eman ta zabal zazu

ESCUELA UNIVERSITARIA DE INGENIERÍA TÉCNICA INDUSTRIAL DE BILBAO

GRADO EN (TITULACIÓN)

TRABAJO FIN DE GRADO

2013 / 2014

FOOTBALL MANAGER APP

MEMORIA TFG

DATOS DE LA ALUMNA O DEL ALUMNO	**DATOS DEL DIRECTOR O DE LA DIRECTORA**
NOMBRE: YERAY	NOMBRE: MIKEL
APELLIDOS: GONZÁLEZ GUTIÉRREZ	APELLIDOS: VILLAMAÑE GIRONÉS
	DEPARTAMENTO: LENGUAJES Y SISTEMAS INFORMÁTICOS
FDO.:	FDO.:
FECHA:	FECHA:

RESUMEN

El objetivo de este proyecto ha sido la realización de una aplicación para tablets con sistema operativo iOS de Apple. La aplicación la puede utilizar cualquier persona con inquietudes sobre el mundo del fútbol, pero está pensada básicamente para entrenadores que busquen almacenar datos relacionados con sus equipos, jugadores, partidos, estadísticas de los equipos y los jugadores, etc.

La información introducida quedará almacenada hasta que el usuario decida lo contrario eliminando los datos que ya no desee tener en su dispositivo de manera fácil e intuitiva.

Aparte de eso, se ha desarrollado una pizarra táctil donde el usuario podrá realizar sus tácticas previas al partido y enseñársela a sus jugadores. Dispondrá de dos colores para diferenciar los diferentes jugadores que se dibujen. Además, se pueden realizar ligas online para que varios usuarios simultáneamente accedan a su contenido o lo modifiquen. Las ligas podrán ser reales a nivel profesional, de amigos o incluso ligas de videojuegos.

Se ha realizado un apartado de mensajería donde los usuarios se podrán comunicar entre ellos sin salir de la aplicación. Bastará con buscar al usuario en el sistema, seleccionarlo y mandarle un mensaje.

Si un usuario decide eliminar todos sus datos del sistema, podrá hacerlo y se eliminará cualquier información relacionada con él.

ÍNDICE

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

1.1. Descripción y situación del trabajo

En este proyecto se va a realizar una aplicación para tablets con sistema operativo iOS. Esta aplicación va a permitir al usuario almacenar datos de sus equipos de fútbol, jugadores, estadísticas tanto de equipos como de jugadores, resultados de los partidos que dispute cada equipo, planificar tácticas en una pizarra táctil, comunicarse con otros usuarios mediante un apartado de mensajería y participar en ligas reales y online.

Para llevar a cabo el proyecto, se partirá de la base obtenida en las asignaturas cursadas en la carrera, así como de libros y tutoriales de otros desarrolladores, ya que en el lenguaje utilizado para la creación de aplicaciones en iOS (Objective-C) parto desde cero.

A día de hoy la tecnología forma parte de nuestras vidas y está presente en nuestras tareas cotidianas. La decisión de hacer una aplicación para tablet con sistema operativo iOS surge por el incremento de ventas y el uso de los usuarios con este tipo de dispositivos.

El objetivo principal de las grandes empresas es estar en lo más alto en la lista de ventas de este tipo de dispositivos y por la calidad de sus productos, *"En el año 2013 Apple fue la empresa que más tablet vendió"* . (Velasco, 2014)

A los usuarios lo que les gusta es probar las aplicaciones nuevas que van desarrollándose y es por ello que *"El número de aplicaciones disponibles para descargar, supera ya la cifra de 1.000.000 en la AppStore".* (JC, 2013)

En cuanto a conocimientos, es interesante adentrarse en el mundo de las aplicaciones hasta ahora desconocido y aprender el método de trabajo a seguir para realizar este tipo de software.

1.2. Razones

Durante estos últimos años se ha visto cómo las tablets, ya sean de sistema operativo iOS o Android, han ido entrando en las vidas de las personas como en su día lo hicieron los smartphones. Hoy en día, la mayoría de las personas tiene una tablet, empezando por los más pequeños de la casa, hasta personas mayores que las utilizan para leer noticias en internet. El motivo por el cual tantas personas dispone de un dispositivo así es por su gran facilidad de uso.

Este es el motivo por el que cada día aparecen más desarrolladores para estos dispositivos y más es la gente que se está adentrando en el mundo de desarrollo de iOS. Es muy importante a día de hoy para un estudiante de informática estar un paso por delante del resto y como Objective-C es un lenguaje que no se aprende en la carrera, creo que eso suma un punto a mi favor a la hora de encontrar trabajo en un futuro.

El realizar una aplicación en este caso de fútbol, puede ayudar a muchas personas a tener sus datos actualizados de manera sencilla y disponibles en todo momento, hace que la motivación sea mayor a la hora de realizar la aplicación y de adquirir los conocimientos necesarios para la realización de la misma.

1.3. Planteamiento del problema

El planteamiento de este proyecto será realizar una aplicación para iOS en la que el usuario podrá almacenar información de equipos de fútbol (tanto si es un entrenador, como si es un aficionado), datos de los jugadores que pertenecen a cada equipo, estadísticas personales y generales tanto de los equipos como de los jugadores, realizar tácticas en una pizarra táctil, enviar mensajes a otros usuarios y participar de manera directa o indirecta en ligas online.

Buscando este tipo de aplicaciones llegué a la conclusión de que no existía algo parecido que ofreciese tantas cosas como se ofrece en esta aplicación. Para poder tener todas estas funcionalidades había que tener varias aplicaciones diferentes por lo que se pensó unir todo en una para dar facilidad al usuario.

Es importante ofrecer al usuario todas las funcionalidades que utilice en un mismo sitio, ya que resulta engorroso el tener que salir de la aplicación para enviar un mensaje o ir a una página web externa para ver un resultado.

Lo que se busca desde el principio es facilidad de uso y que el usuario se adapte rápidamente a la aplicación gracias a los contenidos que se esperan ofrecer.

1.4. Justificación

Con este proyecto se pretende ayudar a los usuarios que necesiten del uso de este tipo de aplicaciones para facilitarle el almacenamiento de datos y el acceso rápido a ellos. Son muchos los entrenadores de fútbol que llevan sus cuadernos a los partidos y al final acaban perdiendo los datos que han anotado. De esta manera, todo lo anotado en la aplicación quedará almacenado de manera permanente hasta que el usuario decida lo contrario.

En cuanto al impacto a corto plazo, se intentará que entrenadores a nivel provincial prueben esta aplicación para así poder ver que mejoras se le podría hacer y que necesidades que como desarrollador no he visto puedan tener. Con esto se espera que con el tiempo se pueda llegar a expandir a más usuarios y sea una aplicación que utilicen en su día a día.

Esta aplicación dentro del mercado es diferente al resto y además permite que los usuarios se comuniquen entre sí sin tener que utilizar otra aplicación para enviar mensajes. Además, si solo somos un usuario con inquietudes y curioso por ver que ocurre en las diferentes ligas que se hayan creado, también se le da esa posibilidad.

Para entrenadores, lo que es más interesante es el hecho de tener una pizarra táctil donde poder mostrar todas las tácticas que quiere que hagan sus jugadores o simplemente mostrar el once inicial en la misma. Esto hace que ya no tenga que llevar una pizarra con un rotulador o un cuaderno para apuntar el resto de cosas. Con una tablet y la aplicación será más que suficiente.

2. OBJETIVOS DEL PROYECTO

2.1. Objetivos

El objetivo principal es aprender a desarrollar aplicaciones para dispositivos de Apple. En este apartado se mostrarán los objetivos del proyecto.

- Realización de una aplicación de fútbol en Objective-C para el sistema operativo iOS.
- Utilización de la aplicación para aquellos usuarios que deseen almacenar los datos de sus equipos y jugadores, así como participar en ligas con otros usuarios y poder tener una comunicación vía mensajería con cada uno de ellos.
- Aprender un nuevo lenguaje de programación desde cero como es Objective-C.
- Aprender a diseñar una aplicación partiendo de las necesidades que un usuario pueda tener para la utilización de una aplicación de este tipo.
- Entrar en el mundo del desarrollo de aplicaciones para dispositivos móviles o tablets.

2.2. Descripción del proyecto

El mundo del fútbol es un mundo en el que participan todo tipo de personas ya sea de manera directa o indirecta y cada día van apareciendo cosas que facilitan la vida a quienes lo disfrutan. Hablamos desde chips integrados en las botas de los futbolistas que cuentan las calorías que va consumiendo cada jugador y el número de kilómetros recorridos, hasta páginas web con información personal de todas las estadísticas posibles de cada jugador de la liga (veces que toca el balón en un partido, hacia donde tira los penalti, etc.).

En este caso se ha pensado en una aplicación de fútbol en la que poder almacenar datos y participar con otros usuarios que quieran interactuar con el resto.

La idea surge por la necesidad de tener que almacenar los datos de cada jugador que participa en el equipo de fútbol que yo entreno. Como yo, el resto de los entrenadores que participamos en la liga de fútbol de categoría infantil en Vizcaya, anotamos las estadísticas que nuestros jugadores han ido haciendo durante el partido y pensé en la posibilidad de poder tener todos esos datos almacenados en un sistema portátil. Además, esta aplicación permite crear una liga online para poder anotar los resultados disputados de cada partido de manera directa sin tener que esperar días hasta que la Federación Vizcaína de Fútbol anote todos los resultados en su página web.

Esta aplicación está pensada para entrenadores de fútbol o para los aficionados que quieran llevar sus propias estadísticas tanto de equipos profesionales como de ligas de amigos reales u online.

El objetivo del proyecto es la realización de un software que permita al usuario realizar todas esas acciones que se echa en falta tener en un solo sistema. Para ello, cada usuario que quiera utilizar la aplicación deberá de darse de alta en el sistema y una vez dentro encontrará un menú con seis apartados; equipos, pizarra, estadísticas, ligas, perfil y mensajes. En cada uno de los apartados se podrán realizar las diversas funciones dependiendo de las necesidades de cada uno. Para poder acceder a la aplicación es necesaria la identificación y por lo tanto conexión a internet ya que la aplicación tendrá funcionalidades como la comunicación entre usuarios, ligas online, etc. que requiere tener que estar conectado.

La aplicación dispondrá de interfaces con iconos muy intuitivos y colores agradables a la vista del usuario para facilitar en la medida de lo posible la navegación por toda la aplicación.

2.3 Alcance del proyecto

Para el desarrollo del proyecto el ciclo de vida elegido es el de prototipos. Esta estrategia de desarrollo se basa en la creación de unos prototipos previos que sirvan para dar apoyo al desarrollador a la hora de la implementación. Este tipo de ciclo de vida tiene la ventaja de ir evaluando cada prototipo una vez finalizado y comprobar si se cumple con lo esperado.

A continuación se explicarán las distintas etapas del proyecto:

- **Captura de requisitos:** Se buscará información relacionada con este tipo de aplicaciones subidas en la AppStore y comprobar si hay alguna que ofrezca lo que se te pretende ofrecer con esta. Información sobre el lenguaje de programación Objective-C que va a ser el que se utilice para realizar la aplicación, así como información relacionada con el sistema operativo iOS. Se tendrá en cuenta que tipo de datos se muestra en aplicaciones y páginas web relacionadas con el mundo del fútbol.

- **Análisis:** En este apartado se tendrá en cuenta los elementos que van a intervenir en la realización del proyecto a partir de los objetivos mencionados con anterioridad. Se determinará la estructura que va a tener el proyecto, las herramientas a utilizar y las distintas funcionalidades.

- **Diseño:** Dentro de esta fase definiremos las funcionalidades que se van a desarrollar en la aplicación, las relaciones de las bases de datos y se diseñará el algoritmo principal. También determinaremos el lenguaje de programación que se va a utilizar para el desarrollo del proyecto.

- **Implementación:** En esta fase se llevará a cabo el desarrollo de los algoritmos definidos en la fase de diseño.

    - ***Prototipo 1:*** Realización de una estructura de diseño orientativa.
    - ***Prototipo 2:*** Como guardar información en la base de datos interna.

- ***Prototipo 3:*** Una vez almacenados los datos en la base de datos interna, mostrarlo en las interfaces.
- ***Prototipo 4:*** Como guardar información en la base de datos externa.
- ***Prototipo 5:*** Una vez almacenados los datos en la base de datos externa, mostrarlo en las interfaces.
- ***Prototipo 6:*** Realización de una pizarra táctil.
- ***Prototipo 7:*** Creación de usuarios.
- ***Prototipo 8:*** Creación de servicio de mensajería e interacción entre usuarios.
- ***Prototipo 9:*** Participación en actividades online y modificación de datos.
- ***Prototipo 10:*** Cambios visuales de la aplicación relacionadas con el diseño de interfaces, iconos y colores.
- ***Prototipo 11:*** Actualizaciones y mejoras de la aplicación una vez terminada.

- ***Debugging:*** Se realizarán pruebas durante todo el desarrollo del proyecto para ir solucionando los posibles errores de código que vayan surgiendo y no tener que buscar todos los errores una vez terminado el proyecto.

- ***Validación:*** El proyecto deberá cumplir con los requisitos exigidos.

- ***Mantenimiento y evolución:*** En este apartado se podrán añadir nuevas funcionalidades al proyecto todavía no definidas y que pueda suponer una mejora a la aplicación, así como corregir los errores que no se hayan detectado con anterioridad y puedan causar problemas en el funcionamiento de la aplicación.

EDT

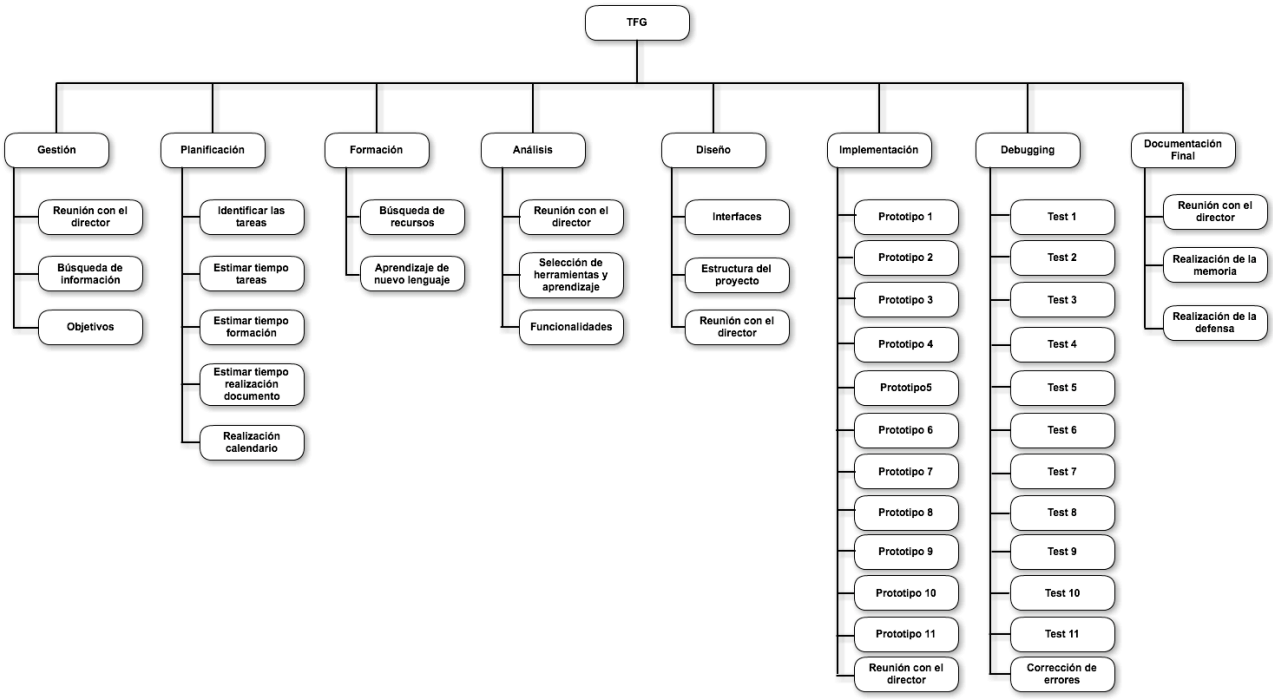


Ilustración 1: EDT

TAREAS

1. GESTIÓN

- Reunión con el director

  Duración: 1 hora
  Descripción: Realización de una reunión con el director para hablar de lo que se pretende realizar en el proyecto con una idea más o menos clara e intentar resolver dudas en cuanto al tipo de proyecto se refiere.

- Búsqueda de información

  Duración: 20 horas
  Descripción: Búsqueda de información sobre lo que se pretende hacer en el proyecto, ver si ya existe algo parecido a lo que queremos hacer y si es así que novedades introducirle para destacar sobre el resto y las herramientas que hay que utilizar para la realización del proyecto.

- Objetivos

  Duración: 6 horas
  Descripción: Dejar claros los objetivos que se quieren conseguir, así como las funcionalidades que queremos desarrollar en el proyecto.

2. PLANIFICACIÓN

- Identificar las tareas

  Duración: 2 horas
  Descripción: Identificación de todas y cada una de las tareas que hay que realizar a lo largo del proyecto.

- Estimar tiempo tareas

  Duración: 5 horas
  Descripción: Explicar detalladamente en que va a consistir cada tarea a realizar y el tiempo estimado que vamos a tardar en realizarla.

- Estimar tiempo formación

  Duración: 2 horas
  Descripción: Estimar aproximadamente con los conocimientos previos en otros lenguajes de programación el tiempo estimado de aprendizaje del nuevo lenguaje Objective-C.

- Estimar tiempo realización documento

  Duración: 2 horas
  Descripción: Estimar el tiempo que vamos a tardar en realizar el documento teniendo en cuenta las dimensiones del proyecto.

- Realización calendario

  Duración: 2 horas
  Descripción: Realización de una tabla con las fechas estimadas para la realización de todas las partes del proyecto.

3. FORMACIÓN

- Búsqueda de recursos

  Duración: 5 horas
  Descripción: Búsqueda de recursos como pueden ser libros, tutoriales, etc. para empezar la auto formación de Objective-C.

- Aprendizaje nuevo lenguaje

  Duración: 50 horas

Descripción: Aprendizaje mediante libros y tutoriales online del nuevo lenguaje de programación Objective-C. Es un tiempo estimado ya que facilitara el aprendizaje el conocer otros lenguajes de programación.

4. ANÁLISIS

- Reunión con el director

  Duración: 1 hora
  Descripción: Realización de una reunión con el director para hablar de lo que se pretende utilizar para realizar la aplicación, así como las funcionalidades que se pretenden hacer.

- Selección de herramientas y aprendizaje

  Duración: 10 horas
  Descripción: Búsqueda de las herramientas que se van a utilizar a lo largo del proyecto, instalación y aprendizaje.

- Funcionalidades

  Duración: 20 horas
  Descripción: Definir claramente las funcionalidades de la aplicación y que elementos intervienen a cada una de ellas, además de cómo utilizarlos.

5. DISEÑO

- Interfaces

  Duración: 10 hora
  Descripción: Diseño de las interfaces de nuestra aplicación mostrando lo que se va a querer mostrar en nuestra aplicación.

- Estructura del proyecto

  Duración: 30 horas
  Descripción: Realización del diseño del algoritmo a implementar.

- Reunión con el director

  Duración: 1 hora
  Descripción: Reunión con el director para mostrarle las interfaces diseñadas y el algoritmo que se ha pensado para realizar la aplicación y corregir errores en caso de no estar de acuerdo con nuestras ideas.

6. IMPLEMENTACIÓN

- Prototipo 1

  Duración: 20 horas
  Descripción: Realización de estructura de diseño.

- Prototipo 2

  Duración: 20 horas
  Descripción: Guardar información base de datos interna.

- Prototipo 3

  Duración: 10 horas
  Descripción: Mostrar datos almacenados de la base de datos interna en las interfaces.

- Prototipo 4

  Duración: 20 horas
  Descripción: Guardar información en base de datos externa.

- Prototipo 5

  Duración: 10 horas
  Descripción: Mostrar datos almacenados de la base de datos externa en las interfaces.

- Prototipo 6

  Duración: 20 horas
  Descripción: Realización de la pizarra táctil.

- Prototipo 7

  Duración: 20 horas
  Descripción: Creación de usuarios.

- Prototipo 8

  Duración: 20 horas
  Descripción: Creación de servicio de mensajería e interacción entre usuarios.

- Prototipo 9

  Duración: 20 horas
  Descripción: Actividades online y modificación de datos.

- Prototipo 10

  Duración: 20 horas
  Descripción: Cambios visuales de la aplicación relacionadas con el diseño de interfaces, iconos y colores.

- Prototipo 11

  Duración: 20 horas
  Descripción: Actualizaciones y mejoras de la aplicación.

- Reunión con el director

  Duración: 1 hora
  Descripción: Realización de una reunión con el director para mostrarle la aplicación y ver si se podría hacer alguna mejora en caso de no haberla tenido en cuenta.

7. DEBUGGING

- Test prototipo 1

  Duración: 2 horas
  Descripción: Test de prueba sobre el prototipo 1.

- Test prototipo 2

  Duración: 2 horas
  Descripción: Test de prueba sobre el prototipo 2.

- Test prototipo 3

  Duración: 2 horas
  Descripción: Test de prueba sobre el prototipo 3.

- Test prototipo 4

  Duración: 2 horas
  Descripción: Test de prueba sobre el prototipo 4.

- Test prototipo 5

  Duración: 2 horas
  Descripción: Test de prueba sobre el prototipo 5.

- Test prototipo 6

  Duración: 2 horas

Descripción: Test de prueba sobre el prototipo 6.

- Test prototipo 7

  Duración: 2 horas
  Descripción: Test de prueba sobre el prototipo 7.

- Test prototipo 8

  Duración: 2 horas
  Descripción: Test de prueba sobre el prototipo 8.

- Test prototipo 9

  Duración: 2 horas
  Descripción: Test de prueba sobre el prototipo 9.

- Test prototipo 10

  Duración: 2 horas
  Descripción: Test de prueba sobre el prototipo 10.

- Test prototipo 11

  Duración: 5 horas
  Descripción: Test de prueba sobre el prototipo 11.

8. DOCUMENTACIÓN FINAL

- Reunión con el director

  Duración: 1 hora
  Descripción: Reunión para hablar sobre las dudas surgidas a la hora de realizar el documento.

- Realización de la memoria

  Duración: 30 hora
  Descripción: Realización de la memoria con todo lo realizado en el proyecto.

- Realización de la defensa

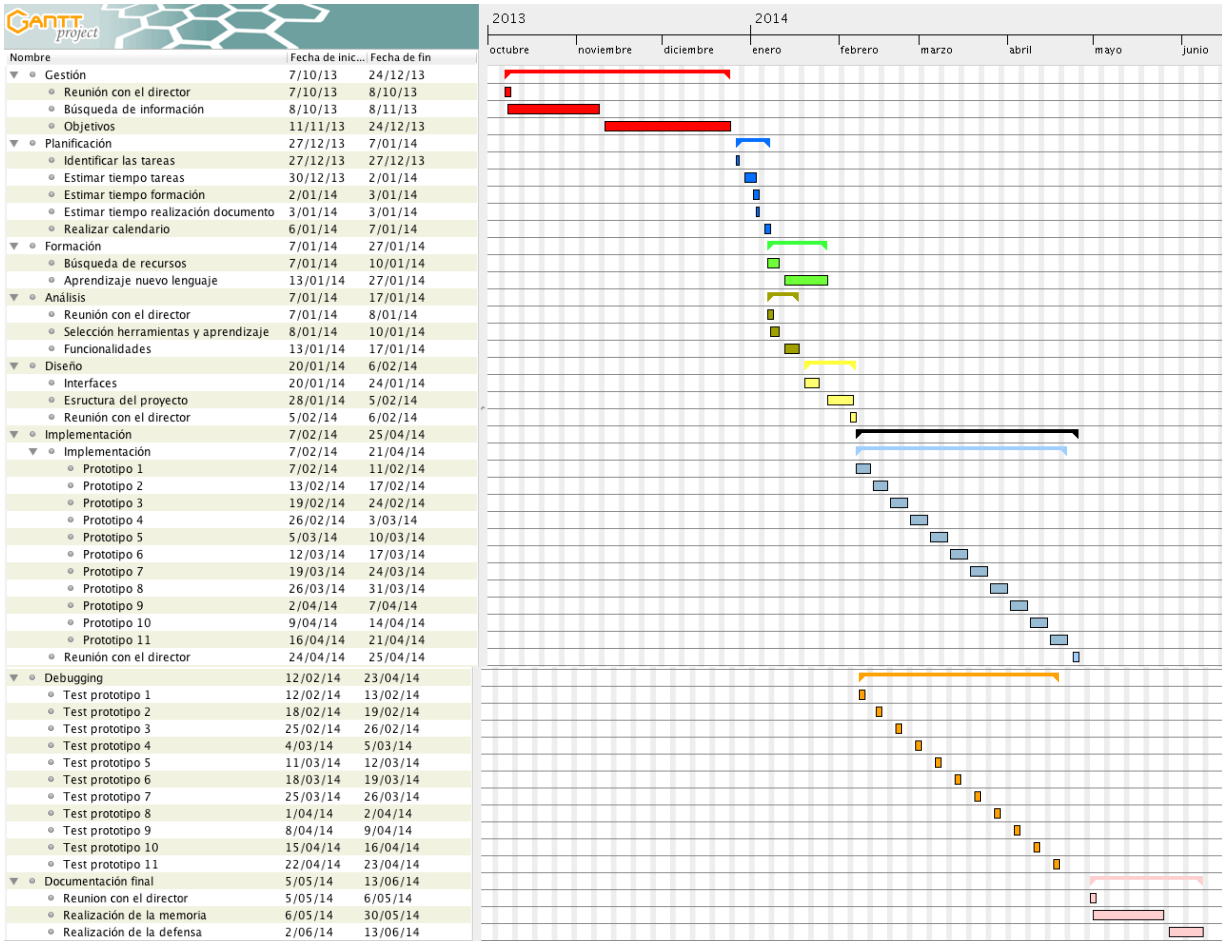
  Duración: 10 horas
  Descripción: Preparación para la defensa del proyecto.

Código	Tarea	Fecha Inicio	Fecha Fin	Horas
1	**Gestión**	**07/10/13**	**24/12/13**	**27**
1.1	Reunión director	07/10/13	08/10/13	1
1.2	Búsqueda información	08/10/13	08/11/13	20
1.3	Objetivos	11/11/13	24/12/13	6
2	**Planificación**	**27/12/13**	**07/01/14**	**13**
2.1	Identificar las tareas	27/12/13	28/12/13	2
2.2	Estimar tiempo tareas	30/12/13	02/01/14	5
2.3	Estimar tiempo formación	02/01/14	03/01/14	2
2.4	Estimar tiempo realización documento	03/01/14	04/01/14	2
2.5	Realizar calendario	06/01/14	07/01/14	2
3	**Formación**	**07/01/14**	**27/01/14**	**55**
3.1	Búsqueda de recursos	07/01/14	10/01/14	5
3.2	Aprendizaje nuevo lenguaje	13/01/14	27/01/14	50
4	**Análisis**	**07/01/14**	**17/01/14**	**31**
4.1	Reunión con el director	07/01/14	08/01/14	1
4.2	Selección herramientas aprendizaje	08/01/14	10/01/14	10
4.3	Funcionalidades	13/01/14	17/01/14	20
5	**Diseño**	**20/01/14**	**06/02/14**	**41**
5.1	Interfaces	20/01/14	24/01/14	10
5.2	Estructura del proyecto	28/01/14	05/02/14	30
5.3	Reunión con el director	05/02/14	06/02/14	1
6	**Implementación**	**07/02/14**	**25/04/14**	**201**
6.1	Prototipo 1	07/02/14	11/02/14	20
6.2	Prototipo 2	13/02/14	17/02/14	20
6.3	Prototipo 3	19/02/14	24/02/14	10
6.4	Prototipo 4	26/03/14	03/03/14	20
6.5	Prototipo 5	05/03/14	10/03/14	10
6.6	Prototipo 6	12/03/14	17/03/14	20
6.7	Prototipo 7	19/03/14	24/03/14	20
6.8	Prototipo 8	26/03/14	31/03/14	20
6.9	Prototipo 9	02/04/14	07/04/14	20
6.10	Prototipo 10	09/04/14	14/04/14	20
6.11	Prototipo 11	16/04/14	21/04/14	20
6.12	Reunión con el director	24/04/14	25/04/14	1
7	**Debugging**	**12/02/14**	**23/04/14**	**25**
7.1	Test prototipo 1	12/02/14	13/02/14	2
7.2	Test prototipo 2	18/02/14	19/02/14	2
7.3	Test prototipo 3	25/02/14	26/02/14	2
7.4	Test prototipo 4	04/03/14	05/03/14	2
7.5	Test prototipo 5	11/03/14	12/03/14	2
7.6	Test prototipo 6	18/03/14	19/03/14	2
7.7	Test prototipo 7	25/03/14	26/03/14	2
7.8	Test prototipo 8	01/04/14	02/04/14	2
7.9	Test prototipo 9	08/04/14	09/04/14	2
7.10	Test prototipo 10	15/04/14	16/04/14	2
7.11	Test prototipo 11	22/04/14	23/04/14	5

Código	Tarea	Fecha Inicio	Fecha Fin	Horas
8	**Documentación final**	**05/05/14**	**13/06/14**	**41**
8.1	Reunión con el director	05/05/14	06/05/14	1
8.2	Realización de la memoria	06/05/14	30/05/14	30
8.3	Realización de la defensa	02/06/14	13/06/14	10
NÚMERO TOTAL DE HORAS				**434**

Tabla 1: GANTT

GANTT



Nombre	Fecha de inic...	Fecha de fin
Gestión	7/10/13	24/12/13
Reunión con el director	7/10/13	8/10/13
Búsqueda de información	8/10/13	8/11/13
Objetivos	11/11/13	24/12/13
Planificación	27/12/13	7/01/14
Identificar las tareas	27/12/13	27/12/13
Estimar tiempo tareas	30/12/13	2/01/14
Estimar tiempo formación	2/01/14	3/01/14
Estimar tiempo realización documento	3/01/14	3/01/14
Realizar calendario	6/01/14	7/01/14
Formación	7/01/14	27/01/14
Búsqueda de recursos	7/01/14	10/01/14
Aprendizaje nuevo lenguaje	13/01/14	27/01/14
Análisis	7/01/14	17/01/14
Reunión con el director	7/01/14	8/01/14
Selección herramientas y aprendizaje	8/01/14	10/01/14
Funcionalidades	13/01/14	17/01/14
Diseño	20/01/14	6/02/14
Interfaces	20/01/14	24/01/14
Esructura del proyecto	28/01/14	5/02/14
Reunión con el director	5/02/14	6/02/14
Implementación	7/02/14	25/04/14
Implementación	7/02/14	21/04/14
Prototipo 1	7/02/14	11/02/14
Prototipo 2	13/02/14	17/02/14
Prototipo 3	19/02/14	24/02/14
Prototipo 4	26/02/14	3/03/14
Prototipo 5	5/03/14	10/03/14
Prototipo 6	12/03/14	17/03/14
Prototipo 7	19/03/14	24/03/14
Prototipo 8	26/03/14	31/03/14
Prototipo 9	2/04/14	7/04/14
Prototipo 10	9/04/14	14/04/14
Prototipo 11	16/04/14	21/04/14
Reunión con el director	24/04/14	25/04/14
Debugging	12/02/14	23/04/14
Test prototipo 1	12/02/14	13/02/14
Test prototipo 2	18/02/14	19/02/14
Test prototipo 3	25/02/14	26/02/14
Test prototipo 4	4/03/14	5/03/14
Test prototipo 5	11/03/14	12/03/14
Test prototipo 6	18/03/14	19/03/14
Test prototipo 7	25/03/14	26/03/14
Test prototipo 8	1/04/14	2/04/14
Test prototipo 9	8/04/14	9/04/14
Test prototipo 10	15/04/14	16/04/14
Test prototipo 11	22/04/14	23/04/14
Documentación final	5/05/14	13/06/14
Reunion con el director	5/05/14	6/05/14
Realización de la memoria	6/05/14	30/05/14
Realización de la defensa	2/06/14	13/06/14

Ilustración 2: GANTT

2.4. Gestión de riesgos

Se van a analizar los posibles riesgos que me puedan surgir a lo largo del proyecto y que supondría un retraso en la realización del mismo. Se identificará el riesgo y se analizará su posible solución y prevención.

