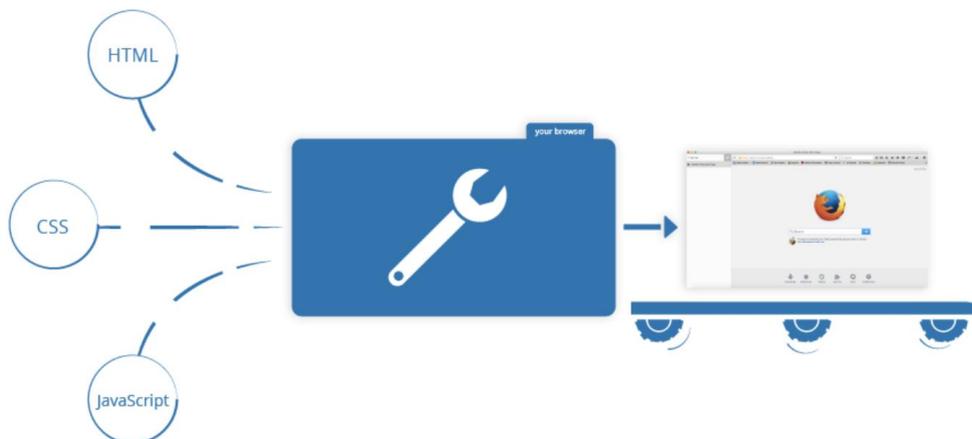


Figura 2.25: JavaScript en funcionamiento

Como hemos mencionado, nos permite realizar numerosas e interesantes funciones como crear contenido dinámico, controlar archivos multimedia o crear imágenes animadas a partir de varias líneas de código.

El lenguaje JavaScript es ejecutado por el motor del navegador de JavaScript, independientemente del código HTML y CSS y una vez estos han sido cargados, como se observa en la Figura 2.25. Esto nos asegura que el estilo y la estructura de la página web se conservarán intactos y funcionando correctamente. Por otro lado, es frecuente el uso del DOM para realizar actualizaciones de la interfaz de usuario complementándolo con JavaScript.

En nuestra aplicación se ha hecho uso del lenguaje JavaScript para el correcto funcionamiento de Bootstrap y para incluir varios efectos que aportan dinamismo a la interfaz de usuario.

2.3 Persistencia de datos

Finalmente definiremos las tecnologías que han sido utilizadas para implementar la persistencia de datos en nuestro proyecto. Primero hablaremos sobre el modelo de base de datos utilizado para la aplicación y posteriormente sobre el sistema de gestión.

2.3.1 Modelo relacional

Para implementar la persistencia de datos en nuestra aplicación se ha hecho uso del modelo relacional, el cual lleva a cabo el modelado y la gestión de base de datos basándose en la lógica de predicados y en la teoría de conjuntos.

Este modelo se basa fundamentalmente en relaciones y la base de datos es considerada como una colección de las mismas. Estas relaciones pueden considerarse como conjuntos de datos (en los que el orden de inserción no tiene relevancia) que reciben el nombre de tuplas. Cada relación se entiende como una tabla la cual está conformada por una serie de registros (filas o tuplas) y columnas (también conocidas como campos).

El modelo relacional ofrece numerosas ventajas frente a otros modelos, entre las que destacan las que enumeramos a continuación:

- Cuenta con herramientas que evitan la duplicidad de registros.
- Garantiza la integridad referencial (es decir, si eliminamos un registro se eliminan el resto de registros asociados al mismo).
- Es un modelo comprensible y aplicable.

En cuanto a la manipulación de la información, utilizamos un lenguaje relacional para su acceso (actualmente se cuenta con el “álgebra relacional”, la cual describe la forma en la que se desea realizar la consulta, y el “cálculo relacional”, el cual indica lo que se desea devolver).

Finalmente, para entender mejor el concepto de modelo relacional, definiremos los conceptos de “esquema” e “instancia”:

- **Esquema:** especifica la estructura de la relación y la información que almacenará (contiene los metadatos de la relación).
- **Instancia:** representa el contenido de una tabla en un momento concreto.

2.3.2 MySQL

MySQL es el sistema de gestión de bases de datos más popular que existe actualmente en el mercado para bases de datos relacionales. Es patrocinado por una empresa privada la cual posee el copyright de la mayor parte del código. Actualmente pertenece a Oracle Corporation y está desarrollada principalmente en C++ y ANSI C (Figura 2.26).



Figura 2.26: Logo de MySQL

Es un sistema gestor de base de datos multihilo y multiusuario, lo cual permite ser utilizada por diferentes usuarios a la vez. Además, es comúnmente utilizado en el desarrollo web debido a que permite a los desarrolladores y diseñadores hacer cambios en sus aplicaciones modificando un solo fichero, simplificando enormemente esta tarea.

Por otro lado, aporta seguridad en el acceso a datos y ofrece la posibilidad de desarrollar programas o aplicaciones que requieran acceso a plataformas de bases de datos rápidas.

MySQL es utilizado en nuestro proyecto para realizar la gestión de todos los objetos existentes en la aplicación web (gestión de los usuarios, las partidas, los mensajes a intercambiar entre los distintos usuarios, las sugerencias creadas por los mismos, etc.) a través de la herramienta “MySQLWorkBench”.

3 HERRAMIENTAS UTILIZADAS

Siempre llega una nueva herramienta.

La tecnología es neutral, depende de cómo se use.

- Rick Smolan -

En esta sección se enumeran y describen cada una de las herramientas que se han utilizado para la elaboración de la aplicación web. Este paso es sin duda uno de los más importantes ya que el hecho de escoger las herramientas que mejor se adapten a nuestras intenciones puede repercutir en la posterior dificultad y en el tiempo de desarrollo del proyecto.

3.1 Spring Tool Suite (STS)

Spring Tool Suite es un entorno de desarrollo basado en Eclipse que se personaliza para desarrollar aplicaciones Spring. Proporciona al usuario un entorno listo para usar que permite realizar una gran variedad de funciones, como ejecutar, depurar, implementar y desplegar aplicaciones Spring. Además, incluye integraciones para Pivotal tc Server, Pivotal Cloud Foundry, Git, Maven, AspectJ, y viene encima de las últimas versiones de Eclipse (Figura 3.1).

Su stack tecnológico abarca cada una de las diferentes capas de una aplicación Java desarrollada prácticamente desde cualquier arquitectura de una manera efectiva, incluyendo las nuevas tendencias en microservicios.

Por otro lado, STS incluye la edición de Pivotal tc Server para desarrolladores, la cual consta de una versión de Apache Tomcat optimizada para Spring. Además, desde su consola se puede observar una visión en tiempo real de los parámetros de rendimiento de las aplicaciones en ejecución la cual ayuda a los desarrolladores en el proceso de depuración cuando existen errores.



Figura 3.1: Logo de Spring Tool Suite

Esta herramienta es completamente open source y está licenciada bajo los términos de la Licencia Pública de Eclipse. Soporta el despliegue de aplicaciones en servidores virtuales, locales o en la nube. Algunas de las características principales de este framework son las siguientes:

- **Correcto entendimiento de la aplicación:** STS analiza los archivos de configuración y muestra información detallada sobre los beans que están siendo definidos, sus dependencias, espacios de nombres utilizados, además de extraer una visión general para ciertos estereotipos.
- **Soporte Refactoring:** además de ofrecer un soporte avanzado para el refactoring, el IDE agrega nuevas refactorizaciones para los elementos de Spring, como el cambio de nombre de los Beans.
- **Asistencia de código:** proporciona corrección de errores para todo tipo de ficheros que estemos escribiendo, ya sean XML, ficheros de configuración, etc.
- **Soporte AOP:** proporciona el soporte para AOP más completo de la actualidad debido a su integración con las herramientas de lenguaje AspectJ para Eclipse.
- **Integración con Cloud Foundry y Pivotal tc Server:** permite el despliegue de aplicaciones directamente en Cloud Foundry o desde una instancia tc Server.
- **Validaciones comprensivas para la configuración:** ofrece un amplio conjunto de validaciones que facilitan la tarea de búsqueda de errores, ya que se aplican de forma automática e indican errores en la configuración desde el IDE mucho antes de ejecutar la aplicación.

Esta herramienta ha sido utilizada para programar el código correspondiente al backend y desarrollo de ciertas partes del cliente.

3.2 Visual Studio Code

Visual Studio Code es un editor de código fuente totalmente personalizable desarrollado por Microsoft. La aplicación ofrece versiones para distintos sistemas operativos como Windows, Linux y macOS. Es una herramienta *open source* que incluye soporte para depuración, control integrado de Git, finalización inteligente de código, resaltado de sintaxis, fragmentos y refactorización de código (Figura 3.2).

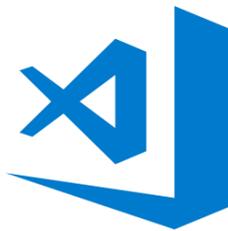


Figura 3.2: Logo de Visual Studio Code

Una de las ventajas que ofrece es que es compatible con distintos lenguajes de programación, por lo que puede ser usado para programar aplicaciones web, diseño de base de datos, etc. En nuestro caso, hemos utilizado esta herramienta para la programación del frontend (ficheros HTML y sus correspondientes estilos CSS) (Figura 3.3).

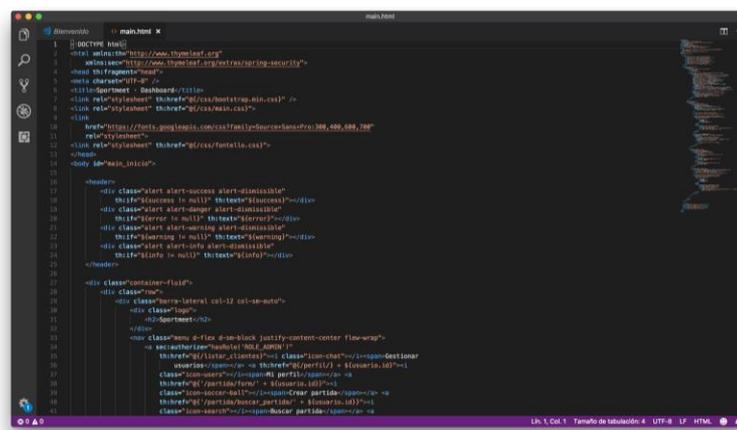


Figura 3.3: Interfaz de usuario de Visual Studio Code

Por otro lado se ha hecho uso de un plugin para editores de texto llamado “Emmet”, ya que se especializa en el diseño web cuyo flujo de trabajo depende básicamente de ficheros HTML, CSS o XML.

Emmet sigue la filosofía de la programación basada en fragmentos, de forma que nos permite escribir código CSS como expresiones que pueden ser analizadas de forma dinámica, produciendo una salida que depende de lo que se escriba en la miniatura. Sin duda es una herramienta bastante útil y que agiliza bastante el proceso de desarrollo del frontend (Figura 3.4).



Figura 3.4: Logo del plugin Emmet

3.3 MySQLWorkBench

MySQLWorkBench es una herramienta visual de diseño de bases de datos que integra el desarrollo software, gestión, mantenimiento, diseño y administración para sistemas de base de datos MySQL (Figura 3.5).



Figura 3.5: Logo de MySQLWorkBench

Es conocido como el sucesor de DBDesigner 4 de fabFORCE.net y reemplaza al anterior conjunto de software, MySQL GUI Tools Bundle. Las principales características de la aplicación se describen a continuación:

- **Exportación de datos:** permite exportar las bases de datos en distintos formatos (PDF, PNG, etc.)
- **Importación de datos:** por otro lado también ofrece la posibilidad de importar datos.
- **Creación de scripts:** permite crear scripts SQL a partir del modelo generado.
- **Ejecución de sentencias:** nos permite ejecutar sentencias desde la misma aplicación para observar el resultado de nuestras consultas.
- **Estilos de notación:** se pueden usar distintos estilos de notación.
- **Calidad visual:** una de las características más importantes ya que la interfaz es sencilla e intuitiva.
- **Flexibilidad:** permite modelar vistas, tablas, rutinas, etc.

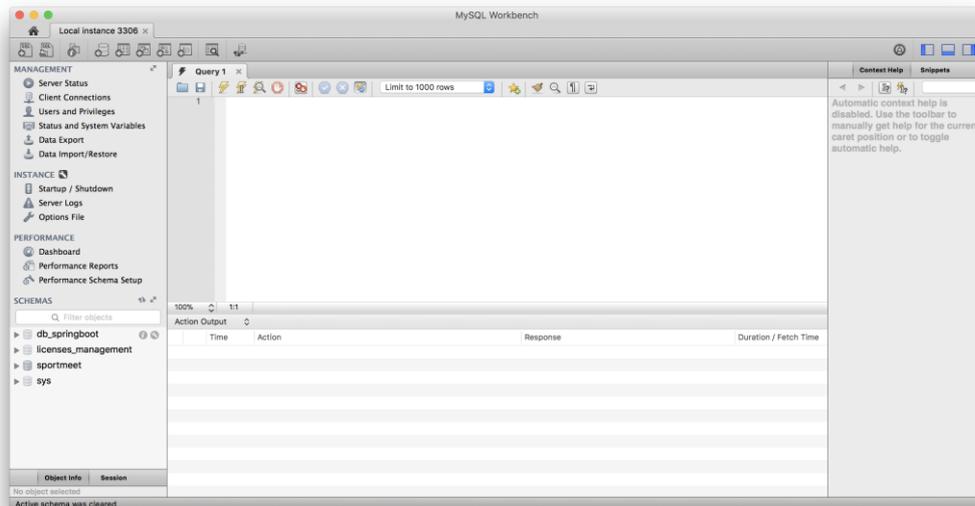


Figura 3.6: Interfaz de usuario de MySQLWorkBench

Esta herramienta ha sido la elegida para las tareas de gestión y administración de la base de datos debido a las numerosas ventajas que ofrece y a su sencillo uso (Figura 3.6).

3.4 GitHub

GitHub es conocida por los desarrolladores por ser una de las plataformas de desarrollo colaborativo más grandes y extendidas del mundo. Se utiliza para alojar proyectos utilizando el sistema de control de versiones Git (Figura 3.7).



Figura 3.7: Logo de Github

Por otro lado, Github destaca por una serie de características que la hacen única e inmejorable para el trabajo en equipo, las cuales se enumeran a continuación:

- **Sistema de seguimiento de problemas:** permite a cualquier miembro de un equipo hacer sugerencias o comentarios acerca de las distintas versiones que pueda abarcar el proyecto, al mismo tiempo que se detecten fallos.
- **Herramienta de revisión de código:** permite añadir anotaciones en cualquier punto de un fichero y debatir sobre determinados cambios llevados a cabo en un “commit” específico.
- **Visor de ramas:** en él se pueden visualizar los distintos caminos o ramas que toma nuestro proyecto, permitiendo así la vuelta a versiones anteriores en caso de fallo o comportamientos inesperados de nuestro código.
- **Wiki de referencia:** opera para el mantenimiento de las distintas versiones de las páginas.

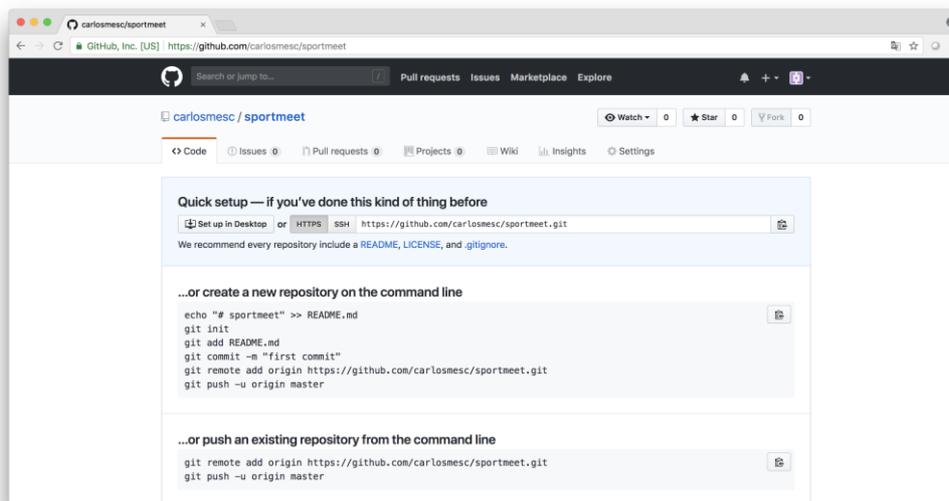


Figura 3.8: Interfaz de usuario de Github

Durante el desarrollo del proyecto se ha utilizado Github para almacenar la aplicación (como un sistema de almacenamiento alternativo y copia de seguridad) y para realizar un control de versiones más sencillo y comprensible (Figura 3.8).

3.5 Google Chrome

Google Chrome es uno de los navegadores web más populares de la actualidad con más de 750 millones de usuarios. Se trata de un navegador de código cerrado desarrollado por Google, aunque deriva de proyectos de código abierto. Su nombre procede del término inglés usado para describir la interfaz gráfica de usuario (Figura 3.9).



Figura 3.9: Logo de Google Chrome

Posee una gran cantidad de características que lo diferencian del resto de navegadores, como lo son las expuestas a continuación:

- **Acceso directo a diferentes aplicaciones:** basta con hacer click derecho en la pestaña de la página web y seleccionar la opción “Crear acceso directo”. De esta forma tendremos acceso directo desde nuestro escritorio a una página concreta.
- **Control de fallos:** como cada pestaña funciona de forma independiente, si una falla no se bloquea todo el navegador. Bastará con cerrar la pestaña defectuosa.
- **Modo incógnito:** para ello solo hace falta activar la opción “Nueva ventana de incógnito”. En ese instante se abrirá una nueva ventana en modo incógnito y al cerrarla se borrarán todas las cookies de las páginas que hayamos visitado en ese modo.

- **Navegación segura:** Google Chrome es capaz de detectar sitios peligrosos o webs sospechosas que puedan alojar software malintencionado. Para ello lanzará avisos para que estemos informados.
- **Traducción de páginas web:** gracias a Google Translate, el navegador ofrece la posibilidad de hacer una traducción automática de páginas web en otros idiomas a nuestro idioma nativo.

Google Chrome ha sido el navegador web elegido durante la producción del proyecto para la comprobación del funcionamiento del cliente debido a su sencillez y herramientas para analizar todo lo relacionado con la programación web.

3.6 StarUML

StarUML es una herramienta UML de licencia gratuita desarrollada en 1996 y modificada posteriormente en 2005 y que es utilizada para el modelado de software, basándose en estándares UML y DMA. Es una aplicación sencilla de utilizar, con una interfaz intuitiva y atractiva. Permite al usuario una rápida percepción de sus objetos, funciones y características (Figura 3.10).



Figura 3.10: Logo de StarUML

Gracias a StarUML se pueden desarrollar distintos tipos de diagramas para entender mejor nuestro código, entre los que se encuentran: diagramas de casos de uso, diagramas de clase, diagramas de secuencia, diagramas de colaboración, diagramas de estados, diagramas de actividad, diagramas de componentes, etc (Figura 3.11).

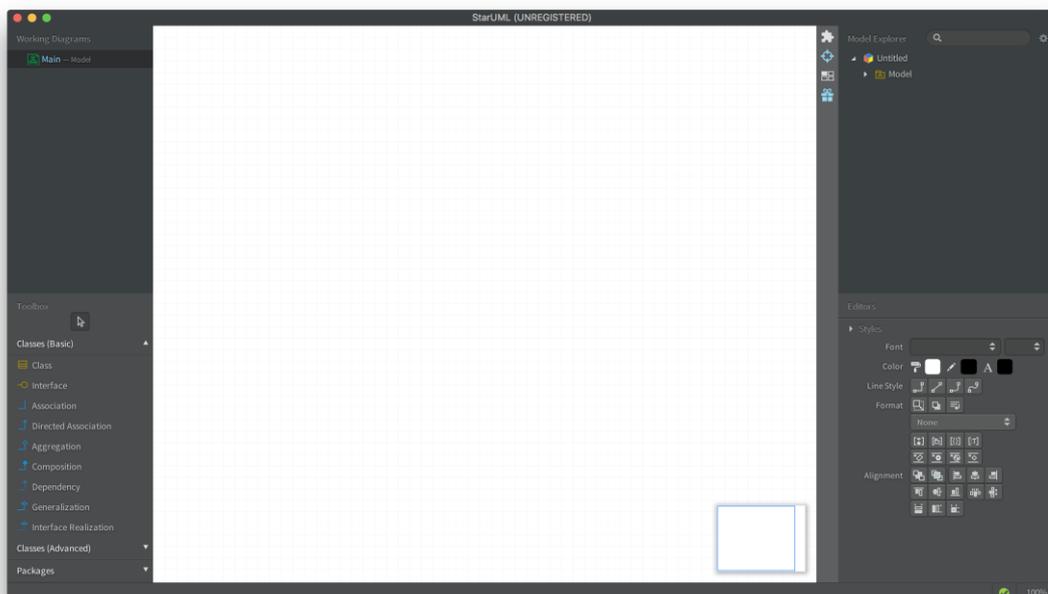


Figura 3.11: Interfaz de usuario StarUML

Además tiene la característica de que es capaz de generar código a partir de los diagramas y viceversa, en los lenguajes de programación C, C++ y Java. Por otro lado también ofrece la posibilidad de generar documentación en formatos Word, Excel o Powerpoint sobre los diagramas.

4 SERVIDOR DE SPORTMEET

La arquitectura es el alcance de la verdad.

- Louis I. Kahn -

En las secciones anteriores hemos visto todos los ingredientes necesarios que fundamentan nuestro proyecto. Ahora es turno de saber cómo se comportan y cómo se relacionan entre sí para conseguir el funcionamiento esperado de nuestra aplicación.

En esta sección se desarrollarán los fundamentos de la arquitectura del servidor de SportMeet, basándonos en una serie de diagramas UML como son los diagramas de clases, diagramas de secuencia y diagramas de casos de uso. Además, mostraremos el esquema de la base de datos implementada a través de un diagrama de Entidad-Relación y se mencionarán cada uno de los elementos que conforman nuestro servidor.

4.1 Introducción al servidor

Para entender de una manera clara el funcionamiento y la estructura que sigue la aplicación, comenzaremos mostrando los distintos diagramas mencionados anteriormente. Posteriormente realizaremos una serie de aclaraciones sobre los distintos paquetes y ficheros que constituyen nuestro servidor, de manera que se asienten todos los conceptos.

4.2 Diagramas de casos de uso

Comenzaremos detallando los distintos casos de uso a los que se puede enfrentar un usuario a la hora de usar la aplicación web. Para ello definiremos previamente los componentes de un diagrama de casos de uso y el esquema que seguiremos para los mismos.

4.2.1 Componentes y esquema del diagrama de casos de uso

Los diagramas de casos de uso se utilizan principalmente para especificar la comunicación y el comportamiento de un sistema a la hora de interactuar con otros usuarios o con otros sistemas externos. Es decir, muestra la relación entre los actores y los casos de uso de un sistema.

A continuación en la Figura 4.1 podemos observar el esquema que seguiremos y en el que se pueden distinguir hasta 4 componentes distintos:

- **Relación:** a través de ella asociamos un caso de uso a los actores que lo realizan.
- **Actor:** persona, organización, dispositivo o componente software externo que interactúa con el sistema.
- **Frontera:** se entiende como frontera al ámbito o sistema en el que nos encontramos (puede ser un pequeño componente software, una aplicación completa, un conjunto de aplicaciones).
- **Caso de uso:** representa las acciones que uno o varios actores realizan para la consecución de un objetivo.