- Pérdida de información

  Probabilidad: Baja

Impacto: Muy alto
Consecuencias: Si se pierde la información que se haya ido almacenando y eso supondría tener que volver a realizarla, el impacto sería muy alto.
Medida Preventiva: Guardar la información en varios dispositivos una vez se vayan modificando.
Plan de contingencia: Recuperar lo perdido y verificar hasta qué punto se ha perdido la información.

- Problemas personales

Probabilidad: Baja
Impacto: Depende del tipo de problema
Consecuencias: En caso de sufrir alguna enfermedad, problemas económicos, etc. habría que modificar la planificación y repartir las horas de otra manera diferente.
Medida Preventiva: Realizar más trabajo a diario, intentando mejorar los tiempos estimados.
Plan de contingencia: Analizar qué tipo de problema se tiene y buscar la solución más eficiente que suponga el menor retraso de tiempo posible.

- Problemas en la implementación

Probabilidad: Alta
Impacto: Medio
Consecuencias: Por falta de experiencia con este lenguaje nuevo de programación, me puedo retrasar a la hora de encontrarme con problemas ya que buscar la solución es más complicado que en un lenguaje con el que ya haya trabajado con anterioridad.
Medida Preventiva: Disponer de libros y tutoriales.
Plan de contingencia: Tener claro mediante libros y tutoriales lo que se va a realizar y en caso de ocurrir un fallo poder identificarlo con mayor rapidez.

- Ordenador estropeado

Probabilidad: Baja
Impacto: Muy alto
Consecuencias: Si se nos estropea el ordenador la consecuencia puede ser muy alta ya que el entorno de programación con el que se trabaja es exclusivo de Apple.
Medida Preventiva: Tener una máquina virtual en un equipo con Windows.
Plan de contingencia: Continuar la implementación desde la máquina virtual.

- Realizar una mala estimación

  Probabilidad: Alta
  Impacto: Medio
  Consecuencias: No trabajar todo el tiempo estimado, pérdida de tiempo en tareas en las que pensábamos que íbamos a hacer más rápido, etc.
  Medida Preventiva: Realizar una organización real para asemejarse más a la realidad con lo que estamos haciendo.
  Plan de contingencia: Invertir más horas de otro sitio en caso de necesitarlas.

2.5. Herramientas

A continuación, se va a explicar cada una de las herramientas utilizadas.

Herramientas hardware:

- Ordenador personal Mac Mini con sistema operativo OS X Mavericks
- Tablet Ipad Mini con sistema Operativo IOS 7.1

Herramientas de software:

- XCode

Xcode es el entorno de desarrollo que se utiliza para realizar aplicaciones para dispositivos de Apple. Es una herramienta de desarrollo fácil de usar y además es lo suficientemente potente como para que los desarrolladores de Apple utilicen esta herramienta.

Xcode tiene las siguientes características:

- Editor de código fuente
- Compilador integrado
- Simulador iOS
- Interface Builder (Storyboard)
- Etc.

Ilustración 3: XCODE

- SQLite Manager

  SQLite es un gestor de base de datos muy ligero y potente. Por sus características se utiliza en una gran variedad de aplicaciones como Mozilla Firefox, Skype, etc.

  Las bases de datos bajo SQLite se almacenan en un archivo que puede ser accedido por el programa bajo el texto denominado "sqlite3". Mediante la aplicación, se pueden efectuar consultas y ediciones utilizando sentencias SQL.

  SQLite Manager permite acceder a la administración y consultad de las bases de datos locales que se hayan creado.

- Microsoft Office

  Se ha utilizado el conjunto de programas que ofrece Microsoft para el desarrollo del proyecto.
  Para la creación de las tablas se ha hecho uso del Microsoft Excel.
  Para el desarrollo de la memoria Microsoft Word.
  Para la presentación de la defensa de la memoria Microsoft Power Point.

Ilustración 4:OFFICE

- Gantt Project

  Gantt Project es una herramienta gratuita disponible para todos los sistemas operativos que sirve para crear una planificación de un proyecto. El proyecto que creamos lo vamos dividiendo en tareas y estas en subtareas, cada una con su fecha de inicio y fecha de fin.

  Lo que se pretende con este tipo de programas es que se pueda obtener una visión general de lo que se pretende hacer y organizarnos mediante las fechas que vayamos estableciendo y las horas que creamos que tenemos que invertir en cada tarea.

Además, se pueden crear dependencia entre tareas para así evitar empezar una tarea hasta haber terminado otra.

Ilustración 5: GANTT

- Cacoo

Cacoo es una herramienta online que permite hacer diagramas de diversos tipos sin tener que instalar ninguna aplicación externa. Bastará con darse de alta en la web y mediante un nombre de usuario y una contraseña se accederá a la sección en la que se podrán realizar los diagramas.

Una de las ventajas es que es totalmente gratuita y permite que el trabajo colaborativo simultaneo, es decir, el mismo diagrama puede ser editado por varias personas a la vez. Además, es muy sencillo de utilizar y muy intuitivo.

2.6. Evaluación económica

Se realizará una evaluación económica sobre el coste del proyecto.

- Personal

He calculado que un programador gana aproximadamente 20€ / hora. El número de horas del proyecto en total son 434 horas, pero se van a quitar las horas de formación para no introducirlas en el coste total del proyecto, por lo tanto, el coste bruto del personal del proyecto son 379 horas x 20€/ hora que sería 7580€.

- Hardware

Para la realización del proyecto se utilizara un Mac Mini valorado en 600€, con una utilización del 80% y una vida media de 5 años. También se utilizará una tablet iPad Mini valorada en 330€ con una vida media de 3 años y un tiempo estimado de utilización del 60%.

Amortización anual del equipo: 600 / 5 = 125€
Amortización anual de la tablet: 330 / 3 = 110€

El proyecto lleva 34 semanas de trabajo, por lo que el gasto de amortización del hardware es:

(125 x 34 semanas x 0.80) / 52 semanas = 65,38€
(110 x 34 semanas x 0.60) / 52 semanas = 43,15€

Por lo tanto, los gastos de amortización del hardware son: 108,53€

- Software

En cuanto al software, el sistema operativo lo ofrece Apple de manera gratuita por lo que no se añadirá ningún coste. El Microsoft Office está valorado en 200€ con una utilidad del 50%, con una duración de vida de 4 años. Además, hay que pagar la licencia de desarrollador de Apple que son 80€ con solo un año de vida.

Amortización Microsoft Office: 200 / 4 = 50€
Amortización licencia desarrollador: 80 / 1 = 80€

(50 x 34 semanas x 0.60) / 52 semanas = 19,61€
(80 x 34 semanas x 1) / 52 semanas = 52,30€

Por lo tanto, los gastos de amortización del software son: 71,91€

- Materiales

Aproximadamente el gasto en otro tipo de materiales son 20€.

MOTIVO	**COSTE**
Coste personal	7.580€
Hardware	108, 53€
Software	71,91€
Materiales	20,00€
COSTE TOTAL	**7.780,44€**

Tabla 2: COSTE

Una vez analizado el coste total del trabajo de fin de grado, se va a proceder a realizar un estudio para estimar la recuperación del dinero invertido.

Para la venta de la aplicación se va a subir a la AppStore. Cualquier desarrollador que publique su aplicación tiene que saber que Apple se queda con el 30% de las ganancias y el otro 70% es para el desarrollador.

A continuación se calcularán los beneficios con los diferentes precios de venta y el número de descargas que se tienen que producir para recuperar el gasto total.

- Si el precio de venta es 0,99€ y el número de descargas son 10.000:
  - Ingresos = (10.000 x 0,99) –(10.000 x 0.99 x 30%) = 7.615,38€
  - Beneficios = Ingresos – Coste Total = 7615,35 – 7780,44= -165€
- Si el precio de venta es 1,99€ y el número de descargas son 8.000:
  - Ingresos = (8.000 x 1,99) –(8.000 x 1.99 x 30%) = 12.246,16€
  - Beneficios = Ingresos – Coste Total = 12,246,16 – 7780,44= 4.465,75€

Se han hecho estas dos estimaciones y como se puede comprobar, si el precio de venta es mayor, el número de descargas puede ser menor y el beneficio aumentará.

Por otro lado, creo que se debería de aumentar el precio de salida de la aplicación al principio para intentar recuperar lo antes posible el dinero invertido y una vez se ha conseguido el objetivo, disminuir el precio a la mitad.

2.7. Arquitectura

Para el almacenamiento de datos personales, (en cuanto a equipos y jugadores se refiere) se ha utilizado la base de datos interna SQLite para poder acceder de manera más rápida a los datos y la base de datos externa MySQL para las funciones que requieren interacción con otros usuarios.

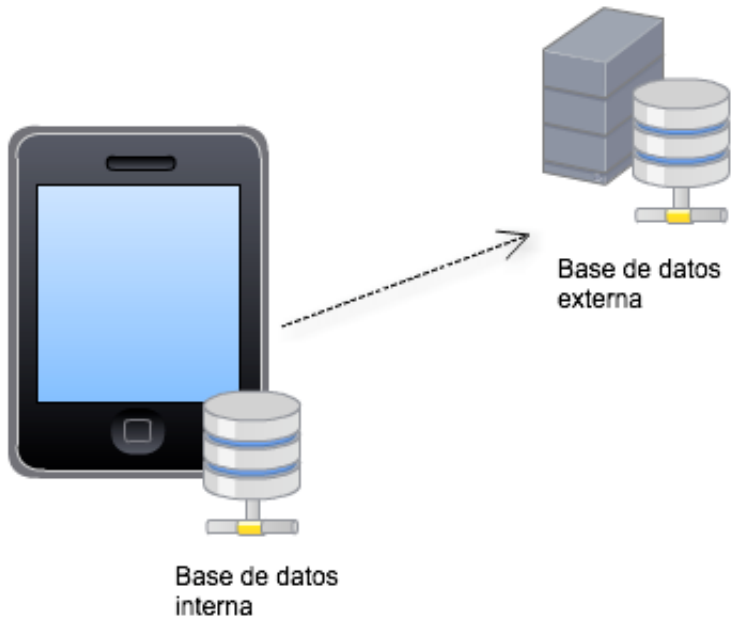


Ilustración 6: ARQUITECTURA

El inconveniente de tener las base de datos interna es que en el momento que se elimine la aplicación del dispositivo, los datos dejarán de estar disponibles y la base de datos volver a estar vacía.

En cuanto a la base de datos externa, estará alojada en un servidor y se accederá a ella mediante los archivos php que se han creado para ello. La mayor ventaja de tener acceso a una base de datos externa es la persistencia de los datos. A menos que se elimine un usuario del sistema, todos los datos almacenados por él estarán disponibles en la base de datos.

Si se decide publicar la aplicación en la AppStore, lo mejor sería que todo lo relacionado con el almacenamiento de bases de datos sea de manera externa. Así los usuarios registrados en el sistema podrían almacenar los datos sin preocuparse que van a ser eliminados en caso de eliminar de manera accidental o no la aplicación de nuestro dispositivo.

3. ANTECEDENTES

A continuación se procederá a explicar cómo es la situación actual del tema a tratar y las diferentes alternativas que se pueden encontrar.

Para llegar a la conclusión final de la aplicación se ha tenido que hacer una búsqueda de las aplicaciones que me pueden aportar ideas para realizar el proyecto.

Para este caso, las aplicaciones que se han buscado han sido de deporte, además de alguna de dibujo para realizar la pizarra. Aun siendo una aplicación de futbol, se ha seguido aplicaciones de deporte en general porque no solo en el ámbito futbolístico se almacena información sobre los jugadores y equipos.

Lo primero que se hizo fue entrar a la AppStore y empezar a buscar aplicaciones que me ofreciera algo similar a lo que se ha querido hacer en este proyecto. Después de varias horas de búsqueda, se llegó a la conclusión de que ninguna aplicación ofrecía tantas funcionalidades como se ha pretendido en esta aplicación. Lo que se encontró fueron aplicaciones con estadísticas de equipos, jugadores, partidos, etc. a nivel profesional y que no permitía añadir nada personalizado pero de las que se han sacado ideas como *The Football App, Athletic Club o Goles App.*

- *The Football App:* Es una aplicación de fútbol muy completa que permite acceder a fotos, videos, noticias, clasificaciones y seguir en directo los partidos que se estén disputando. Además, se pueden ver las estadísticas de los equipos y los jugadores de clubs profesionales.

- *Athletic Club:* Aplicación sobre el club vasco donde se pueden ver noticias sobre el equipo, estadísticas de la plantilla, partidos disputados, clasificación de la Liga BBVA, etc.

- *Goles App:* Esta aplicación muestra exclusivamente resultados, clasificaciones y calendario de las ligas más importantes del mundo.

El analizar estas aplicaciones y ver cómo han mostrado los datos en las interfaces me ha servido de gran ayuda a la hora de interpretar mis datos. A nivel visual ayuda mucho el hecho de ver como lo han hecho otras personas y saber que todos siguen un mismo patrón.

Después se buscaron aplicaciones que ofrecieran una pizarra para poder dibujar y en ese caso se encontraron varias. Se probaron varias aplicaciones y se eligió el diseño y las funcionalidades que se creyeron convenientes para el sistema. Las aplicaciones probadas fueron *Pizarra para iPad* y *Pizarra Fútbol.*

- *Pizarra para iPad:* Pizarra negra con tizas de colores para dibujar sobre ella. Lo que se puede realizar con esta aplicación es elegir un color, dibujar y borrar el contenido de la pizarra.

- *Pizarra fútbol:* Esta aplicación es muy completa para realizar tácticas de fútbol. Da la opción al usuario de elegir una pizarra en la que solo se puede dibujar u otra en la que se pueden añadir objetos como conos, balones, números, etc. que el usuario puede ir moviendo por la pantalla.

Gracias a estas aplicaciones pude probar cómo funcionaba una pizarra en mi tablet y así poder hacer algo parecido para que los usuarios que utilicen la pizarra para realizar tácticas lo haga de la manera más sencilla posible.

Después, para añadir más información de la que ya tenía, se pasó a buscar información externa al mundo de las aplicaciones y empezar a preguntar qué es lo que los usuarios querrían tener en sus dispositivos.

Al ser entrenador de un equipo de fútbol, las cosas fueron mucho más fáciles por que pude hablar con muchos entrenadores de categorías inferiores que almacenan este tipo de datos y me dieron pistas para reflejar esas ideas en el desarrollo.

Me contaron que datos son los que ellos apuntan en sus cuadernos para tener un control total de su equipo y que es lo que ellos hacen una vez iban pasando las semanas. Es muy engorroso ir sumando que jugador ha jugado un número exacto de minutos, cuantos goles ha marcado, etc. así que pensé que esos datos tenían que actualizarse de manera automática.

A parte de hablar con personas del mundo del fútbol, se preguntó a otras personas para que aportasen una idea de cómo harían ellos para hacer una aplicación de este tipo. Al final, se ha conseguido llegar hasta el objetivo final propuesto desde un principio gracias tanto a la búsqueda de información, como a la aportación de usuarios que podrían utilizar la aplicación en un futuro.

4. CAPTURA DE REQUISITOS

En esta fase se va a explicar cómo va a ser el desarrollo del proyecto y que necesidades va a cubrir a los usuarios.

4.1. Objetivos

Para el desarrollo de una aplicación de este tipo hay que tener muy claro lo que se va a desarrollar y que funcionalidades se van a programar para que el usuario cuando utilice la aplicación tenga claro en todo momento que es lo que puede hacer con ella y el uso que le puede dar.

Lo más importante en esta aplicación es el almacenamiento de datos ya que lo que se pretende es poder almacenar información tanto de equipos, jugadores, etc. como de ligas, además de poder mandar mensajes a otros usuarios.

En cuanto al tipo de usuarios para el que se va a desarrollar la aplicación son para entrenadores de fútbol, aunque no quita para que una persona con inquietudes futbolísticas pueda utilizar esta aplicación. El objetivo principal es poder tener datos actualizados tanto para el propio usuario, como para el resto que participe junto a él en ligas online.

En cuanto a las ligas se refiere, un usuario puede crear una liga real y compartirla con el resto, como crear una liga que servirá de referencia para videojuegos como Fifa o Pro Evolution Soccer. Las personas que utilicen esta aplicación para este fin, podrán realizar ligas y colgar sus resultados como si de una página web se tratase.

4.2. Casos de uso

A continuación, el diagrama de casos de uso que se ha seguido para realizar las funcionalidades de la aplicación. Se va a explicar los actores que participan, además de una descripción de cada caso de uso.

Se pueden diferenciar tres tipos de actores diferentes: *No registrado, Registrado y Usuario.*

- **No registrado**
    - *Registrarse:* Cualquier persona no registrada en el sistema podrá darse de alta para poder acceder a todos los contenidos que ofrece la aplicación.

- **Registrado**
    - *Iniciar Sesión:* Un usuario registrado previamente podrá acceder al sistema introduciendo el nombre de usuario y contraseña.

- **Usuario**

    - *Pizarra:* Permite al usuario utilizar una pizarra táctil donde se podrán realizar las diferentes tácticas que van a ser puestas en práctica durante un partido.

    - *Crear equipo:* Permite al usuario crear un equipo. Una vez creado el equipo se podrán añadir jugadores, partidos y estadísticas.

    - *Ver lista equipos:* El usuario podrá ver la lista de equipos que haya creado con anterioridad.

    - *Ver lista jugadores:* El usuario podrá ver la lista de jugadores que haya creado con anterioridad.

    - *Crear jugador:* Permite al usuario crear un nuevo jugador dentro del equipo seleccionado de la lista de equipos.

    - *Ver información jugador:* El usuario podrá ver la información correspondiente a un jugador, así como las estadísticas de los partidos en los que haya participado ese jugador.

    - *Ver información equipo:* El usuario podrá ver la información correspondiente a un equipo, así como las estadísticas de los partidos en los que haya participado ese equipo.

    - *Ver lista partidos:* El usuario podrá ver la lista de partidos que haya creado con anterioridad.

    - *Crear partido:* Permite al usuario crear un partido correspondiente a un equipo. Cuando se crea el partido con el resultado final, se actualizará de manera automática las estadísticas de ese equipo.

    - *Crear estadísticas partido:* Permite al usuario crear estadísticas individuales a cada jugador que participe en cada partido creado. Se añadirán datos como los minutos jugados, tarjetas amarillas, tarjeta roja, asistencias y goles. Una vez introducidas las estadísticas se irán actualizando de manera automática en cada jugador para tener también las estadísticas globales de cada uno.

    - *Ver estadísticas partido:* Permite al usuario ver estadísticas de cada jugador que ha participado en cada partido habiéndolas introducido previamente.

    - *Ver perfil:* El usuario podrá ver su perfil de usuario con los datos personales que haya rellenado en la ficha de registro.

- *Eliminar usuario:* El usuario podrá eliminar su cuenta del sistema y con ello se eliminarán también las ligas que haya creado, además de los mensajes recibidos.

- *Enviar mensaje:* El usuario podrá enviar un mensaje a un usuario que pertenezca al sistema para poder informarle de ligas creadas o simplemente para mandarle cualquier tipo de información.

- *Buscar usuario:* Un usuario podrá buscar a otros usuarios registrados previamente en el sistema.

- *Ver lista mensajes:* Un usuario podrá ver la lista de mensajes recibidos por otros usuarios.

- *Ver ligas:* Un usuario podrá ver una liga que haya creado él o que haya creado otro usuario.

- *Crear liga:* Un usuario podrá crear una liga en donde podrán participar más de un usuario.

- *Identificarse liga:* Un usuario solo podrá participar en una liga y modificar su contenido si dispone de la contraseña que la protege.

- *Crear equipo liga:* Un usuario podrá crear un equipo para que participe en una liga existente.

- *Ver clasificación:* Un usuario podrá ver la clasificación con todos los datos actualizados según los partidos que se hayan disputado en esa liga.

- *Ver lista jornadas:* Un usuario podrá ver la lista de jornadas que se hayan creado para una liga.

- *Crear jornada liga:* Un usuario podrá crear una jornada de una liga en la que participe.

- *Ver lista jornadas:* Un usuario podrá ver la lista de jornadas que se hayan creado para una liga.

- *Crear partido jornada:* Un usuario podrá crear un partido de una jornada que ya exista en la liga.

- *Salir:* Un usuario podrá abandonar la sesión iniciada.

Ilustración 7: CASOS DE USO

4.3. Modelo de dominio

En la aplicación se va a almacenar información sobre los *equipos* que un usuario haya creado, así como de los *jugadores*, *estadísticas* y *partidos*. Hay que tener en cuenta que a un equipo le pueden pertenecer partidos, estadísticas y jugadores, pero que cada una de ellas pertenecen a un mismo equipo. En un partido se pueden encontrar muchas estadísticas pero cada estadística pertenece a un único partido. Un jugador puede tener muchas estadísticas y cada una de ellas pertenece a un único jugador.

En cuanto a la entidad *usuarios,* se almacena la información de los usuarios que están dados de alta en el sistema. La entidad *mensajes* guarda información sobre los mensajes que un usuario haya recibido. Por lo tanto, un usuario puede tener muchos mensajes pero cada uno de ellos pertenece a un solo usuario.

En cuanto a las *ligas*, *equiposLiga*, *jornadas* y *partidos* cada entidad contiene información que los usuarios han ido introduciendo. Un usuario puede crear tantas ligas como quiera y en cada liga puede haber muchas jornadas, partidos y equipos pertenecientes a esa liga. Cada jornada puede disponer de muchos partidos y cada uno de ellos pertenece a una jornada en particular.

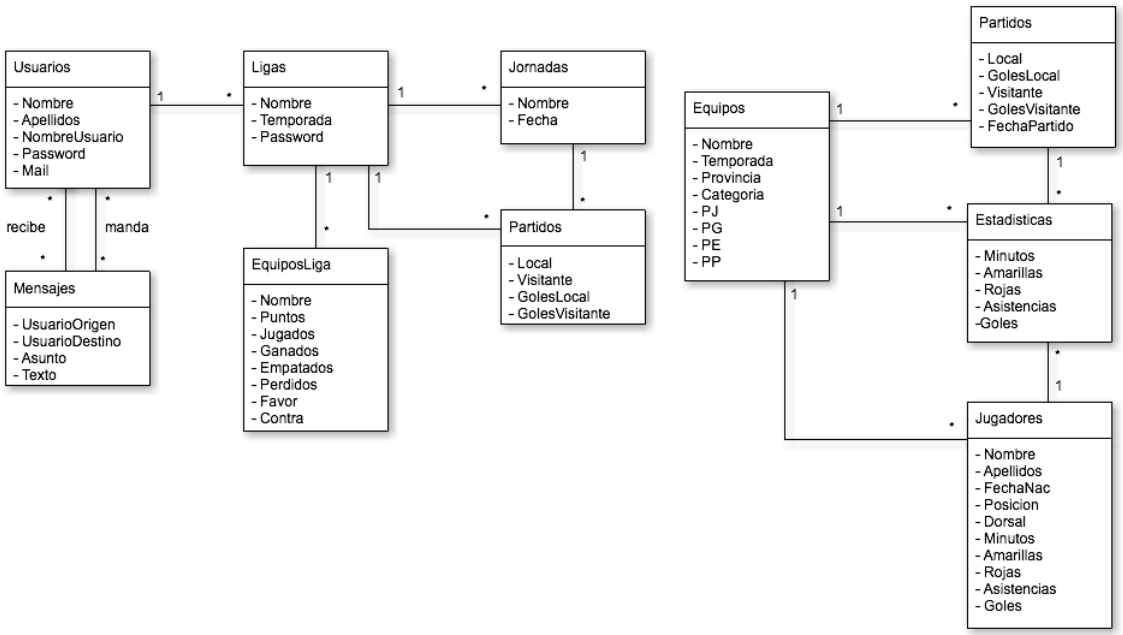


Ilustración 8: MODELO DE DOMINIO

5. ANÁLISIS Y DISEÑO

Para poder entender el diagrama de clases que a continuación se va a mostrar, hace falta conocer cómo funciona Objective-C, ya que lo hace de manera diferente al resto de lenguajes.

Cuando un desarrollador utiliza este lenguaje por primera vez, ve como hay similitud entre "C" y "Java" pero tiene grandes diferencias. Este lenguaje, es un lenguaje orientado a objetos, pero con la gran ventaja de tener más libertad a la hora de utilizar métodos de otros objetos.

Para entender el funcionamiento voy a poner un ejemplo, como puede ser la creación de un nuevo proyecto en el que aparezca una vista con un botón y que al pulsarlo muestre por pantalla el ya conocido *"Hola Mundo".*

Cuando se crea un proyecto en Xcode, para añadir botones, imágenes, texto, etc. a las interfaces hay que ir al *Storyboard* que será el lienzo en el que se hagan las vistas y las conexiones que tiene que haber entre ellas.

La primera vista, es la que se pone el botón que al pulsarlo me llevará a la siguiente vista con el texto *"Hola mundo".* A cada vista hay que asociarle una clase que será la que contendrá el código y hará las funciones yo quiera hacer.

Por eso en el diagrama de clases de la aplicación, existen clases como *Agregar liga, Listar Partidos, etc.* que lo que contiene es el código para llamar a las clases objeto y utilizar los métodos que se hayan creado para realizar esas funciones.

Una de las grandes ventajas es que Objective-C es totalmente compatible con "C" y se puede hacer uso de sus librerías.

5.1. Diagrama de clases

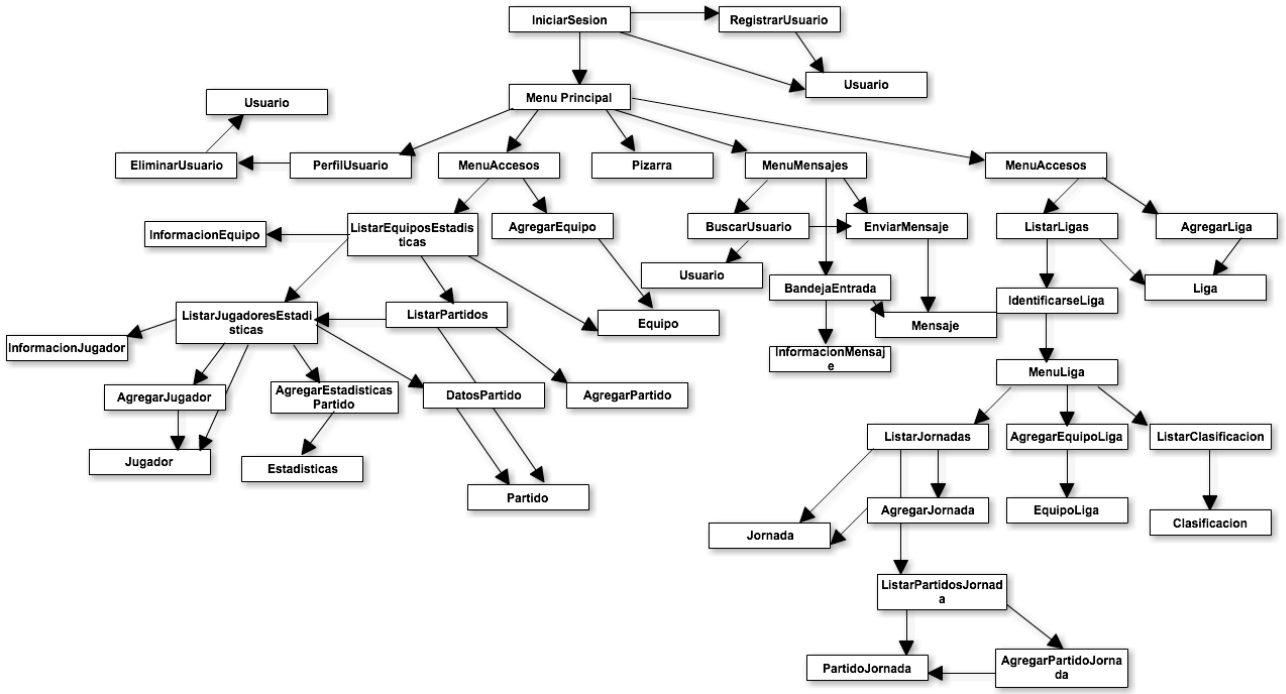


Ilustración 9: DIAGRAMA DE CLASES

5.2. Clases

En este apartado se va a explicar cada clase utilizada con detalle y que métodos se han utilizado.

AppDelegate

Esta clase se encarga de cargar la Base de Datos local (que previamente hemos creado con SQLite Manager) al iniciar la aplicación y que las clases que necesiten acceder a la base de datos puedan acceder para insertar, modificar o eliminar datos.

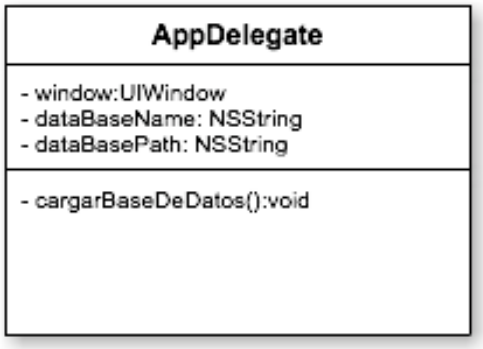


Ilustración 10: APP DELEGATE

Iniciar Sesión

Esta clase pertenece a la pantalla principal. Desde aquí se envían los datos para comprobar si el usuario existe en el sistema y poder acceder al Menú Principal de la aplicación. También se accede a la siguiente vista "Registrarse" mediante un botón para poder registrarnos en el sistema.

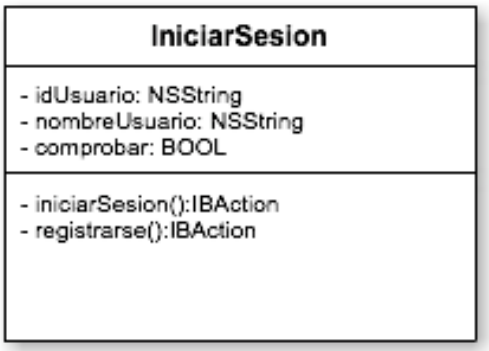


Ilustración 11: INICIAR SESIÓN

Registrar Usuario

Esta clase se encarga de recoger los datos necesarios para el registro de un usuario. Una vez introducidos los datos en los campos se utiliza el método registrarse que será en el encargado de llamar a la clase Usuario y proceder con el registro.

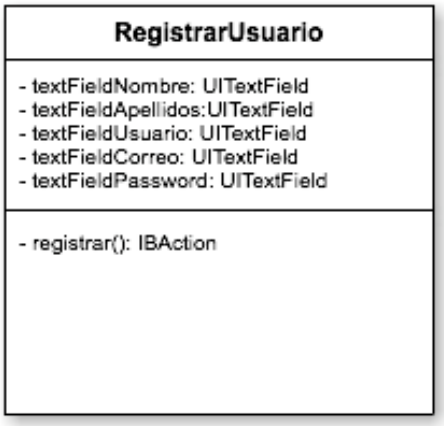


Ilustración 12: REGISTRAR USUARIO

Perfil Usuario

Es la clase encargada de mostrar toda la información relacionada con el usuario que ha iniciado sesión en el sistema. Aquí se manda desde el Menú Principal los datos del usuario obtenido para poder mostrarlos por pantalla.

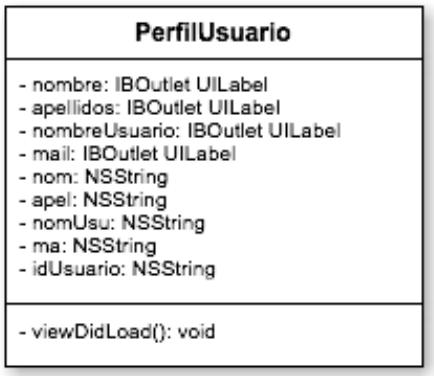


Ilustración 13: PERFIL USUARIO

Eliminar Usuario

Esta clase elimina del sistema al usuario que está activo en la aplicación, así como todas las ligas que el usuario haya creado y los mensajes que haya recibido por parte del resto de participantes del sistema.

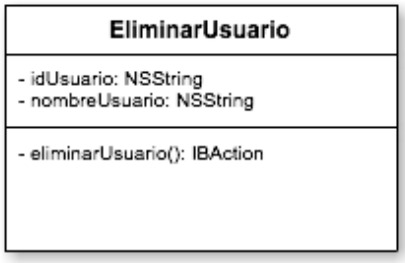


Ilustración 14: ELIMINAR USUARIO

Usuario

En esta clase están todos los métodos relacionados con los usuarios y se accederá a ella desde las diferentes clases que necesite de sus métodos para realizar las funciones que se exijan.

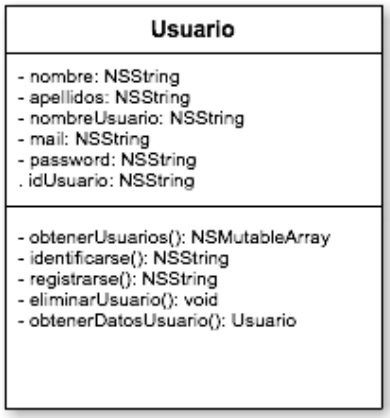


Ilustración 15: USUARIO

Menú Principal

Es la clase utilizada cuando entramos en el sistema una vez nos hayamos identificado correctamente. Desde aquí se manda información a otros controladores que necesiten información del usuario activo, como puede ser su nombre de usuario o el identificador que lo caracteriza.

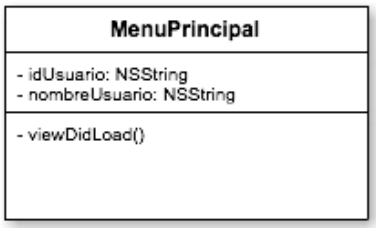


Ilustración 16: MENÚ PRINCIPAL

Pizarra

Es la clase encargada de hacer funcionar la pizarra táctil. Contiene los métodos para poder dibujar los trazos de las líneas con los diferentes colores, además de borrar todo lo que se haya dibujado en pantalla.

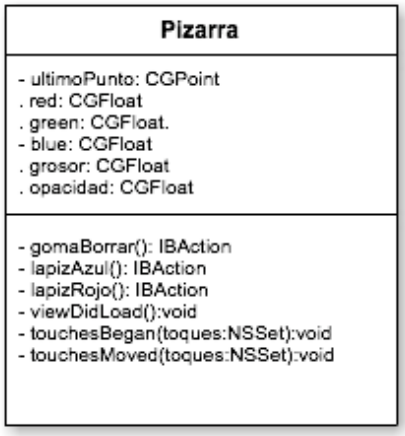


Ilustración 17: PIZARRA

Menú Accesos

Esta clase intermedia se ha creado para utilizarla con diferentes vistas y realizar las mismas funciones. El método *viewDidLoad()* realiza una serie de acciones automáticamente cuando accedemos a diferentes apartados pasando por este menú.

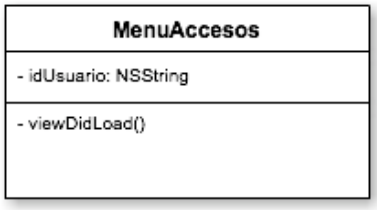


Ilustración 18: MENÚ ACCESOS

Listar Equipos Estadísticas

Esta clase es la encargada de cargar los equipos que el usuario haya creado. Esta clase se utiliza en varias vistas que necesita cargar los equipos para después poder acceder a los jugadores de cada equipo, ver los partidos que ha disputado o ver la información personal de cada equipo.

El método *commitEditingStyle* sirve para eliminar un equipo de la lista.

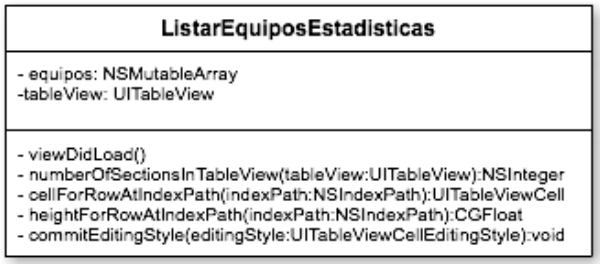


Ilustración 19: LISTAR EQUIPOS ESTADÍSTICAS

Agregar Equipo

Esta clase manda información introducida por el usuario para poder añadir un nuevo equipo. Una vez creado un equipo, se podrán crear partidos y jugadores para después añadir sus estadísticas.

Ilustración 20: AGREGAR EQUIPO

Información Equipo

Esta clase es la encargada de mostrar la información del equipo que se seleccione de la lista de equipos. Se le pasa toda la información del equipo creado anteriormente, además de las estadísticas que corresponda al equipo una vez haya disputado partidos.

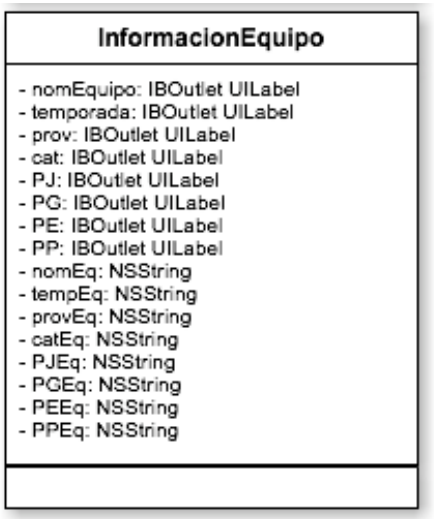


Ilustración 21: INFORMACIÓN EQUIPO

Equipo

En esta clase están todos los métodos relacionados con los equipos y se accederá a ella desde las diferentes clases que necesite de sus métodos para realizar las funciones que se exijan.

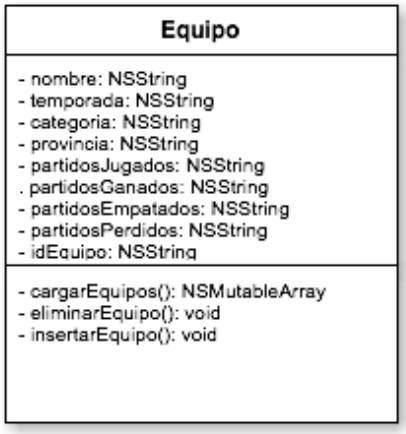


Ilustración 22: EQUIPO

Listar Jugadores Estadísticas

Esta clase es la encargada de cargar los jugadores que el usuario haya creado. Esta clase se utiliza en varias vistas que necesita cargar los jugadores para después poder acceder a las estadísticas generales, como a las particulares de cada partido que haya disputado y añadir estadísticas de cada partido que hayan jugado.

El método *commitEditingStyle* sirve para eliminar un jugador de la lista.

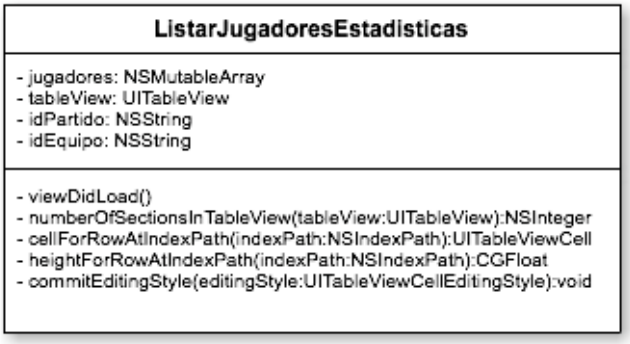


Ilustración 23: LISTAR JUGADORES ESTADÍSTICAS

Agregar Jugador

Esta clase manda información introducida por el usuario para poder añadir un nuevo jugador. Una vez creado un jugador, se podrán insertar estadísticas que hagan referencia a cada jugador que el usuario desee.

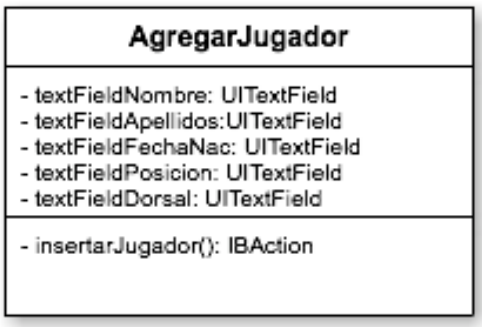


Ilustración 24: AGREGAR JUGADOR

Información Jugador

Esta clase es la encargada de mostrar la información del jugador que se seleccione de la lista de jugadores. Se le pasa toda la información del jugador creado anteriormente, además de las estadísticas que corresponda al jugador una vez haya disputado partidos.

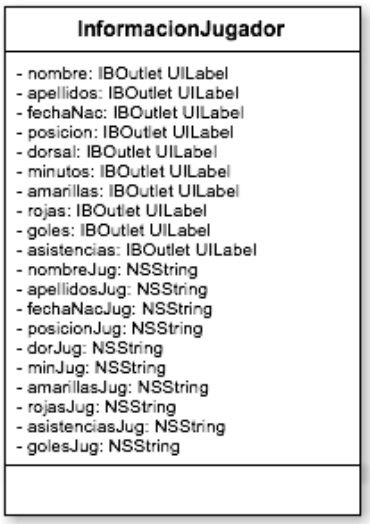


Ilustración 25: INFORMACIÓN JUGADOR

Jugador

En esta clase están todos los métodos relacionados con los jugadores y se accederá a ella desde las diferentes clases que necesite de sus métodos para realizar las funciones que se exijan.

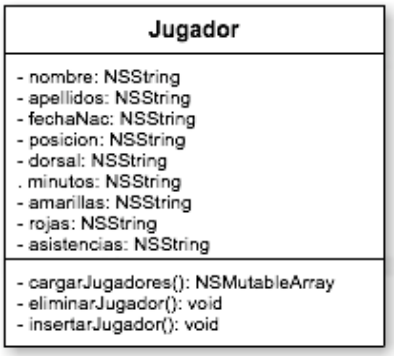


Ilustración 26: JUGADOR

Listar Partidos

Esta clase es la encargada de cargar los partidos que el usuario haya creado para cada equipo en particular.

El método *commitEditingStyle* sirve para eliminar un jugador de la lista.

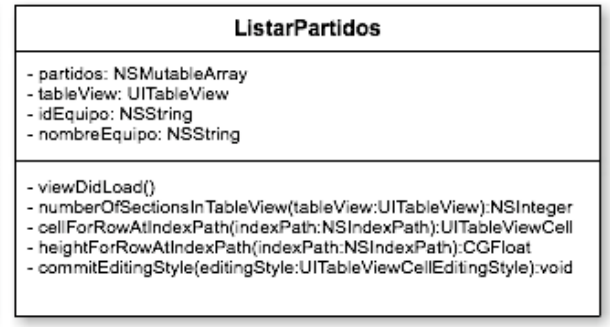


Ilustración 27: LISTAR PARTIDOS

Agregar Partido

Esta clase manda información introducida por el usuario para poder añadir un nuevo partido. Una vez creado un partido, se podrán insertar estadísticas de los jugadores que el usuario seleccione.

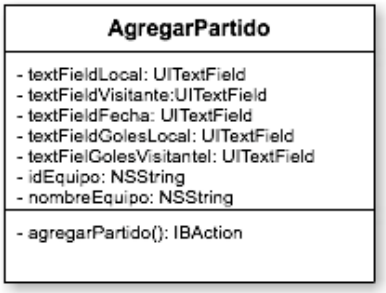


Ilustración 28: AGREGAR PARTIDO

Datos Partido

Esta clase es la encargada de mostrar las estadísticas que un jugador ha tenido en un partido determinado.

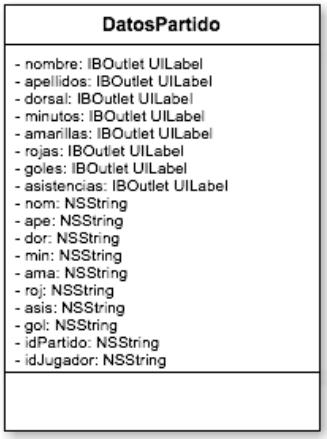


Ilustración 29: DATOS PARTIDO

Partido

En esta clase están todos los métodos relacionados con los partidos y se accederá a ella desde las diferentes clases que necesite de sus métodos para realizar las funciones que se exijan.

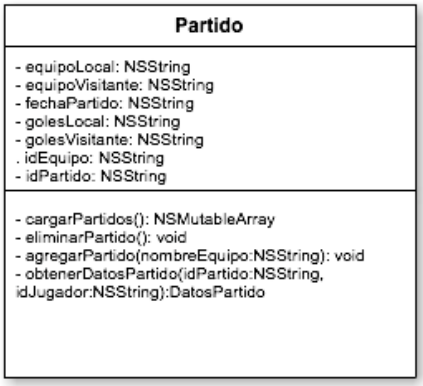


Ilustración 30: PARTIDO

Agregar Estadísticas Partido

Esta clase manda información introducida por el usuario para poder añadir una nueva estadística. Una vez creadas las estadísticas, se actualizarán las personales de cada jugador que haya participado.

Ilustración 31: AGREGAR ESTADÍSTICAS PARTIDO

Estadísticas

En esta clase está el método de agregar una nueva estadística personal de cada jugador que haya jugado un partido determinado.

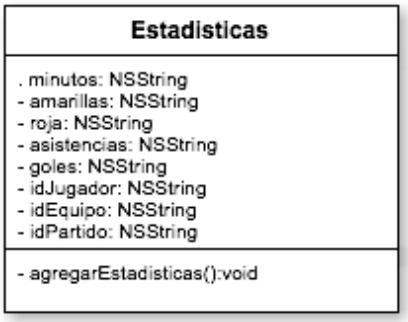


Ilustración 32: ESTADÍSTICAS

Menú Mensajes

Esta clase contiene información del usuario activo para poder acceder a las funcionalidades que ofrece como son el nombre de usuario y el identificador del mismo.

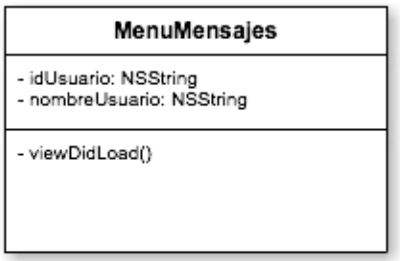


Ilustración 33: MENÚ MENSAJES

Bandeja Entrada

Esta clase es la encargada de cargar los mensajes que el usuario haya recibido por parte del resto de usuarios.

El método *commitEditingStyle* sirve para eliminar un mensaje de la lista.

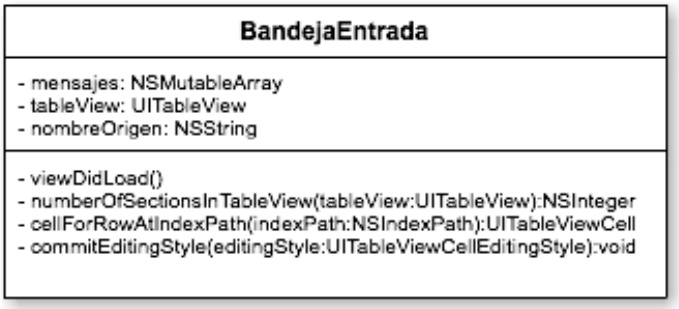


Ilustración 34: BANDEJA ENTRADA

Buscar Usuario

Esta clase nos muestra la lista de usuarios que están registrados en el sistema a través de la introducción de datos en un buscador que utiliza el método *textDidChange.*

Una vez aparecido el usuario que se quiera buscar, se le podrá enviar un mensaje pulsando sobre él, ya que nos mandará a la vista de Enviar Mensaje.

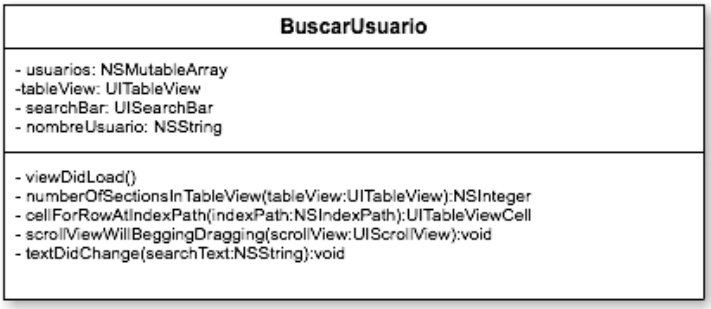


Ilustración 35: BUSCAR USUARIO

Enviar Mensaje

Esta clase será la encargada de enviar los datos que el usuario haya introducido para poder enviar un mensaje a otro usuario.

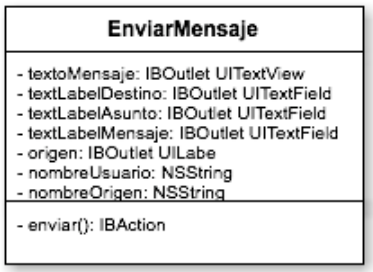


Ilustración 36: ENVIAR MENSAJE

Información Mensaje

Esta clase es la encargada de mostrar un mensaje en particular que se haya seleccionado de la Bandeja de Entrada.

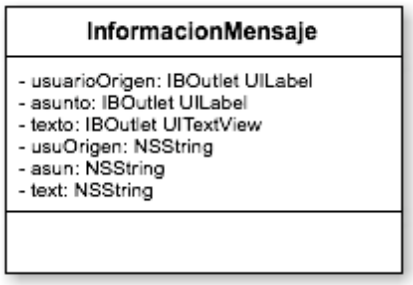


Ilustración 37: INFORMACIÓN MENSAJE

Mensaje

En esta clase están todos los métodos relacionados con los mensajes y se accederá a ella desde las diferentes clases que necesite de sus métodos para realizar las funciones que se exijan.

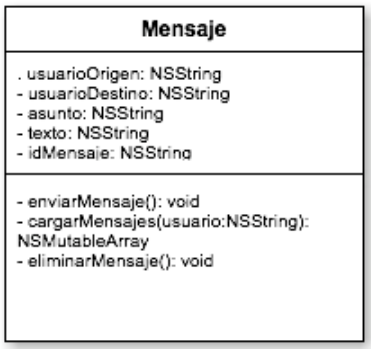


Ilustración 38: MENSAJE

Listar Ligas

Esta clase será la encargada de mostrar todas las ligas que hayan sido creadas por los usuarios.

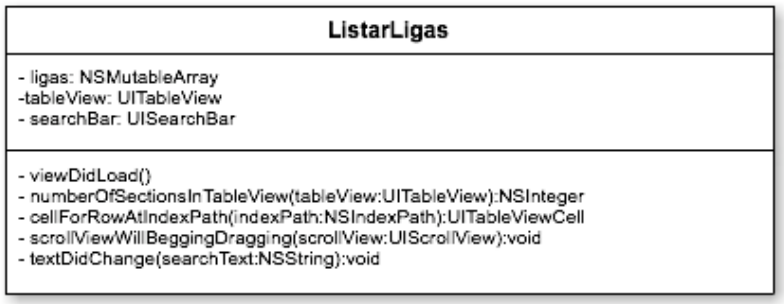


Ilustración 39: LISTAR LIGAS

Agregar Liga

Esta clase será la encarga de enviar los datos que el usuario haya introducido para crear una nueva liga.

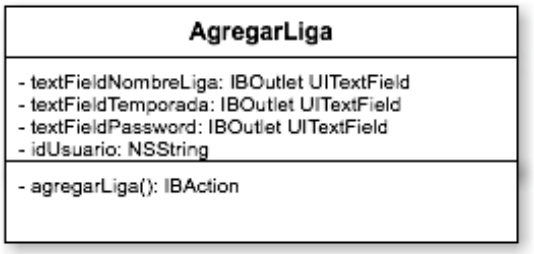


Ilustración 40: AGREGAR LIGA

Identificarse Liga

Esta clase se encarga de identificar el acceso a una liga y poder hacer modificaciones en ella. También se da la opción de entrar sin identificación y solo acceder a su contenido para poder verlo.

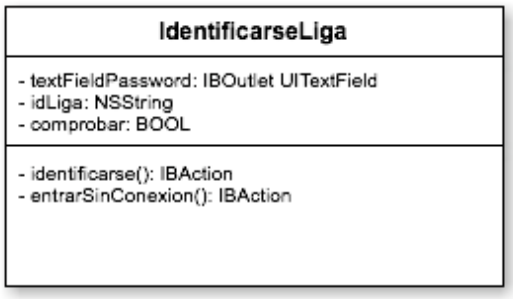


Ilustración 41:IDENTIFICARSE LIGA

Liga

En esta clase están todos los métodos relacionados con la liga y se accederá a ella desde las diferentes clases que necesite de sus métodos para realizar las funciones que se exijan.

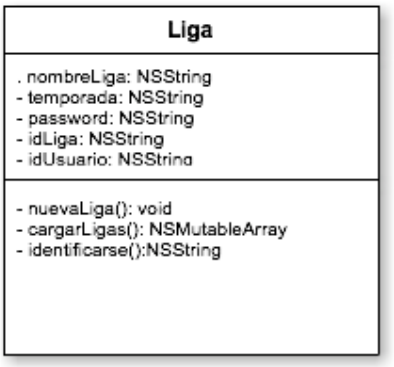


Ilustración 42: LIGA

Agregar Equipo Liga

Esta clase será la encargada de enviar los datos que el usuario haya introducido para crear un nuevo equipo de la liga en la que el usuario se haya identificado.

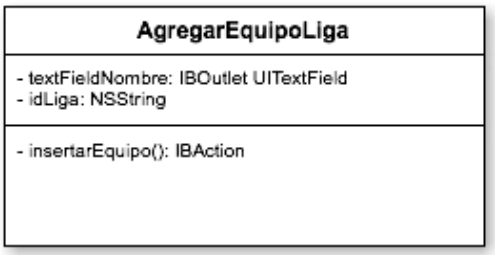


Ilustración 43: AGREGAR EQUIPO LIGA

Equipo Liga

En esta clase está el método que añade un nuevo equipo a la liga seleccionada.

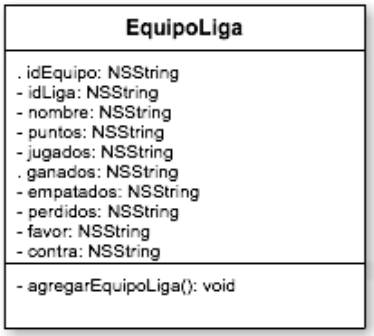


Ilustración 44: EQUIPO LIGA

Listar Jornadas

Esta clase será la encargada de mostrar todas las jornadas pertenecientes a una liga en particular.

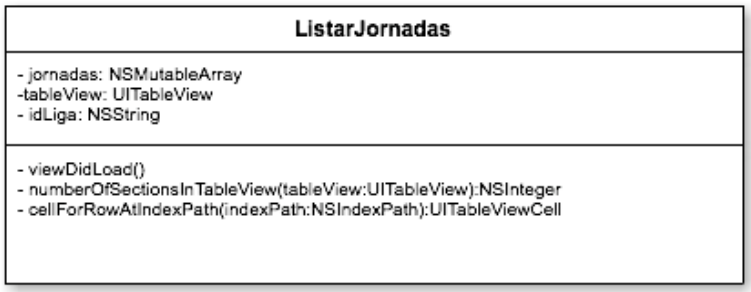


Ilustración 45: LISTAR JORNADAS

Agregar Jornada

Esta clase se encargará de enviar los datos que el usuario haya introducido para añadir una nueva jornada de una liga.

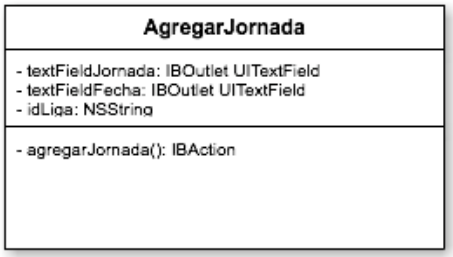


Ilustración 46: AGREGAR JORNADA

Jornada

En esta clase están los métodos relacionados con las jornadas de la liga y se accederá a ella desde las diferentes clases que necesite de sus métodos para realizar las funciones que se exijan.

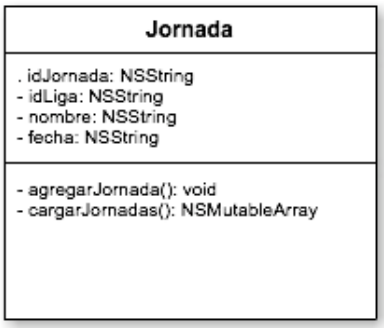


Ilustración 47: JORNADA

Listar Partidos Jornadas

Esta clase se encarga de mostrar los partidos relacionados a cada jornada que pertenece a cada liga en particular.

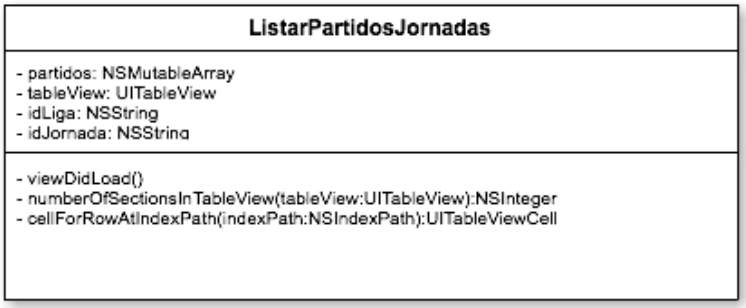


Ilustración 48: LISTAR PARTIDOS JORNADA

Agregar Partido Jornada

Esta clase se encargará de enviar los datos que el usuario haya introducido para añadir un nuevo partido de una jornada de liga.

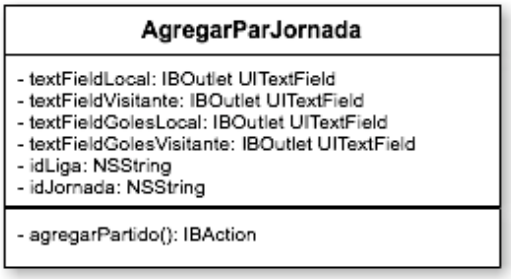


Ilustración 49: AGREGAR PARTIDO JORNADA

Partido Jornada

En esta clase están los métodos relacionados con los partidos de cada jornada de la liga y se accederá a ella desde las diferentes clases que necesite de sus métodos para realizar las funciones que se exijan.

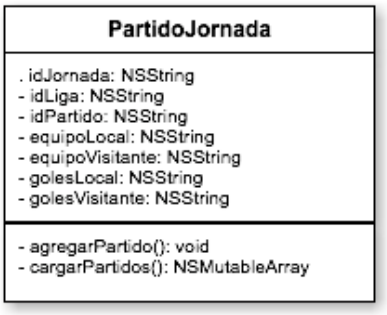


Ilustración 50: PARTIDO JORNADA

5.3. Diagrama relacional de bases de datos

En este apartado se muestra el diagrama relacional de bases de datos. En este diagrama se muestran las diferentes tablas que se han utilizado tanto para la base de datos interna, como para la base de datos externa. Se ha utilizado el color verde para la base de datos interna y el azul para la base de datos externa.

Las tablas de la base de datos interna son las de equipos, jugadores, estadísticas y partidos. Estas tablas se irán modificando según se vayan introduciendo datos en la aplicación, y como bien se ha explicado con anterioridad, estos datos solo estarán disponibles en el dispositivo en el que se utilice la aplicación.

Las tablas de la base de datos externa son las de usuarios, mensajes, equiposLiga, ligas, jornadas y partidos. En este caso todos los datos modificados estarán disponibles para todos los usuarios registrados en el sistema. Las relaciones están hechas de tal manera que en el caso de eliminar un usuario del sistema, no se quede ningún dato que él haya introducido "colgando". Es decir, si un usuario crea una liga, aunque participen otros usuarios por haber conseguido el password y ese usuario se elimina del sistema, la liga quedará eliminada del sistema con todo lo que le relaciona, además de los mensajes que ese usuario haya recibido.

En cada tabla se ha puesto subrayada cual es la clave primaria y qué relación tiene con las diferentes tablas. Es muy importante hacer bien las relaciones para que a la hora de modificar o eliminar un dato, lo haga correctamente.