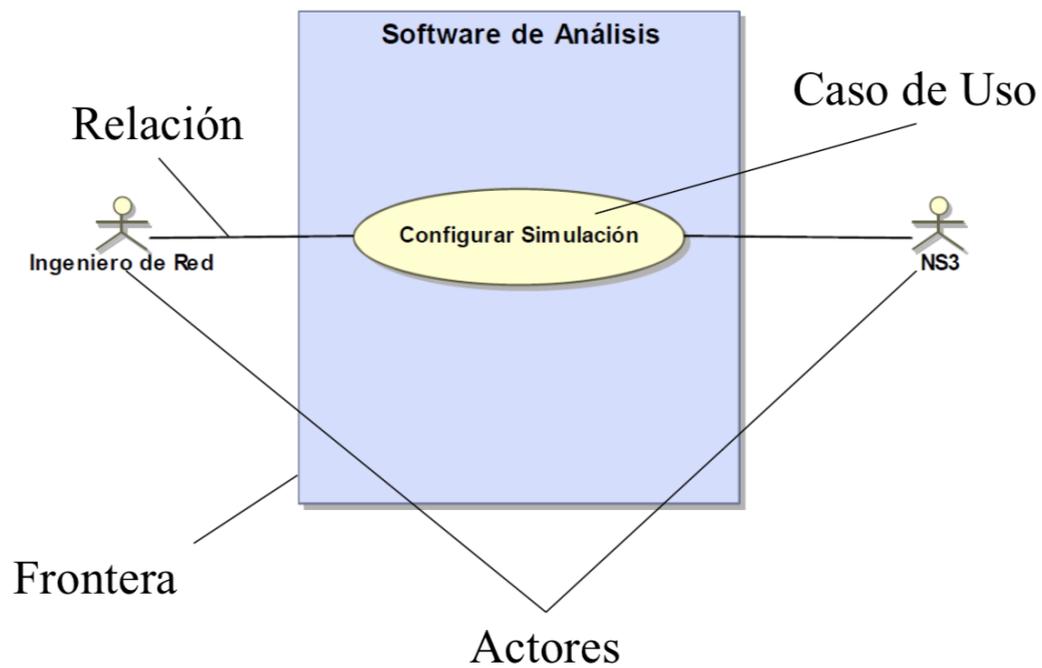


Figura 4.1: Diagrama de casos de uso

Adicionalmente a este esquema añadiremos una tabla con información adicional al caso de uso en cuestión, de forma que quede correctamente definido y se entienda su funcionamiento. El esquema a seguir será el mostrado en la Figura 4.2:

Caso de Uso (CU)		
Actores		
Descripción		
Objetivos		
Precondición		
Secuencia normal	Paso	Acción
Postcondición		
Excepciones	Paso	Acción
Comentarios		

Figura 4.2: Esquema adicional para diagramas de casos de uso

4.2.2 Diagramas de casos de uso de la aplicación

Una vez tenemos ambas herramientas, procedemos a definir los distintos diagramas de casos de uso que representan el funcionamiento de la aplicación.

4.2.2.1 Inicio de sesión

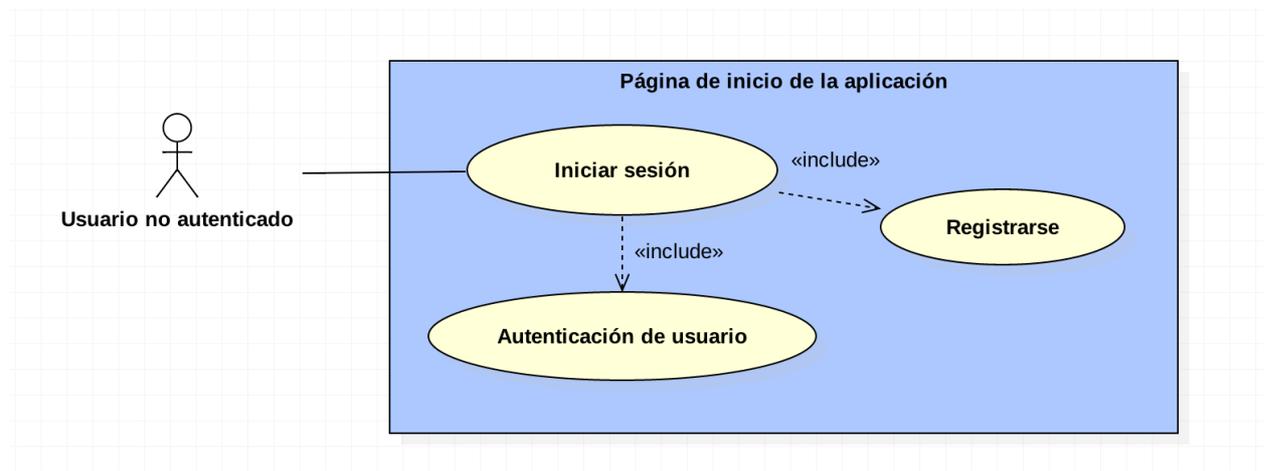


Figura 4.3: CU-01: Inicio de sesión

CU-01	Inicio de sesión	
Actores	Usuario no autenticado	
Descripción	El usuario necesita iniciar sesión para comenzar a usar los servicios de la aplicación web	
Objetivos	Iniciar sesión en la aplicación web	
Precondición	El usuario debe haber creado una cuenta de usuario para empezar a usar la aplicación	
Secuencia normal	Paso	Acción
	1	Introducir “username” y “contraseña” y pulsar en “Iniciar sesión”
	2	Se lleva a cabo la autenticación del usuario
	3	Se asignan los permisos asignados
Postcondición	El usuario se autentica y accede a los servicios de la aplicación web	
Excepciones	Paso	Acción
	2	Si los datos son incorrectos o no aparecen en la base de datos...
	E.1	El sistema informa del fallo que impide iniciar sesión
Comentarios	Para registrarse en la aplicación es necesario pulsar en “Registrarse” e introducir los datos de usuario para crear una nueva cuenta	

4.2.2.2 Usuario autenticado (acceso a servicios)

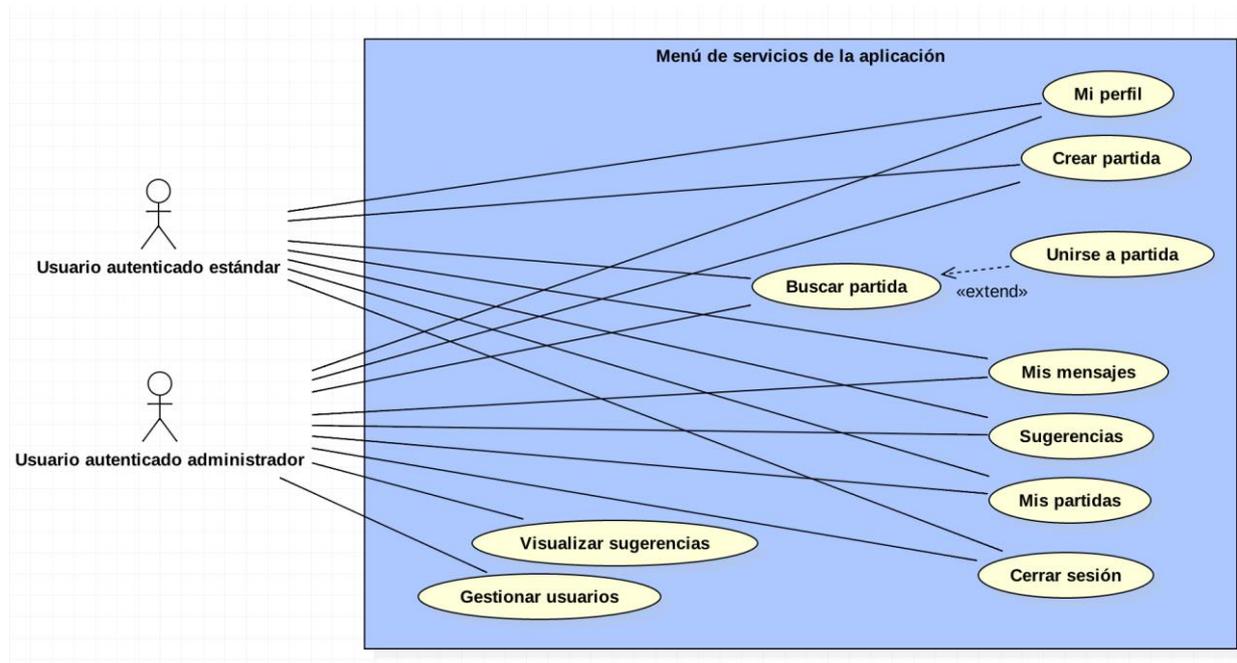


Figura 4.4: CU-02: Usuario autenticado (acceso a servicios)

CU-02	Usuario autenticado (acceso a servicios)	
Actores	Usuarios autenticados	
Descripción	El usuario ha iniciado sesión y puede acceder a los diferentes servicios	
Objetivos	Acceder a los servicios que ofrece la aplicación web	
Precondición	El usuario debe haber iniciado sesión	
Secuencia normal	Paso	Acción
	1	Seleccionar el servicio deseado
Postcondición	El usuario accede al servicio seleccionado	
Excepciones	Paso	Acción
		No hay excepciones
Comentarios	No hay comentarios	

4.2.2.3 Mi perfil

Nota: En el siguiente diagrama el usuario autenticado puede ser usuario administrador o usuario estándar.

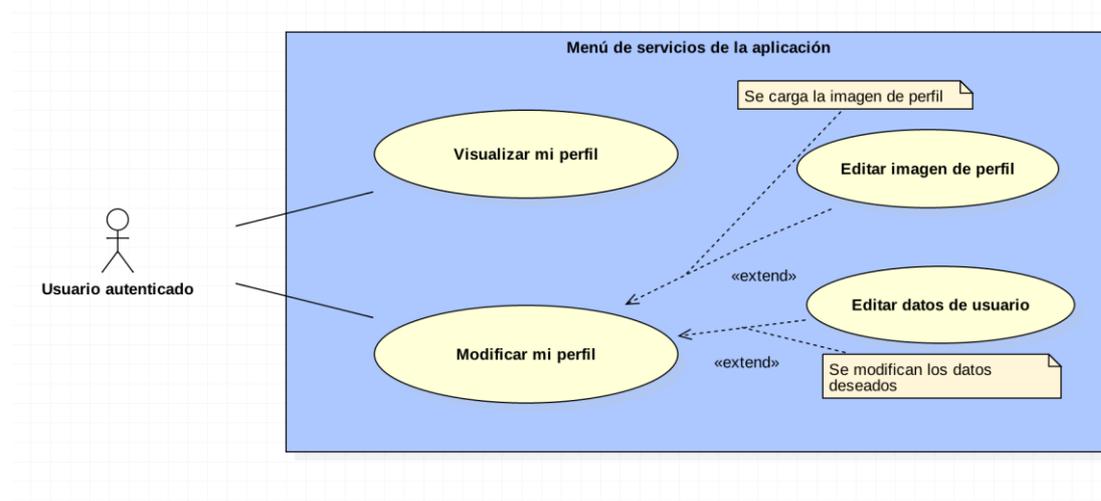


Figura 4.5: CU-03: Mi perfil

CU-03	Mi perfil	
Actores	Usuarios autenticados	
Descripción	El usuario ha iniciado sesión y puede acceder a los diferentes servicios	
Objetivos	Acceder al perfil del usuario para modificar o visualizar sus datos	
Precondición	El usuario debe haber iniciado sesión	
Secuencia normal 1	Paso	Acción (editar datos)
	1	Introducir nuevos datos y pulsar “Modificar datos”
Secuencia normal 2	Paso	Acción (editar imagen de perfil)
	1	Pulsar en “Seleccionar archivo”, cargar imagen y pulsar “Modificar datos”
Secuencia normal 3	Paso	Acción (visualizar perfil)
	1	Acceder a mi perfil
Postcondición	El usuario accede a los datos de su perfil y los actualiza/visualiza	
Excepciones	Paso	Acción
		No hay excepciones
Comentarios	Se actualizan los datos del usuario en la base de datos	

4.2.2.4 Crear partida

Nota: En el siguiente caso de uso no se muestra un diagrama debido a su sencillez. El usuario accede a la opción “Crear partida”, rellena el formulario con los datos de la partida y finalmente pulsa en crear. Detallamos este procedimiento a continuación (Figura 4.6):

CU-04 Crear partida		
Actores	Usuarios autenticados	
Descripción	El usuario ha iniciado sesión y pretende crear una nueva partida	
Objetivos	Crear una nueva partida	
Precondición	El usuario debe haber iniciado sesión	
Secuencia normal	Paso	Acción
	1	Introducir los datos de la partida y pulsar en “Crear partida”
Postcondición	El usuario crea una nueva partida	
Excepciones	Paso	Acción
		No hay excepciones
Comentarios	La partida se añade a la base de datos	

Figura 4.6: Tabla CU-04: Crear partida

4.2.2.5 Buscar partida

Nota: En el siguiente diagrama el usuario autenticado puede ser usuario administrador o usuario estándar.

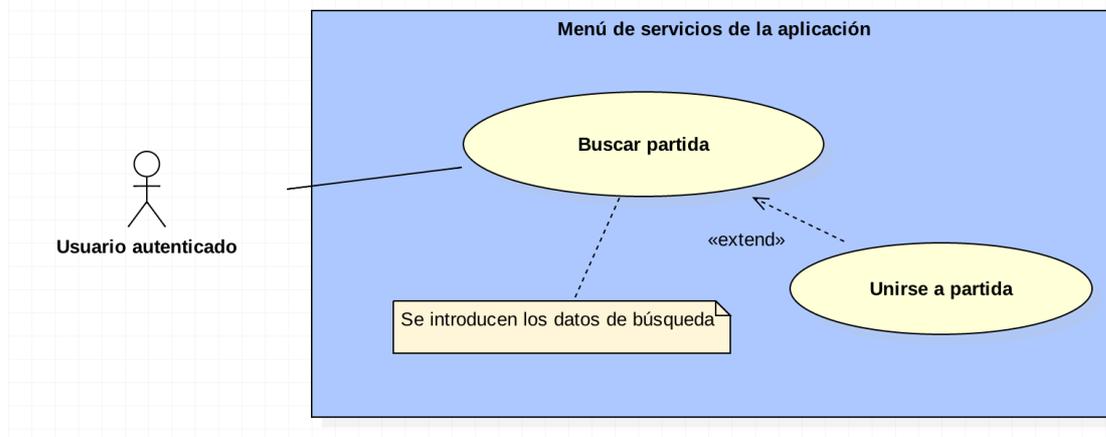


Figura 4.7: CU-05: Buscar partida

CU-05	Buscar partida	
Actores	Usuarios autenticados	
Descripción	El usuario ha iniciado sesión y pretende buscar una nueva partida	
Objetivos	Buscar una partida que reúna ciertos requisitos para unirse a ella	
Precondición	El usuario debe haber iniciado sesión	
Secuencia normal	Paso	Acción
	1	Introducir los datos de búsqueda de la partida y pulsar en “Buscar partida”
Postcondición	El usuario obtiene las partidas que cumplen con sus requisitos de búsqueda	
Excepciones	Paso	Acción
	1	Si no hay partidas, el sistema muestra un mensaje de información al usuario
Comentarios	Existen tres filtros de búsqueda: deporte, ciudad y provincia	

4.2.2.6 Unirse a partida

Nota: En el siguiente caso de uso no se muestra ningún diagrama debido a la sencillez de la acción. Una vez buscada la partida como vimos en el anterior caso de uso, el usuario pulsará en la opción “unirse” y así quedará vinculado a dicha partida (Figura 4.8):

CU-06		Unirse a partida	
Actores	Usuarios autenticados		
Descripción	El usuario ha buscado una partida y pretende unirse		
Objetivos	Unirse a una partida		
Precondición	El usuario debe haber iniciado sesión		
Secuencia normal	Paso	Acción	
	1	Pulsar en “Unirse”	
Postcondición	El usuario se une a la partida		
Excepciones	Paso	Acción	
	1	Si la partida está completa el sistema muestra un mensaje de información al usuario, por lo que el usuario no podrá unirse	
Comentarios	Si la partida no está llena, añade al usuario a la partida y actualiza la base de datos		

Figura 4.8: Tabla CU-06: Unirse a partida

4.2.2.7 Mis partidas

Nota: En el siguiente diagrama el usuario autenticado puede ser usuario administrador o usuario estándar.

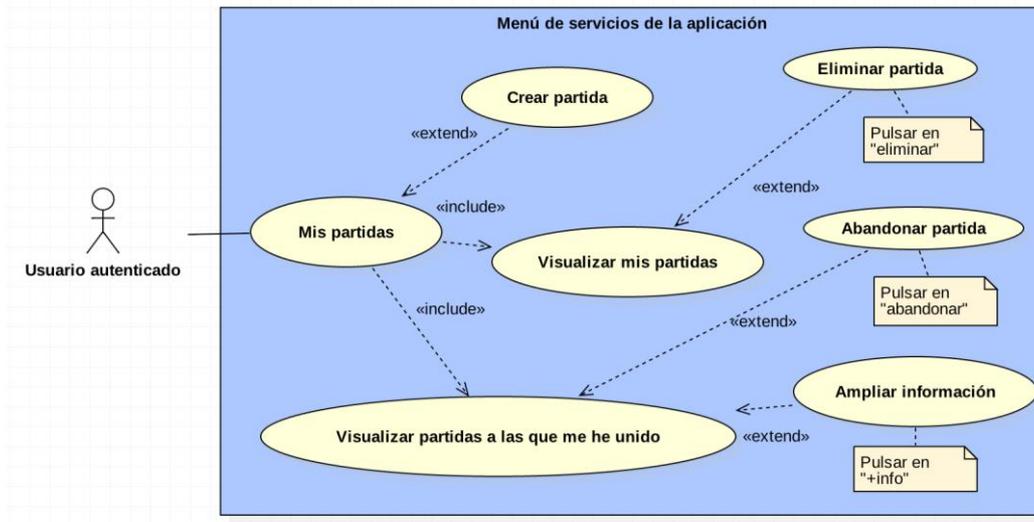


Figura 4.9: CU-07: Mis partidas

CU-07	Mis partidas	
Actores	Usuarios autenticados	
Descripción	El usuario ha iniciado sesión y pretende acceder a sus partidas	
Objetivos	Visualizar, eliminar, abandonar, crear u obtener información sobre mis partidas	
Precondición	El usuario debe haber iniciado sesión	
Secuencia normal 1	Paso	Acción (eliminar partida)
	1	Pulsar el botón “eliminar” de la partida correspondiente creada por el usuario
Secuencia normal 2	Paso	Acción (abandonar partida)
	1	Pulsar el botón “abandonar” de la partida correspondiente
Secuencia normal 3	Paso	Acción (crear partida)
	1	Pulsar el botón “Crear partida”
Secuencia normal 4	Paso	Acción (obtener información sobre una partida)
	1	Pulsar el botón “+ info” de la partida correspondiente
Postcondición	El usuario visualiza, crea, abandona, elimina u obtiene información de sus partidas	
Excepciones	Paso	Acción
		No hay excepciones
Comentarios	Si el usuario elimina, abandona o crea una partida, se actualiza la base de datos	

4.2.2.8 Mis mensajes

Nota: En el siguiente diagrama el usuario autenticado puede ser usuario administrador o usuario estándar.

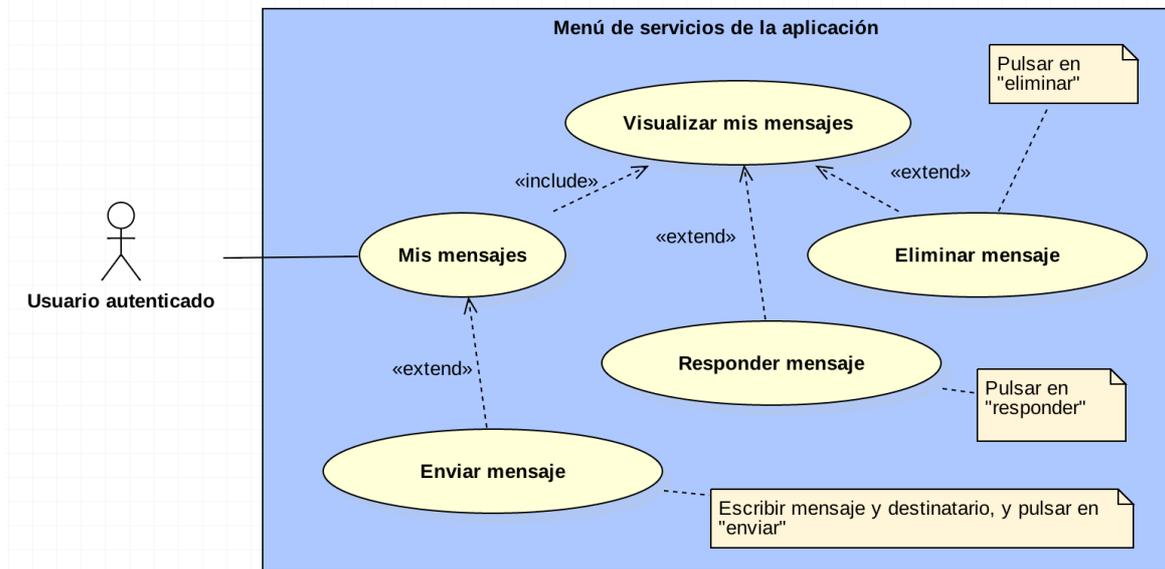


Figura 4.10: CU-08: Mis mensajes

CU-08	Mis mensajes	
Actores	Usuarios autenticados	
Descripción	El usuario ha iniciado sesión y pretende acceder a sus mensajes	
Objetivos	Visualizar, eliminar, responder y enviar mensajes	
Precondición	El usuario debe haber iniciado sesión	
Secuencia normal 1	Paso	Acción (eliminar mensaje)
	1	Pulsar el botón “eliminar” del mensaje correspondiente
Secuencia normal 2	Paso	Acción (responder mensaje)
	1	Pulsar el botón “responder” del mensaje correspondiente
Secuencia normal 3	Paso	Acción (enviar mensaje)
	1	Detallar el destinatario y el contenido del mensaje y pulsar en “enviar”
Postcondición	El usuario eliminar, responde o envía un mensaje	
Excepciones	Paso	Acción
		No hay excepciones
Comentarios	Se actualiza la base de datos si el usuario elimina, responde o envía algún mensaje	

4.2.2.9 Sugerencias

Nota: En el siguiente diagrama el usuario autenticado puede ser usuario administrador o usuario estándar.