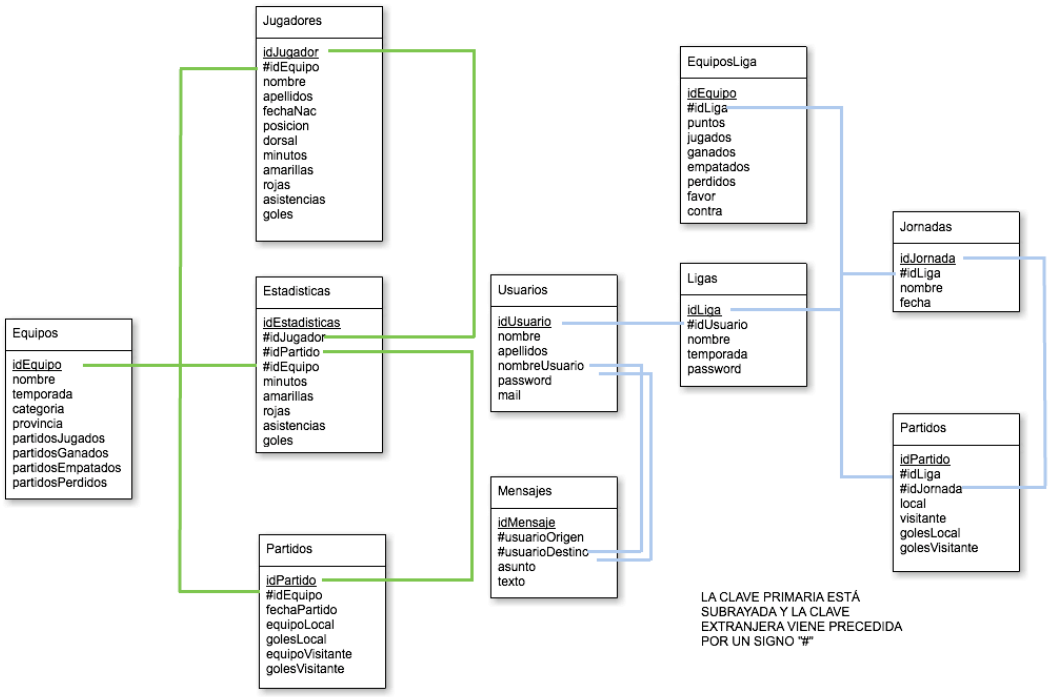


Ilustración 51: DIAGRAMA RELACIONAL BD

6. DESARROLLO

A continuación, se va a explicar la parte de desarrollo de la aplicación y que se ha hecho en cada apartado. Como se ha ido mencionando anteriormente, el lenguaje utilizado para el desarrollo ha sido Objective-C y el entorno de programación en Xcode.

En esta aplicación se va a hacer uso de bases de datos para el almacenamiento de datos. Para este tipo de aplicaciones es muy importante tener bien definidas las bases de datos, ya que lo que se pretende es tener la información guardada para poder acceder de la manera más rápida posible en caso de necesitarla.

Para la creación de la base de datos interna se ha utilizado la extensión ***SQLite Manager*** que proporciona el navegador ***Mozilla Firefox***. Es una herramienta que se instala a través del navegador y que permite la creación de una base de datos. Una vez creada, se crean las tablas con sus relaciones y se añade al proyecto que hemos iniciado con ***Xcode***.

Para poder utilizar este tipo de bases de datos, se utiliza la clase ***AppDelegate***, que va a ser la encargada de que se pueda utilizar en la aplicación. La extensión de la base de datos tiene que ser ***.sqlite*** y a través del método ***cargarBaseDeDatos()*** que se ha definido en la clase mencionada hace un momento, se cargará la base de datos en nuestro dispositivo una vez iniciada la aplicación.

Esta base de datos se utiliza para realizar funciones de almacenamiento local en la aplicación y permite guardar, modificar y eliminar datos de manera rápida y sencilla, ya que no tiene que acceder a ningún servidor externo para tratar la información. Todo lo que se vaya almacenando en la base de datos solo estará disponible para aquellos usuarios que utilicen el dispositivo y para ello se ha implementado varias funcionalidades.

El problema que he encontrado a la hora de utilizar este modo de almacenamiento ha sido que a la hora de querer eliminar varios datos en cascada, no lo soporta y ha habido que eliminar paso por paso con varias llamadas a la base de datos todo lo que hemos relacionado entre tablas.

Es un factor a tener en cuenta porque si todas las relaciones de las tablas están bien hechas y un usuario elimina un tipo de información con el fin de eliminar varias cosas en cascada, si no se da cuenta de ello se va a encontrar con información almacenada sin ningún sentido en su base de datos.

Para poder utilizar esta clase, hay que importarla en las clases en las que va a ser utilizada única y exclusivamente. Para importarla bastará con poner en la cabecera de la clase ***#import "AppDelegate.h"***.

Una vez determinado que datos son los que van a pertenecer a la base de datos interna, se procedió a buscar la forma de añadir una base de datos externa a

nuestra aplicación. Para ello, después de buscar la manera más eficiente me encontré con que existe una clase para hacer las conexiones a través de archivos php. La clase se llama ***CJSONDeserializer*** y como he mencionado, se utiliza para acceder a datos almacenados en una base de datos externa.

Lo que hace básicamente es hacer un parseo de un json y convertirlo en un ***NSDIctionary***. ***Json*** viene de "*JavaScript Object Notation*" y es un formato ligero de intercambio de datos. Por otra parte, NSDictionary es un diccionario en el que se puede almacenar información en forma de objeto y que cada objeto almacenado tiene una clave única. La clave única puede ser de cualquier tipo de objeto aunque por regla general suelen ser cadenas.

En el proceso para obtener los datos de las tablas lo que se hace es guardarlos en un json para después convertirlos en un NSDictionary y poder acceder a ellos de una manera que el programa interpreta.

Para ello, en la clases en las que se va a acceder a la base de datos se definen varias variables. Lo primero que se hace es crear un error por si a la hora de parsear ocurriese algún problema. Después hay que decir cuál es la dirección del servidor y el nombre del php desde el que se va a conectar. Finalmente se crea una variable ***NSData*** que sirve para empaquetar datos y una variable NSDictionary desde la que se va a llamar a CJSONDeserializer para obtener los datos.

```
NSError *error;
NSString *strURL = [NSString stringWithFormat:
@"http://galan.ehu.es/ygonzalez014/DAS/listaLigas.php?"];
NSData *dataURL = [NSData dataWithContentsOfURL:[NSURL
URLWithString:strURL]];
NSDictionary * dict = [[CJSONDeserializer
deserializer]deserializeAsDictionary:dataURL error:&error];
```

Estos pasos hay que realizarlos siempre que se quiera recoger información que esté almacenada en la base de datos externa y se quiera mostrar en algún apartado de la aplicación.

Esta clase, al igual que la utilizada para la base de datos interna, se utiliza importándola en las clases en las que va a ser utilizada escribiéndola en la cabecera con ***#import "CJSONDeserializer.h"***

Esto ha sido todo en cuanto a las bases de datos y ahora se va a explicar que se ha hecho con todos los datos que se han ido almacenando y de qué manera se han utilizado.

En la aplicación, se han hecho varios apartados en los que se muestran listas. Estas listas se cargan o bien de la base de datos interna o desde la base de datos externa.

En Objective-C existen dos maneras de poder recoger los datos y guardarlos en lo que se conoce en todos los lenguajes de programación como arrays. Los dos tipos a mencionar son los NSArray y los NSMutableArray.

Los array del tipo NSArray una vez instanciados, no será posible añadir o quitar objetos de ellos, aunque sí que se podrá acceder a dichos objetos. Por su parte, los NSMutableArray que son una subclase de NSArray, permiten acceder a los objetos de forma dinámica para poder eliminarlos o añadir otros nuevos.

En Objective-C, un array no contiene realmente los objetos, lo que se almacena es una referencia o un puntero asociado a cada uno de ellos y cuando se añade un objeto a un array, lo que se está guardando en el array será la dirección de memoria del objeto.

```
[array addObject:object]
```

La diferencia de la utilización de arrays en este lenguaje comparado con el resto es que se pueden tener objetos de distintos tipos almacenados, ya que solo se guardan los punteros asociados a cada uno de ellos y eso da mucha libertad al programador.

Una vez obtenidos los datos necesarios lo que se quiere hacer con ellos es mostrarlos y para eso se ha hecho el uso de tablas. Para las tablas se utilizan los denominados UITableView. Este componente es esencial ya que en mi caso muestro las plantillas de jugadores, los equipos creados, la clasificación, etc. Además, para personalizar cada celda de la tabla se han utilizado UITableViewCell, y sirve para personalizar la celda de la tabla de la manera que se crea conveniente.

Para empezar a trabajar con las tablas, primero se deben conocer muchos de los métodos propios que tienen estas tablas. Estos métodos en realidad pertenecen a UITableViewDelegate y UITableViewDataSource.

El delegate contiene la información de cómo se tiene que comportar la tabla a la hora de mostrarla con los datos obtenidos, como por ejemplo, el tamaño de las celdas, el método que se utiliza cuando se selecciona una de las celdas, etc. Por otro lado, el datasource contiene la información del contenido que tiene que mostrar la tabla, como el número de celdas, el número de secciones, el contenido de las celdas, etc.

Para entender estos métodos se van a ir explicando con detalle para el que se entienda lo que estos hacen. Se ha utilizado como ejemplo la lista de equipos creados.

Si se quieren mostrar las secciones que va a contener la tabla, se tiene que definir el número de secciones que va a contener la tabla.

```
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView{
   return 1;
}
```

Si lo que se quiere definir son el número de filas que tendrá cada sección de la tabla teniendo en cuenta el número de elementos que contienen los arrays, hay que utilizar el array con los datos obtenidos.

```
- (NSInteger)tableView:(UITableView *)tableView
numberOfRowsInSection:(NSInteger)section{
   return [_equipos count];
}
```

Para personalizar la tabla de los datos que se van a mostrar existe un método en el que se podrá definir cómo va a ser la celda que se va a mostrar en la tabla, que atributos va a mostrar, etc. Lo que se hace es instanciar la celda y utilizarla dentro de este método. Cada celda tiene un identificador para diferenciarlas del resto.

```
- (CeldaEquipo *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath{

   static NSString *CellIdentifier = @"Celda";

   Equipo *eq = [_equipos objectAtIndex:indexPath.row];

   CeldaEquipo *cell = [self.tableView
dequeueReusableCellWithIdentifier:CellIdentifier];
   if(cell==nil){
      cell = [[CeldaEquipo alloc]initWithStyle:UITableViewCellStyleDefault
reuseIdentifier:CellIdentifier];
   }
   cell.nombreEquipo.text = eq.nombre;
   return cell;
}
```

Además, se puede personalizar también el tamaño de la celda definiendo la altura que se le quiera dar.

```
- (CGFloat)tableView:(UITableView *)tableView
heightForRowAtIndexPath:(NSIndexPath *)indexPath{
   return 44;
}
```

Si lo que se quiere es eliminar un objeto de la tabla, existe un método que además es la única manera de poder eliminarlo manualmente. Este método una vez configurado permite al usuario deslizando el dedo sobre la celda seleccionada eliminar ese objeto en particular. Este se encargará de llamar al método adecuado para eliminar su contenido de la base de datos tanto interna, como externa.

```
-(void) tableView:(UITableView *)tableView
commitEditingStyle:(UITableViewCellEditingStyle)editingStyle
forRowAtIndexPath:(NSIndexPath *)indexPath{

   Equipo *eq=[_equipos objectAtIndex:indexPath.row];
```

```
   [_equipos removeObjectAtIndex:indexPath.row];
   [tableView deleteRowsAtIndexPaths:[NSArray
arrayWithObject:indexPath]withRowAnimation:UITableViewRowAnimationAutomati
c];

   [eq eliminarEquipo];

}
```

Estos han sido muchos de los métodos existentes que se han utilizado en el desarrollo de la aplicación aunque existan muchos más para realizar diferentes funciones.

En la aplicación, aparte de lo ya mencionado, se muestra información particular de un objeto como puede ser el caso de un equipo. Para ver la información de un equipo, si el usuario elige un equipo de la lista de equipos ya obtenida, hay que utilizar un método para pasarle esos datos.

Cuando se crean las conexiones entre vistas, lo que se hace es asignar identificadores. En este método hay que instanciar cual va a ser la clase de destino y los datos que se van a enviar para que lo muestre en otra vista.

```
-(void) prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender{
   NSIndexPath *indexPath = [self.tableView indexPathForCell:sender];
      Equipo *eq = [_equipos objectAtIndex:indexPath.row];
        if([[segue identifier] isEqualToString:@"datosEquipo"]){
      InformacionEquipo *info = [segue destinationViewController];
      info.nomEq=eq.nombre;
      info.catEq=eq.categoria;
}
}
```

En resumen, lo que se pretende es que una vez hemos cargado la lista de equipos, se seleccione un equipo del que ya tenemos toda la información sobre él y se envíen los datos que se crean convenientes para mostrar en la siguiente vista.

Como bien se ha nombrado esta aplicación está basada sobre todo en la posibilidad de almacenar datos de equipos, jugadores, partidos, etc. Para ello dependiendo de la funcionalidad se hace uso de la base de datos interna o de la base de datos externa. A continuación se explicará como lo hace cada una de ellas.

En el caso de los equipos, se utiliza SQLite para el almacenamiento de los datos. Se accede a la vista correspondiente navegando mediante los menús, en la que se tendrá que rellenar los UITextField que aparezcan y pulsando el botón aceptar se añadirá un nuevo equipo.

A partir de aquí el controlador manda los datos correspondientes que se van a añadir en la base de datos a la clase Equipo, que es la que contiene el método para añadir un nuevo equipo.

Lo primero que se hace es instanciar la clase AppDelegate que se ha explicado

antes para poder tener acceso a la base de datos. Si todo va bien y se realiza el acceso sin ningún problema, se creará la sentencia para añadir un nuevo equipo y se accederá a la base de datos para modificar la tabla con los datos recibidos por el controlador.

A parte de eso, se ha añadido un UIAlertVIew que es una alerta que se muestra en caso de que el equipo se haya añadido correctamente. Después, se cierra la base de datos y el dato habrá sido añadido correctamente.

```
-(void)insertarEquipo{
   AppDelegate *appDelegate = (AppDelegate *)[[UIApplication
sharedApplication]delegate];

   sqlite3 *database;
   sqlite3_stmt *sentencia;

   if (sqlite3_open([appDelegate.dataBasePath UTF8String], &database)==
SQLITE_OK){

   const char *sql1 = "insert into equipos
(nombre,temporada,categoria,provincia,partidosJugados,partidosGanados,partidosE
mpatados,partidosPerdidos) VALUES (?,?,?,?,0,0,0,0)";

if(sqlite3_prepare_v2(database, sql1, -1, &sentencia, NULL)==SQLITE_OK){
      NSLog(@"entra");
      sqlite3_bind_text(sentencia, 1, [self.nombre UTF8String], -1, SQLITE_TRANSIENT);
      sqlite3_bind_text(sentencia, 2, [self.temporada UTF8String], -1,
SQLITE_TRANSIENT);
      sqlite3_bind_text(sentencia, 3, [self.categoria UTF8String], -1,
SQLITE_TRANSIENT);
      sqlite3_bind_text(sentencia, 4, [self.provincia UTF8String], -1,
SQLITE_TRANSIENT);

UIAlertView *alerta= [[UIAlertView alloc]init];
      [alerta setTitle:@"Correcto"];
      [alerta setMessage:@"Equipo añadido"];
      [alerta addButtonWithTitle:@"Aceptar"];

[alerta show];
      while(sqlite3_step(sentencia) == SQLITE_OK){
}
   }else{
      NSLog(@"Error en la creacion del insert" );
   }
      sqlite3_finalize(sentencia);
}else{
   NSLog(@"no se ha podido abrir la BD" );
}
sqlite3_close(database);
}
```

Para el caso de la base de datos externa se va poner el ejemplo de añadir una

nueva liga. Al igual que en el caso anterior, aquí también se envían los datos a la clase que los está solicitando para añadirlos después.

Lo primero es crear un NSString que contenga la dirección URL en la que está el php que contiene la sentencia para acceder a la base de datos. Se le mandan los parámetros recibidos (tantos como sean necesarios y hayamos definido en el php). Después, se ejecuta el php y este devolverá un resultado que se mostrará en consola para saber si la sentencia se ha ejecutado correctamente.

```
-(void)nuevaLiga{

NSString *strURL = [NSString
stringWithFormat:@"http://galan.ehu.es/ygonzalez014/DAS/ligas.php?nombre=%@
&temporada=%@&password=%@&idUsuario=%@",self.nombreLiga,self.temporada,
self.password,self.idUsuario];
NSString *strURL2 = [strURL stringByReplacingOccurrencesOfString:@" "
withString:@"$$"];
NSData *dataURL = [NSData dataWithContentsOfURL:[NSURL
URLWithString:strURL2]];
NSString *strResult = [[NSString alloc] initWithData:dataURL
encoding:NSUTF8StringEncoding];
NSLog(@"%@", strResult);
}
```

Estas son las dos opciones que tenemos de añadir datos a nuestras bases de datos.

Ahora se va a hablar de los usuarios y de cómo se pensó en el acceso a la aplicación.

Para acceder a esta aplicación se ha creído conveniente la inserción de usuarios en el sistema. Al principio se pensó en hacer la aplicación para uso de cualquiera sin tener restricciones pero cuando se introdujo la parte de creación de ligas online y el servicio de mensajería, hacía falta realizar este apartado.

Cuando un usuario accede por primera vez a la aplicación, se ve de manera muy clara cuales son las dos opciones que tiene, que son iniciar sesión o registrarse. Para ello, al igual que en el apartado anterior de introducir nuevos equipos, jugadores, ligas, etc. se sigue el mismo proceso de inserción.

La diferencia es que a la hora de realizar el registro de un usuario se comprueba si el nombre de usuario que se está insertando existe ya en el sistema y se avisa en caso de existir con un UIAlertView para que sea sustituido por otro.

En el registro y para posteriormente acceder al sistema se pide que se introduzca una contraseña. Para la seguridad de los usuarios una vez se introduzca en el sistema, esta quedará encriptada y cuando un usuario inicie sesión y mediante el php se envié la contraseña introducida en el UITextField de la vista de iniciar sesión, esta se encriptará también y se comparará con la almacenada en la tabla de usuarios de la base de datos.

Una vez dentro del sistema, el usuario puede acceder a una sección llamada "Perfil" en la que se puede ver sus datos personales, además de eliminar por completo el usuario del sistema.

Si se decide eliminar un usuario del sistema, se eliminarán todas las ligas que ese mismo usuario haya creado y los mensajes que haya recibido por parte de otros usuarios.

Como hemos dicho, un usuario puede crear ligas y esta es una de las partes más importantes, ya que aquí entra en juego la participación con otros usuarios. Aquí se da la opción de crear nuevas ligas, participar en una ya existente o simplemente curiosear ligas de otros usuarios.

Para ello, el usuario tiene la opción de crear una nueva liga introduciendo un nombre para la liga, el año y un password. En caso de querer que otros usuarios participen tendrá que compartir el password mediante el servicio de mensajería que se ha creado y que después se explicará.

Una vez dentro del apartado de ligas, se le da al usuario la posibilidad de buscar una liga que haya creado o que ya exista en el sistema y pulsará sobre la celda correspondiente de la lista de ligas. Una vez realizados estos pasos se llega a una vista en la que el usuario tiene la opción de si tiene el password identificarse y poder modificar datos de ella, o simplemente entrar a curiosear el contenido de la misma.

Aquí surgieron problemas. Desde el principio se desarrolló este apartado con el acceso a las ligas sin identificación y que cualquier usuario pudiera modificar lo que le diera la gana. Esto no era viable ya que si hay usuarios que participan en una liga no tienen por qué verse envueltos en problemas de datos si un usuario cualquiera decide alterar los datos de la competición de manera errónea.

Los métodos utilizados tanto en la inserción, eliminación o de cómo mostrar los datos que previamente se han cargado, se harán de la misma manera que se ha explicado anteriormente.

Para la comunicación entre los diferentes usuarios del sistema se ha creado un servicio de mensajería al que podemos denominarlo como un "email" dentro de la aplicación.

Como la aplicación da la oportunidad a los usuarios de crear ligas online y participar modificando los datos de ellas para crear un entorno en el que los usuarios son los protagonistas, se ha creado este sistema para que la comunicación sea desde dentro de la misma aplicación.

Para poder enviar un mensaje a un usuario hace falta que el usuario esté dado de alta en el sistema. Lo que el usuario ve cuando entra a la opción de buscar usuario es una tabla en blanco y un buscador. El objetivo es que según vaya introduciendo el nombre de un usuario si empieza por la letra "S", vayan

apareciendo en la tabla todos los usuarios que contengan esa letra en su nombre de usuario. Una vez encontrado, se puede pulsar sobre la celda del usuario y automáticamente se nos enviará a una vista en la que podemos enviarle a un usuario un mensaje.

```
-(void)searchBar:(UISearchBar *) searchBar textDidChange:(NSString *)searchText{
 [usuariosSel removeAllObjects];
 for (int i=0;i<_usuarios.count;i++){
 Usuario *usua = [_usuarios objectAtIndex:i];
NSRange nameRange = [usua.nombreUsuario rangeOfString:searchText
options:NSCaseInsensitiveSearch];
        if(nameRange.location !=NSNotFound){
         [usuariosSel addObject:[_usuarios objectAtIndex:i]];
      }
   }
   [self.tableView reloadData];
}
```

Además de esto, el usuario dispondrá también de una bandeja de entrada donde podrá ver todos los mensajes que haya recibido, además de tener la opción de eliminar mediante el método explicado anteriormente los mensajes que se hayan leído.

Por último, se ha desarrollado una pizarra táctil en la que los usuarios pueden dibujar sus tácticas. Desde el principio se pensó en aprovechar las dimensiones de la tablet para la realización de una pizarra. Es muy cómodo para los entrenadores poder mostrar a sus jugadores lo que quieren que hagan sin salir de la aplicación.

Una vez pensado como iba a ser, se creó una clase Pizarra en la que se definieron los métodos que después se iban a implementar. Como toda pizarra, lo que se buscaba era tener una serie de colores y un sistema que borrase todo el contenido de la misma.

Buscando información de cómo hacerlo, me di cuenta de cómo tenían que ser los métodos. Lo primero era definir una serie de métodos para crear los lápices de colores y la goma de borrar. En ellos se ha definido tanto el grosor de la línea como su opacidad.

Para la zona de dibujo se ha utilizado una UIImageVIew, que es una imagen transparente en la que se va a mostrar las líneas que el usuario trace. Para realizar esos trazos, se crea un UITouch que sirve para representar un movimiento de un dedo en la pantalla para un evento en particular. Después, se crea la variable último punto que se inicializa en el punto donde el dedo toca la pantalla.

```
-(void)touchesBegan:(NSSet *)toques withEvent:(UIEvent *)event{
UITouch *toque = [toques anyObject];
 CGPoint ultimoPunto = [toque locationInView:self.view];
}
```

Una vez hecho esto, hay que interpretar el movimiento del dedo cuando este

está deslizándose por la pantalla. Para ello se hace uso de un método que merece la pena explicarlo para poder entenderlo. Lo que se hace en él es que cuando toquemos desde un punto de partida a un punto final, se dibuje la línea tal y como la hayamos dibujado con el dedo. Además, según el grosor y la opacidad definida, lo hará de esa manera. Para finalizar, guarda como último punto el punto actual para saber dónde finaliza el trazo.

```
-(void)touchesMoved:(NSSet *)toques withEvent:(UIEvent *)event{

   UITouch *toque = [toques anyObject];
   CGPoint puntoActual = [toque locationInView:self.view];


   UIGraphicsBeginImageContext(self.view.frame.size);
  [self.zonaDibujo.image drawInRect:CGRectMake(0, 0, self.view.frame.size.width,
self.view.frame.size.height)];


   CGContextMoveToPoint(UIGraphicsGetCurrentContext(), _ultimoPunto.x,
_ultimoPunto.y);
   CGContextAddLineToPoint(UIGraphicsGetCurrentContext(), puntoActual.x,
puntoActual.y);


   CGContextSetLineCap(UIGraphicsGetCurrentContext(), kCGLineCapRound);
   CGContextSetLineWidth(UIGraphicsGetCurrentContext(), _grosor);
   CGContextSetRGBStrokeColor(UIGraphicsGetCurrentContext(), _red, _green, _blue,
1.0);
   CGContextSetBlendMode(UIGraphicsGetCurrentContext(), kCGBlendModeNormal);
  CGContextStrokePath(UIGraphicsGetCurrentContext());
   self.zonaDibujo.image = UIGraphicsGetImageFromCurrentImageContext();
   [self.zonaDibujo setAlpha:_opacidad];
   UIGraphicsEndImageContext();

   _ultimoPunto = puntoActual;
}
```

En este apartado se han explicado las partes más importantes de la fase de desarrollo.

7. VERIFICACION Y EVALUACIÓN

En este apartado de la memoria se va a proceder a especificar las pruebas que se han ido realizando para comprobar que la aplicación funciona correctamente y cumple con las funcionalidades que se han descrito anteriormente.

Todas estas pruebas se han ido realizando de manera periódica, pero una vez terminada la aplicación se va a comprobar que siguen superándose todas las pruebas de manera satisfactoria.

Si alguna de las pruebas ha resultado fallida, se explicará el motivo de cada una de ellas.

La siguiente tabla se va a utilizar para exponer las pruebas de una forma más clara:

ID	PROPÓSITO	PASOS	RESULTADO

Tabla 3: EJEMPLO PRUEBAS

- ID: es un identificador único de la prueba, el cual comenzará en 1 e irá aumentando en una unidad por cada prueba realizada.
- PROPOSITO: descripción de la prueba y lo que se espera conseguir al realizar dicha prueba.
- PASOS: pequeñas pautas que sirve para ayudar al usuario a realizar la prueba.
- RESULTADO: indica si la prueba se ha sido superada satisfactoriamente y puede tener dos valores: éxito o fracaso.

1. Registrarse

ID	PROPÓSITO	PASOS	RESULTADO
1.1	Abrir la aplicación y que pulsando en el botón de registrarse cambie de vista	1.Abrir aplicación 2.Pulsar botón Registrarse	Éxito
1.2	Rellenar todos los datos y registrar a un usuario en el sistema	1.Rellenar todos los campos sin dejar ninguno en blanco 2.Pulsar botón Aceptar	Éxito
1.3	Rellenar todos los datos y que de error en caso de existir el nombre de usuario	1.Rellenar todos los campos sin dejar ninguno en blanco 2.Poner un nombre de usuario ya existente	Éxito

Tabla 4: REGISTRARSE

2. Iniciar sesión

ID	PROPÓSITO	PASOS	RESULTADO
2.1	Abrir la aplicación y que rellenado los datos correctamente inicie sesión.	1.Abrir aplicación 2.Introducir nombre de usuario y contraseña y pulsar botón Aceptar	Éxito
2.2	Abrir aplicación e introducir un nombre de usuario y contraseña erróneo para que salga la alerta	1.Abrir aplicación 2.Introducir un nombre de usuario y contraseña erróneo	Éxito

Tabla 5: INICIAR SESIÓN

3. Equipos

ID	PROPÓSITO	PASOS	RESULTADO
3.1	Crear un equipo y ver que aparece en la lista de equipos	1.Pulsar icono Equipos del menú principal 2.Pulsar icono Crear Equipo 3.Rellenar datos y pulsar botón Aceptar 4.Pulsar icono Mis Equipos y ver que se ha creado correctamente	Éxito
3.2	Ver la información de un equipo en particular	1.Pulsar el icono Estadísticas del menú principal 2.Pulsar el icono Info 3.Seleccionar un equipo de la lista de equipos 4.Ver la vista con toda la información sobre el equipo seleccionado	Éxito
3.3	Eliminar un equipo de la lista de equipos	1.Pulsar icono Equipos del menú principal 2.Pulsar el icono Mis Equipos 3.Deslizar el dedo sobre el equipo seleccionado y pulsar eliminar 4.Ver si el equipo ha sido eliminado de la lista de equipos	Éxito

ID	PROPÓSITO	PASOS	RESULTADO
3.4	Eliminar un equipo de la lista de equipos y comprobar en la base de datos que no se queda ningún dato relacionado sin eliminar	1.Eliminar un equipo de la lista de equipos. 2.Acceder a la base de datos y comprobar si todo se ha eliminado correctamente	Fallo

Tabla 6: EQUIPOS

Este fue prácticamente un error que no entendía el por qué ocurría cuando las relaciones de las tablas de la base de datos interna estaban bien hechas. El problema viene de que en SQLite como he comentado con anterioridad, no funciona la eliminación de datos en cascada y hay que hacer las llamadas individuales para cada dato que se quiera eliminar.

Una vez solventado este fallo me sirvió para las demás ocasiones en las que se eliminan datos relacionados.

4. Jugadores

ID	PROPÓSITO	PASOS	RESULTADO
4.1	Crear un jugador para un equipo concreto	1.Seleccionar un equipo de la lista de equipos 2.Pulsar el botón + 3.Rellenar todos los campos 4.Pulsar el botón Aceptar 5.Ver que el jugador aparece en la lista de jugadores de ese equipo	Éxito
4.2	Ver información particular de un jugador	1.Seleccionar un equipo de la lista de equipos 2.Seleccionar un jugador de la lista de jugadores 3.Ver la vista con toda la información del jugador seleccionado	Éxito

ID	PROPÓSITO	PASOS	RESULTADO
4.3	Eliminar un jugador de un equipo	1.Seleccionar un equipo de la lista de equipos 2.Deslizar el dedo sobre el jugador seleccionado y pulsar Eliminar 3.Ver que el jugador no aparece en la lista de jugadores	Éxito
4.4	Crear un jugador y recargar la tabla de la lista de jugadores	1.Crear un nuevo jugador y pulsar Aceptar 2.Deslizar el dedo hacia abajo en la lista de jugadores	Fallo

Tabla 7: JUGADORES

Este fallo lo que hacía a mi aplicación era que cuando recargaba la lista de jugadores, desaparecían todos y me mostraba una lista en blanco. El problema era que el NSMutableArray que obtenía los jugadores una vez se añadía uno nuevo lo dejaba vacío sin darme cuenta y no me mostraba nada por pantalla.

5. Partidos

ID	PROPÓSITO	PASOS	RESULTADO
5.1	Elegir un equipo y crear un partido	1.Pulsar el icono Estadísticas del menú principal 2.Pulsar el icono Partidos 3.Elegir un equipo de la lista de equipos 4.Pulsar el botón + 5.Rellenar todos los campos y pulsar Aceptar 6.Ver que se ha creado el partido en la lista de partidos	Éxito

ID	PROPÓSITO	PASOS	RESULTADO
5.2	Eliminar un partido de la lista de partidos	1.Pulsar el icono Estadísticas del menú principal 2.Pulsar el icono Partidos 3.Deslizar el dedo sobre el partido seleccionado y pulsar Eliminar 4.Ver que el partido se ha eliminado de la lista de partidos	Éxito

Tabla 8: PARTIDOS

6. Pizarra

ID	PROPÓSITO	PASOS	RESULTADO
6.1	Seleccionar uno de los colores y dibujar cualquier cosa sobre la pizarra	1.Pulsar el icono Pizarra del menú principal 2.Seleccionar uno de los colores existentes 3.Tocar la pantalla con el dedo haciendo movimientos	Éxito
6.2	Borrar un dibujo realizado en la pizarra	1.Pulsar el icono Pizarra del menú principal 2.Seleccionar uno de los colores existentes 3.Tocar la pantalla con el dedo haciendo movimientos 4.Pulsar el botón Borrar	Éxito

Tabla 9: PIZARRA

7. Ligas

ID	PROPÓSITO	PASOS	RESULTADO
7.1	Crear una nueva liga y ver que se ha creado correctamente	1.Pulsar el icono Ligas del menú principal 2.Pulsar el icono Crear Liga 3.Rellenar los datos y pulsar Aceptar 4.Pulsar el icono Lista de Ligas y ver que se ha creado correctamente	Éxito
7.2	Identificarse en una liga mediante una contraseña	1.Seleccionar una de las ligas existentes 2.Introducir la contraseña correctamente 3.Acceder al menú de la liga	Éxito
7.3	Crear un nuevo equipo para que participe en la liga	1.Pulsar el botón nuevo equipo del menú de la liga 2.Introducir el nombre del equipo y pulsar Aceptar 3.Ver que se ha creado el equipo correctamente	Éxito
7.4	Crear una nueva jornada para la liga	1.Pulsar el botón Jornadas del menú de la liga 2.Pulsar el botón + 3.Rellenar los campos y pulsar Aceptar 4.Ver que la jornada se ha creado en la lista de jornadas	Éxito

ID	PROPÓSITO	PASOS	RESULTADO
7.5	Crear un nuevo partido perteneciente a una jornada	1.Seleccionar una de las jornadas existentes 2.Pulsar el botón + 3.Rellenar los campos y pulsar Aceptar 4.Ver que el partido se ha creado en la lista de partidos	Éxito
7.6	Ver la clasificación y comprobar que se actualizan los resultados introducidos	1.Pulsar el botón Clasificación del menú de la liga 2.Comprobar que los datos se han actualizado correctamente	Éxito
7.7	Ver la lista de partidos de más de una jornada perteneciente a una liga	1.Seleccionar una jornada de la lista de jornadas. 2.Ver los partidos que hay 3.Volver a seleccionar otra jornada diferente 4.Ver si los partidos son diferentes	Fallo

Tabla 10: LIGAS

Uno de los fallos encontrados fue que cuando pulsaba en diferentes jornadas de la misma liga me mostraba los mismos partidos con los mismos resultados en todas.

Los partidos se creaban bien y la clasificación se actualizaba de manera correcta pero me daba ese error. El problema era la conexión que había creado en el StoryBoard entre vistas. El id que decía lo que tenía que hacer al pulsar sobre una jornada eran diferentes en el código y en el Storyboard.

8. Mensajes

ID	PROPÓSITO	PASOS	RESULTADO
8.1	Ver uno de los mensajes que el usuario activo ha recibido	1.Pulsar el botón Mensajes del menú principal 2.Pulsar el botón Bandeja de Entrada 3.Seleccionar un mensaje de la lista 4.Ver el contenido del mensaje en la nueva vista	Éxito
8.2	Eliminar un mensaje de la lista de mensajes	1.Pulsar el botón Mensajes del menú principal 2.Pulsar el botón Bandeja de Entrada 3.Deslizar sobre el mensaje seleccionado y pulsar Eliminar 4.Comprobar que el mensaje ha sido eliminado de la lista de mensajes	Éxito
8.3	Enviar un mensaje a un usuario	1.Pulsar el botón Mensajes del menú principal 2.Pulsar el botón Enviar Mensaje 3.Rellenar los campos y pulsar Aceptar	Éxito
8.4	Enviar un mensaje a un usuario y que el texto contenga tildes	1.Pulsar el botón Mensajes del menú principal 2.Pulsar el botón Enviar Mensaje 3.Rellenar los campos y pulsar Aceptar 4.Comprobar que el mensaje ha llegado	Fallo

Tabla 11: MENSAJES

Este error me ocurrió por primera vez a la hora de enviar un mensaje y tener que poner tildes en el texto que enviaba. Había veces que enviaba mensajes y no llegaban y no entendía la razón. Me di cuenta que la codificación que había puesto en la columna "Texto" en la tabla de la base de datos no era correcta y había que poner *utf8_general_ci.*

9. Usuarios

ID	PROPÓSITO	PASOS	RESULTADO
9.1	Ver el perfil del usuario activo	1.Pulsar el botón Perfil del menú principal 2.Comprobar que en la nueva vista que los datos son correctos	Éxito
9.2	Buscar usuarios dados de alta en el sistema	1.Pulsar el botón Mensajes del menú principal 2.Pulsar el botón buscar usuario 3.Introducir un nombre y comprobar si existe	Éxito
9.3	Eliminar un usuario del sistema e intentar iniciar sesión con el para comprobar que ha sido eliminado por completo	1.Pulsar el botón Perfil del menú principal 2.Pulsar el botón Eliminar Usuario 3.Iniciar sesión con los datos del usuario eliminado y comprobar que no se puede	Éxito
9.4	Eliminar un usuario y ver que sus ligas y sus mensajes han sido eliminados	1.Eliminar el usuario 2.Comprobar las ligas y los mensajes en la base de datos	Fallo

Tabla 12: USUARIOS

Cuando hice esta funcionalidad todo se eliminaba correctamente menos los mensajes. No se me ocurría como hacer que se eliminasen los mensajes que perteneciesen a un usuario por que al hacer la relación no podía utilizar el id del usuario. Al final pensé, que se tenían que eliminar los mensajes de aquellos usuarios que se eliminen relacionándolo con el nombre de usuario.

8. CONCLUSIONES Y FUTURO TRABAJO

En este apartado se van a exponer las conclusiones personales sobre el desarrollo de la aplicación, así como un trabajo futuro que se podría realizar en caso de querer insertar nuevas funcionalidades o mejorar las ya existentes.

8.1. Conclusiones

Una vez finalizado el proyecto puedo decir que estoy realmente satisfecho con el trabajo realizado y la manera en las que he hecho las cosas. En cuanto a la aplicación desarrollada, el resultado ha sido muy bueno y considero que es una aplicación realmente útil para aquellos usuarios que estén dispuestos a utilizarla.

En cuanto a los objetivos planteados en un principio, se han superado con creces ya que he podido añadir más funcionalidades de las planteadas en un primer momento. Además, el diseño a nivel visual de la aplicación se ha realizado con especial cuidado y se ha intentado que sea divertida para el usuario por lo llamativo que son los iconos para acceder a las funcionalidades. A parte de esto, creo que la aplicación está al nivel de poder estar en la AppStore para que quien lo deseé se la pueda descargar en su dispositivo.

En cuanto a la metodología, el proyecto ha sido basado en prototipos y se ha podido ir viendo los resultados que se iban consiguiendo una vez realizados. La ventaja ha sido la libertad que esto nos da para trabajar y en caso de haberme quedado atascado en algún momento, poder avanzar por otro lado solucionando después del problema encontrado.

Si hablamos de la dificultad del proyecto, creo que ha sido bastante alta ya que se desconocía por completo el lenguaje de programación. Ha habido que empezar desde cero, buscando libros que me diesen pautas para saber por dónde empezar y como interpretar todas las cosas que me iba encontrando por el camino. Además, el hecho de aprender un nuevo programa con el que se iba a pasar tanto tiempo trabajando es un hándicap añadido. Lo más favorable ha sido todo lo que he aprendido durante la realización de este proyecto, ya que cuando me encontraba con errores de programación, he tenido que buscar la solución de la mejor manera posible. Creo que eso da madurez para a la hora de salir al mercado laboral y encontrarme con estos obstáculos, saber salir del paso.

A la hora de la implementación, los problemas principales han venido dados de las bases de datos, ya que no existe demasiada información acerca de SQLite para dispositivos móviles. De primeras, tuve problemas con cargar la base de datos en el dispositivo porque no me mostraba ningún dato. Todo parecía estar correcto y el problema era que se había copiado mal en el proyecto. No había manera de encontrar el problema y se tardaron varios días en encontrar la solución porque en el simulador del Mac funcionaba todo perfectamente. Otro de los problemas ha sido la eliminación en cascada. Después de mucho investigar me quedó claro que si se querían eliminar datos relacionados, había que hacerlo paso a paso.

Sobre la gestión del proyecto, no se ha cumplido del todo por el aumento de horas que he sufrido en cada tarea. Todo esto viene por la inexperiencia de realizar un trabajo de estas dimensiones yo solo y de no saber calcular los tiempos que se iba a tardar en realizar cada apartado.

Donde se ha perdido más tiempo ha sido en la parte de la formación, implementación y en la de la realización de la memoria. Se ha invertido más tiempo del esperado en la formación porque he tenido que hacer muchos programas de prueba antes de ponerme con el proyecto para entender cómo funcionaba cada cosa que pretendía hacer. A la hora de implementar, se ha perdido mucho tiempo en pequeños fallos en los que no encontraba la solución o no se me ocurría la manera de solucionarlos.

En cuanto a la memoria, pensé que iba a tardar menos tiempo en realizar cada apartado. El hecho de hacer tablas, insertar imágenes después de hacer las capturas en la aplicación, corregir fallos de escritura, etc. ha hecho que aumenten las horas reales en la memoria.

En el siguiente gráfico se muestra de manera más visual la comparativa entre la planificación estimada y la real.

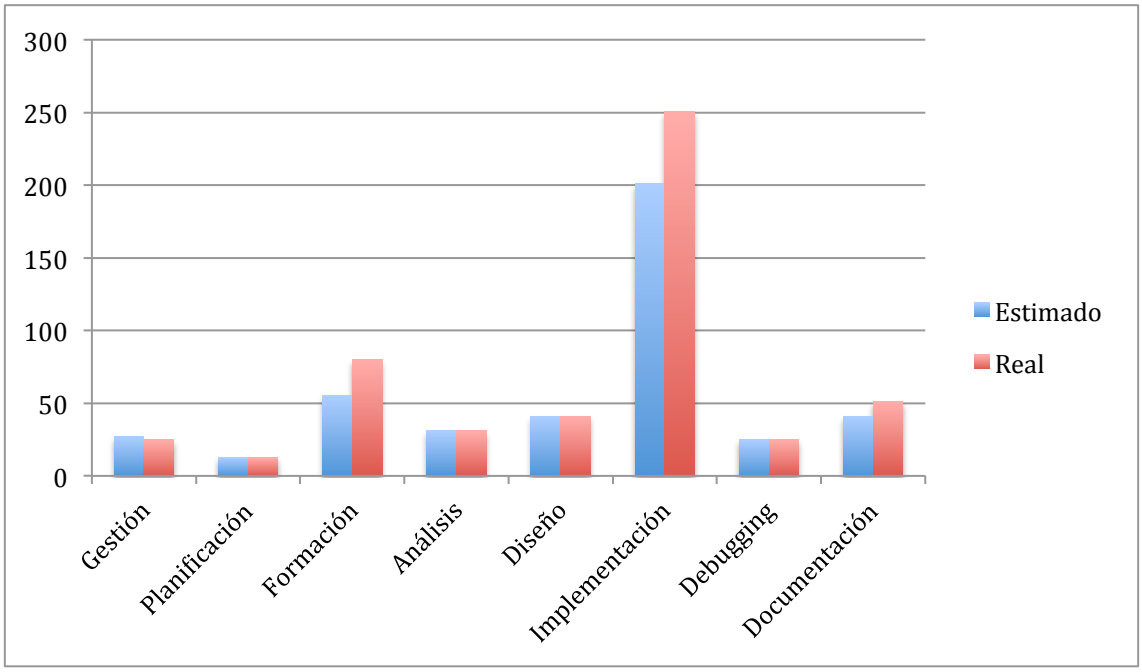


Ilustración 52: GRÁFICO PLANIFICACIÓN

Sobre los riesgos que se podían producir, tengo que decir que no he sufrido ninguno de ellos. Se han tomado precauciones para ello y en el caso de la información se ha ido almacenando en varios sitios a la vez para evitar la pérdida parcial o total del trabajo realizado a diario.

Como conclusión personal, pienso que este proyecto me ha aportado mucho y me ha hecho aprender un lenguaje que considero importante para un

programador. Las posibilidades que te ofrece y la manera de programar son muy interesantes. Además, el hecho de haber trabajado en casa sin ningún tipo de presión añadida más que la mía propia, me ha hecho ser más responsable y saber qué puedo hacer todo lo que me proponga.

Por último decir, que este trabajo me puede abrir puertas a un futuro laboral próximo por haber aprendido una nueva metodología y un nuevo lenguaje de programación.

8.2.Trabajo futuro

En este apartado se va a proceder a explicar el posible futuro de esta aplicación. Aunque la aplicación tiene todo lo que se quería conseguir desde el principio, siempre se pueden añadir mejoras u otras funcionalidades para aplicarlas a esta aplicación.

- Imágenes y videos

  Una de las cosas que igual se puede echar en falta es el uso de imágenes en la aplicación. A muchas personas les gusta el hecho de sacar fotos y subirlas a las aplicaciones. Por ejemplo, a la hora de crear un equipo poder poner la foto del escudo o si se añade un jugador, poder añadir una foto de su cara como se ve en muchas páginas web de deportes.

  También se podría añadir el uso de videos para resumir alguno de los partidos que hayamos creado en la aplicación. De ese modo, aparte de las estadísticas de los jugadores y el resultado del partido, se podría tener un apartado visual y ver un resumen de dicho partido.

  Realizar esta mejora no supondría mucho trabajo ya que el proceso de almacenamiento de estos archivos es parecido al de almacenar datos en una base de datos alojada en un servidor externo.

- Parsear datos de una web

  Otra posible funcionalidad sería el parsear datos de alguna web de deporte para que los usuarios tengan las clasificaciones reales de las ligas más importantes del mundo y así no tengan que hacer uso de otra aplicación externa para acceder a este tipo de datos.

  La única desventaja que tiene esto es que si en algún momento cambian la página web, esto dejaría de funcionar en nuestra aplicación.

  Para realizar esta función sería necesario buscar información de cómo interpretar los datos que están alojados en una página web y como poder extraerlos para mostrarlos en la aplicación.

- Aplicación para el iPhone

Una de las cosas que igual se echa en falta es la opción de tener la misma aplicación también en iPhone. Para hace este cambio de dispositivo habría que rediseñar la aplicación y acoplarla al tamaño del iPhone. Además, habría que modificar el tamaño de las imágenes utilizadas por si alguna no se viese como se tiene que ver.

Ilustración 53: DISPOSITIVOS APPLE

Hajek, M. (2012) De: http://www.technobuffalo.com/

El realizar la aplicación para iPhone supondría tener que volver a empezar la aplicación desde el principio, con la diferencia de que ya se tienen todos los métodos hechos y todas las conexiones para en caso de duda poder mirar el proyecto que se ha realizado para iPad.

- Otros sistemas operativos

Otra de las opciones que sería realmente buena, sería que la aplicación estuviera disponible para otros sistemas operativos como pueden ser Windows Phone, Android o incluso Firefox, ya que cada vez hay más variedad de sistemas operativos en los móviles que están saliendo al mercado.

La mayor desventaja de añadir esta nueva funcionalidad sería mucho más complicada que las anteriores ya que habría que implementar de nuevo todo desde cero y la forma de programar en iOS comparada con el resto de lenguajes es totalmente diferente.

Además, hacer la aplicación para los sistemas operativos mencionados supondría invertir una gran cantidad de horas ya que habría que adaptarse a la manera de programar que se utiliza para cada sistema y utilizar sus respectivas herramientas, las cuales habría que aprender a utilizar.

Ilustración 54: OTROS SO

Rosillo, P. (2013) De: http://andro4all.com/

9.BIBLIOGRAFÍA

A continuación los recursos electrónicos, las citas bibliográficas y los libros utilizados para la realización del trabajo de fin de grado.

9.1. Libros

- Joe Conway & Aaron Hillegas (2010), Desarrollo de aplicaciones para iPhone & iPad, Anaya
- Rory Lewis (2011), Aplicaciones iPhone e iPad para Principiantes, Apress
- Juan Manuel Cigarran Recuero (2013), Aprende iOS: Primeros pasos

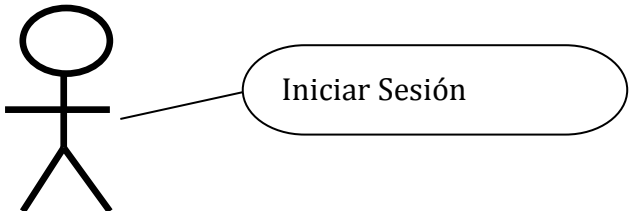
9.2. Recursos electrónicos

- Uso y manejo de JSON: http://davidcruz127.blogspot.com.es/2014/01/ios-7-obtener-un-json-y-mostrar-los.html
- Tutoriales Manzana Mágica- iOS Formación: http://www.manzanamagica.com/desarrollo/
- Tutorial SQLite, Ray Wenderlich: http://www.raywenderlich.com/913/sqlite-tutorial-for-ios-making-our-app
- Curso de Objective-C iPhone/iPad: http://codigofacilito.com/cursos/objective-c
- Recursos iOS, Miguel Diaz Rubio: http://www.migueldiazrubio.com/category/desarrolloios/
- Videotutoriales de desarrollo iOS: http://www.desarrolloweb.com/videos/

9.3. Citas bibliográficas

- Velasco, J. (3 de Marzo de 2014). *http://alt1040.com*. Recuperado el 28 de Mayo de 2014
- JC. (9 de Diciembre de 2013). *http://iphoneate.com*. Recuperado el 28 de Mayo de 2014

ANEXO I: CASOS DE USO EXTENDIDOS



Nombre: Iniciar Sesión

Descripción: Un usuario registrado previamente podrá acceder al sistema introduciendo el nombre de usuario y contraseña.

Actores: Registrado

Precondiciones: Estar registrado en el sistema

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. Para poder acceder al sistema el usuario introducirá su nombre de usuario y contraseña en la pantalla inicial. (Ilust. 55)
   1.1. Si la identificación es correcta, se iniciará sesión y se accederá al menú principal. (Ilust. 56)
   1.2. Si el nombre de usuario, la contraseña o algún campo está sin rellenar, se mostrará una alerta (Ilust. 57)

Postcondiciones: Ninguna

Interfaz Gráfica:

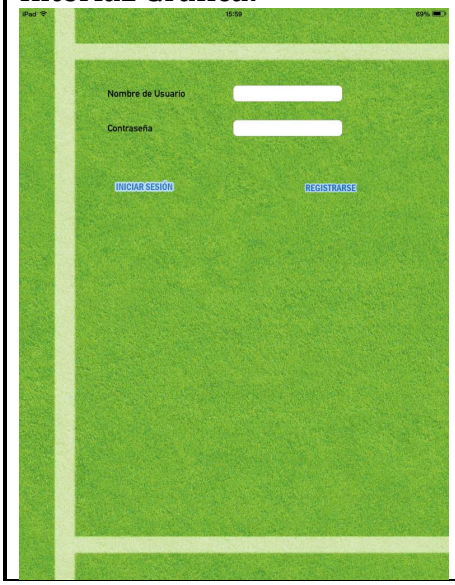


Ilustración 55: INICIAR SESIÓN

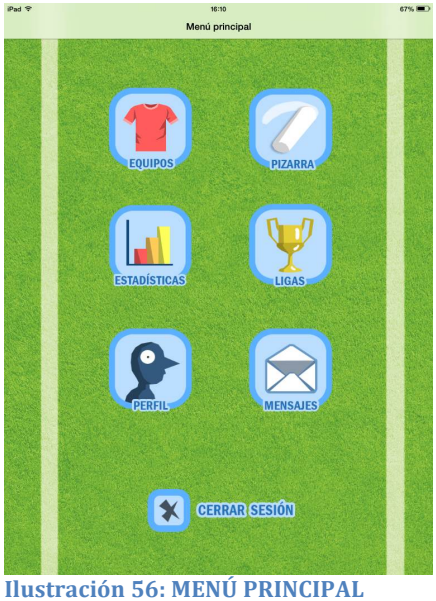
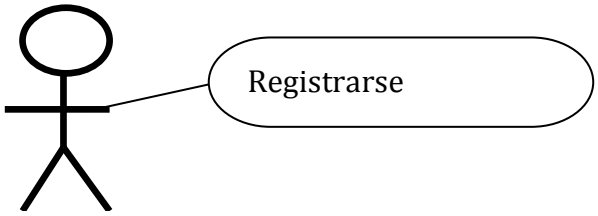


Ilustración 56: MENÚ PRINCIPAL

Ilustración 57: ALERTA



Nombre: Registrarse

Descripción: Cualquier persona no registrada en el sistema podrá darse de alta para poder acceder a todos los contenidos que ofrece la aplicación.

Actores: No registrado

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario no registrado pulsará el botón de "Registrarse" en la pantalla inicial. (Ilust. 58)
2. En esta pantalla se rellenarán todos los campos con los datos necesarios para el registro. (Ilust. 59)
   2.1. Si el nombre de usuario ya existe en el sistema, se avisará al usuario para que busque otro. (Ilust. 60)
   2.2. Si todo es correcto, el usuario quedará dado de alta en el sistema.

Postcondiciones: Ninguna

Interfaz Gráfica:

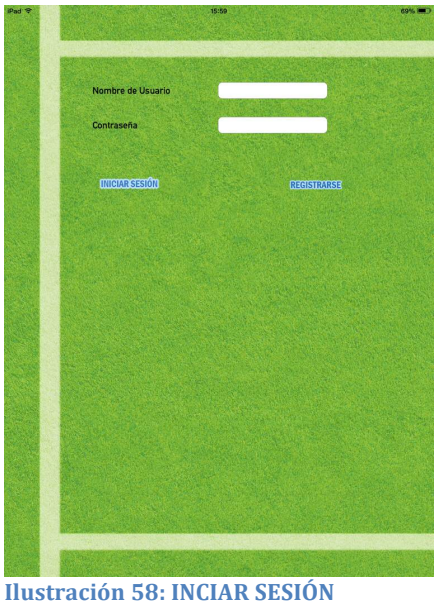


Ilustración 58: INCIAR SESIÓN

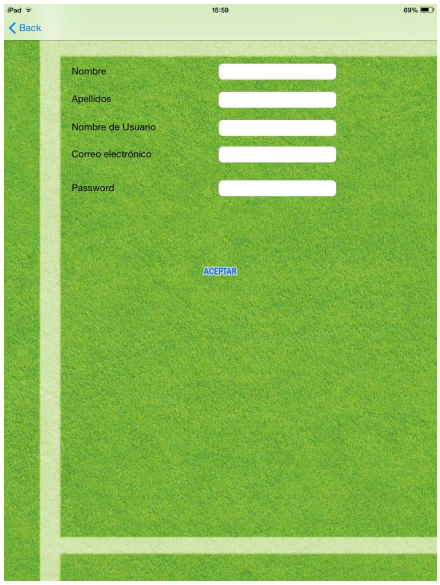
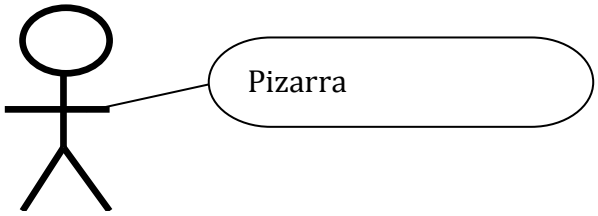


Ilustración 59: REGISTRARSE

Ilustración 60: ALERTA



Nombre: Pizarra

Descripción: Permite al usuario utilizar una pizarra táctil donde se podrán realizar las diferentes tácticas que van a ser puestas en práctica durante un partido.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario selecciona el icono "Pizarra" del Menú principal. (Ilust. 61)
2. El usuario podrá realizar las siguientes funciones: (Ilust. 62)
   2.1. Dibujar tácticas en la pizarra.
   2.2. Cambiar de color para diferenciar los equipos entre rojo y azul.
   2.3. Borrar todo el contenido de la pantalla pulsando el botón "goma".

Postcondiciones: Ninguna

Interfaz Gráfica:

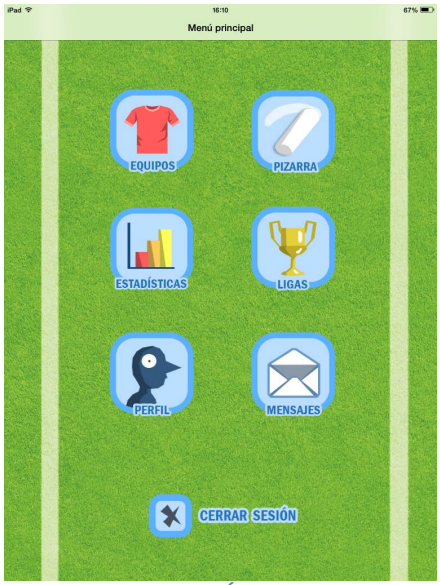


Ilustración 61: MENÚ PRINCIPAL

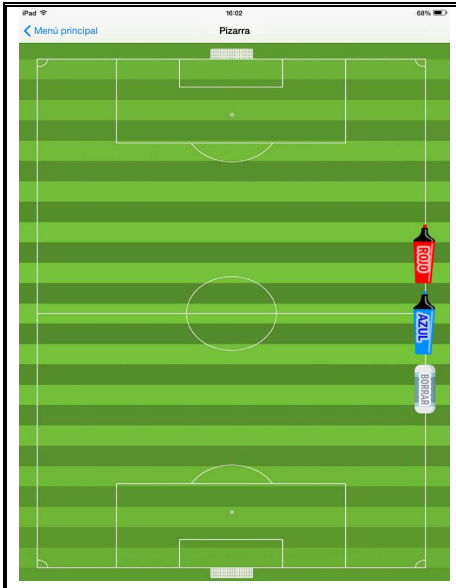
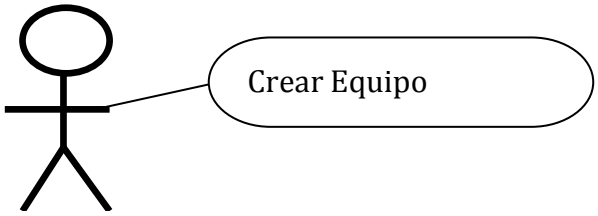


Ilustración 62: PIZARRA



Nombre: Crear Equipo

Descripción: Permite al usuario crear un equipo. Una vez creado el equipo se podrán añadir jugadores, partidos y estadísticas.

Actores: Usuario

Precondiciones: Estar identificado en el sistema

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsara el icono de "Equipos" en el Menú principal. (Ilust. 63)
2. El usuario seleccionará el icono "Crear Equipo" en el Menú de Equipos. (Ilust. 64)
3. Se introducirá el nombre, temporada, categoría y provincia en los campos a rellenar y se pulsará el botón "Aceptar" para crear un nuevo equipo. (Ilust. 65)

Postcondiciones: Ninguna

Interfaz Gráfica:

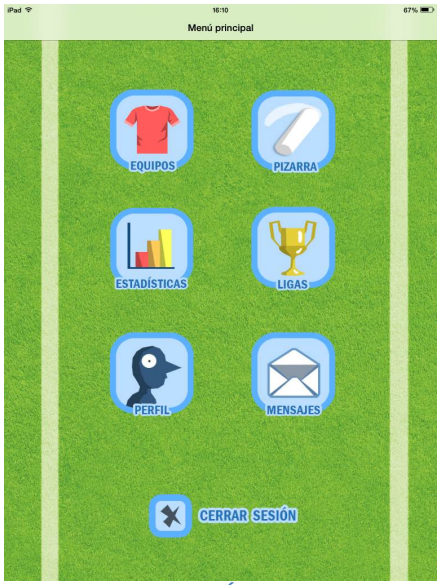


Ilustración 63: MENÚ PRINCIPAL

Ilustración 64: MENÚ EQUIPOS

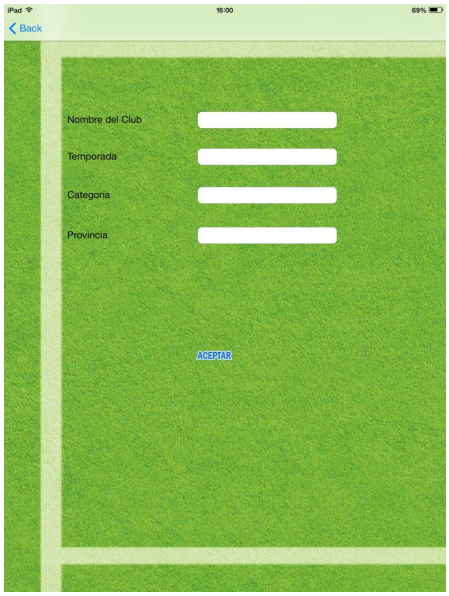
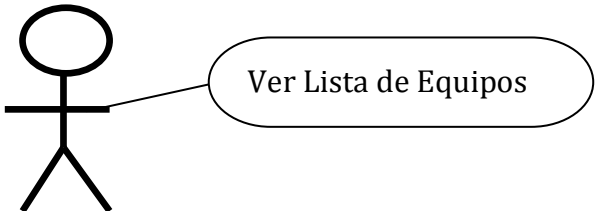


Ilustración 65: AGREGAR EQUIPO



Nombre: Ver Lista de Equipos

Descripción: El usuario podrá ver la lista de equipos que haya creado con anterioridad.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará el icono "Equipos" del Menú Principal. (Ilust. 66)
2. Seleccionar el icono "Lista de Equipos" del Menú de Equipos. (Ilust. 67)
3. El usuario podrá ver la lista de equipos que se hayan creado. (Ilust. 68)

[si quiere eliminar un equipo]

4. Deslizará de derecha a izquierda sobre el equipo que quiera eliminar y pulsará "Eliminar". (Ilust. 69)

Postcondiciones: En caso de eliminar un equipo, se eliminarán también los jugadores que pertenezcan a cada equipo, así como sus partidos y todas las estadísticas relacionadas entre ellos.

Interfaz Gráfica:

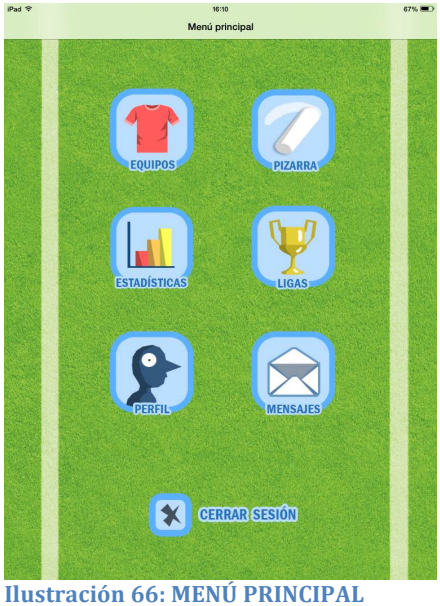


Ilustración 66: MENÚ PRINCIPAL

Ilustración 67: MENÚ EQUIPOS

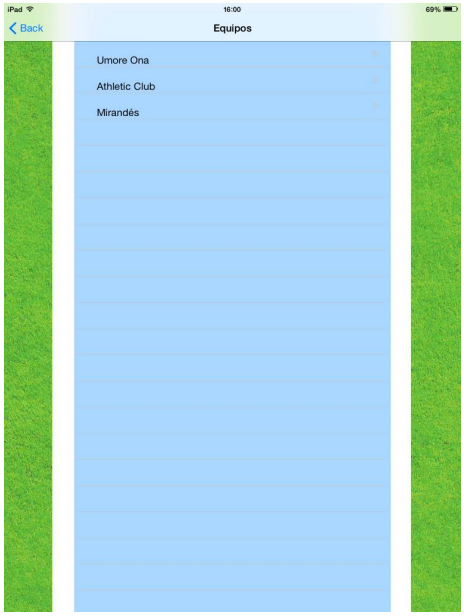
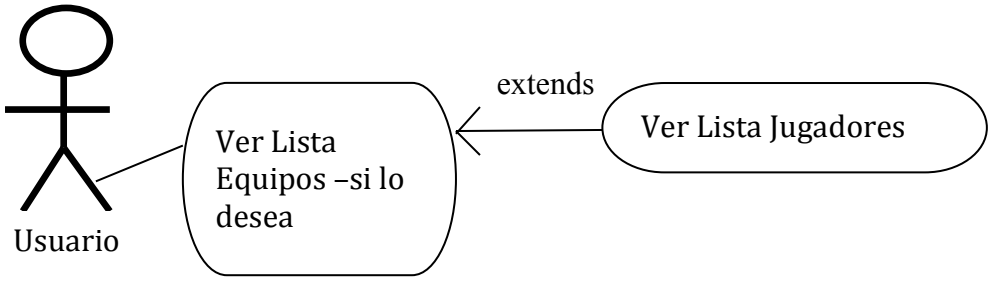


Ilustración 68: LISTA EQUIPOS

Ilustración 69: ELIMINAR EQUIPO



Nombre: Ver Lista de Jugadores

Descripción: El usuario podrá ver la lista de jugadores que haya creado con anterioridad.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:
Se puede acceder a la lista de jugadores por dos caminos distintos, ya que se podrán realizar cosas diferentes dependiendo el camino.
[Si el usuario entra mediante equipos]

1. El usuario selecciona un equipo de la lista de equipos creada. (Ilust. 70)
2. El usuario podrá ver la lista de jugadores que se hayan creado. (Ilust. 71)
   [si quiere eliminar un jugador]
   4. Deslizará de derecha a izquierda sobre el jugador que quiera eliminar y pulsará "Eliminar". (Ilust. 72)

[Si se pulsa el icono +]

a. EXTENDS "Añadir Jugador"

[Si lo hace entrando desde estadísticas]

1. El usuario selecciona un equipo de la lista de equipos creada. (Ilust. 73)
2. El usuario podrá ver la lista de jugadores que se hayan creado. (Ilust. 74)
   [Si pulsa el icono "i"]
   a. EXTENDS "Ver estadísticas partido".
   [Si pulsa sobre la celda de un jugador]
   b. EXTENDS "Crear estadísticas partido".