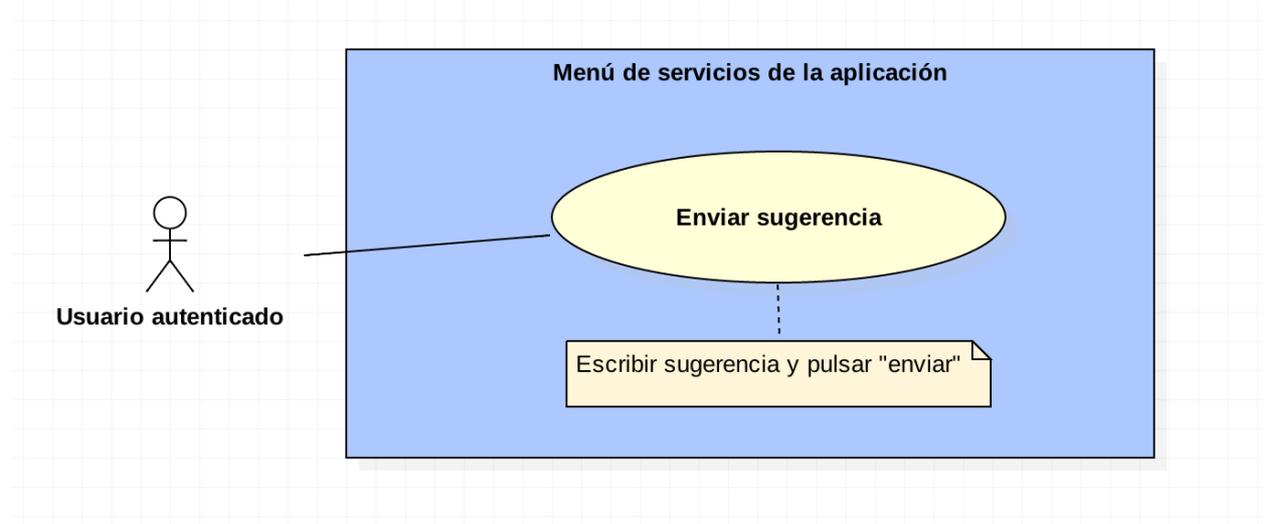


Figura 4.11: CU-09: Sugerencias

CU-09		Sugerencias	
Actores	Usuarios autenticados		
Descripción	El usuario ha iniciado sesión y pretende escribir una sugerencia de mejora		
Objetivos	Escribir una sugerencia		
Precondición	El usuario debe haber iniciado sesión		
Secuencia normal	Paso	Acción	
	1	Escribir la sugerencia y pulsa "enviar"	
Postcondición	El usuario manda su sugerencia al administrador		
Excepciones	Paso	Acción	
		No hay excepciones	
Comentarios	Se inserta una sugerencia en su tabla correspondiente de la base de datos		

4.2.2.10 Gestionar usuarios

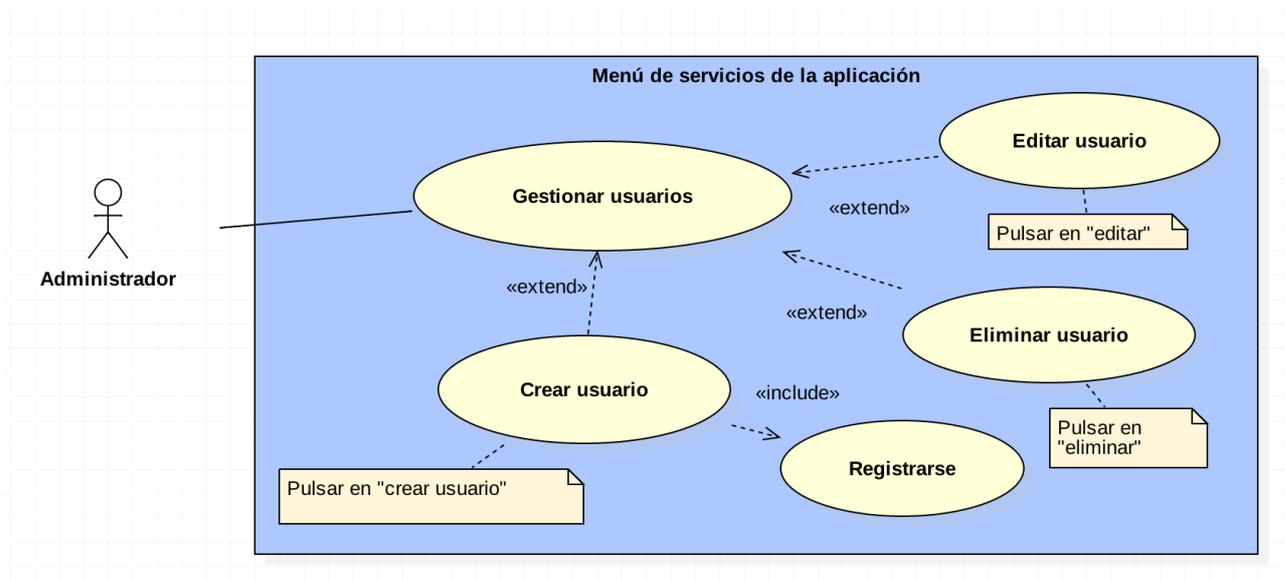


Figura 4.12: CU-10: Gestionar usuarios

CU-10	Gestionar usuarios	
Actores	Administrador	
Descripción	El administrador ha iniciado sesión y pretende gestionar los usuarios	
Objetivos	Crear, eliminar o editar usuarios	
Precondición	El administrador debe haber iniciado sesión	
Secuencia normal 1	Paso	Acción (eliminar usuario)
	1	Pulsar el botón “eliminar” del usuario correspondiente
Secuencia normal 2	Paso	Acción (editar usuario)
	1	Pulsar el botón “editar” del usuario correspondiente
Secuencia normal 3	Paso	Acción (crear usuario)
	1	Pulsar en “crear usuario”
	2	Rellenar los datos para crear el nuevo usuario
Postcondición	El administrador gestiona los usuarios según sus necesidades	
Excepciones	Paso	Acción (crear usuario)
	2	Las mismas que en CU-01 para el registro de usuarios
Comentarios	Se actualiza la base de datos si el administrador edita, crea o elimina algún usuario	

4.2.2.11 Visualizar sugerencias

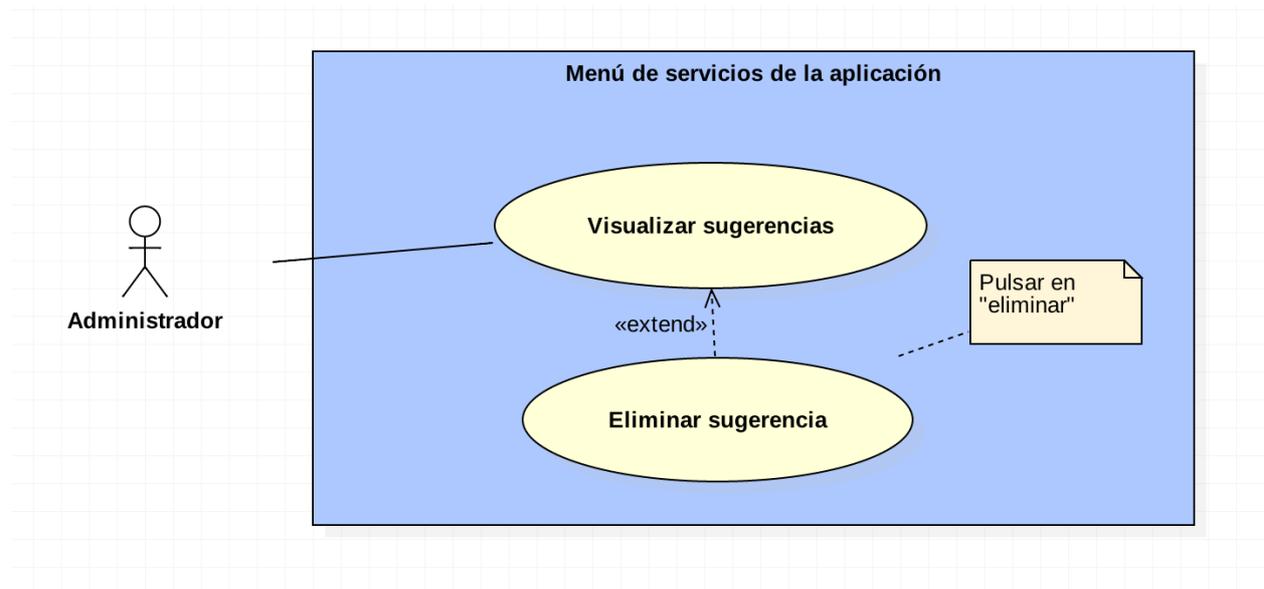


Figura 4.13: CU-11: Visualizar sugerencias

CU-11	Visualizar sugerencias	
Actores	Administrador	
Descripción	El administrador ha iniciado sesión y pretende visualizar las sugerencias	
Objetivos	Visualizar o eliminar las sugerencias	
Precondición	El administrador debe haber iniciado sesión	
Secuencia normal	Paso	Acción (eliminar sugerencia)
	1	Pulsar el botón "eliminar" para la sugerencia deseada
Postcondición	El administrador visualiza o elimina las sugerencias	
Excepciones	Paso	Acción
		No hay excepciones
Comentarios	Se actualiza la base de datos si el administrador borra alguna sugerencia	

4.2.2.12 Cerrar sesión

Nota: En el siguiente diagrama el usuario autenticado puede ser usuario administrador o usuario estándar.

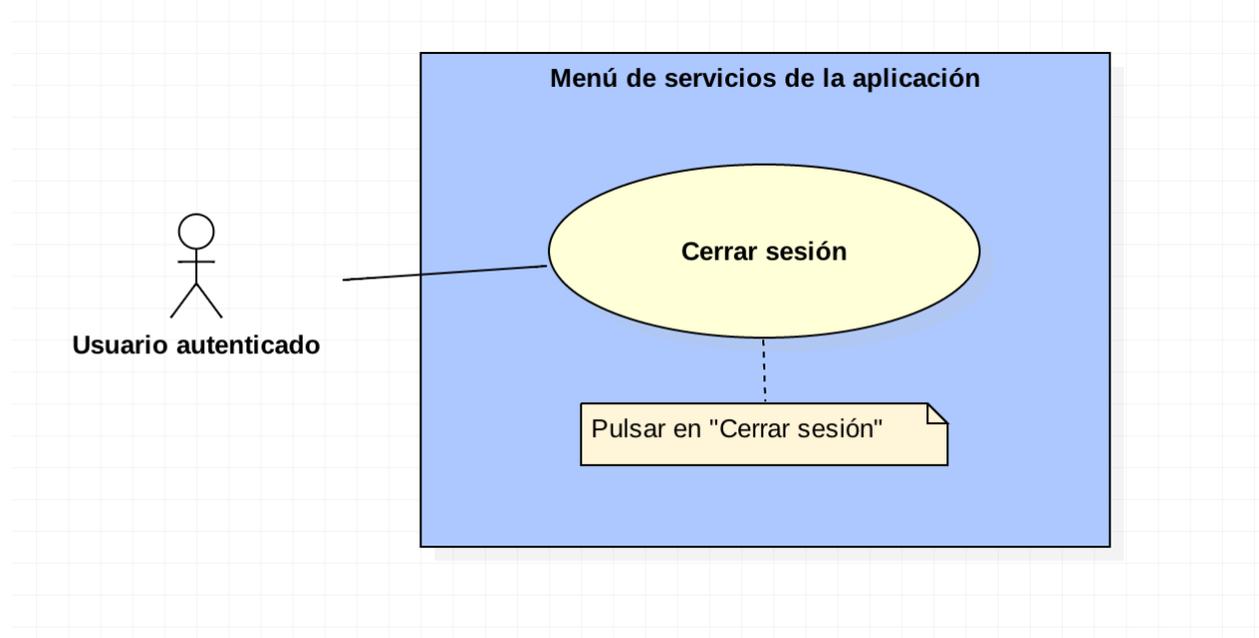


Figura 4.14: CU-12: Cerrar sesión

CU-12	Cerrar sesión	
Actores	Usuario autenticado	
Descripción	El usuario ha iniciado sesión y pretende cerrarla	
Objetivos	Cerrar la sesión del usuario	
Precondición	El usuario debe haber iniciado sesión	
Secuencia normal	Paso	Acción
	1	Pulsar el botón "Cerrar sesión"
Postcondición	El usuario cierra su sesión satisfactoriamente	
Excepciones	Paso	Acción
		No hay excepciones
Comentarios	El usuario vuelve a la página de inicio de la aplicación	

4.3 Diagrama de clases

Los diagramas de clases son diagramas de estructura estática principalmente utilizados para describir los tipos de datos existentes y sus relaciones independientemente de su implementación. Es decir, muestran la estructura del sistema describiendo sus clases, sus correspondientes atributos, sus métodos y las relaciones entre objetos. De esta forma, conseguimos centrar la atención en los aspectos lógicos de la clase en vez de en su implementación.

En nuestra aplicación existen 4 tipos de objetos principales en torno a los que girará esta sección: usuarios, partidas, mensajes y sugerencias (Figura 4.15):

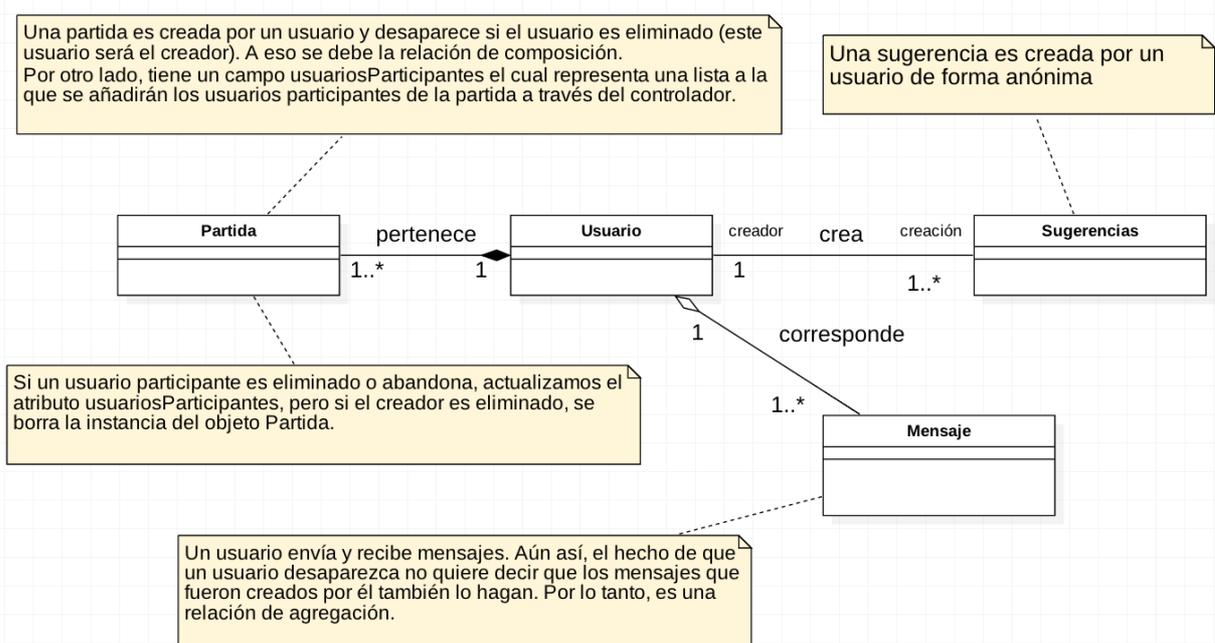


Figura 4.15: Diagrama de clases de la aplicación web

Por otro lado, a continuación (Figuras 4.16A-B) podemos observar los atributos y métodos de los que se componen las clases mostradas en la anterior figura:

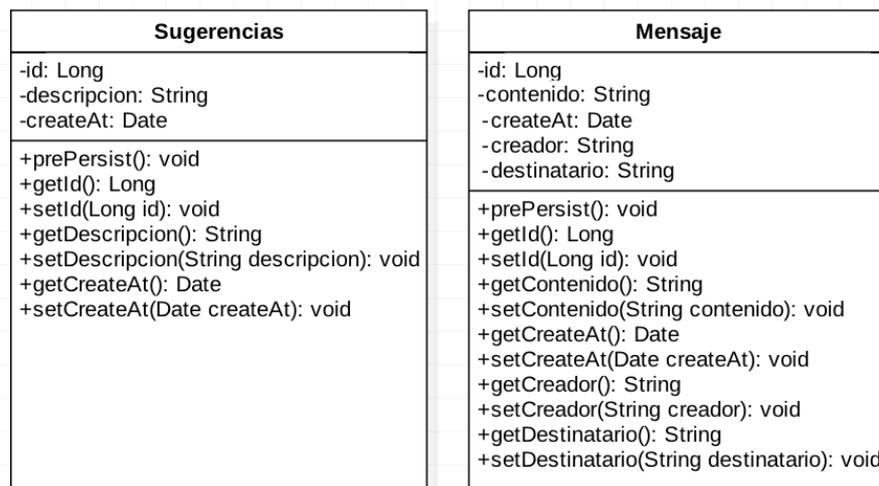


Figura 4.16-A: Atributos y métodos de las clases (1)

Partida	Usuario
-id: Long -creador: String -deporte: String -provincia: String -ciudad: String -fecha: Date -horaComienzo: Date -horaFinal: Date -participantes: int -suplentes: int -usuariosParticipantes: List<Usuario>	-id: Long -username: String -password: String -repPassword: String -enabled: Boolean -rol: String -nombre: String -apellido: String -edad: int -foto: String -email: String -create_at: String -partidas: List<Partida>
+getId(): Long +setId(Long id): void +getCreador(): String +setCreador(String creador): void +getDeporte(): String +setDeporte(String deporte): void +getProvincia(): String +setProvincia(String provincia): void +getCiudad(): String +setCiudad(String ciudad): void +getFecha(): Date +setFecha(Date fecha): void +getHoraComienzo(): Date +setHoraComienzo(Date horaComienzo): void +getHoraFinal(): Date +setHoraFinal(Date horaFinal): void +getParticipantes(): int +setParticipantes(int participantes): void +getSuplentes(): int +setSuplentes(int suplentes): void +getComentarios(): String +setComentarios(String comentarios): void +getUsuario(): Usuario +setUsuario(Usuario usuario): void +getUsuariosParticipantes(): List<Usuario> +setUsuariosParticipantes(List<Usuario>usuariosParticipantes): void +addParticipantes(Usuario usuario): void	+Usuario() +prePersist(): void +getNombre(): String +setNombre(String nombre): void +getApellido(): String +setApellido(String apellido): void +getEdad(): int +setEdad(int edad): void +getEmail(): String +setEmail(String email): void +getCreateAt(): Date +setCreateAt(Date createAt): void +getId(): Long +setId(Long id): void +getUsername(): String +setUsername(String username): void +getPassword(): String +setPassword(String password): void +getRepPassword(): String +setRepPassword(String repPassword): void +getEnabled(): Boolean +setEnabled(Boolean enabled): void +getRol(): String +setRol(String rol): void +getFoto(): String +setFoto(String foto): void +getPartidas(): List<Partida> +setPartidas(List<Partida>partidas): void +addPartida(Partida partida): void

Figura 4.16-B: Atributos y métodos de las clases (2)

4.4 Diagramas de secuencia

El objetivo principal de los diagramas de secuencia es modelar las distintas interacciones que se llevan a cabo entre los componentes que componen la aplicación. En esta sección mostraremos una serie de diagramas de secuencia en los que describimos los procesos más significativos del funcionamiento de la aplicación.

4.4.1 Creación de un usuario

El primer paso para empezar a utilizar la aplicación será la creación de una nueva cuenta de usuario. A la hora de crear la cuenta, el usuario puede encontrarse con tres situaciones: no completa los datos correctamente, utiliza un username o email ya en uso o introduce los datos correctamente y se crea el usuario. En el siguiente diagrama (Figura 4.17) se refleja este proceso:

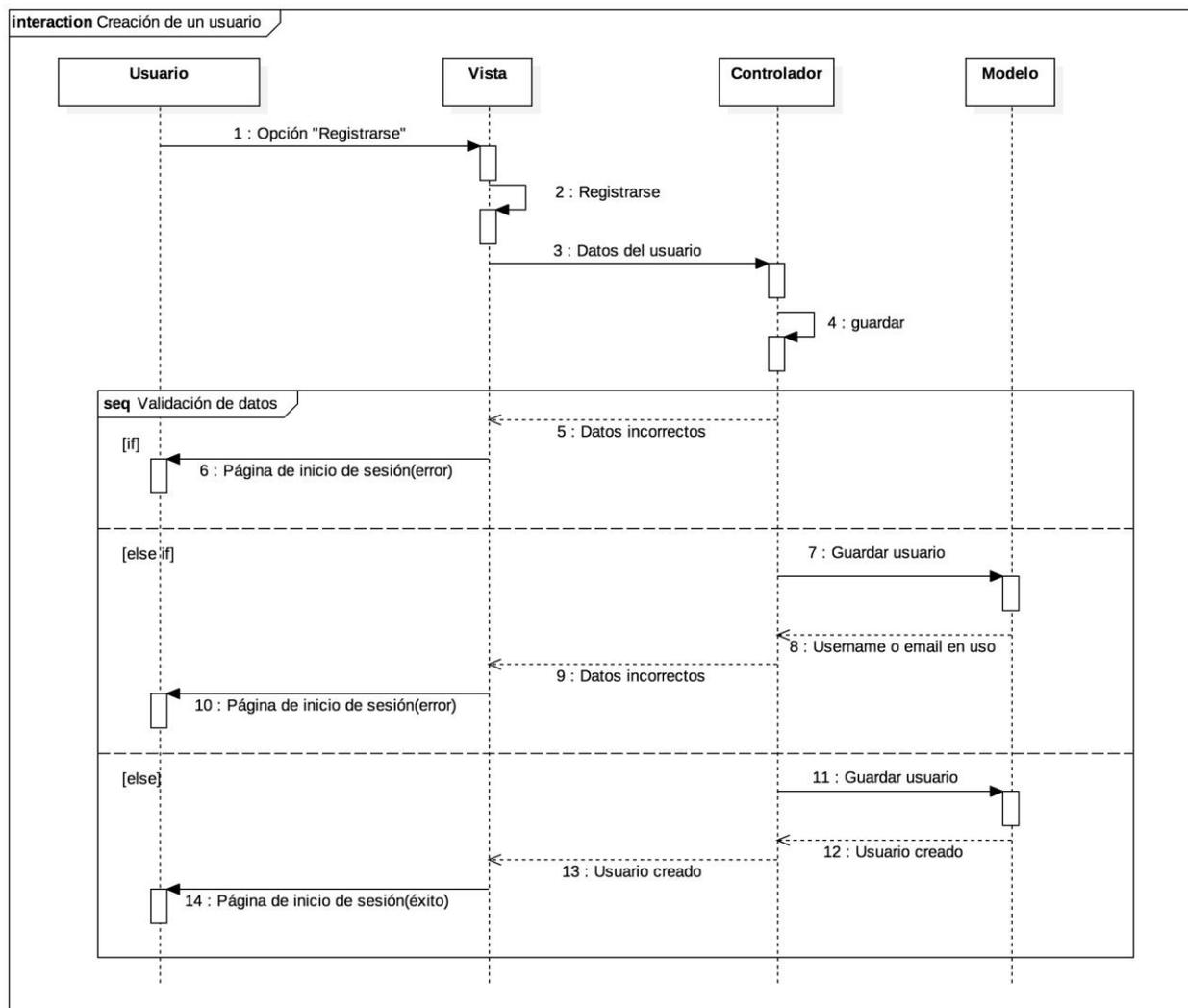


Figura 4.17: Diagrama de secuencia de creación de usuario

4.4.2 Inicio de sesión

A continuación se muestra un diagrama de secuencia que simboliza el inicio de sesión de un usuario en la aplicación. Podemos observar como la autenticación del usuario puede acabar de dos formas distintas: con su correcta autenticación y acceso, o bien con una denegación de acceso a la aplicación (Figura 4.18):

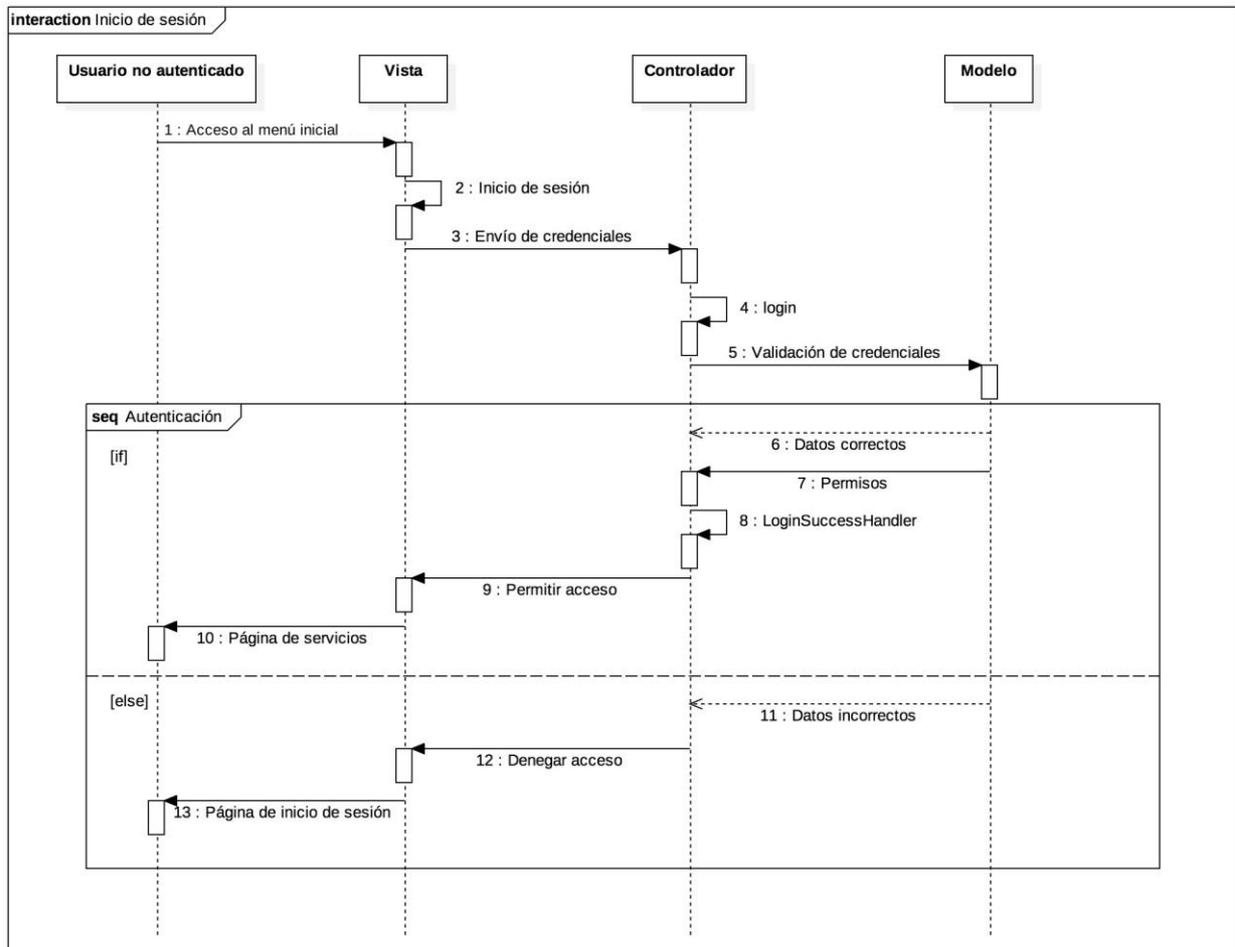


Figura 4.18: Diagrama de secuencia del inicio de sesión

4.4.3 Creación de una partida

En el siguiente diagrama (Figura 4.19) observaremos el proceso de creación de una partida. Este proceso será prácticamente similar al proceso de creación de mensajes y sugerencias, distinguiéndose en los métodos utilizados en el controlador para gestionar los distintos eventos. El resto del diagrama será similar:

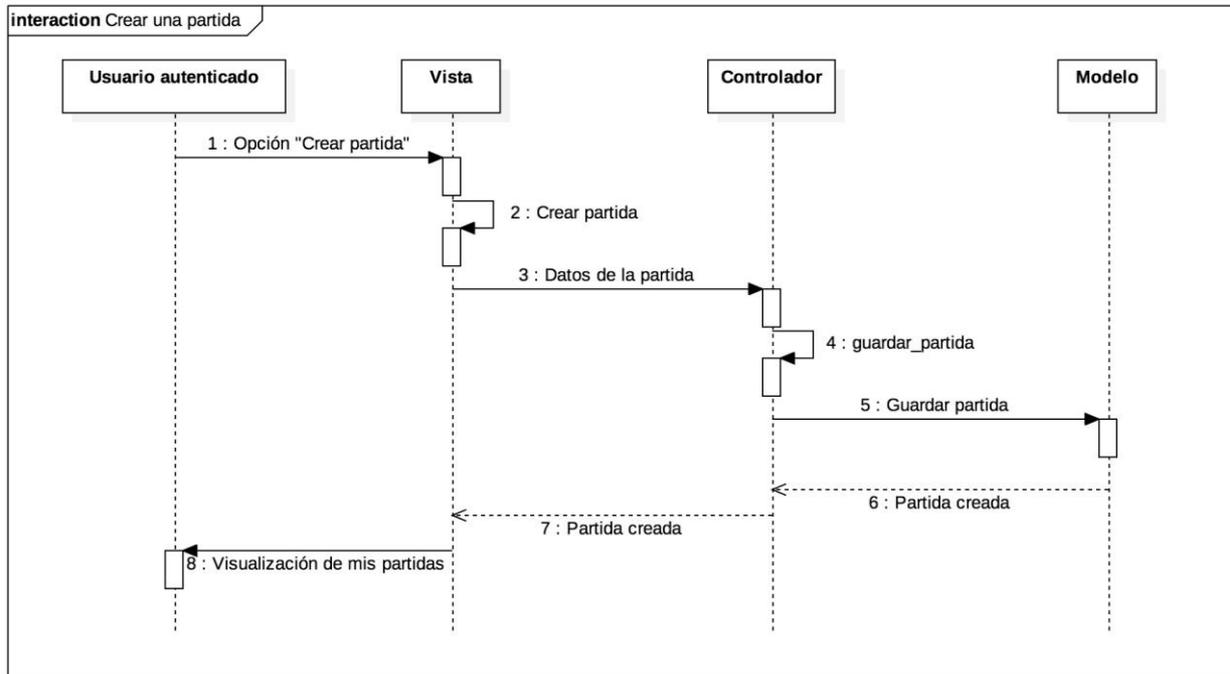


Figura 4.19: Diagrama de secuencia de creación de una partida

4.4.4 Borrado de una partida

A continuación se detalla el proceso de borrado de una partida el cual será similar al proceso de borrado de una sugerencia, un usuario o un mensaje, diferenciándose en los métodos usados en su respectivo controlador. Podemos observar este proceso en la Figura 4.20:

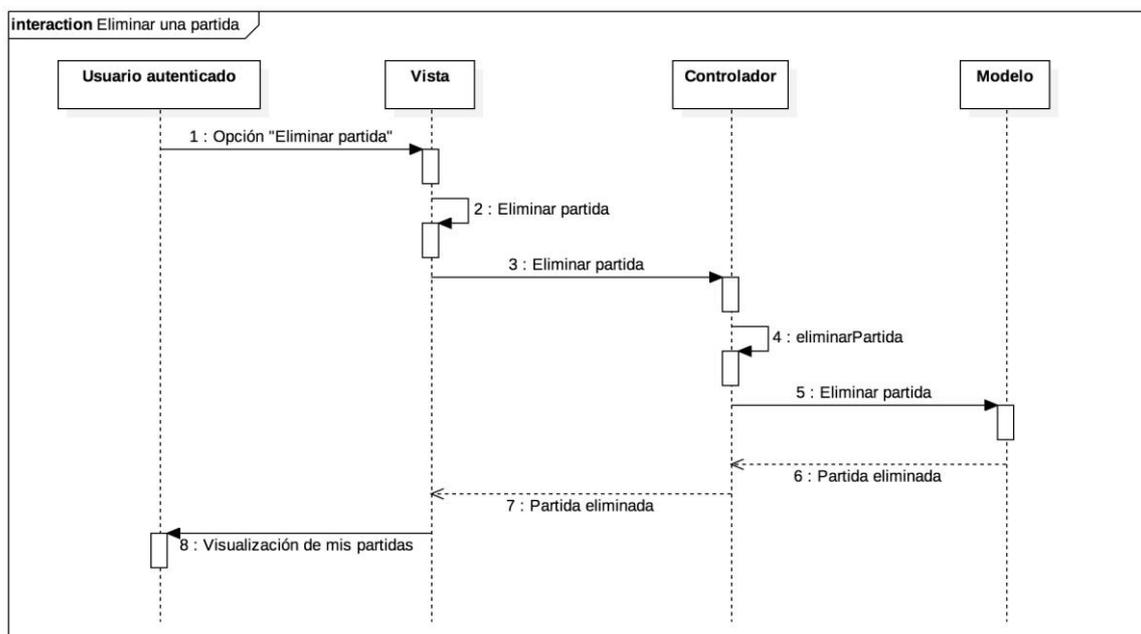


Figura 4.20: Diagrama de secuencia de borrado de una partida

4.4.5 Búsqueda y unión a una partida

En este diagrama (Figura 4.21) veremos el proceso a seguir para buscar una partida y la posterior unión del usuario a la misma:

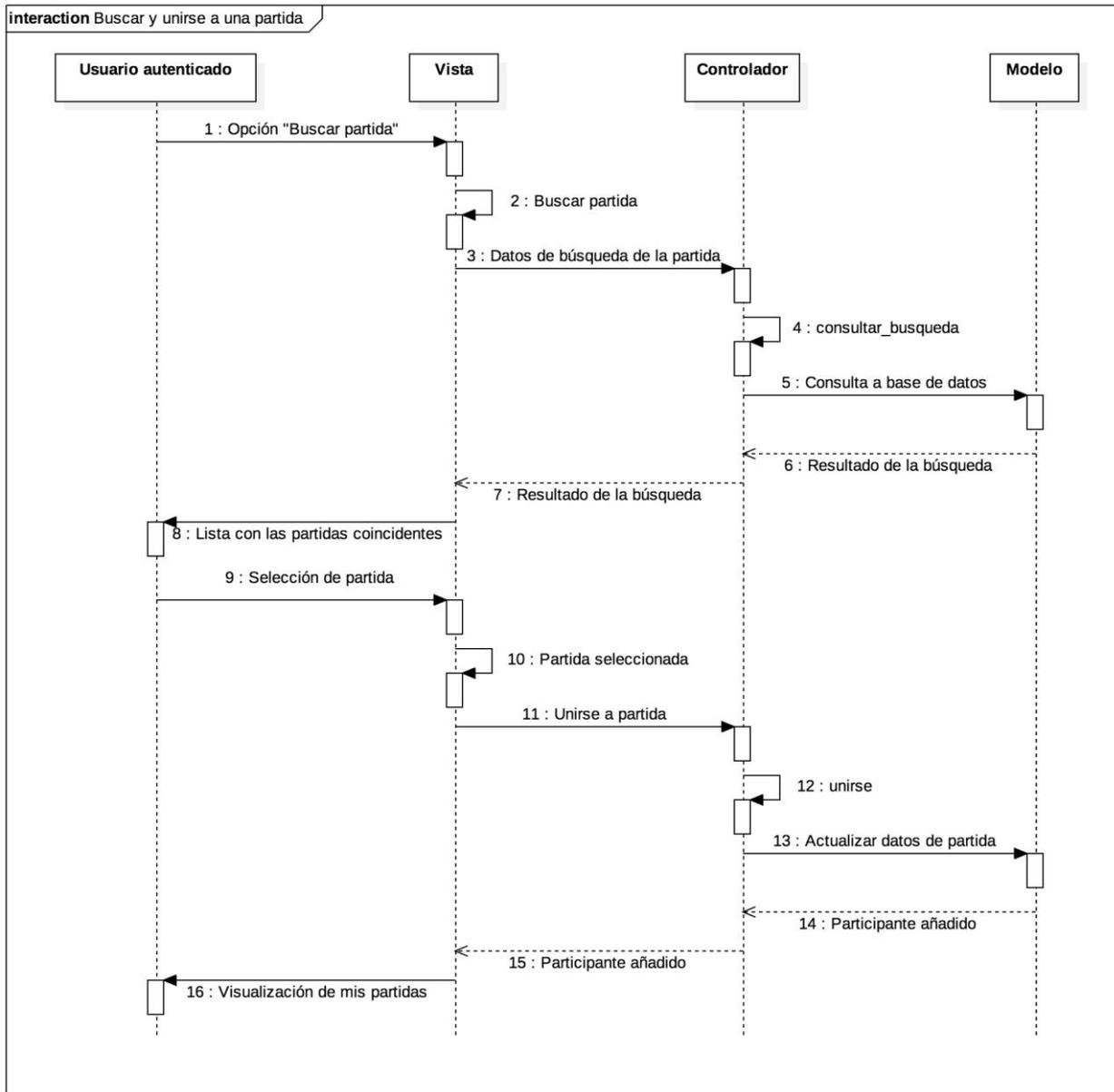


Figura 4.21: Diagrama de secuencia de la búsqueda y unión a una partida

4.5 Arquitectura de la base de datos

En este apartado comentaremos cuál es la estructura que sigue la base de datos de la aplicación, definiendo el esquema que sigue y los distintos elementos que la conforman.

4.5.1 Diagrama de entidad-relación de la base de datos (DER)

En la Figura 4.22 se muestra el diagrama entidad-relación que representa la arquitectura de la base de datos de la aplicación, donde se pueden observar las distintas relaciones entre los distintos componentes (“ManyToMany” entre las tablas “usuarios” y “partidas”, y “OneToMany” desde la tabla “usuarios” a las tablas “mensajes” y “sugerencias”):

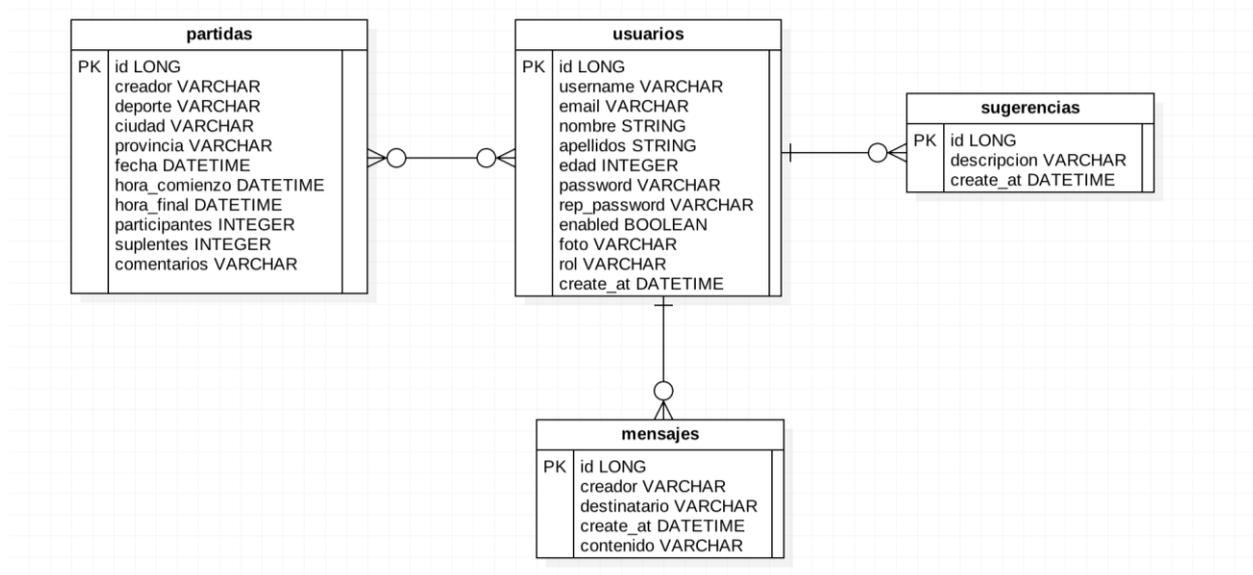


Figura 4.22: Diagrama entidad-relación de la aplicación

4.5.2 Elementos del diagrama E-R

A continuación describiremos los campos contenidos en cada una de las tablas mencionadas en el anterior diagrama.

4.5.2.1 Tabla “usuarios”

Esta tabla contiene toda la información correspondiente a los usuarios que están utilizando la aplicación.

Atributo	Descripción
id	Atributo que identifica a un usuario. Actúa como llave primaria.
username	Nombre de la cuenta del usuario. Dimensión máxima de 30 caracteres. Debe ser único.
email	Email del usuario. Debe ser único.
nombre	Nombre del usuario.
apellido	Apellido del usuario.
edad	Edad del usuario.
password	Contraseña del usuario. Tiene una dimensión máxima de 60 caracteres.
rep_password	Valida que la contraseña se ha introducido correctamente en el registro.
create_at	Fecha de creación del usuario.
enabled	Se pone a “true” cuando el usuario ha sido creado y activado.
foto	Contiene la foto de perfil del usuario.
rol	Representa el rol del usuario para temas de autorización.

4.5.2.2 Tabla “partidas”

Esta tabla contiene toda la información correspondiente a las partidas que son creadas por los usuarios durante el uso de la aplicación.

Atributo	Descripción
id	Atributo que identifica a una partida. Actúa como llave primaria.
usuario_id	Identificador del usuario creador de la partida.
deporte	Nombre del deporte de la partida.
ciudad	Ciudad en la que se disputará la partida.
provincia	Provincia en la que se disputará la partida.
creador	Username del creador de la partida.
fecha	Fecha en la que se llevará a cabo la partida. Sigue el patrón “dd-MM-yyyy”
hora_comienzo	Hora de comienzo de la partida. Sigue el patrón “HH:mm”
hora_final	Hora de finalización de la partida. Sigue el patrón “HH:mm”
participantes	Número de participantes que disputarán la partida.
suplentes	Número de suplentes para la partida.
comentarios	Contiene información adicional sobre la partida.

4.5.2.3 Tabla “usuarios_partidas”

Esta tabla nace debido a la relación “ManyToMany” entre las tablas “usuarios” y “partidas”. Se genera automáticamente a partir del código y relaciona los usuarios con las partidas de la aplicación, de forma que podamos gestionar qué usuarios pertenecen a qué partidas.

Atributo	Descripción
Usuarios_participantes_id	Atributo que identifica a un usuario. Actúa como llave foránea referenciando a la tabla “usuarios”.
partidas_id	Atributo que identifica a una partida. Actúa como llave foránea referenciando a la tabla “partidas”.

4.5.2.4 Tabla “mensajes”

Esta tabla contiene toda la información correspondiente a los mensajes que los usuarios intercambian entre sí durante el uso de la aplicación.

Atributo	Descripción
id	Atributo que identifica a un mensaje. Actúa como llave primaria.
contenido	Corresponde al mensaje en sí. Tiene una dimensión máxima de 255 caracteres.
creador	Identifica al creador del mensaje a través de username.
destinatario	Identifica al destinatario del mensaje a través de su username.
create_at	Establece la fecha de creación del mensaje.

4.5.2.5 Tabla “sugerencias”

Esta tabla contiene toda la información correspondiente a las sugerencias que los usuarios crean acerca de la aplicación con el fin de mejorarla.

Atributo	Descripción
id	Atributo que identifica a una sugerencia. Actúa como llave primaria.
descripcion	Corresponde a la sugerencia en sí. Tiene una dimensión máxima de 255 caracteres.
create_at	Establece la fecha de creación de la sugerencia.

4.6 Arquitectura de los componentes

4.6.1 Componentes de la aplicación

Como mencionamos anteriormente, para el correcto funcionamiento de nuestro proyecto es necesario realizar una relación adecuada entre los distintos componentes que lo conforman. Para el desarrollo de la aplicación se ha seguido el patrón de diseño Modelo-Vista-Controlador como pudimos ver en la sección 3 (“Tecnologías utilizadas”). Tras implementarlo, la arquitectura de la aplicación tiene el aspecto que muestra la Figura 4.23:

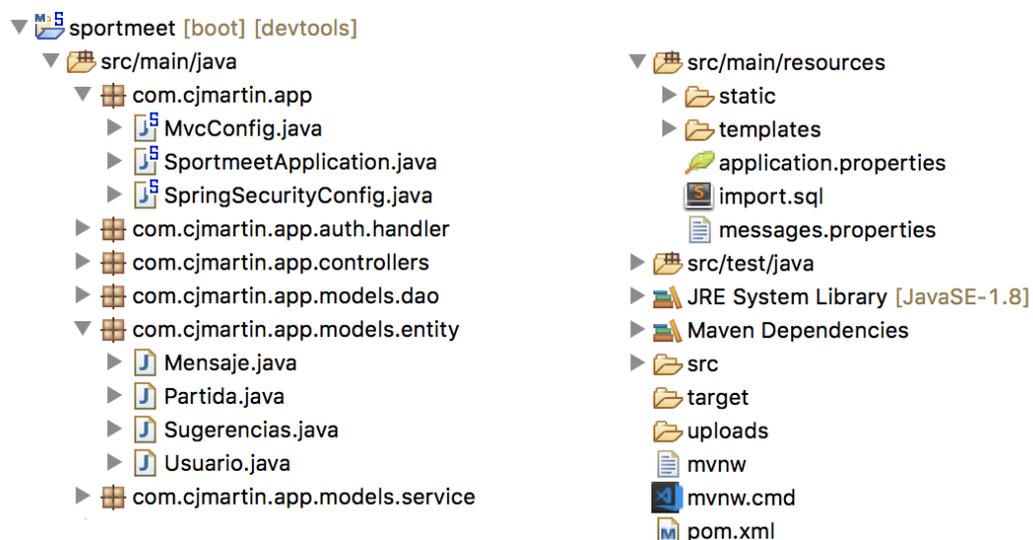


Figura 4.23: Arquitectura de la aplicación

Comenzaremos comentando cada uno de estos elementos para tener así una idea clara de qué funciones realiza cada componente y el por qué de su necesidad:

- **Carpeta “src/main/java”:** dentro de esta carpeta encontramos los siguientes paquetes relevantes:
 - **com.cjmartin.app:** este paquete contiene las clases necesarias para configurar varios aspectos del patrón modelo-vista-controlador, poner en funcionamiento la aplicación e implementar la autenticación con Spring Security, como podemos observar en la Figura 4.24:

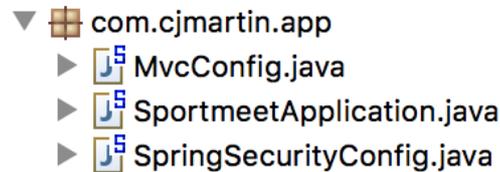


Figura 4.24: Contenido del paquete “com.cjmartin.app”

- **com.cjmartin.app.auth.handler:** este paquete contiene la clase que se encarga de implementar correctamente el proceso de autorización de la aplicación una vez la autenticación ha sido exitosa, como vemos en la Figura 4.25:



Figura 4.25: Contenido del paquete “com.cjmartin.app.auth.handler”

- **com.cjmartin.app.controllers:** este paquete contiene los controladores asignados a cada uno de los objetos que posee la aplicación, los cuales se muestran en el paquete “com.cjmartin.app.models.entity”. Se encargan de gestionar los sucesos relacionados con estas entidades que llegan directamente desde la interfaz de usuario y sus posteriores consultas y acciones con el modelo. Además, contiene el controlador que se encarga del proceso de inicio de sesión (LoginController). Vemos el contenido del paquete en la Figura 4.26:



Figura 4.26: Contenido del paquete “com.cjmartin.app.controllers”

- **com.cjmartin.app.models.dao:** este paquete contiene las interfaces que especifican las consultas a la base de datos asignadas a los métodos implementados en las clases del paquete “com.cjmartin.app.models.service”. Se ha hecho uso del “CRUD Repository” para implementar las funciones básicas del CRUD (crear, leer, actualizar y borrar). Podemos verlas en la Figura 4.27:



Figura 4.27: Contenido del paquete “com.cjmartin.app.models.dao”

- **com.cjmartin.app.models.entity:** este paquete contiene la definición de los atributos y métodos de las distintas entidades u objetos que intervienen en la aplicación. En la Figura 4.28 se muestran las entidades existentes:



Figura 4.28: Contenido del paquete “com.cjmartin.app.models.entity”

- **com.cjmartin.app.models.service:** esta carpeta contiene varias interfaces donde se definen varios servicios ofrecidos para cada entidad (además de los servicios que ofrece el CRUD) y las clases que contienen la implementación de estos servicios y los definidos en el paquete “com.cjmartin.app.models.dao”. Podemos ver su contenido en la Figura 4.29:

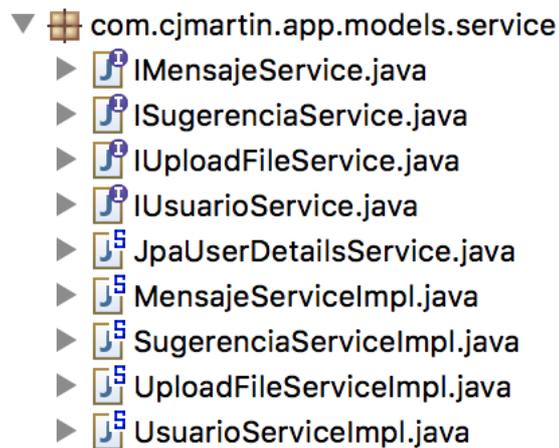


Figura 4.29: Contenido del paquete “com.cjmartin.app.models.service”

- **Carpeta “src/main/resources”:** dentro de esta carpeta encontramos las siguientes subcarpetas y ficheros relevantes:
 - **Carpeta “static”:** contiene subcarpetas con los ficheros necesarios para los estilos y la apariencia visual de la aplicación web (carpeta “css” y “font” para estilos, carpeta “images” con las imágenes utilizadas en la aplicación y carpeta “js” con el código JavaScript usado) como se observa en la Figura 4.30:

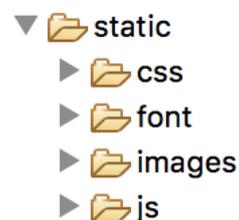


Figura 4.30: Contenido de la carpeta “static”

- **Carpeta “templates”:** contiene las plantillas con el código HTML de las vistas de la aplicación web (se observa también una carpeta “partida” con varias plantillas relacionadas con las partidas y una carpeta “layout” que contiene un esquema con los elementos básicos desde los que parten todas las plantillas) como observamos en la Figura 4.31:

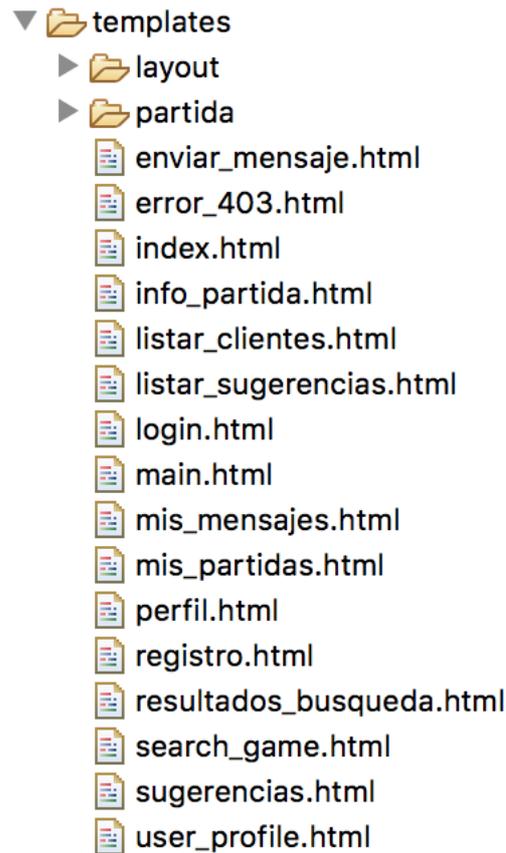


Figura 4.31: Contenido de la carpeta “templates”

- **Fichero “application.properties”:** en este fichero se describen distintas propiedades de la aplicación, como por ejemplo la base de datos a utilizar o características relacionadas con la subida de ficheros, como se puede observar en la Figura 4.32:

```

1 spring.datasource.url=jdbc:mysql://localhost/sportmeet?useSSL=false
2 spring.datasource.username=root
3 spring.datasource.password=root
4 spring.datasource.driver-class-name=com.mysql.jdbc.Driver
5 spring.jpa.database = mysql
6 spring.jpa.database-platform= org.hibernate.dialect.MySQL5Dialect
7 logging.level.org.hibernate.SQL = debug
8 spring.servlet.multipart.max-file-size= 10MB
9 spring.servlet.multipart.max-request-size= 10MB

```

Figura 4.32: Fichero “application.properties”

- **Fichero “import.sql”:** este fichero ha sido utilizado para realizar consultas y operaciones de prueba con la aplicación a través de sentencias SQL.
 - **Fichero “messages.properties”:** en este fichero se definen los distintos mensajes que la aplicación muestra al usuario tras la realización de distintas acciones, como por ejemplo los requisitos a la hora de crear una nueva cuenta de usuario.
- **Otros componentes relevantes:**
 - **Carpeta “uploads”:** contiene las imágenes cargadas en la aplicación por los usuarios a la hora de establecer una imagen de perfil.
 - **Maven Dependencies:** contiene las dependencias necesitadas por Maven para una correcta ejecución y funcionamiento de la aplicación.
 - **Fichero “pom.xml”:** contiene información acerca del proyecto necesaria para Maven, como las fuentes, test, dependencias, plugins, versión, etc.