Postcondiciones: Si elimina un jugador se eliminaran las estadísticas de los partidos que haya disputado ese jugador.

Interfaz Gráfica:

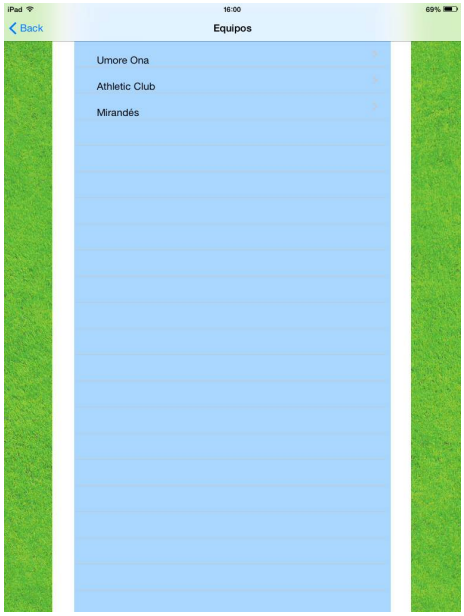


Ilustración 70: LISTA EQUIPOS

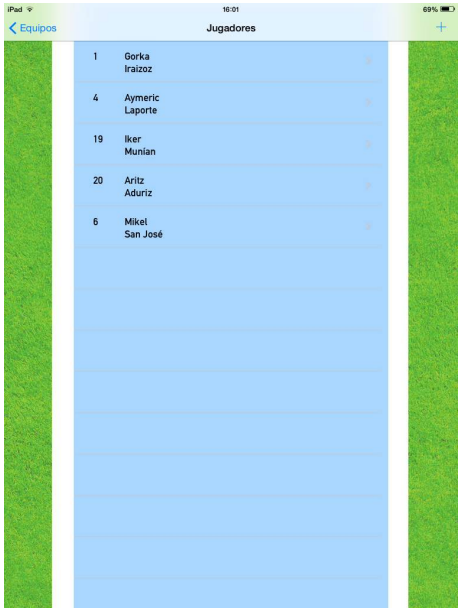


Ilustración 71: LISTA JUGADORES

Ilustración 72: ELIMINAR JUGADOR

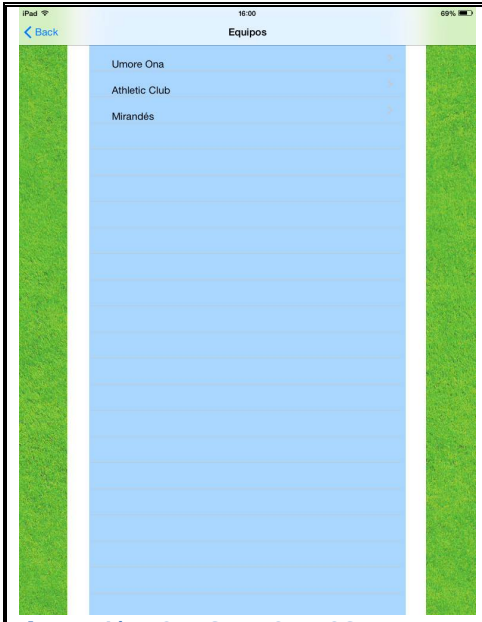


Ilustración 73: LISTA EQUIPOS

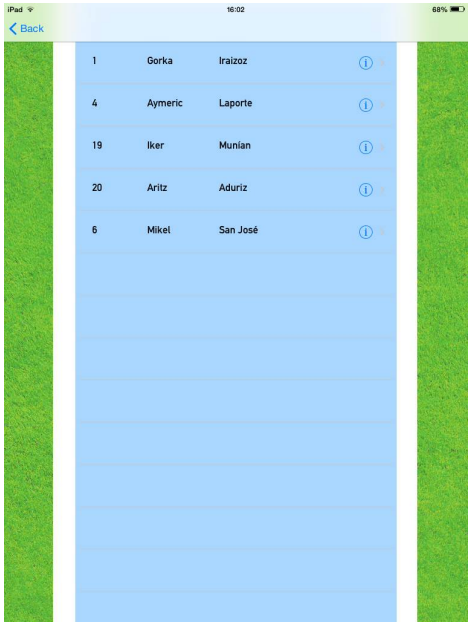
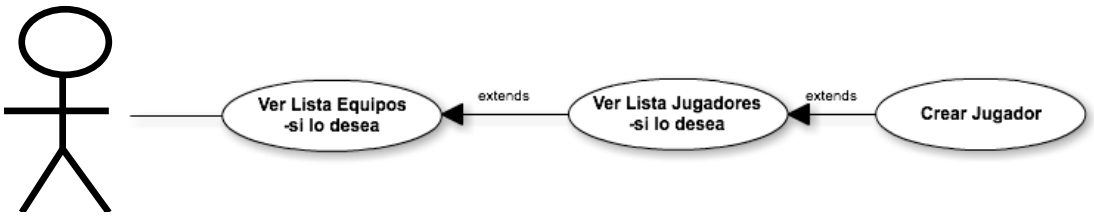


Ilustración 74: LISTA JUGADORES



Nombre: Crear Jugador

Descripción: Permite al usuario crear un nuevo jugador dentro del equipo seleccionado de la lista de equipos.

Actores: Usuario

Precondiciones: Haber creado al menos un equipo

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario seleccionará un equipo de la lista de equipos. (Ilust. 75)
2. El usuario pulsará el botón "+" situado en la parte superior derecha para añadir un nuevo jugador. (Ilust. 76)
3. Se introducirá el nombre, apellidos, fecha de nacimiento, posición y dorsal en los campos a rellenar y se pulsará el botón "Aceptar" para crear un nuevo jugador. (Ilust. 77)

Postcondiciones: Ninguna

Interfaz Gráfica:

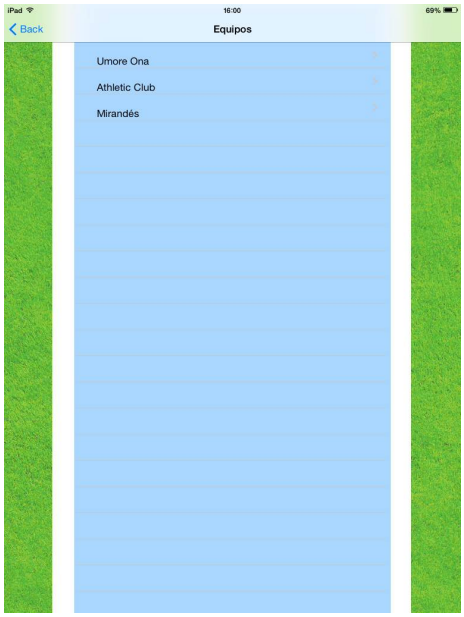


Ilustración 75: LISTA EQUIPOS

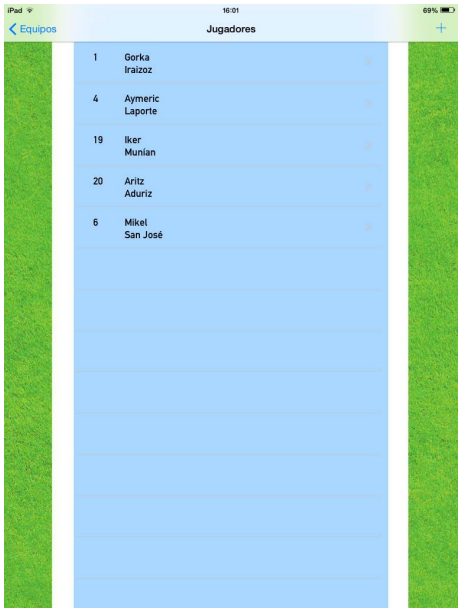


Ilustración 76: LISTA JUGADORES

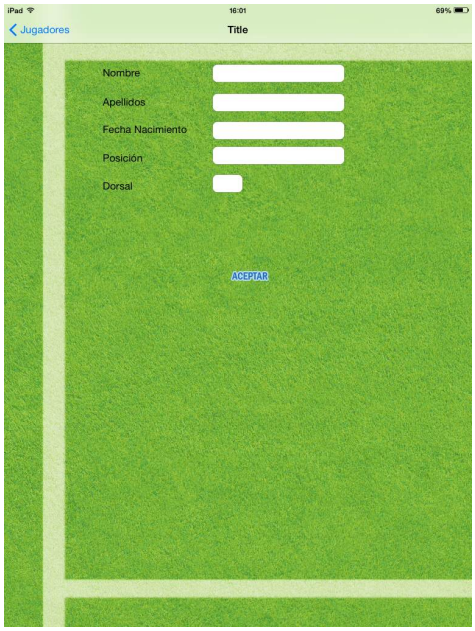
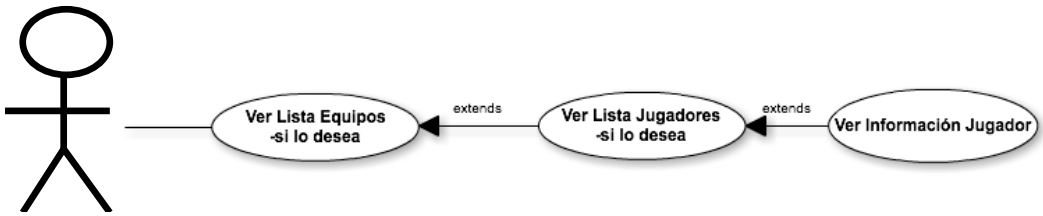


Ilustración 77: AGREGAR JUGADOR



UsuarioRegist

Nombre: Ver Información Jugador

Descripción: El usuario podrá ver la información correspondiente a un jugador, así como las estadísticas de los partidos en los que haya participado ese jugador.

Actores: Usuario

Precondiciones: Haber creado al menos un jugador.

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario seleccionará un jugador de la lista de jugadores. (Ilust. 78)
2. El usuario podrá ver toda la información relacionada a ese jugador. (Ilust. 79)

Postcondiciones: Ninguna

Interfaz Gráfica:

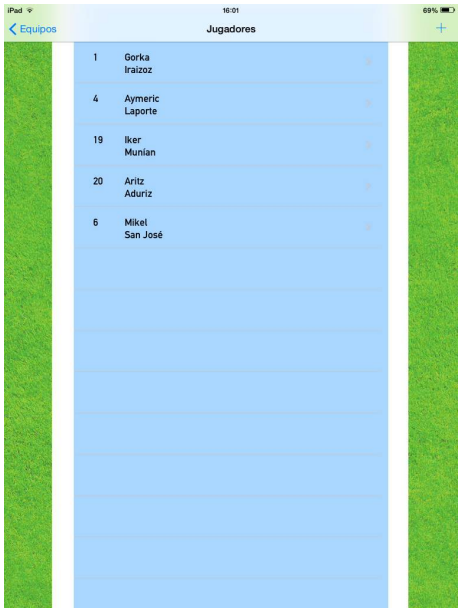
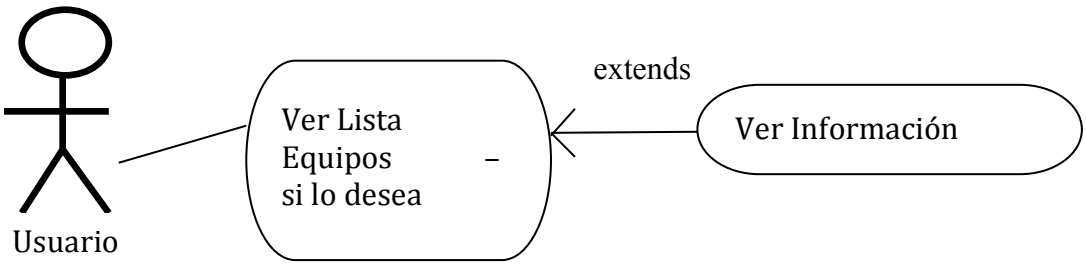


Ilustración 78: LISTA JUGADORES

Ilustración 79: INFORMACIÓN JUGADOR



Nombre: Ver Información Equipo

Descripción: El usuario podrá ver la información correspondiente a un equipo, así como las estadísticas de los partidos en los que haya participado ese equipo.

Actores: Usuario

Precondiciones: Haber creado al menos un equipo.

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará el icono "Estadísticas" del Menú Principal. (Ilust. 80)
2. El usuario pulsará el icono "Info" dentro del Menú de Estadísticas. (Ilust. 81)
3. Seleccionar un equipo de la lista de equipos. (Ilust. 82)
4. El usuario podrá ver toda la información relacionada a ese equipo. (Ilust. 83)

Postcondiciones: Ninguna

Interfaz Gráfica:

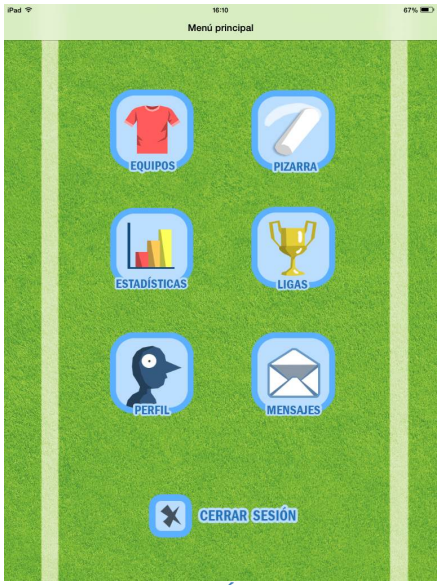


Ilustración 80: MENÚ PRINCIPAL

Ilustración 81: MENÚ ESTADISTICAS

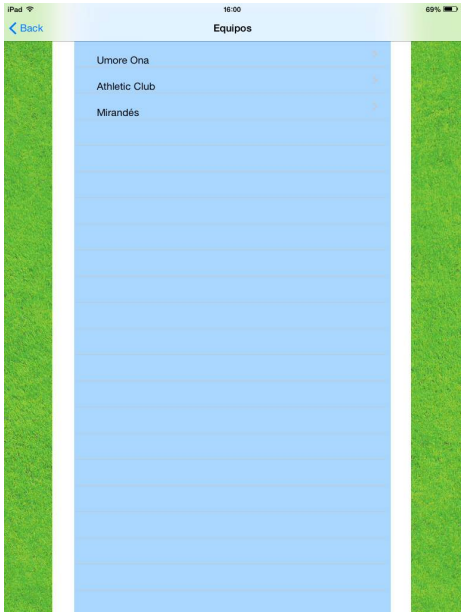
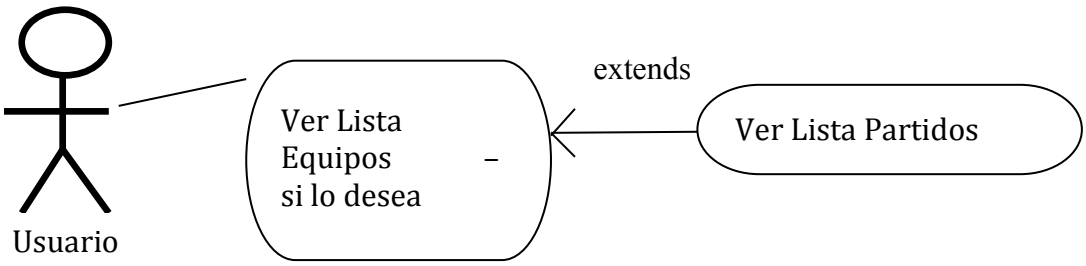


Ilustración 82: LISTA EQUIPOS

Ilustración 83: INFORMACIÓN EQUIPO



Nombre: Ver Lista Partidos

Descripción: El usuario podrá ver la lista de partidos que haya creado con anterioridad.

Actores: Usuario

Precondiciones: Haber creado al menos un equipo.

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará el icono "Estadísticas" del Menú Principal. (Ilust. 84)
2. Seleccionar el icono "Partidos" del Menú de Estadísticas. (Ilust. 85)
3. Seleccionar un equipo de la lista de equipos. (Ilust. 86)
4. El usuario podrá ver la lista de partidos que se hayan creado. (Ilust. 87)
   [si desea eliminar un partido]
5. Deslizará de derecha a izquierda sobre el jugador que quiera eliminar y pulsará "Eliminar". (Ilust. 88)
   [Si pulsa el botón "+"]
   a. EXTENDS "Crear Partido"

Postcondiciones: Ninguna

Interfaz Gráfica:

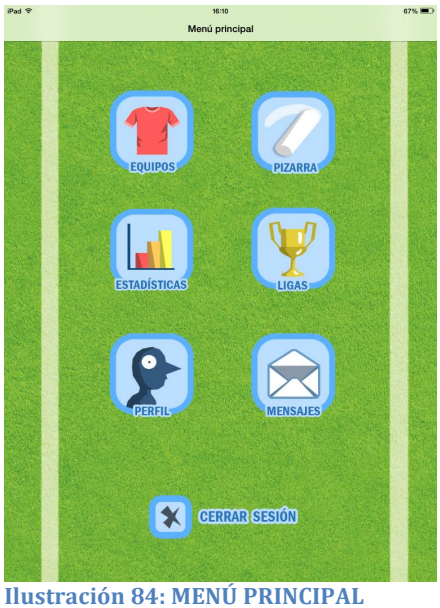


Ilustración 84: MENÚ PRINCIPAL

Ilustración 85: MENÚ ESTADÍSTICAS

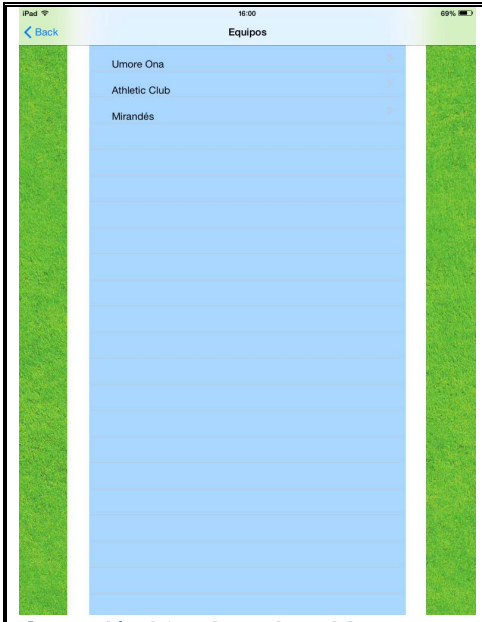


Ilustración 86: LISTA EQUIPOS

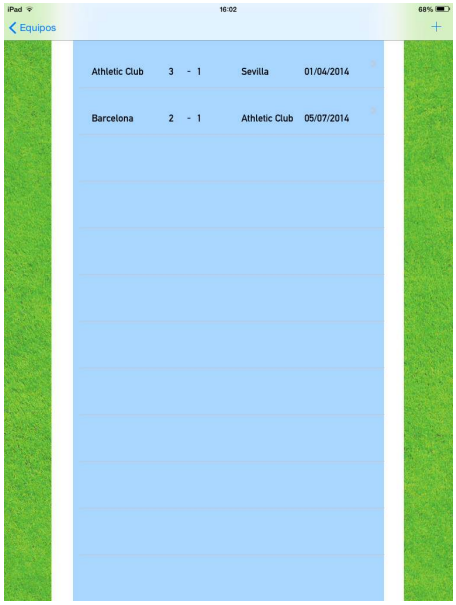
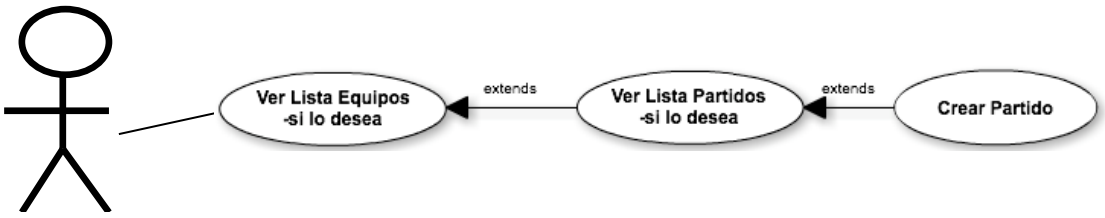


Ilustración 87: LISTA PARTIDOS

Ilustración 88: ELIMINAR PARTIDO



Nombre: Crear Partido

Descripción: Permite al usuario crear un partido correspondiente a un equipo. Cuando se crea el partido con el resultado final, se actualizará de manera automática las estadísticas de ese equipo.

Actores: Usuario

Precondiciones: Haber creado al menos un equipo

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará el botón "+" situado en la parte superior derecha para añadir un nuevo partido. (Ilust. 89)
2. Se introducirá el equipo local y sus goles, el equipo visitante y sus goles, además de la fecha del partido y se pulsará el botón "Aceptar". (Ilust. 90)

Postcondiciones: Ninguna

Interfaz Gráfica:

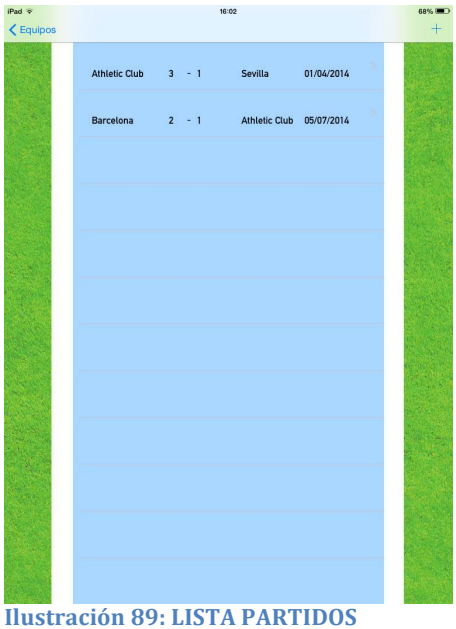


Ilustración 89: LISTA PARTIDOS

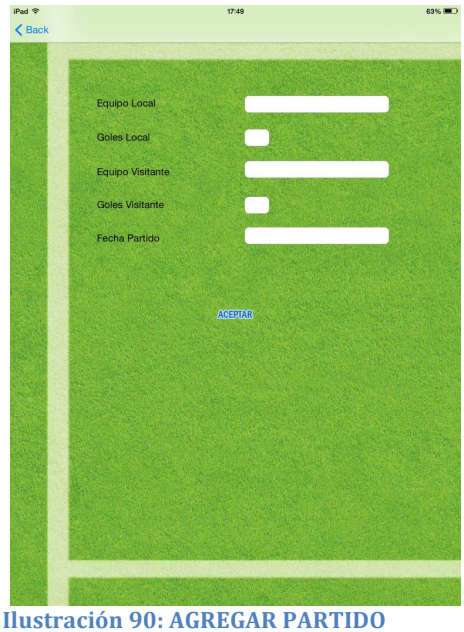
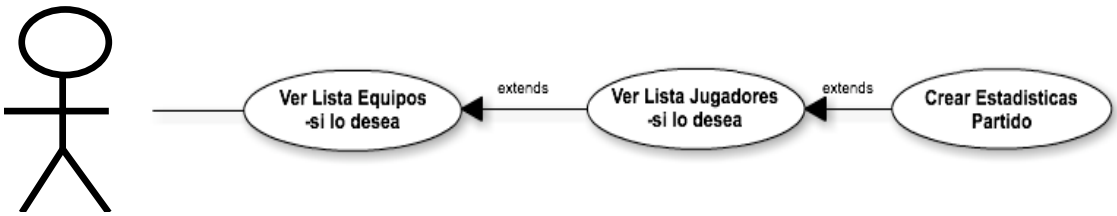


Ilustración 90: AGREGAR PARTIDO



Usuario

Nombre: Crear Estadísticas Partido

Descripción: Permite al usuario crear estadísticas individuales a cada jugador que participe en cada partido creado. Se añadirán datos como los minutos jugados, tarjetas amarillas, tarjeta roja, asistencias y goles. Una vez introducidas las estadísticas se irán actualizando de manera automática en cada jugador para tener también las estadísticas globales de cada uno.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará sobre la celda del jugador al que se le quiera añadir las estadísticas. (Ilust. 91)
2. Se introducirán los minutos jugados, tarjetas amarillas, tarjeta roja, asistencias y goles y se pulsará el botón “Aceptar”. (Ilust. 92)

Postcondiciones: Se actualizarán las estadísticas de los jugadores que hayan participado en el partido.

Interfaz Gráfica:

Ilustración 91: LISTA JUGADORES

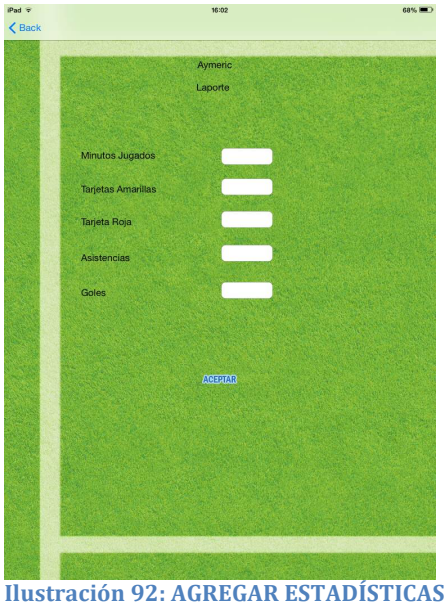
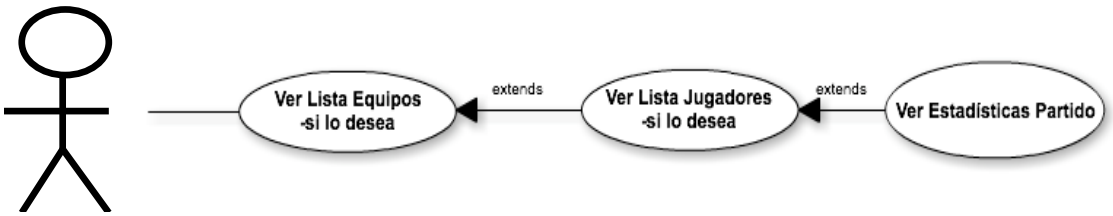


Ilustración 92: AGREGAR ESTADÍSTICAS



Usuario

Nombre: Ver Estadísticas Partido

Descripción: Permite al usuario ver estadísticas de cada jugador que ha participado en cada partido habiéndolas introducido previamente.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará sobre "i" de la celda del jugador que se quiera ver las estadísticas. (Ilust. 93)
2. Se mostrarán todas las estadísticas relacionadas con ese jugador en ese partido en particular. (Ilust. 94)

Postcondiciones: Se actualizarán las estadísticas de los jugadores que hayan participado en el partido.

Interfaz Gráfica:

Ilustración 93: LISTA JUGADORES

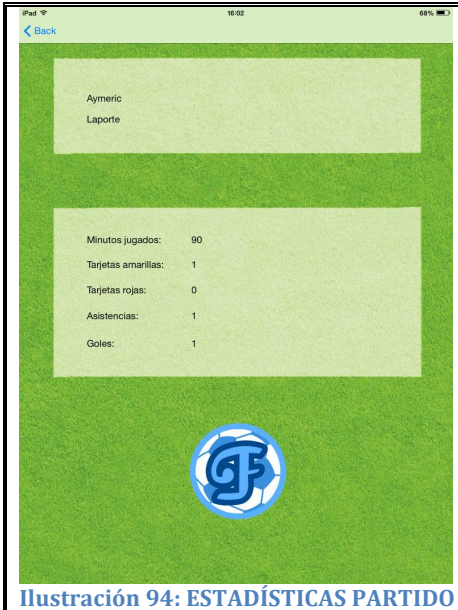
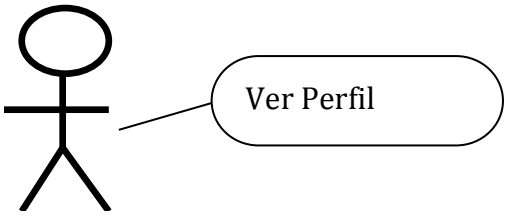


Ilustración 94: ESTADÍSTICAS PARTIDO



Nombre: Ver Perfil

Descripción: El usuario podrá ver su perfil de usuario con los datos personales que haya rellenado en la ficha de registro.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará el icono "Perfil" del Menú Principal. (Ilust. 95)
2. El usuario podrá ver la información personal de usuario con los datos insertados en el registro al darse de alta en el sistema. (Ilust. 96)
   [Si pulsa el botón "eliminar usuario"]
   a. EXTENDS "Eliminar Usuario".

Postcondiciones: Ninguna

Interfaz Gráfica:

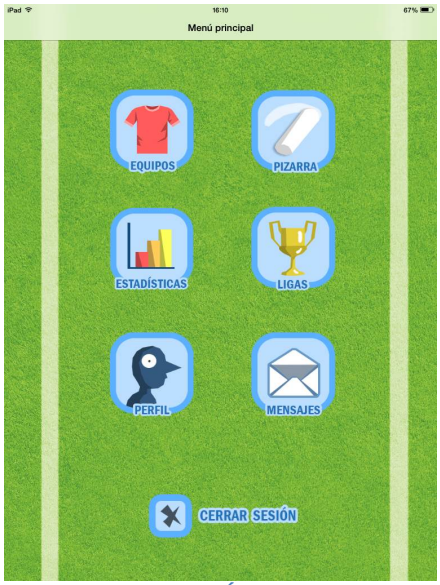
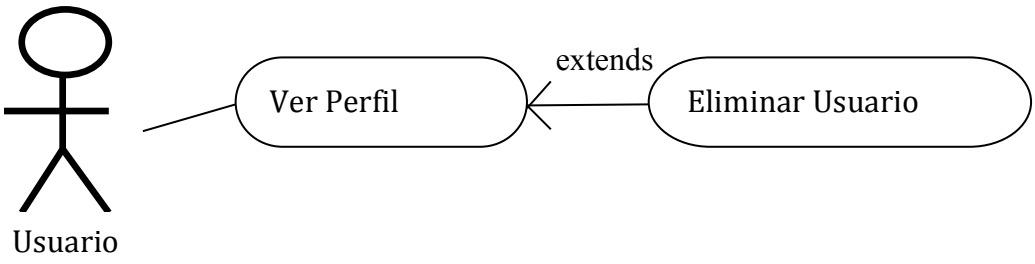


Ilustración 95: MENÚ PRINCIPAL

Ilustración 96: PERFIL USUARIO



Nombre: Eliminar Usuario

Descripción: El usuario podrá eliminar su cuenta del sistema y con ello se eliminarán también las ligas que haya creado, además de los mensajes recibidos.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

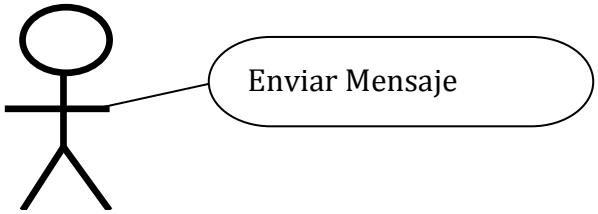
1. El usuario pulsará el botón "Eliminar Usuario" situado en la parte inferior del perfil. (Ilust. 97)
2. Si el usuario está seguro, pulsará el botón "Aceptar" y la cuenta quedará completamente eliminada. (Ilust. 98)

Postcondiciones: Se eliminarán todas las ligas creadas por el usuario además de los mensajes recibidos.

Interfaz Gráfica:

Ilustración 97: PERFIL USUARIO

Ilustración 98: ELIMINAR USUARIO



Nombre: Enviar Mensaje

Descripción: El usuario podrá enviar un mensaje a un usuario que pertenezca al sistema para poder informarle de ligas creadas o simplemente para mandarle cualquier tipo de información.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará el icono "Mensajes" del Menú Principal. (Ilust. 99)
2. Pulsar el botón "Enviar Mensaje" del Menú de Mensajes. (Ilust. 100)
3. Rellenar todos los campos y pulsar el botón "Enviar" para mandar el mensaje al usuario. (Ilust. 101)

Postcondiciones: Ninguna

Interfaz Gráfica:

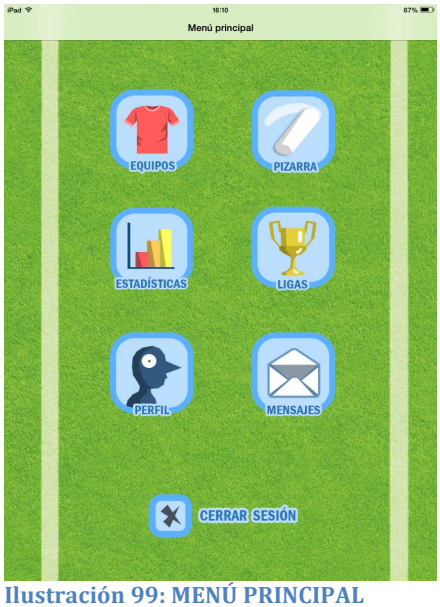


Ilustración 99: MENÚ PRINCIPAL

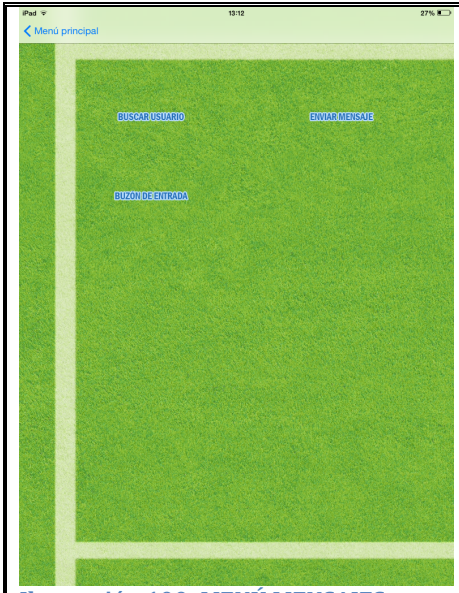
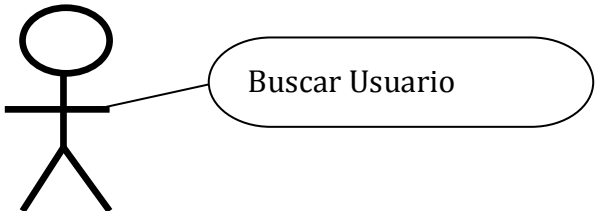


Ilustración 100: MENÚ MENSAJES

Ilustración 101: ENVIAR MENSAJE



Nombre: Buscar Usuario

Descripción: Un usuario podrá buscar a otros usuarios registrados previamente en el sistema.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario seleccionará el icono "Mensajes" del Menú principal. (Ilust. 102)
2. Pulsar el botón "Buscar Usuario" del Menú de Mensajes (Ilust. 103)
3. El usuario introducirá el nombre del usuario que desee buscar en el buscador. En caso de existir el usuario aparecerá en una de las celdas de la parte inferior. (Ilust. 103)
   [Si pulsa sobre un usuario]
   a. EXTENDS "Enviar Mensaje"

Postcondiciones: Ninguna

Interfaz Gráfica:

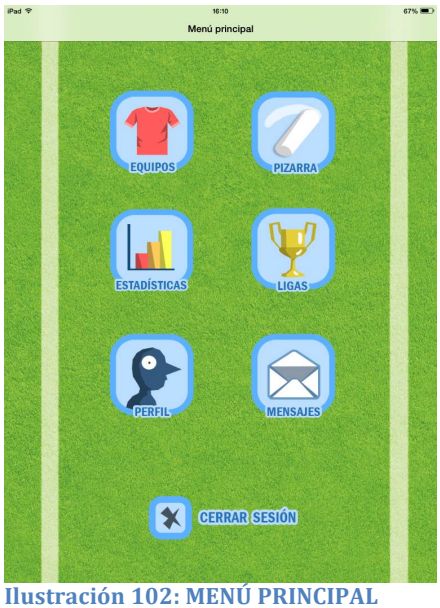


Ilustración 102: MENÚ PRINCIPAL

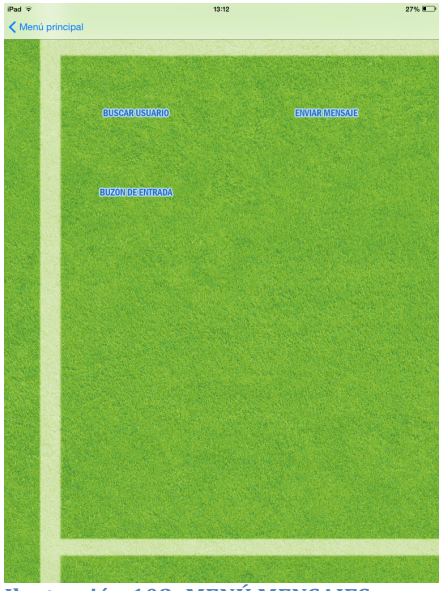


Ilustración 103: MENÚ MENSAJES

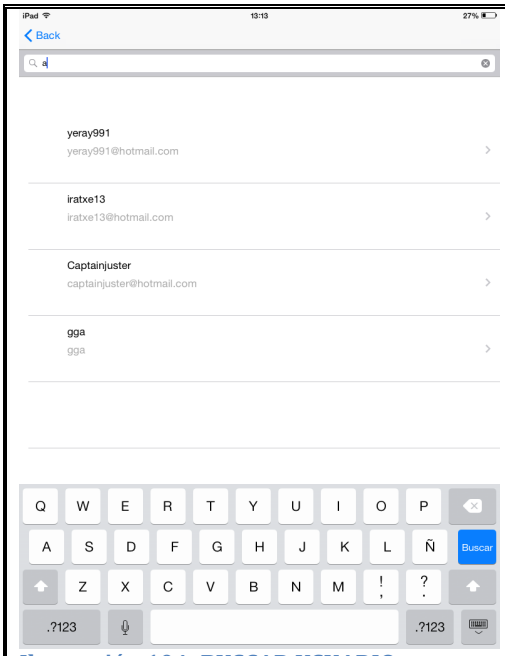
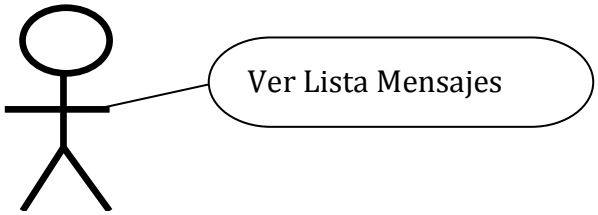


Ilustración 104: BUSCAR USUARIO



Nombre: Ver Lista Mensajes

Descripción: Un usuario podrá ver la lista de mensajes recibidos por otros usuarios.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará sobre el icono "Mensajes" del Menú Principal. (Ilust. 105)
2. Pulsará sobre el botón "Bandeja de Entrada" del Menú de Mensajes. (Ilust. 106)
3. Se mostrará una lista con todos los mensajes que haya recibido ese usuario. (Ilust. 107)

   [si quiere eliminar un mensaje]

   4. Deslizará de derecha a izquierda sobre el mensaje que quiera eliminar y pulsará "Eliminar". (Ilust. 108)

Postcondiciones: Ninguna.

Interfaz Gráfica:

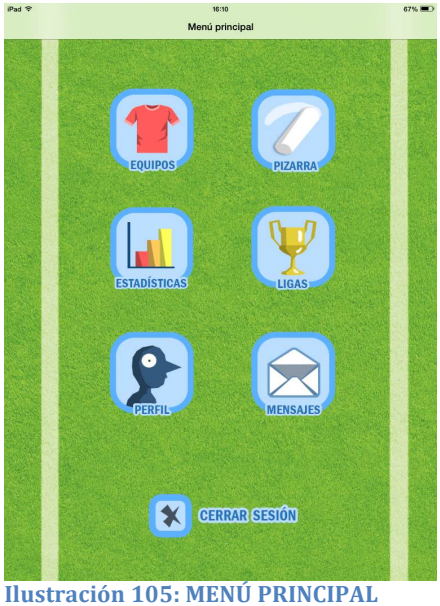


Ilustración 105: MENÚ PRINCIPAL

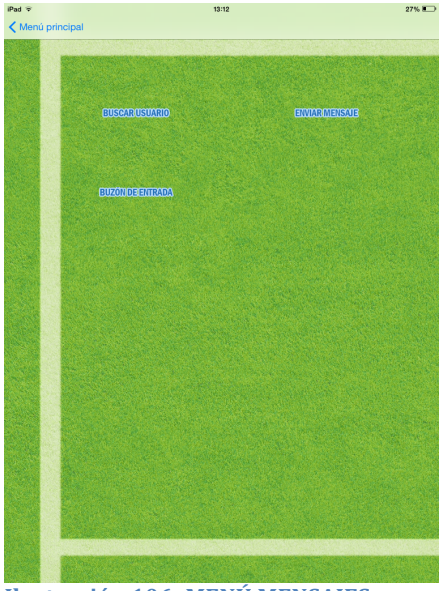


Ilustración 106: MENÚ MENSAJES

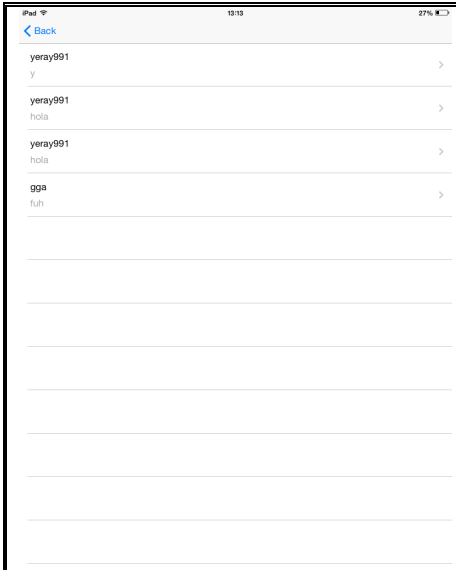


Ilustración 107: LISTA MENSAJES

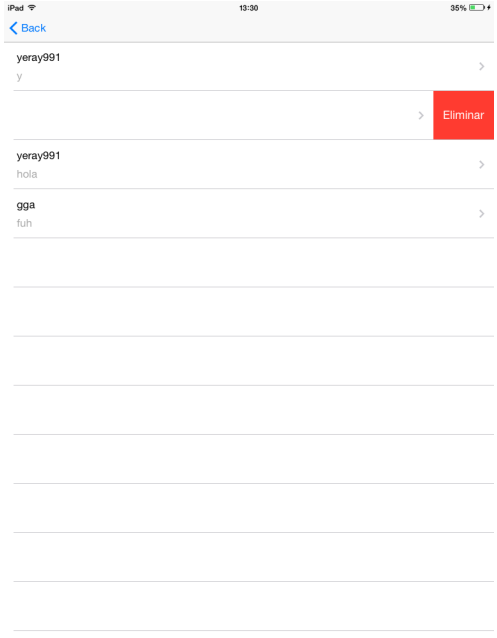
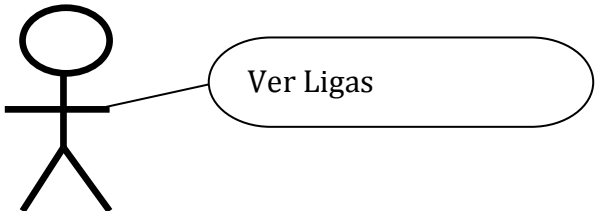


Ilustración 108: ELIMINAR MENSAJE



Nombre: Ver ligas

Descripción: Un usuario podrá ver una liga que haya creado él o que haya creado otro usuario.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará sobre el icono "Ligas" del Menú Principal. (Ilust. 109)
2. Seleccionará el icono "Lista de Ligas" del Menú de Ligas. (Ilust. 110)
3. Introducirá el nombre de la liga que deseé buscar en el buscador y aparecerá en la parte inferior. (Ilust. 111)

Postcondiciones: Ninguna.

Interfaz Gráfica:

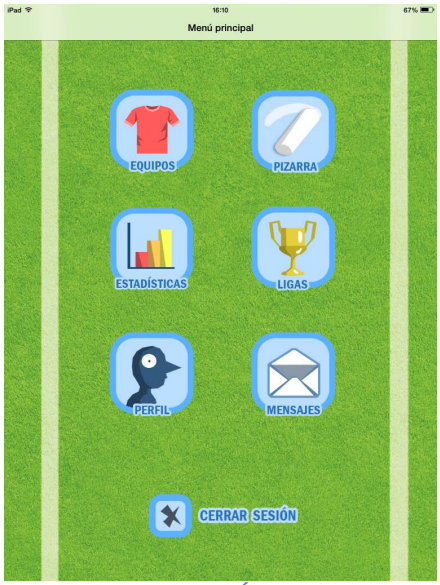


Ilustración 109: MENÚ PRINCIPAL

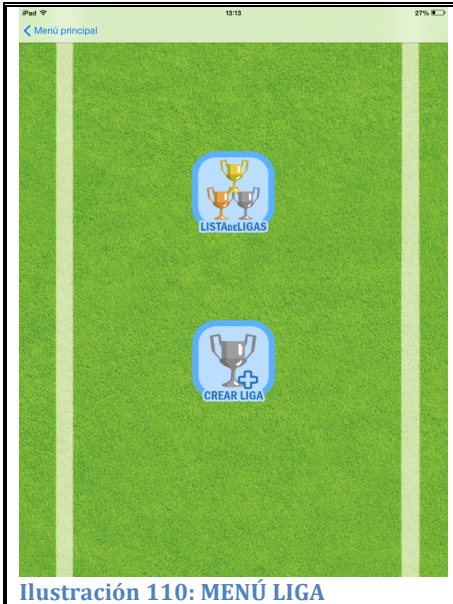


Ilustración 110: MENÚ LIGA

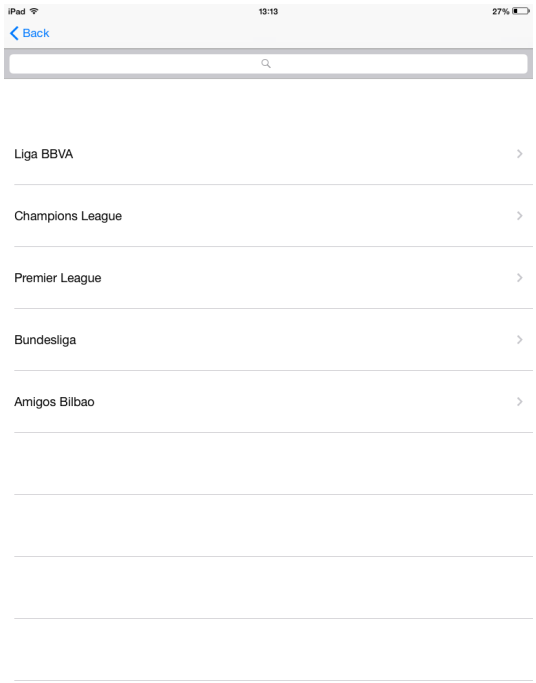
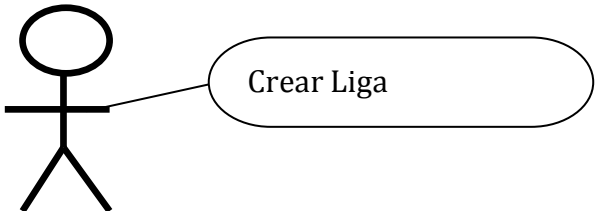


Ilustración 111: LISTA LIGAS



Nombre: Crear Liga

Descripción: Un usuario podrá crear una liga en donde podrán participar más de un usuario.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsara sobre el icono "Ligas" del Menú de Ligas. (Ilust. 112)
2. Pulsará sobre el icono "Crear Liga" del Menú de Ligas. (Ilust. 113)
3. Rellenará los datos correspondientes a los campos y pulsará el botón "Aceptar" para crear la liga. (Ilust. 114)

Postcondiciones: Una liga solo será eliminada al eliminar el usuario que la haya creado.

Interfaz Gráfica:

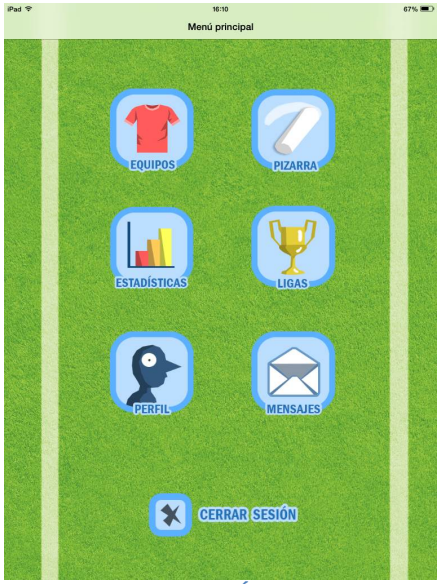


Ilustración 112: MENÚ PRINCIPAL

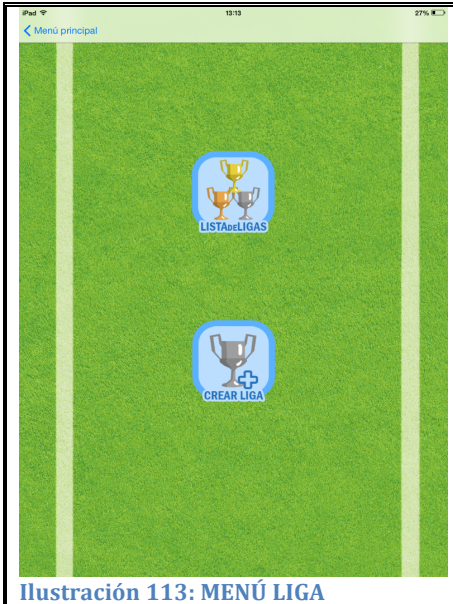


Ilustración 113: MENÚ LIGA

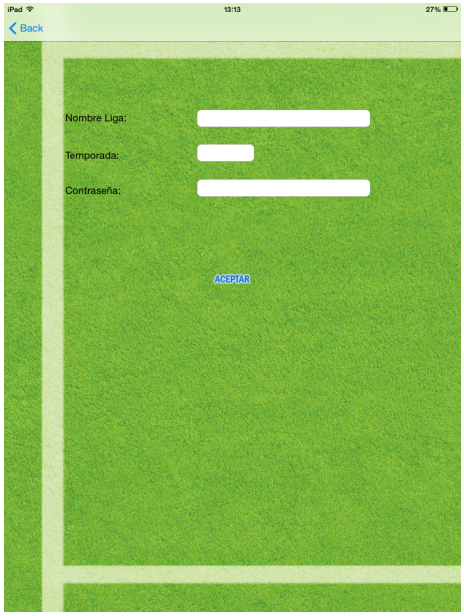


Ilustración 114: AGREGAR LIGA

Usuario

Nombre: Identificarse Liga

Descripción: Un usuario solo podrá participar en una liga y modificar su contenido si dispone de la contraseña que la protege.

Actores: Usuario

Precondiciones: Tener la contraseña de la liga.

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario seleccionará una liga de la lista de ligas. (Ilust. 115)
   [si tiene la contraseña y pulsa "Identificarse"]
   2.a. Introducirá la contraseña en el campo correspondiente y pulsará el botón "Aceptar". (Ilust. 116)
   3.a Accederá a un Menú donde podrá crear equipos, jornadas y ver la clasificación general de los equipos que participan. (Ilust. 117)
   [si no tiene la contraseña y pulsa "Entrar sin conexión"]
   2.b. Accederá a un Menú donde podrá ver las jornadas y la clasificación general de los equipos que participan. (Ilust. 118)

Postcondiciones: Ninguna

Interfaz Gráfica:

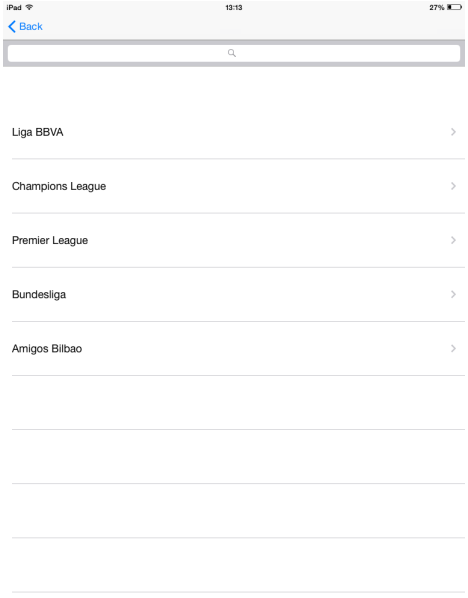


Ilustración 115: LISTA LIGAS

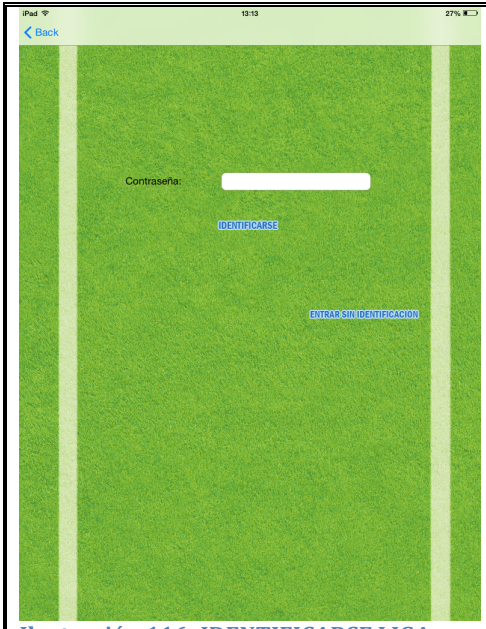


Ilustración 116: IDENTIFICARSE LIGA

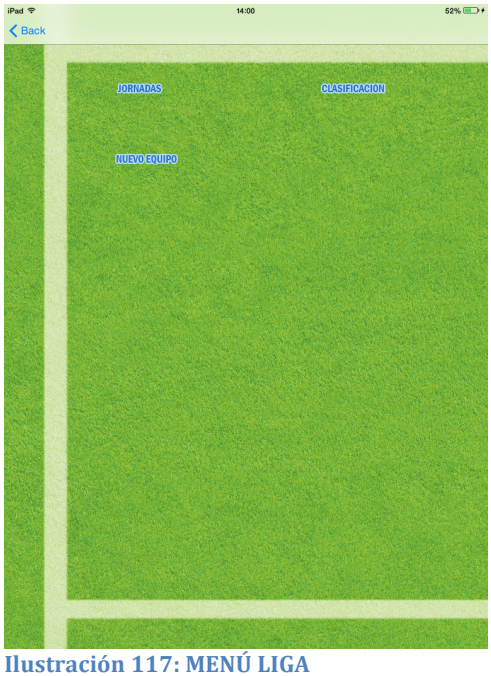


Ilustración 117: MENÚ LIGA

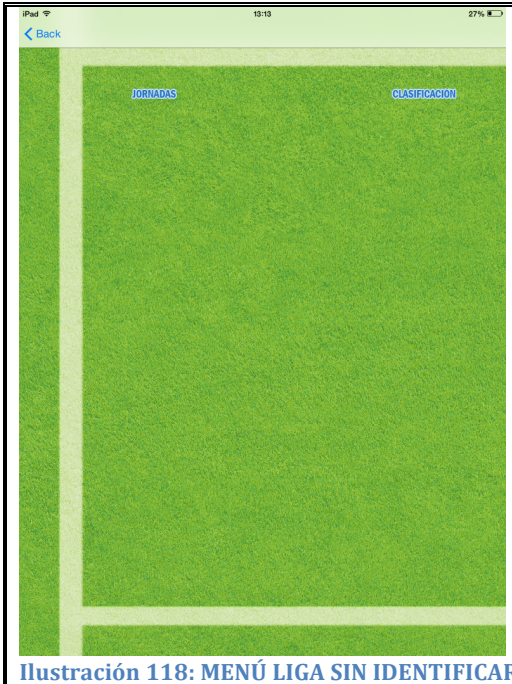
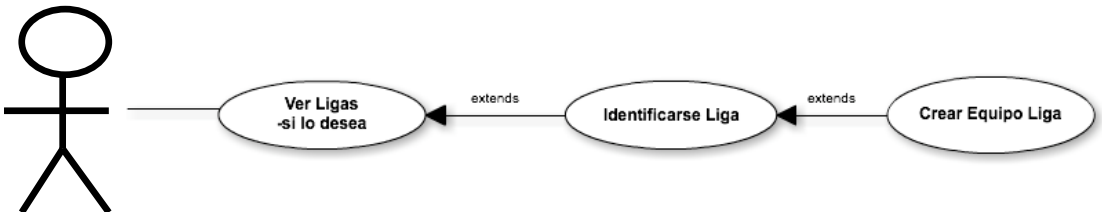


Ilustración 118: MENÚ LIGA SIN IDENTIFICAR



Usuario

Nombre: Crear Equipo Liga

Descripción: Un usuario podrá crear un equipo para que participe en una liga existente.

Actores: Usuario

Precondiciones: Estar identificado en la liga.

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará sobre el botón "Nuevo Equipo" del Menú de la Liga. (Ilust. 119)
2. Introducirá el nombre del equipo que quiera añadir a esta liga y pulsará "Aceptar". (Ilust. 120)

Postcondiciones: Ninguna.