4.6.2 Recursos de los controladores

Como hemos mencionado anteriormente, los controladores se encargan de gestionar los sucesos o eventos procedentes de la interfaz de usuario y realizan las tareas pertinentes. Es importante aclarar que en nuestra aplicación, cada uno de los métodos implementados por los controladores devuelven un String con el nombre de la vista lógica a cargar. Posteriormente, Spring usa un objeto del tipo ModelAndView para cargar la vista HTML, en nuestro caso con Thymeleaf pasando todos los parámetros de la vista. Thymeleaf analizará estos parámetros junto a la plantilla y genera el contenido HTML, el cual se pasa a la respuesta y se manda al usuario. Todo este proceso se realiza en el servidor de la aplicación de forma transparente.

Una vez aclarado este punto, pasamos a detallar los métodos contenidos en cada uno de los controladores y su finalidad.

4.6.2.1 Recursos de Usuario

Método	Nombre	URI	Parámetros	Descripción
GET	verFoto	"/uploads/{filename:.+}"	String filename	Ver la foto de perfil
GET	listar	"/listar_clientes"	Model model, Authentication authentication, HttpServletRequest request	Listar todos los usuarios
POST	editar	"/perfil/{id}"	Long id, Model model, RedirectAttributes flash	Editar un usuario
POST	crear	"/registro"	Model model	Crear un nuevo usuario
POST	guardar	"/registro"	Usuario usuario, BindingResult result, Model model, MultipartFile foto, RedirectAttributes flash, SessionStatus status	Procesar el formulario
DELETE	eliminar	"/eliminar/{id}"	Long id, RedirectAttributes flash	Eliminar un usuario
GET	index	/index	Model model	Mostrar la portada
GET	main	/main	Model model, Authentication authentication	Mostrar la página inicial

4.6.2.2 Recursos de Partida

Método	Nombre	URI	Parámetros	Descripción
GET	mis_partidas	"/mis_partidas/{id}"	Long id, Model model, RedirectAttributes flash	Listar partidas del usuario
POST	crear	"/form/{usuarioId}"	Long usuarioId, Model model, RedirectAttributes flash	Crear una nueva partida
DELETE	eliminar	"/eliminar/{id}"	Long id, RedirectAttributes flash	Eliminar una partida
DELETE	abandonar	"/abandonar/{partida_id}/{usuario_id}"	Long partida_id, Long usuario_id, RedirectAttributes flash	Abandonar una partida
GET	informacion	"/informacion/{partida_id}/{usuario_id}"	Long partida_id, Long usuario_id, RedirectAttributes flash, Model model	Obtiene información sobre las partidas
POST	unirse	"/unirse/{partidaId}/{usuarioId}"	Long partida_id, Long usuario_id, RedirectAttributes flash	Unirse a una partida
POST	guardar	/form	Partida partida, BindingResult result, Model model, RA flash, SessionStatus status	Guardar la partida
GET	buscar	"/buscar_partida/{usuarioId}"	Model model, Long usuarioId, RedirectAttributes flash	Buscar una partida
GET	Mostrar_busqueda	"/buscar_partida"	Partida partida, BindingResult result, Model model, RedirectAttributes flash, SessionStatus status	Mostrar resultado de la búsqueda

4.6.2.3 Recursos de Sugerencias

Método	Nombre	URI	Parámetros	Descripción
GET	listar_sugerencias	"/listar_sugerencias"	Model model	Listar sugerencias
GET	sugerencias	"/sugerencias"	Model model	Vista de sugerencias
POST	guardar_sugerencia	"/guardar_sugerencia"	Sugerencias sugerencia, BindingResult result, Model model, RedirectAttributes flash, SessionStatus status	Guardar sugerencia
DELETE	eliminar_sugerencia	"/eliminar_sugerencia/{id}"	Long id, RedirectAttributes flash	Eliminar una sugerencia

4.6.2.4 Recursos de Mensaje

Método	Nombre	URI	Parámetros	Desc.
GET	mis_mensajes	"/mis_mensajes"	Model model, Authentication authentication	Listar mensajes
POST	guardar_mensaje	"/guardar_mensaje"	Mensaje mensaje, BindingResult result, Model model, RedirectAttributes flash, SessionStatus status	Guardar un mensaje
POST	enviar_mensaje	"/enviar_mensaje/{username_remitente}/{username_destinatario}"	String username_remitente, String username_destinatario, Model model, RedirectAttributes flash	Enviar un mensaje
DELETE	eliminar_mensaje	"/eliminar_mensaje/{id}"	Long id, RedirectAttributes flash	Eliminar un mensaje

5 INTERFAZ DE USUARIO

*El diseño no es sólo lo que ves,
sino como funciona.
- Steve Jobs -*

En esta sección mostraremos en funcionamiento cada una de las funcionalidades que fueron mencionadas anteriormente en el apartado 4.2 de la sección 1 (“Funcionalidades de la aplicación”) de forma que cualquier usuario con o sin experiencia en el uso de aplicaciones web sea capaz de utilizarla y sacarle el máximo rendimiento posible. Se expondrá la interfaz de usuario, mostrando cada una de las vistas con las que cuenta la aplicación. Ésta es una de las partes más importantes ya que el usuario interactuará con cada una de las funcionalidades a través de ella.

Por lo tanto, uno de los objetivos más importantes a trabajar y conseguir en el desarrollo de una aplicación web es el hecho de obtener una interfaz atractiva y sencilla para el cliente. Una interfaz que anime al usuario a hacer uso de la aplicación.

5.1 Introducción a la interfaz de usuario

Cuando el usuario se dirige a la aplicación Web SportMeet a través de un navegador web, lo primero que obtendrá será una portada donde se describen los servicios que ofrece la aplicación a través de distintas secciones, como podemos observar en la Figura 5.1:

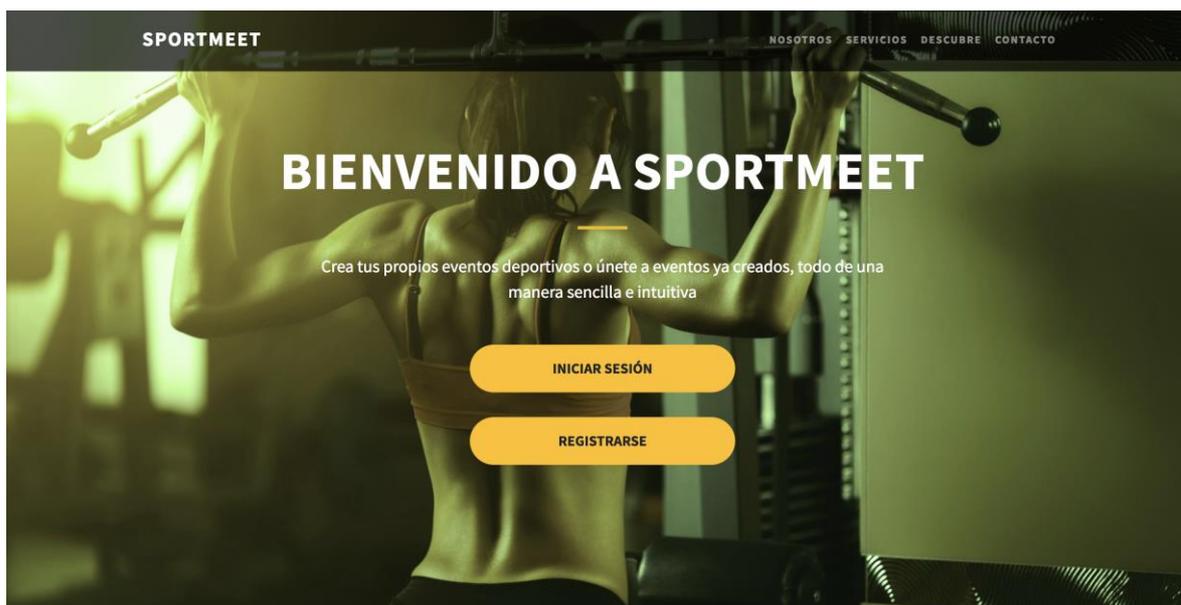


Figura 5.1: Portada de la aplicación

Si vamos deslizando la web hacia abajo podremos encontrar información sobre cada uno de las secciones definidas en la barra superior.

5.2 Registro de usuario

Lo primero que debemos hacer será registrar a un nuevo usuario en la aplicación. Para ello pulsamos en el botón “Registrarse”. Tras esta acción, la aplicación nos redirigirá a un formulario el cual deberemos cumplimentar con nuestros datos personales como observamos en la Figura 5.2:

- Nombre de usuario
- Nombre
- Apellidos
- Edad
- Email
- Contraseña
- Foto de perfil (opcional)

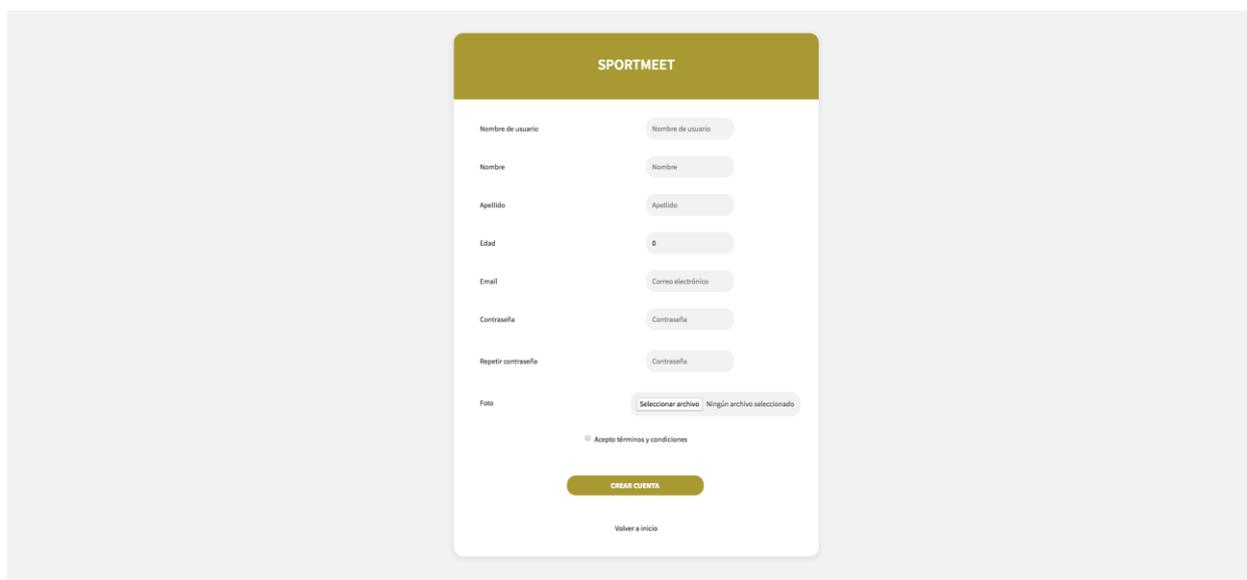


Figura 5.2: Registro de un nuevo usuario en la aplicación

Como estamos usando HTML5, nos aseguramos de que todos los campos son correctamente cumplimentados. Si algún campo no se completa, la aplicación nos mostrará el siguiente mensaje (Figura 5.3):

Apellido

Apellido

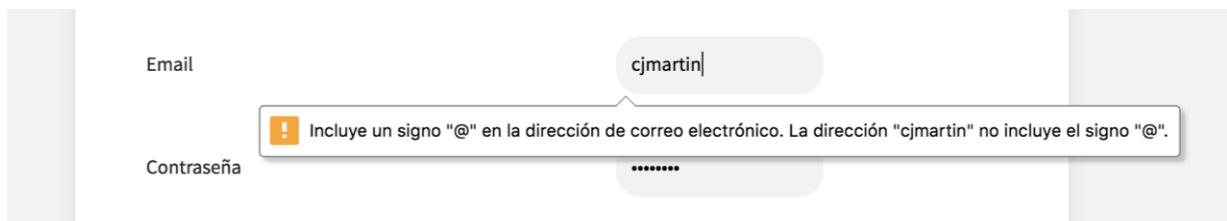
Edad

! Completa este campo

23

Figura 5.3: Aviso de campo sin rellenar en formulario de registro

Por otro lado también se comprueba que el correo se escriba en un formato correcto (Figura 5.4) y que las contraseñas introducidas sean las mismas (Figura 5.5):



The image shows a registration form with two input fields: 'Email' and 'Contraseña'. The 'Email' field contains the text 'cjmartin|'. Below the 'Email' field, a red error message box is displayed with a warning icon and the text: 'Incluye un signo "@" en la dirección de correo electrónico. La dirección "cjmartin" no incluye el signo "@'. The 'Contraseña' field contains a series of dots representing a password.

Figura 5.4: Email introducido incorrecto en formulario de registro



The image shows a registration form with a red error message box at the top that reads 'Las contraseñas no son iguales'. Below the error message, there is a green button with the text 'SPORTMEET'.

Figura 5.5: Contraseñas incorrectas en formulario de registro

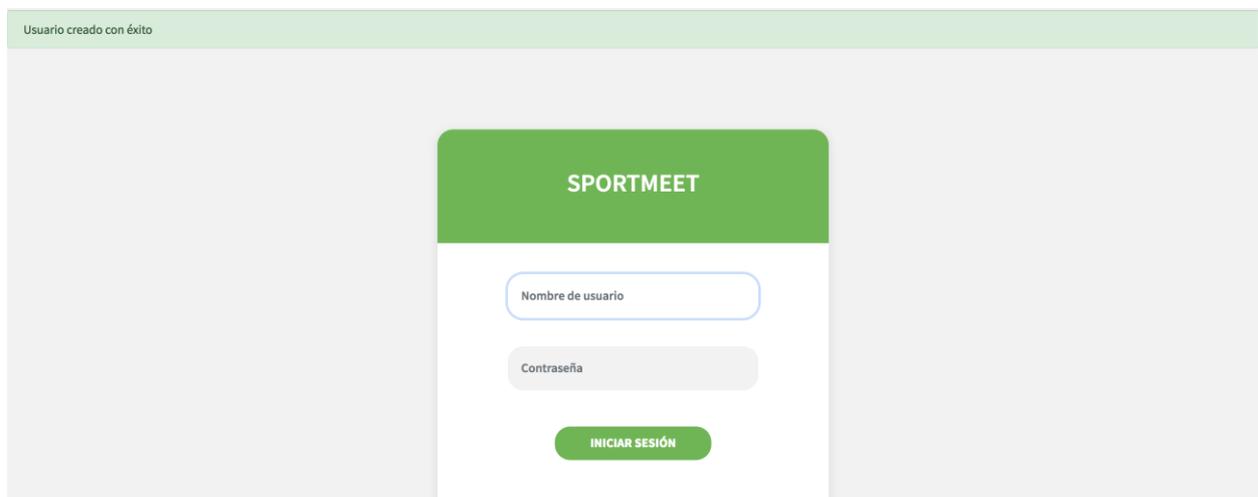
Por último, si el email introducido o el nombre de usuario introducidos están ya en uso, la aplicación nos mostrará el mensaje de error que podemos ver en la Figura 5.6 y no creará el nuevo usuario:



The image shows two separate red error message boxes. The first one contains the text 'El email ya está en uso'. The second one contains the text 'El username ya está en uso'.

Figura 5.6: Username o email en uso en formulario de registro

Finalmente, si el usuario cumple con todos los requisitos, se crea correctamente y la aplicación nos redirige a la opción de inicio de sesión (Figura 5.7):



The image shows a registration form with a green success message at the top that reads 'Usuario creado con éxito'. Below the success message, there is a green button with the text 'SPORTMEET'. Below the button, there are two input fields: 'Nombre de usuario' and 'Contraseña'. Below the input fields, there is a green button with the text 'INICIAR SESIÓN'.

Figura 5.7: Usuario creado con éxito

5.3 Inicio de sesión en la aplicación

Una vez hemos creado nuestra cuenta de usuario procedemos a iniciar sesión en la aplicación. Para ello pulsamos en el botón “Iniciar sesión” de la página principal.

A continuación introduciremos nuestro nombre de usuario y nuestra contraseña para realizar el proceso de autenticación e inicio de sesión (Figura 5.8).

En el caso de que introduzcamos correctamente los datos, la aplicación mostrará un error no permitiendo el inicio de sesión al usuario (Figura 5.9).

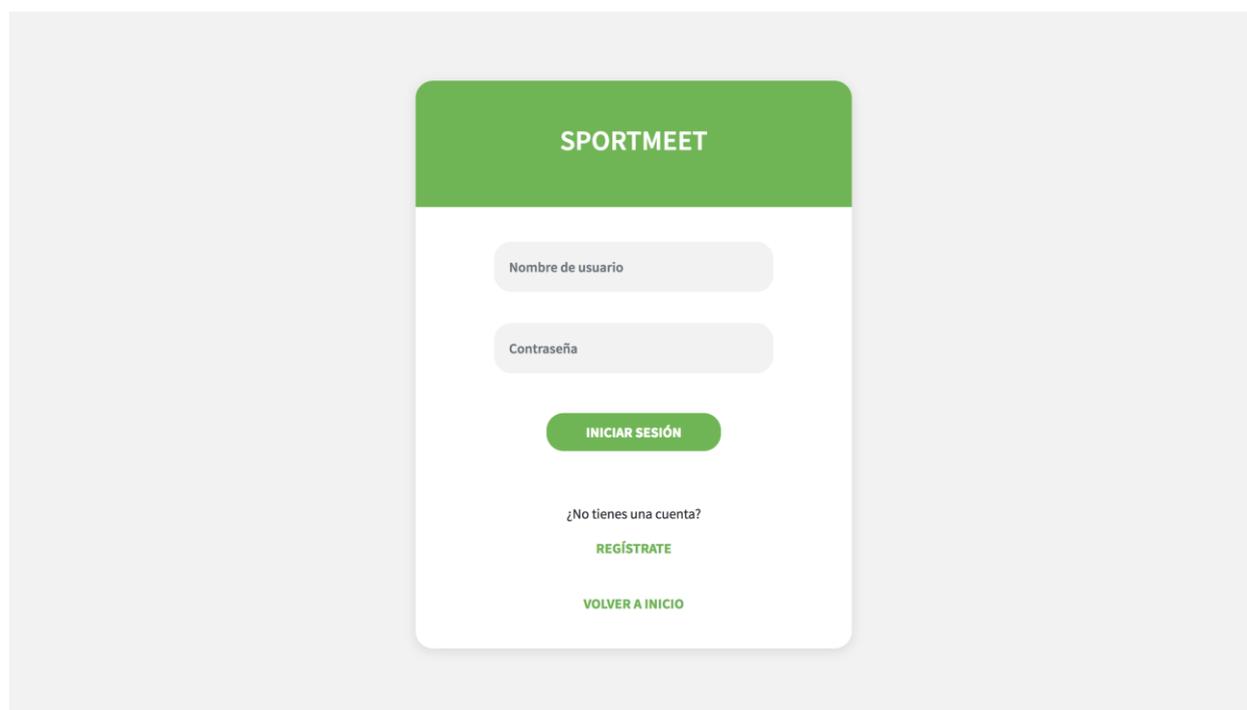


Figura 5.8: Inicio de sesión en la aplicación

Error en el login: Nombre de usuario o contraseña incorrecta, por favor vuelva a intentarlo!

Figura 5.9: Nombre de usuario o contraseña incorrectos en inicio de sesión

Si el usuario introduce los datos correctamente, el inicio de sesión será realizado con éxito, accediendo así a la página principal de la aplicación, como vemos en la Figura 5.10

Hola carlos, has iniciado sesión con éxito

Sportmeet

- Mi perfil
- Crear partida
- Buscar partida
- Mis partidas
- Mis mensajes
- Sugerencias
- Cerrar sesión

Bienvenid@ carlos

Próximas partidas **3**

Deporte	Ciudad	Hora de comienzo
Volleyball	Algeciras, Cádiz	10:00:00
Fútbol	San Roque, Cádiz	16:00:00
Tenis	Brenes, Sevilla	18:00:00

Sugerencias

Escribe tu sugerencia aquí...

Enviar

Estadísticas

3 Mis partidas	1 Partidas creadas	2 Partidas unidas
--------------------------	------------------------------	-----------------------------

Mis mensajes **1**

Creador	Mensaje	Responder
andres	¿Cuánto dinero tenemos que llevar?	responder

Figura 5.10: Inicio de sesión con éxito en la aplicación

5.4 Página inicial de la aplicación

Cuando el usuario ha iniciado sesión, tendrá acceso a la página inicial de la aplicación. Desde aquí podrá visualizar un menú a su izquierda con las distintas funcionalidades que se ofrecen, como podemos observar en la Figura 5.11:

Sportmeet

- Mi perfil
- Crear partida
- Buscar partida
- Mis partidas
- Mis mensajes
- Sugerencias
- Cerrar sesión

Bienvenid@ carlos

Próximas partidas **3**

Deporte	Ciudad	Hora de comienzo
Volleyball	Algeciras, Cádiz	10:00:00
Fútbol	San Roque, Cádiz	16:00:00
Tenis	Brenes, Sevilla	18:00:00

Sugerencias

Escribe tu sugerencia aquí...

Enviar

Estadísticas

3 Mis partidas	1 Partidas creadas	2 Partidas unidas
--------------------------	------------------------------	-----------------------------

Mis mensajes **1**

Creador	Mensaje	Responder
andres	¿Cuánto dinero tenemos que llevar?	responder

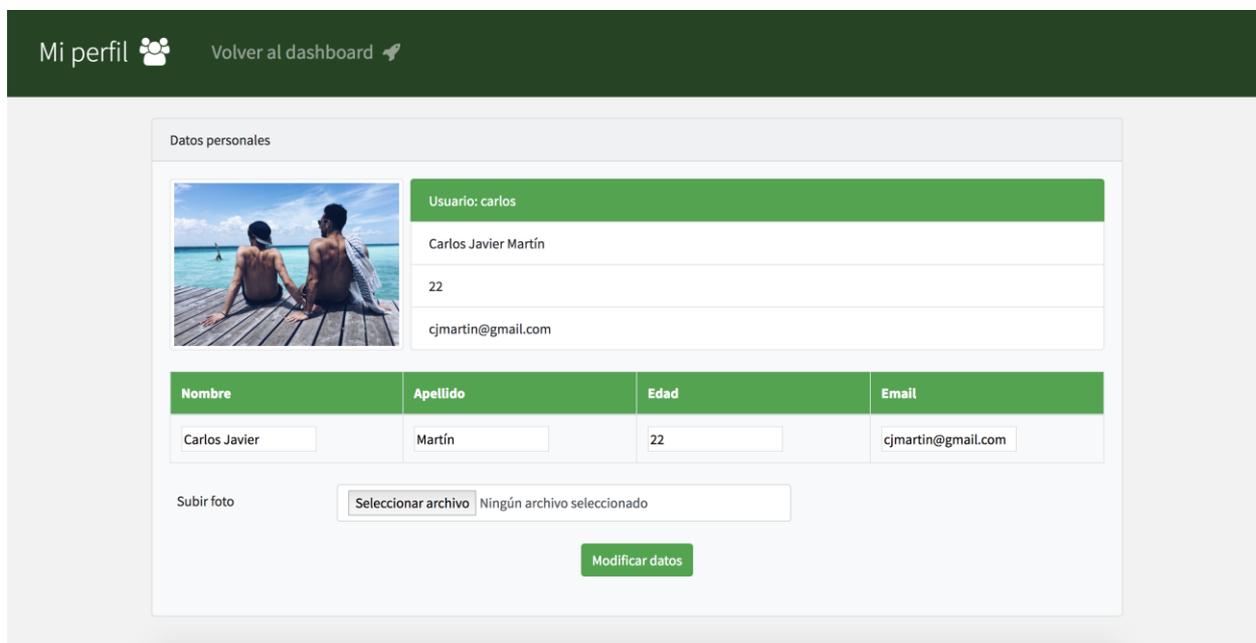
Figura 5.11: Página inicial de la aplicación

Desde aquí se pueden observar también distintas funciones rápidas accesibles fácilmente, como son:

- Próximas partidas: donde se muestra una lista con las siguientes partidas a disputar.
- Sugerencias: opción alternativa para escribir una sugerencia de una forma más rápida.
- Estadísticas: muestra información sobre las partidas totales en las que estamos inscritos, las partidas que hemos creado y las partidas a las que nos hemos unido.
- Mis mensajes: muestra los últimos mensajes recibidos y ofrece la posibilidad de responder a los mismos.

5.5 Mi perfil

Una de las funcionalidades que ofrece la aplicación es la opción “Mi perfil”. Desde aquí los usuarios podrán gestionar y modificar sus datos personales con total libertad (Figura 5.12):



Mi perfil  [Volver al dashboard](#) 

Datos personales



Usuario: carlos

Carlos Javier Martín

22

cjmartin@gmail.com

Nombre	Apellido	Edad	Email
Carlos Javier	Martín	22	cjmartin@gmail.com

Subir foto Ningún archivo seleccionado

Figura 5.12: Funcionalidad “Mi perfil”

5.6 Crear partida

Desde este menú el usuario tendrá la posibilidad de crear sus nuevos eventos o partidas de una forma totalmente personalizable, ya que podrá establecer distintos parámetros como son el deporte, la ciudad, la fecha, la hora, los participantes, etc. Finalmente, basta con pulsar en el botón “Crear partida” para que ésta esté disponible para el resto de los usuarios (Figura 5.13):

Sportmeet · Crear Partida

« Mis partidas

Creador

Deporte

Ciudad

Provincia

Fecha

Hora de comienzo

Hora de finalización

Participantes

Suplentes

Descripción

Figura 5.13: Funcionalidad “Crear partida”

5.7 Buscar partidas

Esta funcionalidad permite buscar partidas a través de determinados filtros (deporte, ciudad, provincia) y su posterior unión a aquellas que cumplan los requisitos de búsqueda. Para buscar una partida basta con rellenar alguno (o varios) de los campos de filtrado y pulsar en el botón “Buscar partida”. Posteriormente aparecerán las partidas disponibles que cumplen los criterios de búsqueda. Para unirse a una partida es necesario hacer click en la opción “Unirse” (Figura 5.14):

Buscar partida

Sportmeet · Buscar Partida

« Mis partidas

Deporte

Ciudad

Provincia

Resultados de búsqueda

Sportmeet · Resultados de la búsqueda

Creador	Deporte	Ciudad	Fecha	Hora de comienzo	Participantes	Comentarios	Eliminar
carlos	Fútbol	San Roque, Cádiz	2018-07-06	16:00:00	10	partida	<input type="button" value="unirse"/>

Figura 5.14: Funcionalidad “Buscar partida”

Si la unión a la partida se realiza correctamente, obtendremos el siguiente mensaje de éxito (Figura 5.15):

Te has unido a la partida!

Figura 5.15: Unión con éxito a una partida

En cambio, si la partida está completa y no puede admitir a más participantes, obtendremos el siguiente mensaje de error (Figura 5.16):

La partida está completa!

Figura 5.16: Partida completa

Finalmente, si intentamos unirnos a una partida a la que ya pertenecemos, obtenemos el siguiente mensaje (Figura 5.17):

Ya eres miembro de la partida

Figura 5.17: Participante existente en la partida

5.8 Mis partidas

Para gestionar todas las partidas a las que nos hemos unido y todas las partidas que hemos creado, existe la funcionalidad “Mis partidas”. Desde esta opción podremos tener un resumen con todas las partidas en las que estamos inmersos.

Los usuarios podrán eliminar las partidas creadas por ellos mismos, o bien, abandonar u obtener información acerca de las partidas a las que se han unido (Figura 5.18):

Mis partidas [Volver al dashboard](#)

Mis partidas

[Crear partida](#)

Partida	Deporte	Ciudad	Fecha	Hora de comienzo	Participantes	Eliminar
1	Fútbol	San Roque, Cádiz	2018-07-06	16:00:00	2/10	eliminar

Partidas a las que me he unido

Creador	Deporte	Ciudad	Fecha	Hora de comienzo	Participantes	Abandonar	Información
andres	Volleyball	Algeciras, Cádiz	2018-07-14	10:00:00	2/6	abandonar	+info
andres	Tennis	Brenes, Sevilla	2018-07-12	18:00:00	2/2	abandonar	+info

Figura 5.18: Funcionalidad “Mis partidas”

Si pulsamos en eliminar partida, se nos mostrará un mensaje de confirmación para eliminarla definitivamente (Figura 5.19) y posteriormente un mensaje que confirma su eliminación, como vemos en la Figura 5.20:

localhost:8080 dice

¿Esta seguro que desea eliminar la partida?

Cancelar

Aceptar

Figura 5.19: Borrado de una partida

Partida eliminada con éxito!

Figura 5.20: Confirmación de borrado de una partida

Por otro lado, si queremos abandonar una partida, el proceso será el mismo pero esta vez pulsando en el botón “abandonar” (Figura 5.21, Figura 5.22):

localhost:8080 dice

¿Esta seguro que desea abandonar la partida?

Cancelar

Aceptar

Figura 5.21: Abandono de una partida

Partida abandonada con éxito!

Figura 5.22: Confirmación de abandono de una partida

Finalmente, si queremos obtener más información sobre una partida a la que nos hemos unido bastará con pulsar en la opción “+ info” (Figura 5.23):

Información de la partida  [Volver al dashboard](#) 

Creador	Deporte	Ciudad	Fecha	Comienzo	Final	Participantes	Suplentes
carlos	Fútbol	San Roque, Cádiz	2018-07-06	19:00:00	20:00:00	2/10	0

Comentarios

Partido de fútbol sala en Manolo Mesa.
Camisetas azules y rojas.
1€ por persona.

Figura 5.23: Información de la partida

5.9 Mis mensajes

Gracias a esta funcionalidad los usuarios podrán ponerse en contacto con los demás para aclarar cualquier tipo de dudas sobre los eventos, proponer nuevos eventos o cualquier tema de conversación que se les pueda ocurrir.

Este apartado está dividido en tres subapartados a su vez, como podemos comprobar en la Figura 5.24:

- Mis mensajes: donde se muestran los mensajes recibidos por el resto de usuarios.
- Partidas a las que me he unido: de esta forma podremos escribir directamente un mensaje al creador de alguno de los eventos a los que estamos unidos.
- Enviar mensaje: podemos también enviar un mensaje a cualquier usuario a través de su nombre de usuario.

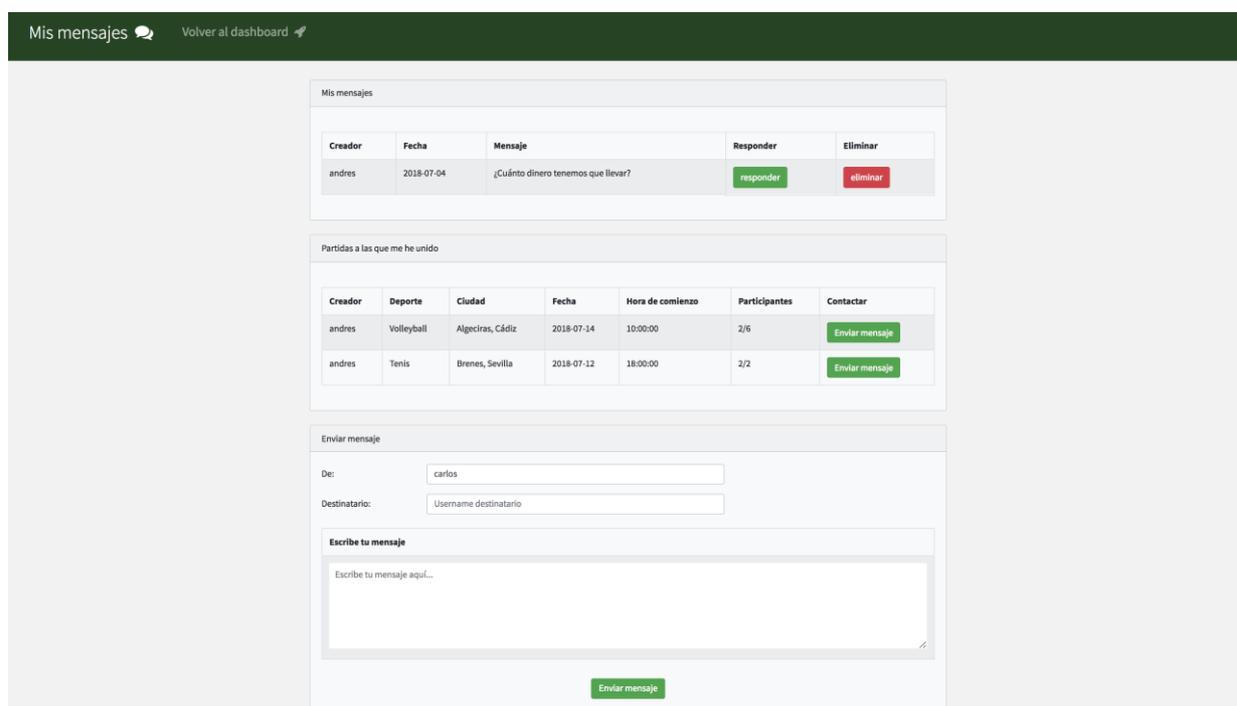


Figura 5.24: Funcionalidad “Mis mensajes”

Para enviar un mensaje desde el primer apartado “Mis mensajes”, es tan sencillo como pulsar el botón “responder”. De esta forma la aplicación nos redirigirá a un formulario desde el que podremos escribir el mensaje deseado al usuario de destino (Figura 5.25):

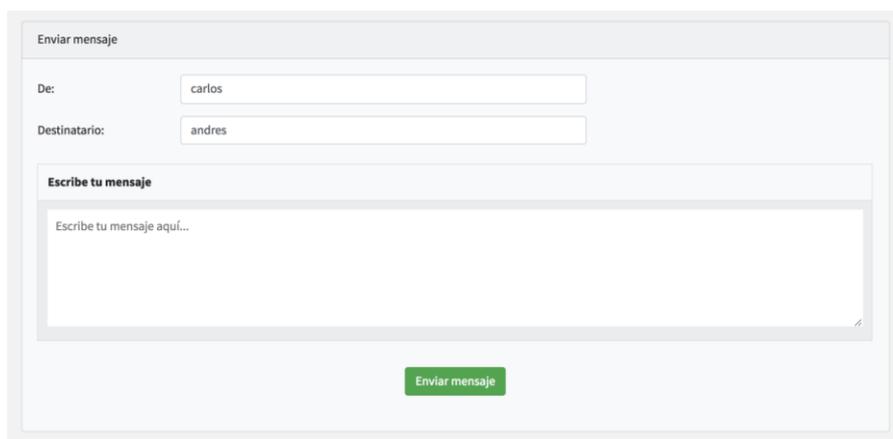


Figura 5.25: Enviar un mensaje

Cuando hayamos acabado de escribir nuestro mensaje, pulsaremos en el botón “Enviar mensaje” y si todo ha funcionado correctamente, recibiremos el siguiente mensaje de confirmación (Figura 5.26):

A green rectangular notification box with the text "Mensaje enviado con éxito" in a dark font.

Figura 5.26: Confirmación de mensaje enviado

Si queremos mandar un mensaje directamente a un creador de una partida a la que estamos unidos, es tan fácil como pulsar en el botón “Enviar mensaje” de la partida correspondiente en la sección “Partidas a las que me he unido”, redirigiéndonos al mismo formulario que antes.

Finalmente, la última opción es mandar un mensaje a cualquier usuario siempre y cuando conozcamos su nombre de usuario desde el subapartado “Enviar mensaje”.

Por otro lado, también existe la posibilidad de eliminar los mensajes recibidos. Para ello pulsaremos en el botón “Eliminar”. Acto seguido aparecerá un menú emergente que nos pide confirmación para borrar definitivamente el mensaje (Figura 5.27). Si pulsamos en aceptar, aparecerá un mensaje de confirmación de que el mensaje se ha eliminado correctamente (Figura 5.28):

localhost:8080 dice

¿Estás seguro de que deseas eliminar el mensaje?

Cancelar

Aceptar

Figura 5.27: Borrado de un mensaje



Figura 5.28: Confirmación de mensaje eliminado

5.10 Sugerencias

SportMeet es una aplicación cuyo objetivo principal es facilitar y mejorar la calidad de vida de sus usuarios. Es por ello que posee una funcionalidad para que estos puedan expresar su opinión respecto a la aplicación y proponer mejoras futuras que serán leídas por el administrador.

Para proponer una nueva sugerencia, solo hay que pulsar en la sección “Sugerencias”, escribir nuestra sugerencia y enviarla pulsando en “Enviar” (Figura 5.29):

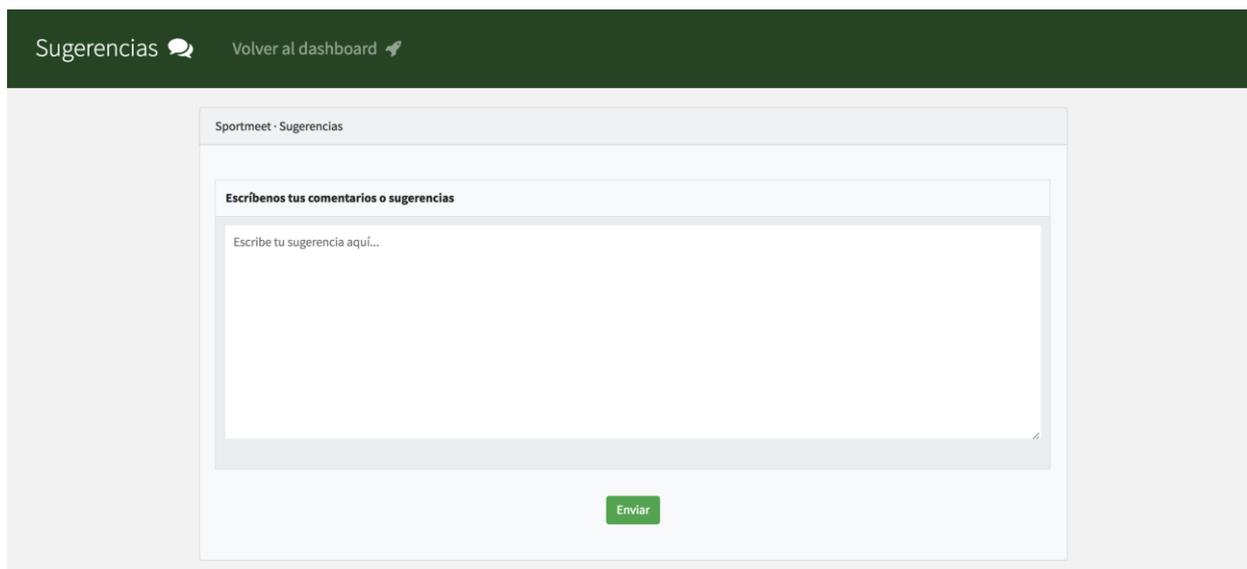


Figura 5.29: Funcionalidad “Sugerencia”

Cuando un usuario envía una sugerencia, recibe un mensaje de éxito para saber que su sugerencia ha sido enviada con éxito (Figura 5.30):

Sugerencia añadida con éxito. Gracias por ayudarnos a mejorar ;)

Figura 5.30: Confirmación de sugerencia enviada

5.11 Funciones del administrador

Existe la posibilidad de iniciar sesión como administrador de la aplicación. Este administrador tendrá acceso a las mismas funcionalidades que los usuarios normales, y además a varias funcionalidades extra como la gestión de usuario, gestión de partidas y visualización de sugerencias (Figura 5.31):

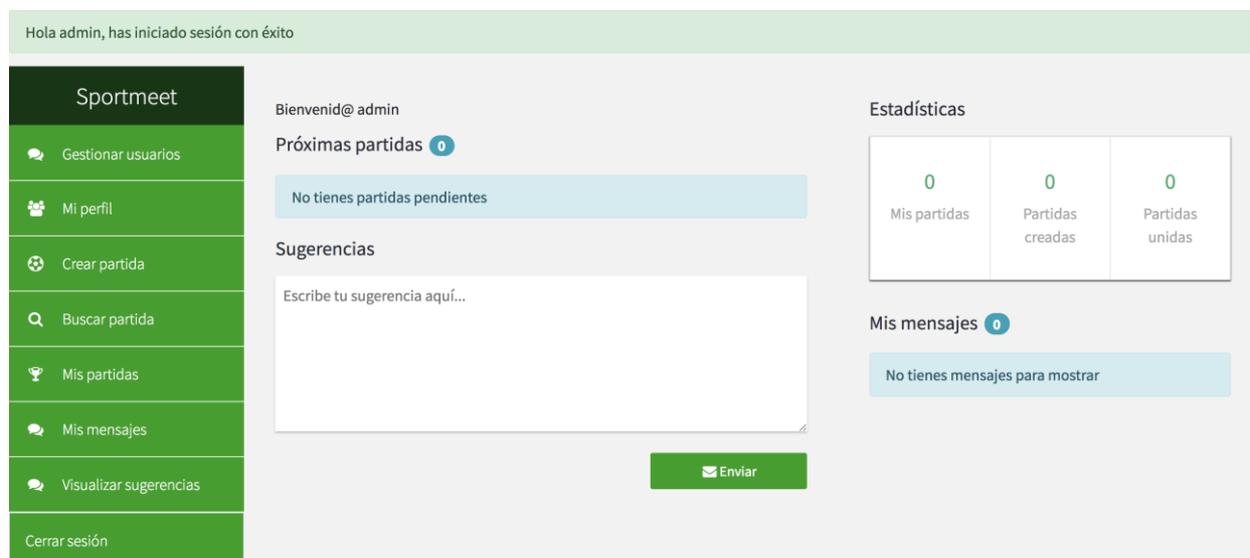
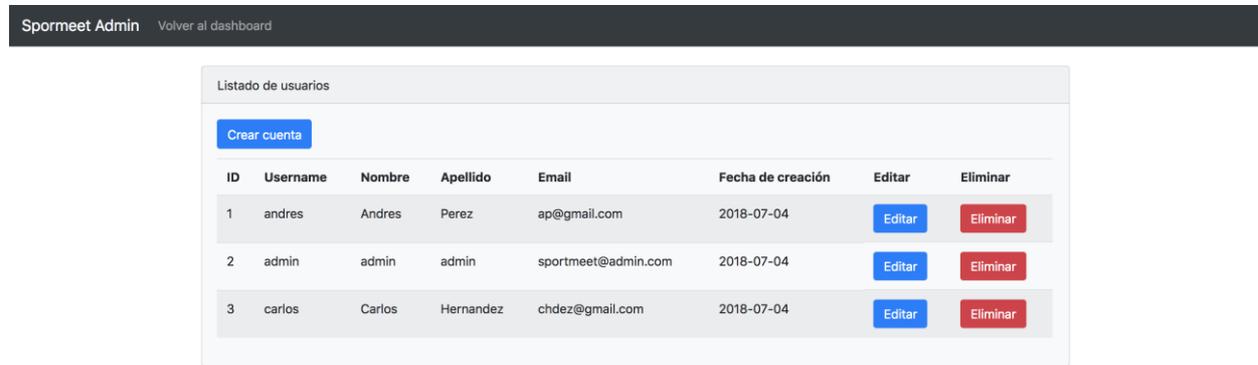


Figura 5.31: Interfaz de usuario del administrador

Si pulsamos en la opción “Gestionar usuarios”, tendremos acceso a una lista con todos los usuarios existentes en la aplicación, y podremos modificar e incluso eliminarlos si alguno de ellos lleva a cabo un mal uso de la aplicación. También existe la posibilidad de crear una nueva cuenta desde el botón “Crear cuenta” (Figura 5.32):



Spormeet Admin [Volver al dashboard](#)

Listado de usuarios

[Crear cuenta](#)

ID	Username	Nombre	Apellido	Email	Fecha de creación	Editar	Eliminar
1	andres	Andres	Perez	ap@gmail.com	2018-07-04	Editar	Eliminar
2	admin	admin	admin	sportmeet@admin.com	2018-07-04	Editar	Eliminar
3	carlos	Carlos	Hernandez	chdez@gmail.com	2018-07-04	Editar	Eliminar

Figura 5.32: Gestión de usuarios (modo administrador)

Por otro lado, si pulsamos en la opción “Visualizar sugerencias” podremos observar todas las sugerencias creadas por los usuarios, clasificadas por su fecha de creación. Estas sugerencias se muestran de forma anónima para preservar la intimidad de los usuarios. Gracias a esta sección el administrador puede saber qué piensan los usuarios de su aplicación y qué aspectos se pueden mejorar (Figura 5.33):



Spormeet Admin [Volver al dashboard](#)

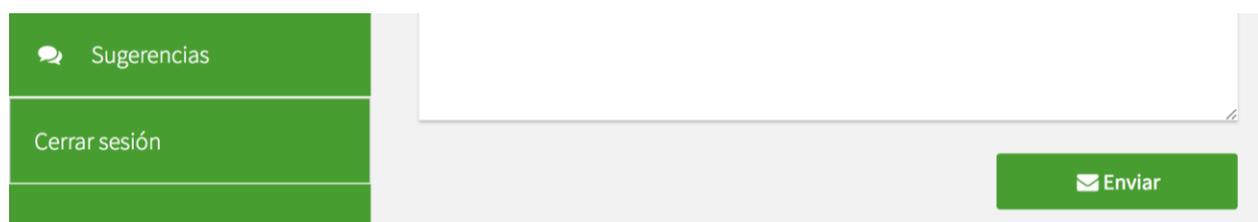
Sportmeet · Sugerencias

Fecha de creación	Descripción	Eliminar
2018-07-04	Me gustaría que añadiráis una nueva sección para compartir fotos de las partidas. Gracias.	Eliminar
2018-07-05	Un tal usuario llamado sensoker está haciendo un uso inadecuado de la aplicación. Rogaría que le echarais un vistazo. Gracias.	Eliminar

Figura 5.33: Visualización de sugerencias (modo administrador)

5.12 Cerrar sesión

Finalmente, el usuario dispondrá de la opción “Cerrar sesión” la cual le permite cerrar su sesión de una forma rápida y segura. Para ello, bastará con pulsar el botón “Cerrar sesión” y la aplicación nos redirigirá a la portada de la aplicación (Figura 5.34):



Sugerencias

Cerrar sesión

[Enviar](#)

Figura 5.34: Cierre de sesión de usuario

6 LÍNEAS DE MEJORA Y CONCLUSIONES

*Después de escalar una montaña muy alta,
descubrimos que hay muchas otras montañas por escalar.
- Nelson Mandela -*

Finalmente, una vez expuestos y desarrollados los conceptos en los que se fundamenta el presente proyecto, es turno de valorar el trabajo y el esfuerzo realizado y los resultados que se han obtenido. A continuación detallaremos qué conclusiones se han deducido durante la realización de la aplicación y cómo ha sido el proceso de aprendizaje ante la inmersión en un nuevo mundo de tecnologías, así como las dificultades encontradas durante su desarrollo.

Por otro lado, estudiaremos las posibles líneas de mejora a implementar en futuras versiones con el objetivo de generar un producto más completo y que se adapte a todas las expectativas y necesidades de los usuarios, al mismo tiempo que resulte llamativo y cómodo a la hora de su uso.

6.1 Líneas de mejora

A continuación se detallan las posibles líneas de mejora a seguir e implementar en el proyecto con el objetivo de adaptar el producto final a las expectativas y la demanda del mercado actual por parte de los usuarios.

6.1.1 Mejoras de funcionalidad de la aplicación Web

En cuanto a las actuales funcionalidades de la aplicación, se enumera una lista de cambios o mejoras que podrían aportar una serie de resultados interesantes:

- **Añadir amigos:** con el objetivo de convertir la aplicación totalmente en una red social, sería interesante ofrecer la posibilidad de añadir amigos, de forma que tengamos un contacto directo con nuevos usuarios que conozcamos a través de la aplicación o que ya sean conocidos por nosotros. De esta forma será posible compartir nuestros perfiles públicamente o aceptar su accesibilidad por parte de otros usuarios a través de peticiones de amistad.
- **Sección “sobre mí”:** esta función se plantea con el objetivo de que los usuarios puedan compartir con los demás información extra acerca de sus propios intereses. Gracias a esto, los usuarios podrán conocer qué personas son más afines a sus gustos y ponerse en contacto para organizar eventos en común.
- **Compartir imágenes sobre las partidas:** otra de las posibles funcionalidades a implementar y que convertiría el uso de la aplicación en algo más atractivo sería la inclusión de imágenes que describan cómo se ha desarrollado la partida. Gracias a esta funcionalidad los usuarios podrán valorar el evento en el que han participado y compartir imágenes del mismo con el resto de usuarios, de forma que existan referencias para eventos posteriores.

- **Añadir sistema de notificación y alarmas:** actualmente, cada evento importante se muestra en la página inicial de la aplicación de forma que el usuario pueda informarse lo antes posible de los cambios que se producen. Aún así, sería interesante ofrecer al usuario la posibilidad de ser notificado ante cualquier tipo de evento (ya sean partidas completas, abandono de usuarios o mensajes recibidos) sin necesidad de haber iniciado sesión en la aplicación, por ejemplo vía email, siempre y cuando él lo desee y lo indique. De esta forma, los usuarios estarán actualizados en cada momento sobre lo que está pasando en la aplicación.
- **Añadir sistema de valoración de participantes:** se estudia la posibilidad de añadir un sistema de evaluación de los participantes por parte del creador de una partida. De esta forma tratamos de evitar problemas de abandonos reiterados o la inclusión de usuarios conflictivos. Así, podemos saber si un participante finalmente no ha asistido a un gran número de partidas o si su comportamiento durante las mismas no es el adecuado, y añadir un cierto filtro a nuestros propios eventos.

6.1.2 Adaptación a otros entornos

Uno de los objetivos de SportMeet es estar disponible para todos los usuarios desde cualquier lugar y en cualquier momento, pero hasta ahora solo está disponible su versión para aplicación Web. Este hecho hace que la aplicación sea solo accesible desde la red, necesitando por lo tanto un navegador web para acceder a la misma.

El diseño se ha llevado a cabo con el framework Bootstrap y se ha adaptado para que sea “responsive”, es decir, para que se adapte a todas las plataformas y tamaños posibles. Aún así, sería interesante llevar a cabo el desarrollo de la aplicación para otros tipos de dispositivos y sistemas operativos, como es el caso de Android o iOS. De esta forma obtendremos una interfaz más dinámica y amigable para el usuario, que podrá adaptar la aplicación Web a cualquier dispositivo desde el que quiera acceder.

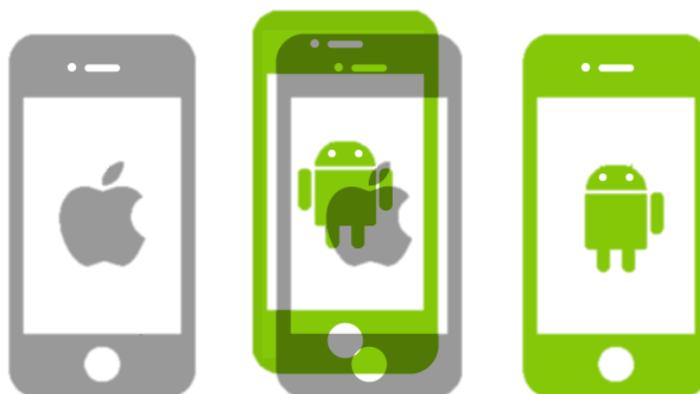


Figura 6.1: Sistemas operativos iOS y Android

Además, hoy en día la mayoría de los usuarios suelen descargar las aplicaciones en sus teléfonos móviles. Esto quiere decir que desarrollando la aplicación para sistemas Android e iOS (Figura 6.1) se conseguiría alcanzar a un mayor número de usuarios, extendiendo así la aplicación de una forma más óptima.

6.1.3 Implementación con la API de Youtube

A la hora de crear una partida, sería interesante ofrecer la posibilidad de añadir un vídeo sobre la actividad deportiva que se va a desarrollar. De esta forma, aquellos usuarios que no conozcan al creador de la partida o bien no hayan practicado nunca una actividad concreta pueden tener una primera impresión sobre el evento, de forma que les ayude a tomar una decisión.

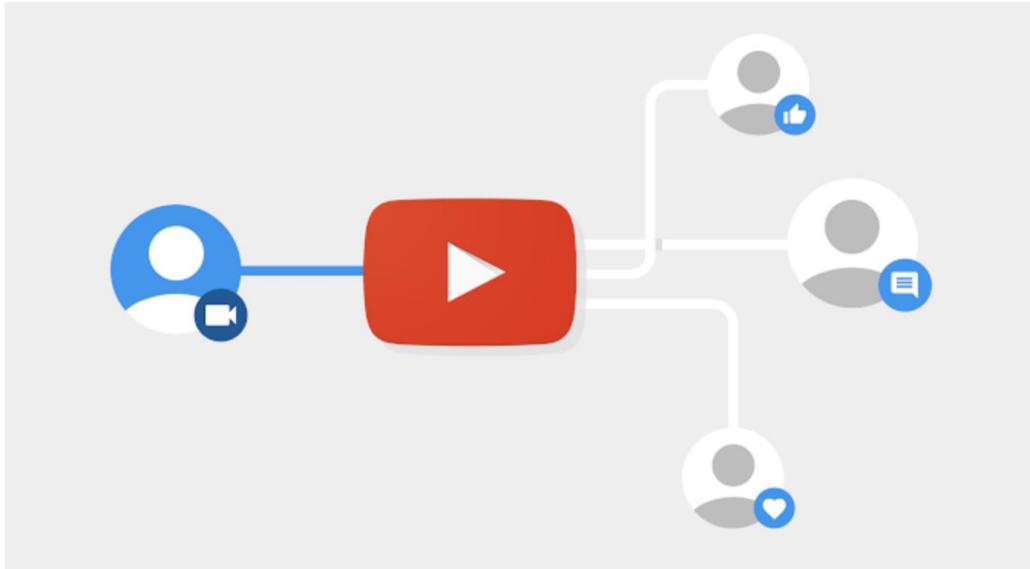


Figura 6.2: API de Youtube

Para ello se podría hacer uso de la API de Youtube, la cual ofrece distintas funcionalidades a los usuarios para gestionar sus videos y su canal, como se puede observar en la Figura 6.2. La idea final sería la implementación de la API de Youtube en la aplicación para que los usuarios puedan compartir sus propios videos en la descripción de la partida y compartir la experiencia.

6.1.4 Implementación con mapas

La idea inicial era la implementación de la API de Google Maps en nuestro proyecto, pero desde hace unos meses esta es de pago para desarrolladores.



Figura 6.3: Logo de “Here Technologies”

Aún así, sería interesante buscar opciones alternativas a Google Maps, como por ejemplo la mostrada en la Figura 6.3, ya que el objetivo es que los usuarios puedan establecer el lugar de encuentro o ubicación del evento directamente desde un mapa a la hora de crear la partida. De esta forma el resto de participantes podrán buscar las mejores rutas hacia el evento y situarlo mucho mejor directamente desde la aplicación sin necesidad de otras páginas web o ayuda de terceros.

6.1.5 Spring Security

Spring Security es uno de los frameworks de seguridad J2EE más populares y completos actualmente en el mercado y es usado por todo tipo de instituciones. El objetivo principal de añadirlo a nuestro proyecto es el de aportar más seguridad a la aplicación. En nuestro caso, gracias a este módulo de Spring, llevamos a cabo la autenticación de los usuarios en la aplicación y la distribución de permisos de acceso a determinadas acciones o funcionalidades en función de los roles asignados (autorización), usando a su vez el motor de plantillas Thymeleaf el cual nos ha facilitado la perfecta implementación de la seguridad en nuestros ficheros HTML.

Como podemos observar, los campos en los que se divide un JWT son los siguientes:

- Header: corresponde a la primera cadena. Contiene el tipo y el algoritmo de codificación usado.
- Payload: corresponde a la segunda cadena. Algunos atributos importantes son:
 - sub: identifica al sujeto del token
 - iat: identifica la fecha de creación del token
 - exp: identifica la fecha de expiración del token
- Signature: corresponde a la tercera parte de la cadena. Contiene la firma, formada por los dos componentes anteriores cifrados en Base64 con una clave secreta.

En resumen, se considera como objetivo la implementación de este modelo de autenticación para futuras versiones debido a las grandes prestaciones que ofrece y al aumento de la seguridad que supone en nuestra aplicación, tanto a nivel de autenticación como a nivel de autorización.

6.1.7 Implementación de AngularJS

Para el desarrollo de la aplicación se ha hecho uso del framework de frontend Bootstrap, el cual es un framework CSS que facilita la creación de la interfaz de usuario de las aplicaciones web y que usa librerías JavaScript que dependen de JQuery.

Es por ello que se ha hecho uso de JQuery, una biblioteca multiplataforma de JavaScript que permite simplificar la interacción con los documentos HTML, manipular el fichero DOM o manejar eventos. Es decir, nos permite acceder y modificar el estado de cualquiera de los elementos de la página web. A través de él y de los selectores de CSS podemos llegar a los distintos elementos y modificarlos libremente, a la vez que nos suscribimos a los eventos que puedan ocurrir en los mismos.

Por otro lado, AngularJS (Figura 6.6) pasa de ser una librería a considerarse un framework de aplicaciones web. Nos ofrece un gran número de funcionalidades como mecanismos de acceso y modificación de elementos, ahorro de líneas de código JavaScript y mecanismos de extensión del HTML convirtiéndolos en ficheros más semánticos. Pero el principal motivo por el que es importante considerar su implementación es que AngularJS nos marca una serie de pautas y normas a seguir en el ámbito de la programación, centrándose principalmente en el patrón MVC (modelo-vista-controlador).



Figura 6.6: Logo de AngularJS

Estas dos tecnologías son diferentes pero no son mutuamente excluyentes, aunque es cierto que AngularJS va más allá. En el actual diseño de la aplicación, el uso de JQuery es más que válido ya que los requisitos son sencillos. Sin embargo, si se pretende seguir trabajando en el desarrollo de la aplicación aumentando la carga de JavaScript es recomendable dar el paso a AngularJS por las ventajas anteriormente descritas.

6.2 Conclusiones

Tras el desarrollo de la aplicación Web se han obtenido una serie de conclusiones que pueden ser utilizadas de cara al futuro en nuevos proyectos de forma que se consiga corregir los errores cometidos y optimizar su evolución.

Por un lado, el hecho de diseñar y desarrollar una aplicación Web por primera vez desde cero y con tecnologías totalmente nuevas significa un gran reto. Durante el proyecto ha sido necesaria de forma constante la búsqueda de información acerca de estas tecnologías y de ejemplos de uso para saber cómo ponerlas en práctica. Aún así, resulta bastante satisfactorio realizar un proyecto de tal envergadura debido a la cantidad de conceptos y tecnologías que se aprenden a manejar, como ha sido el caso de frameworks como Spring o Bootstrap, frameworks bastante potentes y que simplifican de una manera considerable la programación web.

Por otro lado, la consecución del objetivo inicial ha supuesto un continuo enfrentamiento ante problemas y fallos de programación de diversos tipos, originados en su mayor parte por una falta de entendimiento de los conceptos. Sin embargo, se suele decir que de los fallos se aprende y en este caso no ha sido menos, ya que gracias a ellos se ha conseguido entender sus orígenes y cómo solventarlos para futuros diseños.

Otro de los puntos importantes que se puede deducir es la trascendencia de la toma de requisitos antes de comenzar a desarrollar la aplicación. Es necesario hacer un estudio previo de qué requisitos debe cumplir nuestro proyecto y qué soluciones deben implementarse para obtener los resultados esperados. De esta forma, conseguiremos disminuir en gran medida el tiempo empleado en la producción de la aplicación web, evitando a su vez cambios de giro bruscos en cuanto a patrones de diseño o tecnologías se refiere.

Finalmente, en cuanto al resultado final de la aplicación soy consciente de que hay bastantes aspectos que mejorar para obtener un producto más realista y adecuado a los actuales requisitos del mercado como hemos expuesto en el apartado 6.1 del presente documento. Aún así, creo que se ha conseguido desarrollar un producto competente en función al tiempo y el esfuerzo dedicado, con una interfaz sencilla, intuitiva y atractiva y que puede resolver uno de los problemas con el que se encuentran los usuarios deportistas en la sociedad actual.

Una de las razones por las que decidí empezar a estudiar el Grado en Ingeniería de las Tecnologías de Telecomunicación es la posibilidad de mejorar el estado actual de las cosas, aumentando por consecuente el nivel de calidad de vida de la sociedad. Considero que la existencia de aplicaciones Web como SportMeet permite mejorar la condición física y las relaciones sociales de los usuarios, consiguiendo así el objetivo principal por el que me propuse empezar a estudiar esta carrera.

Como última conclusión, gracias a este proyecto queda reflejado el fruto de la unión de mis principales pasiones como son las tecnologías de la información, las relaciones sociales y el deporte, ámbitos totalmente diferentes pero que considero claves en el día a día y en la evolución de la sociedad.

REFERENCIAS

Sección “Tecnologías utilizadas”:

- [1] [Documentación de Spring Framework, STS y módulos](#)
- [2] [Características de Spring Framework](#)
- [3] [JDBC y JPA: diferencias](#)
- [4] [Funcionamiento de Maven](#)
- [5] [Documentación sobre Thymeleaf](#)
- [6] [Patrón de diseño MVC](#)
- [7] [Web oficial de Bootstrap](#)
- [8] [Guía de W3schools para HTML5](#)
- [9] [Guía de W3schools para CSS3](#)
- [10] [Documentación de la API de JQuery](#)
- [11] [Funcionamiento de JavaScript](#)
- [12] [Modelo relacional](#)
- [13] [Web oficial de MySQL \(y documentación de MySQLWorkBench\)](#)

Sección “Herramientas utilizadas”:

- [14] [Características de Github](#)
- [15] [Características de Google Chrome](#)
- [16] [Web oficial de StarUML](#)

Sección “Servidor de SportMeet”

- [17] Apuntes de la asignatura “Ingeniería del Software”
- [18] [Esquema para los diagramas de casos de uso](#)
- [19] Apuntes de la asignatura “Diseño de Bases de Datos”
- [20] [Modelo E-R para bases de datos](#)

Sección “Lineas de mejora y conclusiones”:

- [21] [Documentación sobre Json Web Tokens](#)
- [22] [Web oficial de AngularJS](#)

ANEXO A: MANUAL DE INSTALACIÓN Y DESPLIEGUE DE LA APLICACIÓN

En este anexo se describen los pasos a seguir para la configuración y puesta a punto del entorno necesario para el correcto funcionamiento de la aplicación web.

Todo este procedimiento se ha llevado a cabo en un ordenador con sistema operativo **macOS High Sierra versión 10.13.3**

A.1 Instalación de las herramientas necesarias

A continuación se detalla el proceso de instalación de las herramientas necesarias. Para el correcto desarrollo de este proyecto son necesarias las siguientes herramientas:

- JAVA SE Development Kit 8 (JDK 8)
- Spring Tool Suite
- Maven
- MySQLWorkBench

A.1.1 JAVA SE Development Kit 8

Para el correcto funcionamiento de nuestro framework Spring Tool Suite, antes es necesario tener instalado el kit de herramientas de desarrollo de Java en nuestro sistema operativo. Si aún no lo tenemos instalado, hacemos click en el siguiente enlace para descargarlo:

[Descarga de JDK8](#)

Una vez dentro, aceptamos el acuerdo de licencia y descargamos la versión que se adapte a los requisitos de nuestro sistema operativo. En nuestro caso descargaremos el fichero “MAC OS X x64”, como vemos en A.1:

Java SE Development Kit 8u171		
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.		
<input checked="" type="radio"/> Accept License Agreement <input type="radio"/> Decline License Agreement		
Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.97 MB	jdk-8u171-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.89 MB	jdk-8u171-linux-arm64-vfp-hflt.tar.gz
Linux x86	170.05 MB	jdk-8u171-linux-i586.rpm
Linux x86	184.88 MB	jdk-8u171-linux-i586.tar.gz
Linux x64	167.14 MB	jdk-8u171-linux-x64.rpm
Linux x64	182.05 MB	jdk-8u171-linux-x64.tar.gz
Mac OS X x64	247.84 MB	jdk-8u171-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	139.83 MB	jdk-8u171-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.19 MB	jdk-8u171-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	140.6 MB	jdk-8u171-solaris-x64.tar.Z
Solaris x64	97.05 MB	jdk-8u171-solaris-x64.tar.gz
Windows x86	199.1 MB	jdk-8u171-windows-i586.exe
Windows x64	207.27 MB	jdk-8u171-windows-x64.exe

Figura A.1: Descarga de JDK8

Una vez descargado el fichero **.dmg**, es necesario ejecutarlo para llevar a cabo la instalación. Hacemos doble click en el fichero descargado y nos aparece la siguiente ventana que vemos en la Figura A.2:



Figura A.2: Instalación del paquete JDK8

A continuación hacemos doble click en el paquete y se arrancará el asistente de instalación. Pulsamos en “Continuar” y finalmente en instalar para realizar la instalación satisfactoriamente (Figura A.3):

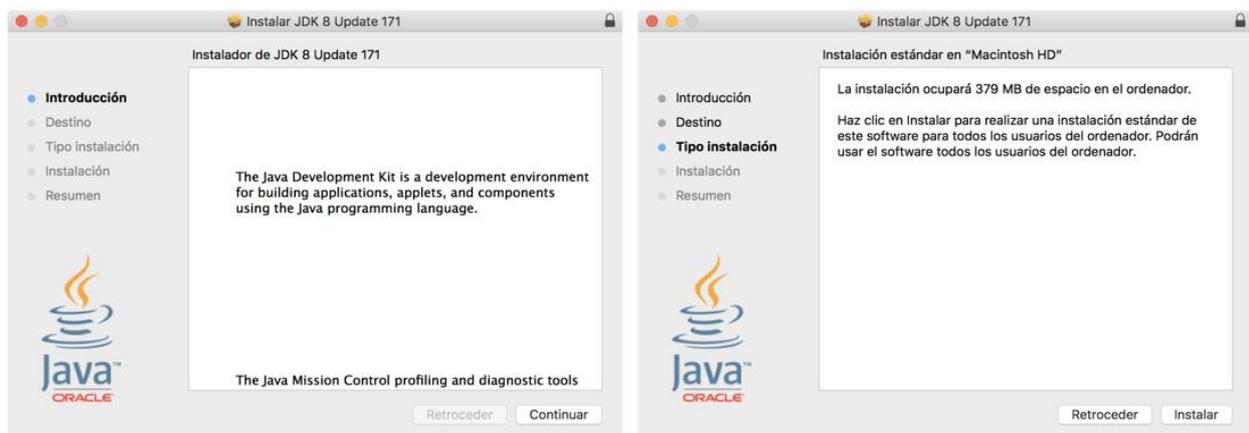


Figura A.3: Asistente de instalación JDK8

Finalmente comprobamos su instalación desde un terminal introduciendo el siguiente comando (Figura A.4):

```
carlosmartinescobia — -bash — 92x9
MacBook-Pro-de-Carlos:~ carlosmartinescobia$ java -version
java version "1.8.0_172"
Java(TM) SE Runtime Environment (build 1.8.0_172-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.172-b11, mixed mode)
MacBook-Pro-de-Carlos:~ carlosmartinescobia$
```

Figura A.4: Comprobación de instalación de JDK8

A.1.2 Spring Tool Suite (STS)

A continuación procedemos a instalar el framework Spring. Para ello hacemos click en el siguiente enlace para descargar el ejecutable:

[Descarga de Spring Tool Suite](#)

Una vez hagamos click descargaremos la versión que corresponda con nuestro sistema operativo, en nuestro caso descargamos la siguiente (Figura A.5):

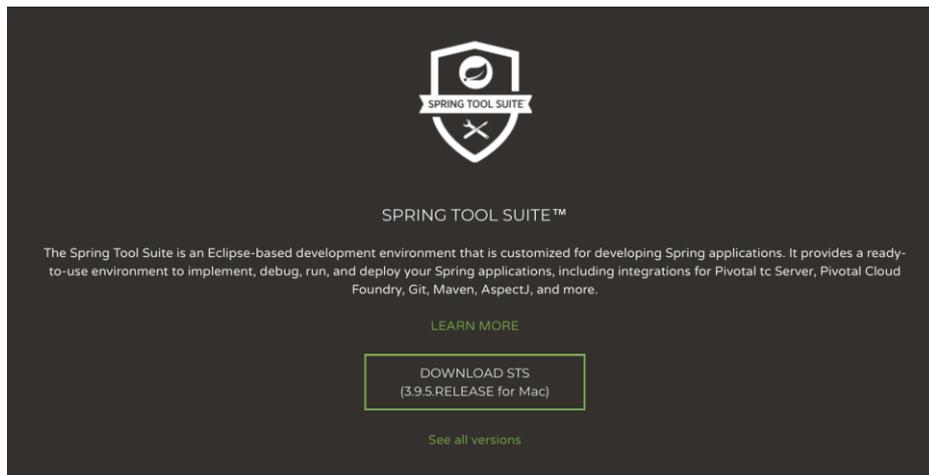


Figura A.5: Descarga de Spring Tool Suite

Una vez descargada la aplicación, ejecutamos el fichero **.dmg** descargado y arrastramos el logo de la aplicación hasta la carpeta “Aplicaciones”, como vemos en la Figura A.6:

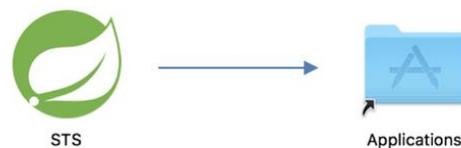


Figura A.6: Instalación de Spring Tool Suite

Tras hacer esto, ejecutamos la aplicación y debe funcionar correctamente como vemos a continuación en la Figura A.7:

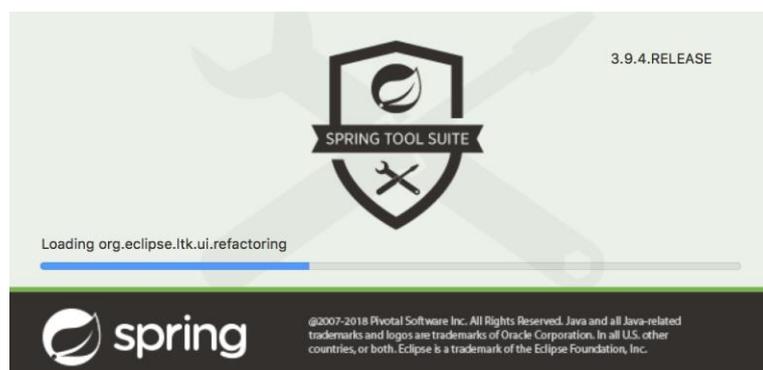
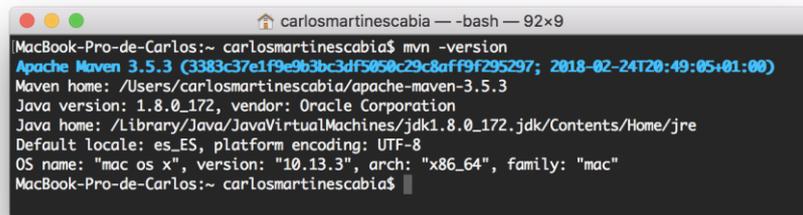


Figura A.7: Ejecución de Spring Tool Suite

A.1.3 Maven

Maven viene incluido al instalar el framework Spring, ya que se encargará de descargar las dependencias de las librerías. Para comprobar la actual versión de Maven en nuestro sistema operativo podemos ejecutar el siguiente comando en un terminal (Figura A.8):



```
MacBook-Pro-de-Carlos:~ carlosmartinescabi$ mvn -version
Apache Maven 3.5.3 (3383c37e1f9e9b3bc3df5650c29c8aff9f295297; 2018-02-24T20:49:05+01:00)
Maven home: /Users/carlosmartinescabi/apache-maven-3.5.3
Java version: 1.8.0_172, vendor: Oracle Corporation
Java home: /Library/Java/JavaVirtualMachines/jdk1.8.0_172.jdk/Contents/Home/jre
Default locale: es_ES, platform encoding: UTF-8
OS name: "mac os x", version: "10.13.3", arch: "x86_64", family: "mac"
MacBook-Pro-de-Carlos:~ carlosmartinescabi$
```

Figura A.8: Comprobación de versión de Maven

A.1.4 MySQLWorkbench

Por último instalaremos la siguiente aplicación para facilitar la tarea de administración y gestión de la base de datos. Para descargarla, hacemos click en el siguiente enlace:

[Descarga de MySQLWorkbench](#)

Una vez hagamos click en el enlace seleccionaremos la versión que mejor se adapte a nuestro sistema operativo. En nuestro caso descargaremos la opción para macOS como podemos ver en la siguiente imagen (Figura A.9):

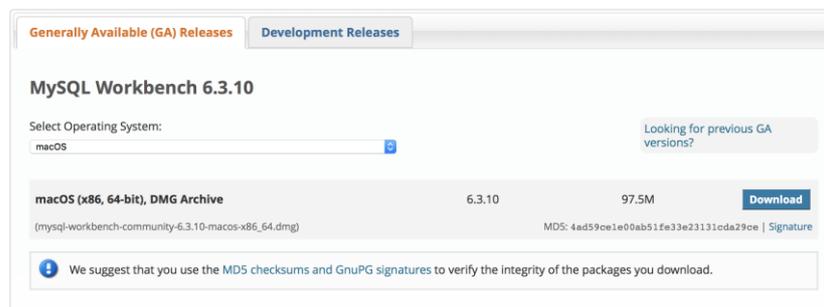


Figura A.9: Descarga de MySQLWorkBench

Finalmente, ejecutamos el fichero .dmg descargado y movemos el icono de la aplicación a la carpeta “Aplicaciones” como se observa en la Figura A.10. La aplicación estará lista para su uso.

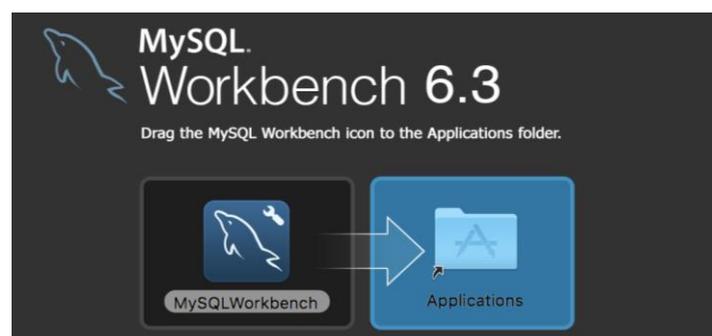


Figura A.10: Instalación de MySQLWorkBench

A.2 Despliegue de la aplicación

A continuación detallaremos los pasos a seguir para el despliegue de la aplicación desde el propio entorno del framework Spring.

A.2.1 Despliegue de la base de datos

Para ello abrimos la aplicación “MySQLWorkBench”. Dentro del apartado “MySQL Connections” pulsaremos en la instancia local 3306, ya que nuestra aplicación se conectará a la misma. Por otro lado, nuestro fichero “application.properties” estará configurado de la siguiente forma (Figura A.11):

```
application.properties
1 spring.datasource.url=jdbc:mysql://localhost/sportmeet?useSSL=false
2 spring.datasource.username=root
3 spring.datasource.password=root
4 spring.datasource.driver-class-name=com.mysql.jdbc.Driver
5 spring.jpa.database = mysql
6 spring.jpa.database-platform= org.hibernate.dialect.MySQL5Dialect
7 logging.level.org.hibernate.SQL = debug
8 spring.servlet.multipart.max-file-size= 10MB
9 spring.servlet.multipart.max-request-size= 10MB
10
```

Figura A.11: Fichero application.properties

Hacemos click en la instancia 3306 e introducimos la contraseña para conectarnos (Figura A.12). Esta será la misma que la que tenemos escrita en el fichero de la Figura A.11, concretamente el campo “spring.datasource.password”:



Figura A.12: Conexión a la base de datos

A continuación pulsamos en la opción “Open a SQL script file”, buscamos el fichero “sportmeet.sql” el cual contiene las sentencias SQL para la creación de la base de datos con todas las tablas correspondientes, y pulsamos en “open” para abrirlo en MySQLWorkBench (Figura A.13):

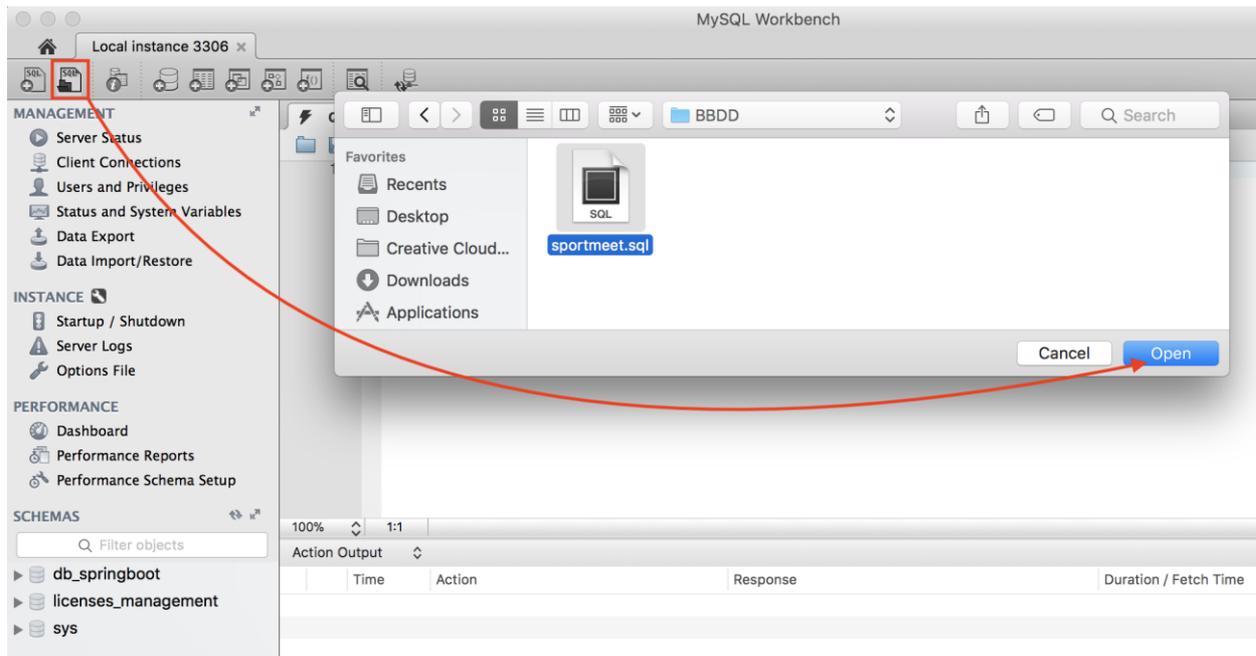


Figura A.13: Importación del fichero SQL para la creación de la base de datos

A continuación pulsamos en el botón “Execute”, refrescamos la secciones “Schemas” y veremos como ahora aparece nuestra base de datos con todas y cada una de las tablas (Figura A.14):

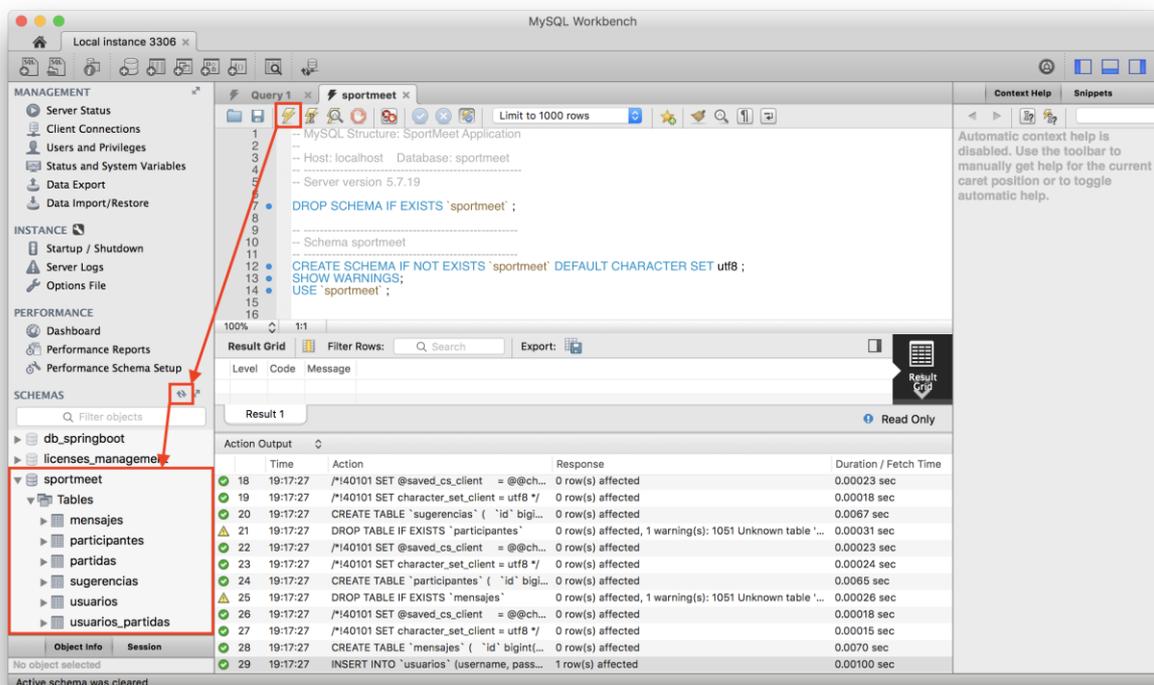


Figura A.14: Creación del esquema “SportMeet” y de sus correspondientes tablas

A.2.2 Puesta en funcionamiento de la aplicación

Ahora que ya tenemos lista la base de datos, pasaremos a poner en funcionamiento nuestra aplicación. Para ello lo primero que debemos hacer es abrir Spring Tool Suite. Una vez abierto, estableceremos la carpeta que funcionará como espacio de trabajo, en la que deberemos tener almacenada nuestra aplicación. La seleccionamos y pulsamos en “Launch”, como se observa en la Figura A.15:

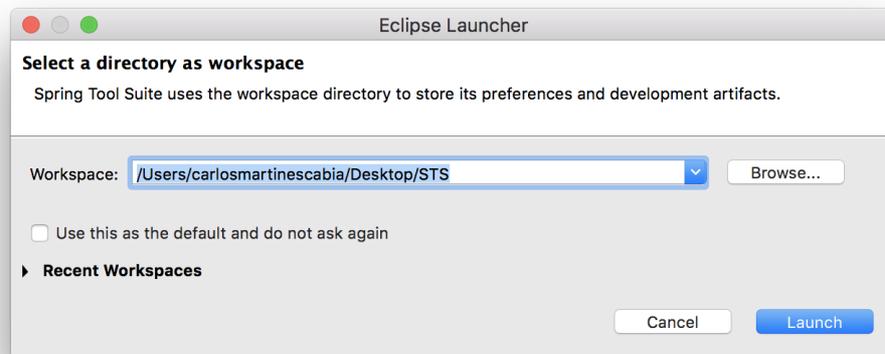


Figura A.15: Estableciendo el espacio de trabajo en Spring Tool Suite

A continuación procedemos a importar el proyecto. Para ello pulsamos en “File” -> “Open projects from File System”, como vemos a continuación en la Figura A.16:

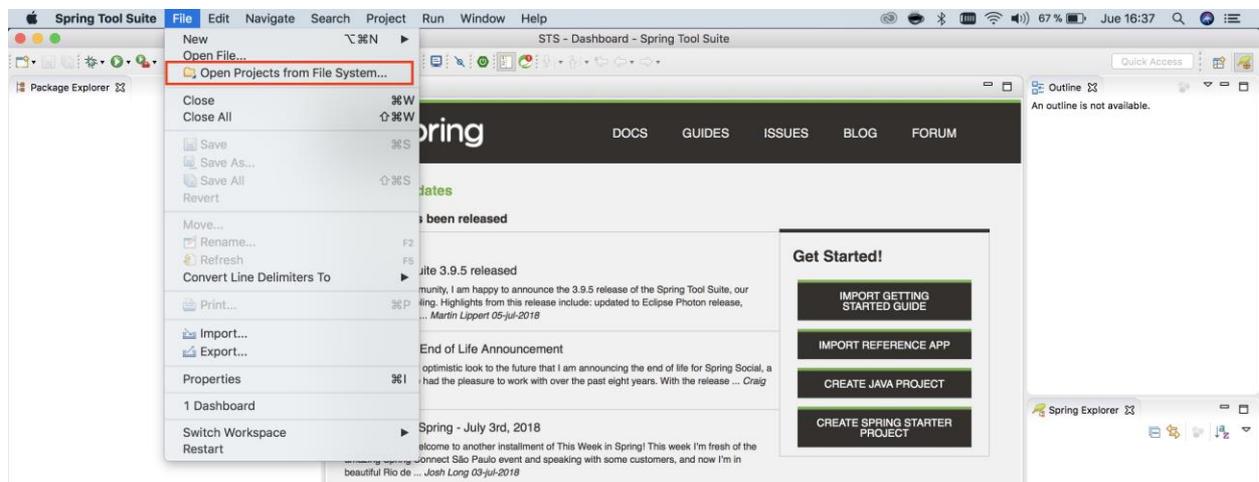


Figura A.16: Importar proyecto a Spring Tool Suite – 1

En el nuevo menú, pulsamos en el botón “Directory” para seleccionar el directorio que contiene nuestro proyecto. Posteriormente seleccionamos el directorio, pulsamos en “Open” y finalizamos con “Finish”, como podemos ver en la Figura A.17:

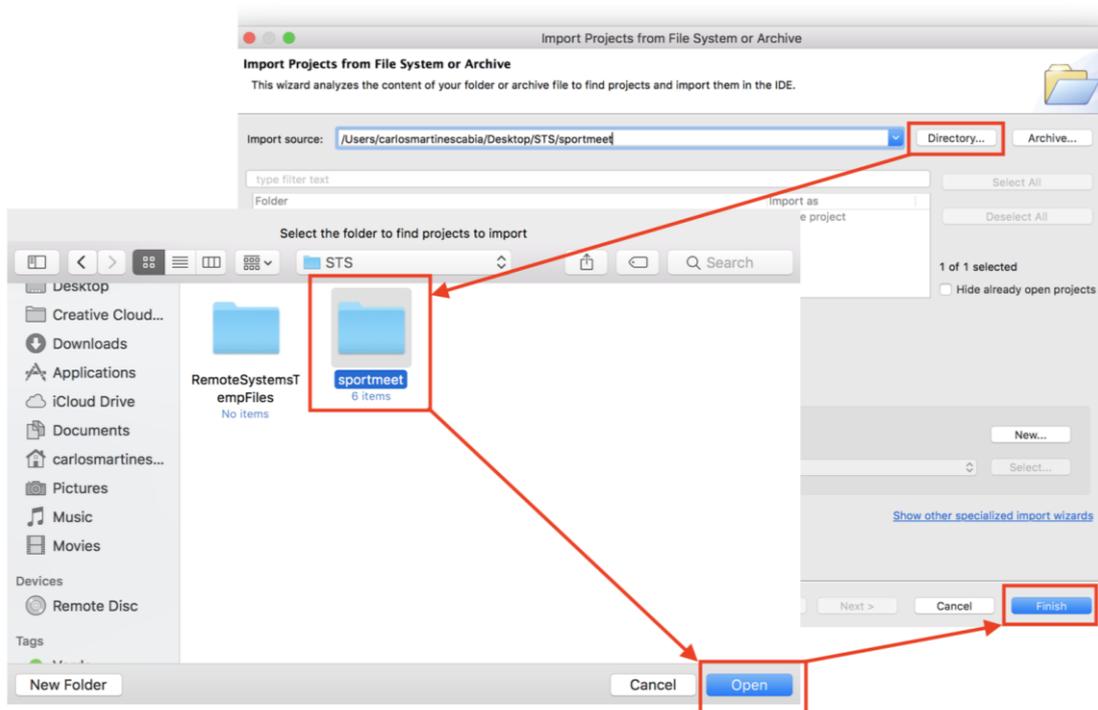


Figura A.17: Importar proyecto a Spring Tool Suite – 2

Finalmente, el proyecto estará importando y listo para ejecutar. Pulsamos en el botón “Start/Restart” y la aplicación se pondrá en funcionamiento (Figura A.18):

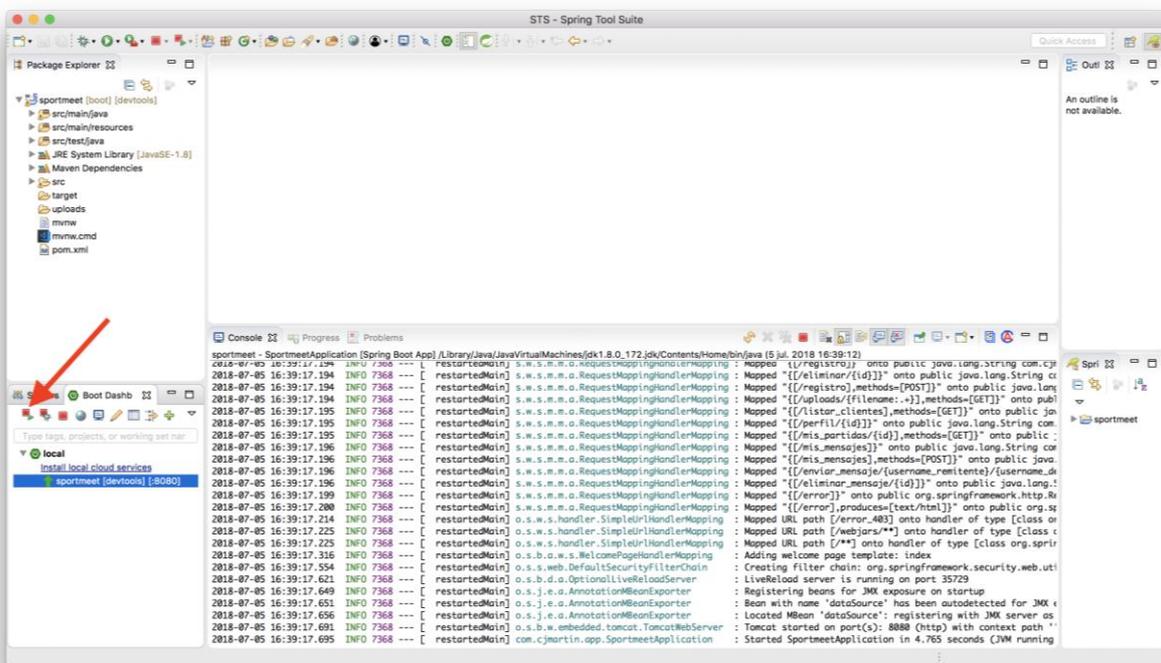


Figura A.18: Aplicación en funcionamiento desde Spring Tool Suite

Por último, basta con abrir un navegador web y escribir la url: <http://localhost:8080/>, como vemos a continuación en la Figura A.19:

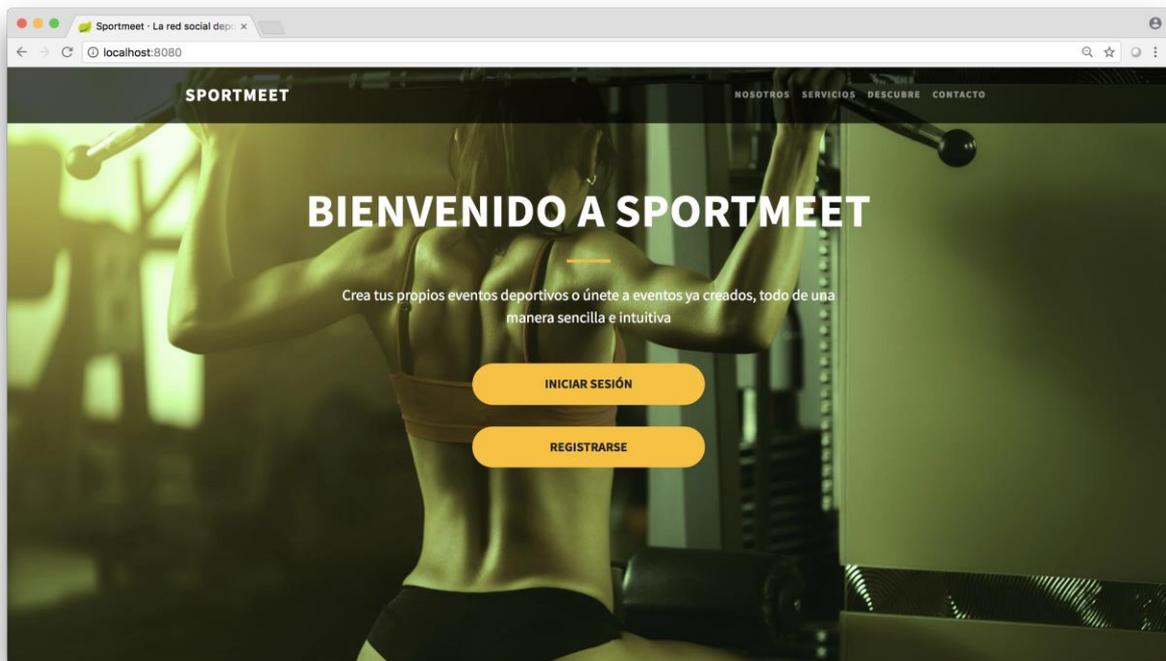


Figura A.19: Aplicación en funcionamiento desde navegador web