Interfaz Gráfica:

Ilustración 119: MENÚ LIGA

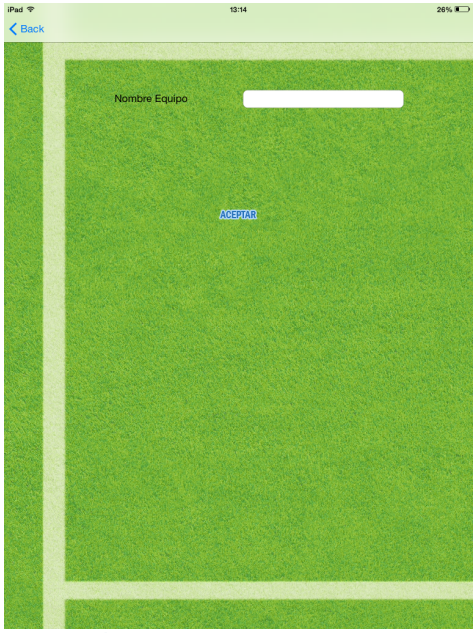
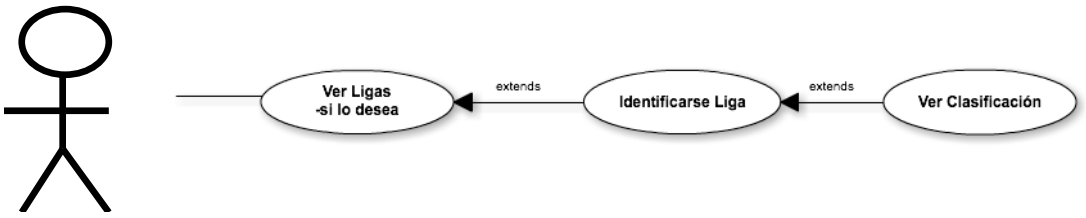


Ilustración 120: AGREGAR EQUIPO LIGA



Nombre: Ver Clasificación

Descripción: Un usuario podrá ver la clasificación con todos los datos actualizados según los partidos que se hayan disputado en esa liga.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará sobre el botón "Clasificación" del Menú de la Liga. (Ilust. 121)
2. Se mostrará la clasificación con todos los datos actualizados según hayan ocurrido a lo largo de la temporada. (Ilust. 122)

Postcondiciones: Ninguna

Interfaz Gráfica:

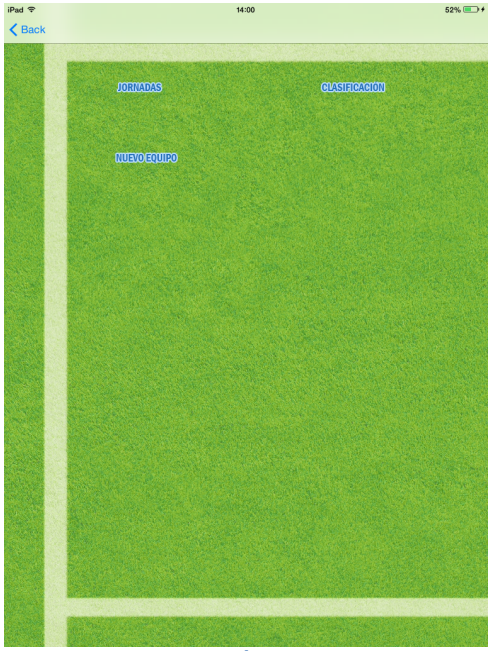
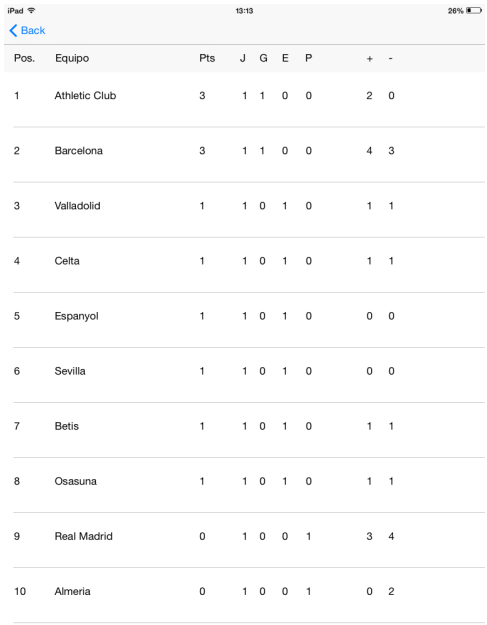
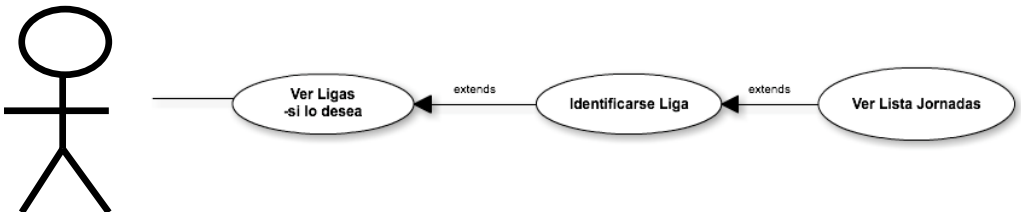


Ilustración 121: MENÚ LIGA



Pos.	Equipo	Pts	J	G	E	P	+	-
1	Athletic Club	3	1	1	0	0	2	0
2	Barcelona	3	1	1	0	0	4	3
3	Valladolid	1	1	0	1	0	1	1
4	Celta	1	1	0	1	0	1	1
5	Espanyol	1	1	0	1	0	0	0
6	Sevilla	1	1	0	1	0	0	0
7	Betis	1	1	0	1	0	1	1
8	Osasuna	1	1	0	1	0	1	1
9	Real Madrid	0	1	0	0	1	3	4
10	Almeria	0	1	0	0	1	0	2

Ilustración 122: CLASIFICACIÓN



Usuario

Nombre: Ver Lista Jornadas

Descripción: Un usuario podrá ver la lista de jornadas que se hayan creado para una liga.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará sobre el botón "Jornadas" del Menú de la Liga. (Ilust. 123)
2. Se mostrará la lista con todas las jornadas que se hayan creado para la liga en particular. (Ilust. 124)
   [Si se ha identificado en la liga y pulsa el botón "+"]
   a. EXTENDS "Crear jornada"

Postcondiciones: Ninguna

Interfaz Gráfica:

Ilustración 123: MENÚ LIGA

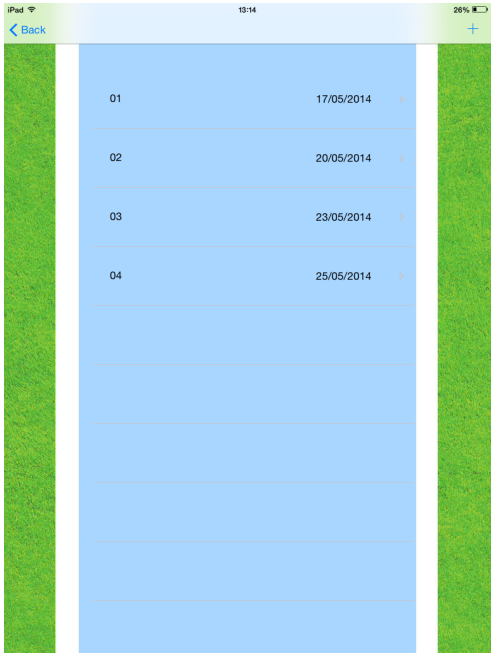
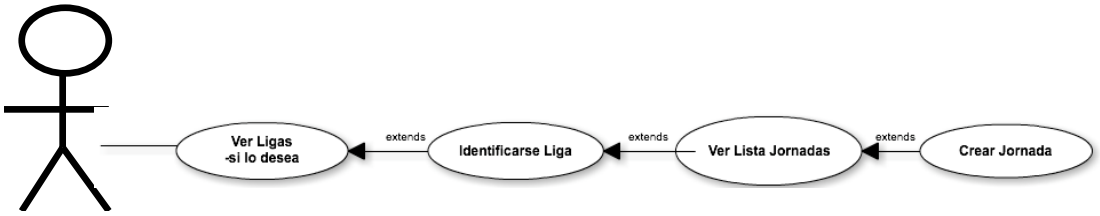


Ilustración 124: LISTA JORNADAS



Nombre: Crear Jornada Liga

Descripción: Un usuario podrá crear una jornada de una liga en la que participe.

Actores: Usuario

Precondiciones: Estar identificado en la liga

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. En la lista de jornadas el usuario pulsará sobre el botón "+" situado en la parte superior derecha para crear una nueva jornada. (Ilust. 125)
2. Rellenará los datos correspondientes a cada campo y pulsara el botón "Añadir" (Ilust. 126)

Postcondiciones: Ninguna

Interfaz Gráfica:

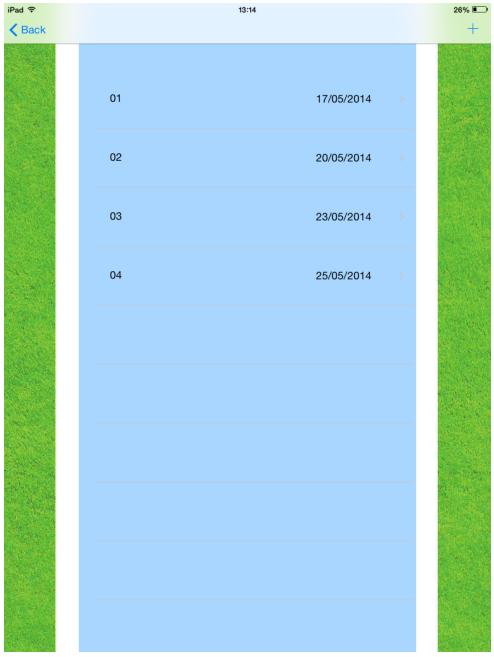


Ilustración 125: LISTA JORNADAS

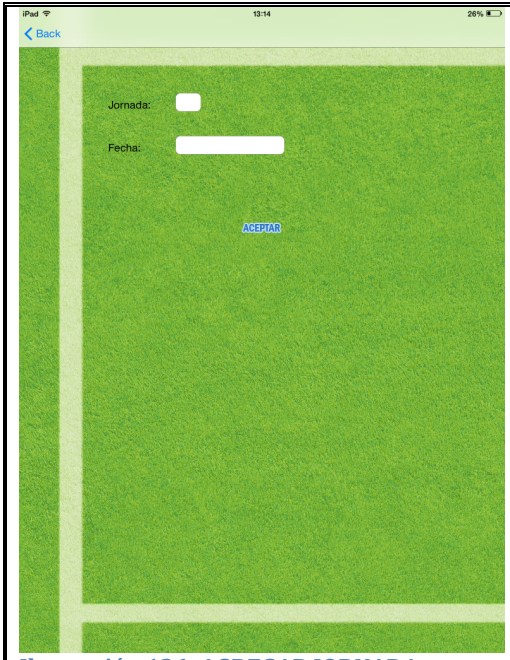
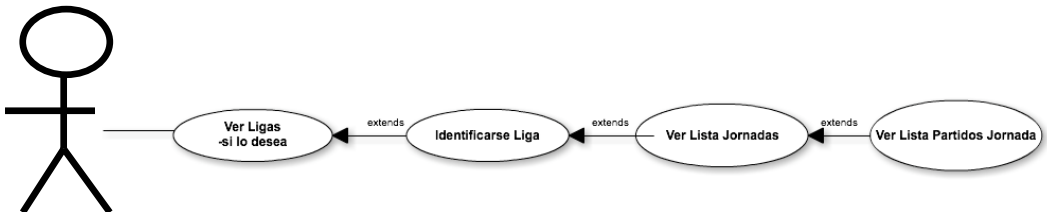


Ilustración 126: AGREGAR JORNADA



Usuario

Nombre: Ver Lista Jornadas

Descripción: Un usuario podrá ver la lista de jornadas que se hayan creado para una liga.

Actores: Usuario

Precondiciones: Ninguna

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará sobre una jornada de la lista de jornadas. (Ilust. 127)
2. Se mostrará la lista con todas las jornadas que se hayan creado para la liga en particular. (Ilust. 128)
   [Si se ha identificado en la liga y pulsa el botón "+"]
   a. EXTENDS "Crear partido jornada"

Postcondiciones: Ninguna

Interfaz Gráfica:

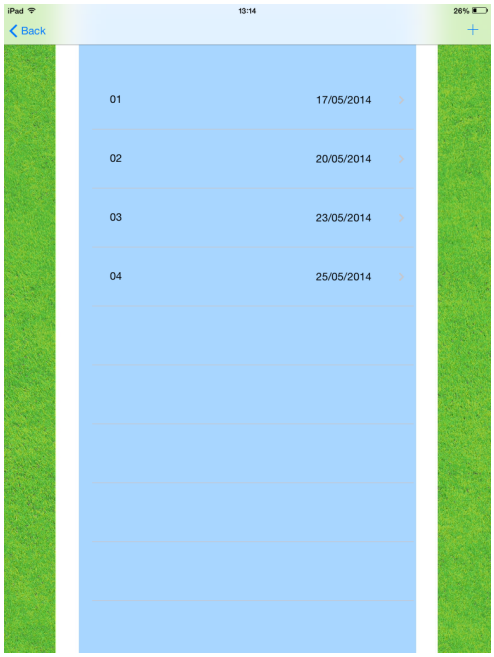


Ilustración 127: LISTA JORNADAS

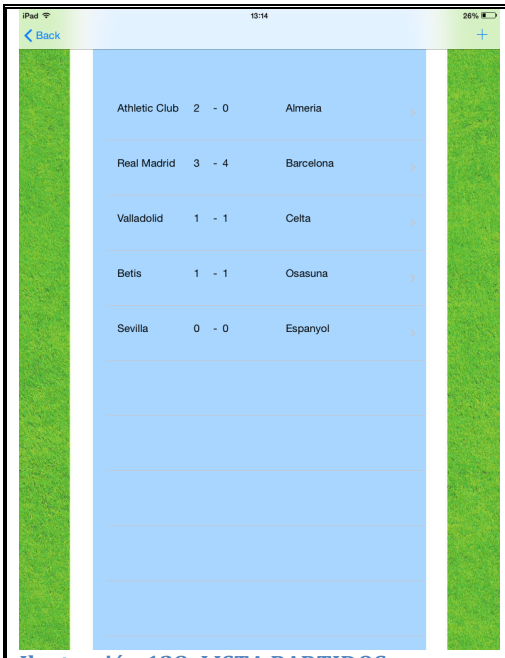
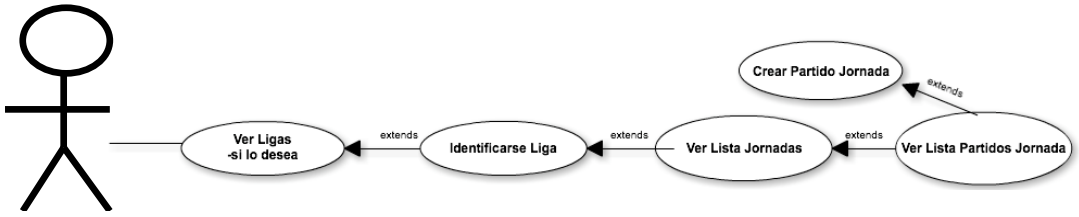


Ilustración 128: LISTA PARTIDOS



Nombre: Crear Partido Jornada

Descripción: Un usuario podrá crear un partido de una jornada que ya exista en la liga.

Actores: Usuario

Precondiciones: Que haya al menos una jornada creada y que el usuario este identificado en la liga.

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. En la lista de jornadas el usuario pulsará sobre el botón "+" situado en la parte superior derecha para crear un nuevo partido. (Ilust. 129)
2. Rellenará los datos correspondientes a cada campo y pulsara el botón "Aceptar" (Ilust. 130)

Postcondiciones: Ninguna

Interfaz Gráfica:

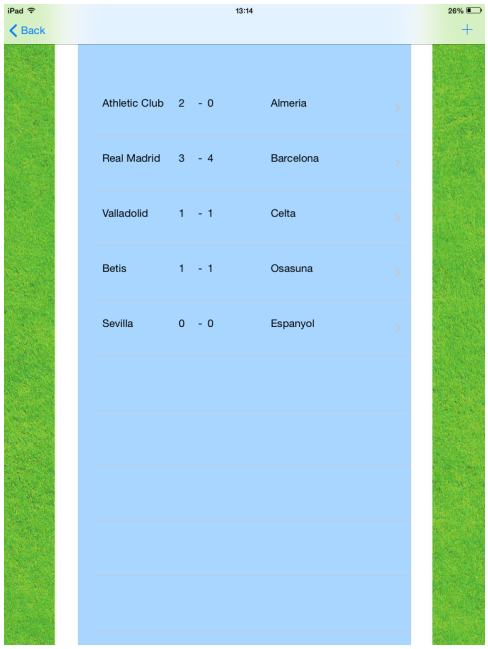


Ilustración 129: LISTA PARTIDOS

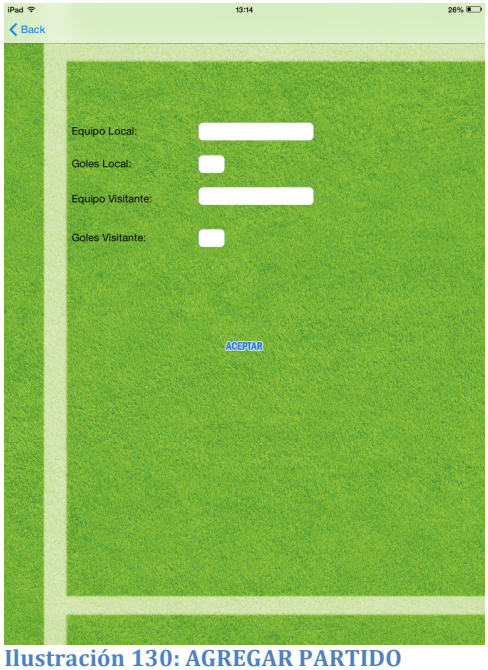
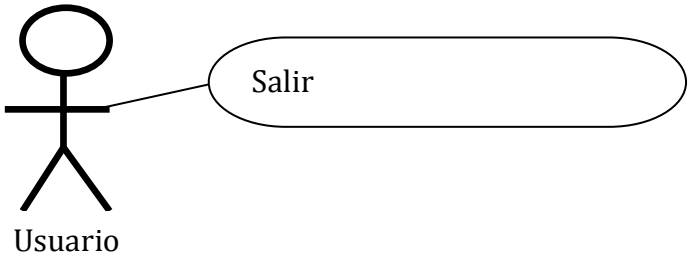


Ilustración 130: AGREGAR PARTIDO



Nombre: Salir

Descripción: Un usuario podrá abandonar la sesión iniciada.

Actores: Usuario

Precondiciones: Estar identificado en el sistema.

Requisitos no funcionales: Ninguno

Flujo de Eventos:

1. El usuario pulsará sobre el botón "Cerrar Sesión" del Menú Principal. (Ilust. 131)
2. La sesión se cerrará y se volverá a la pantalla inicial. (Ilust. 132)

Postcondiciones: Ninguna

Interfaz Gráfica:

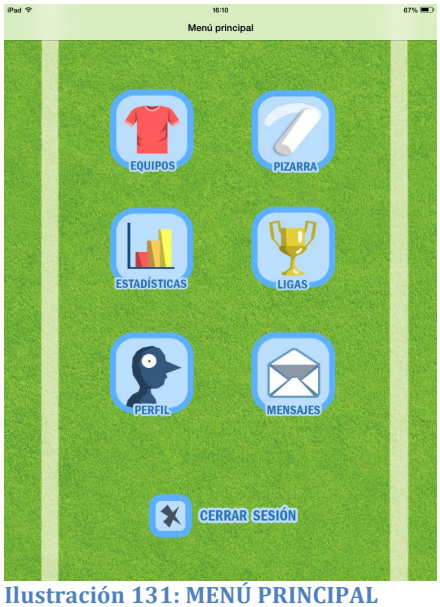


Ilustración 131: MENÚ PRINCIPAL

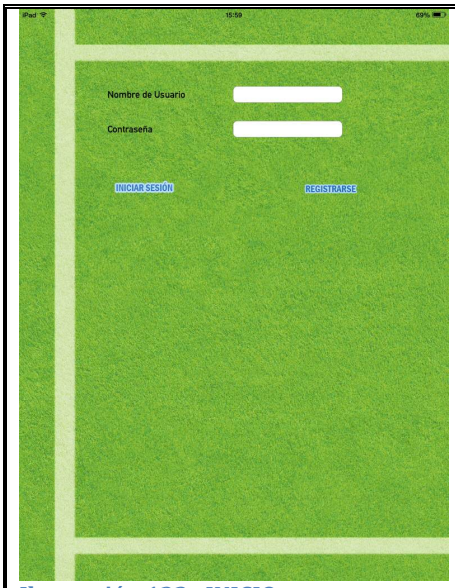


Ilustración 132: INICIO

ANEXO II. DIAGRAMAS DE SECUENCIA

1. Ver lista de equipos

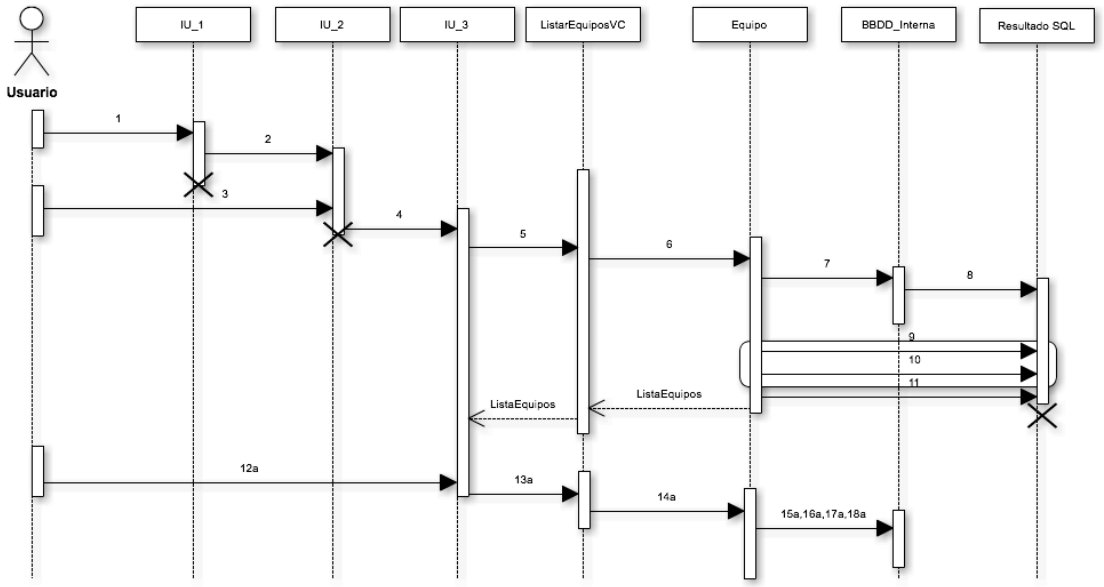


Ilustración 133: SECUENCIA VER LISTA DE EQUIPOS

1. El usuario pulsa sobre el icono Equipos del Menú Principal
2. new()
3. El usuario pulsa sobre el icono Mis Equipos del Menú de Equipos
4. new()
5. inicializar()
6. cargarEquipos():ListaEquipos
7. execSQL("SELECT * FROM Equipos")
8. new()
9. next()
10. getString("Nombre"):String
11. close()

    [Si el usuario quiere eliminar un equipo]

    12a. El usuario desliza el dedo sobre el equipo a eliminar

    13a. EliminarEquipo(idEquipo):void

    14a. EliminarEquipo(idEquipo):void

    15a. execSQL("DELETE FROM Jugadores WHERE idEquipo=%idEquipo%)

    16a. execSQL("DELETE FROM Partidos WHERE idEquipo=%idEquipo%)

    17a. execSQL("DELETE FROM Estadisticas WHERE idEquipo=%idEquipo%)

    18a. execSQL("DELETE FROM Equipos WHERE idEquipo=%idEquipo%)

2. Ver lista de jugadores

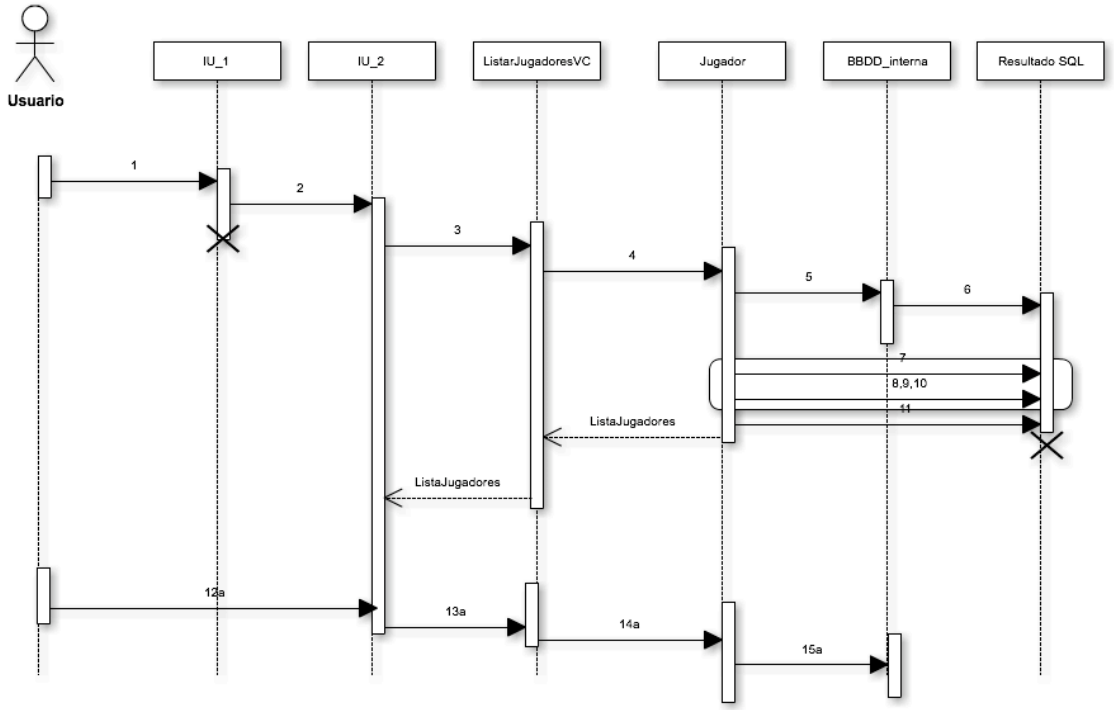


Ilustración 134: SECUENCIA VER LISTA DE JUGADORES

1. El usuario selecciona un equipo de la lista de equipos
2. new(idEquipo)
3. inicializar()
4. cargarJugadores(idEquipo):ListaJugadores
5. execSQL("SELECT * FROM Jugadores WHERE idEquipo=%idEquipo%")
6. new()
7. next()
8. getString("Nombre"):String
9. getString("Apellidos"):String
10. getInt("Dorsal"):int
11. close()

    [Si el usuario quiere eliminar un jugador]

    12a. El usuario desliza el dedo sobre el jugador a eliminar

    13a. EliminarJugador(idJugador):void

    14a. EliminarJugador(idJugador):void

    15a. execSQL("DELETE FROM Jugadores WHERE idJugador=%idJugador%)

3. Ver lista partidos

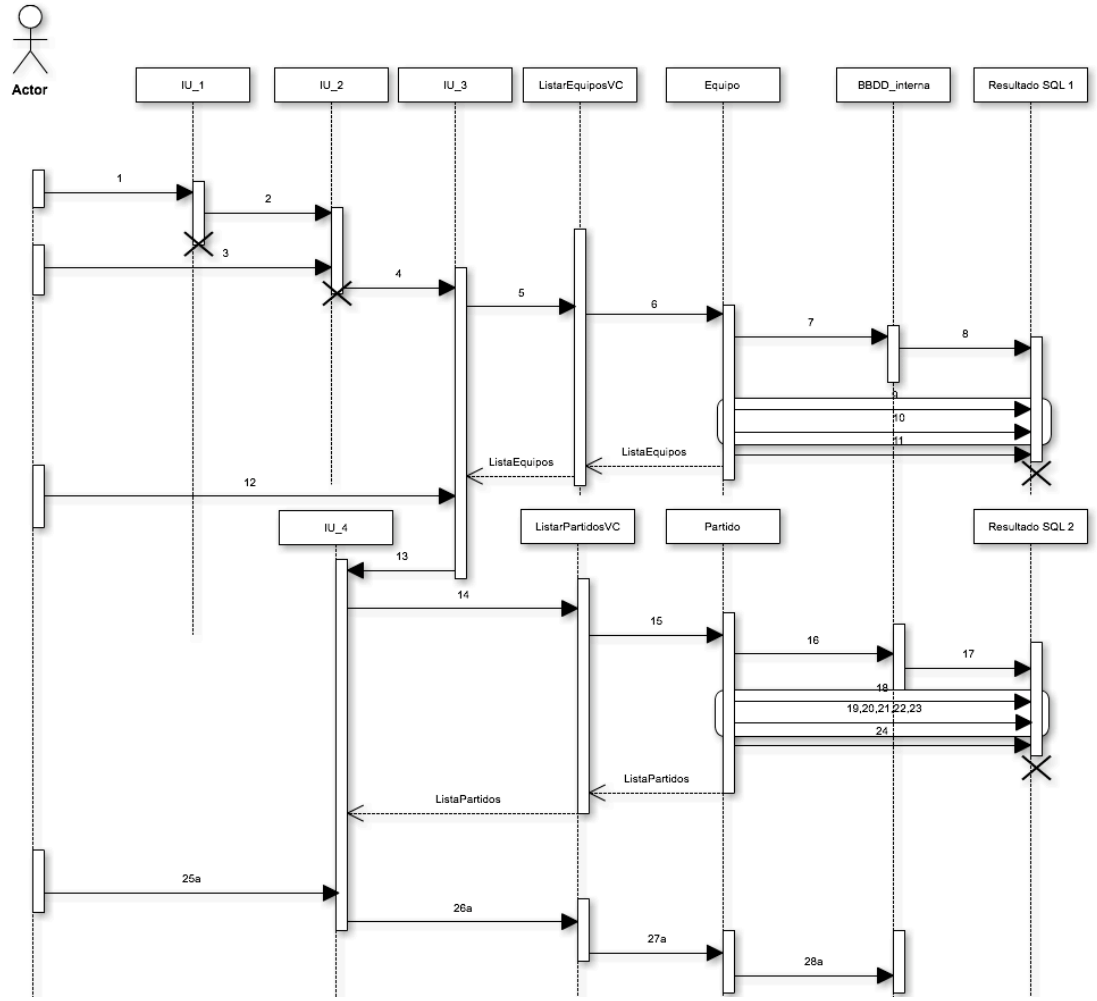


Ilustración 135: SECUENCIA VER LISTA DE PARTIDOS

1. El usuario pulsa sobre el icono Estadisticas del Menú Principal
2. new()
3. El usuario pulsa sobre el icono Partidos del Menú de Estadísticas
4. new()
5. inicializar()
6. cargarEquipos():Lista Equipos
7. execSQL("SELECT * FROM Equipos")
8. new()
9. next()
10. getString("Nombre"):String
11. close()
12. El usuario selecciona un equipo de la lista de equipos
13. new(idEquipo)
14. viewDidLoad()

15. cargarPartidos(idEquipo):ListaPartidos
16. execSQL(“SELECT * FROM Partidos WHERE idEquipo=%idEquipo%”)
17. new()
18. next()
19. getString(“NombreLocal”):String
20. getInt(“GolesLocal”):int
21. getString(“NombreVisitante”):String
22. getInt(“GolesVisitante”):int
23. getString(“FechaPartido”):String
24. close()

    [Si el usuario quiere eliminar un partido]

    25a. El usuario desliza el dedo sobre el partido a eliminar

    26a. EliminarPartido(idPartido):void

    27a. EliminarPartido(idPartido):void

    28a. execSQL(“DELETE FROM Partidos WHERE idPartido=%idPartido%)

4. Ver información Equipo

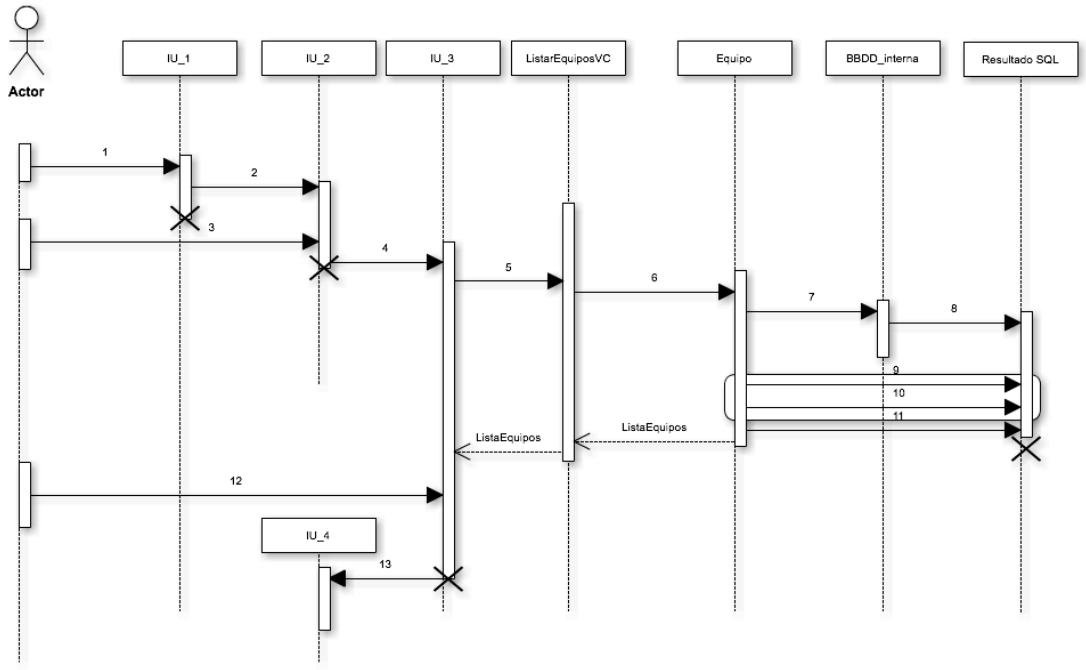


Ilustración 136: SECUENCIA VER INFORMACIÓN EQUIPO

1. El usuario pulsa sobre el icono Estadísticas del Menú Principal
2. new()
3. El usuario pulsa sobre el icono Partidos del Menú de Estadísticas
4. new()
5. inicializar()
6. cargarEquipos():Lista Equipos
7. execSQL("SELECT * FROM Equipos")
8. new()
9. next()
10. getString("Nombre"):String
11. close()
12. El usuario selecciona un equipo de la lista de equipos
13. new(idEquipo,nombre,categoría,provincia,temporada,partidosJugados,partidosGanados,partidosEmpatados,partidosPerdidos)

5. Ver información jugador

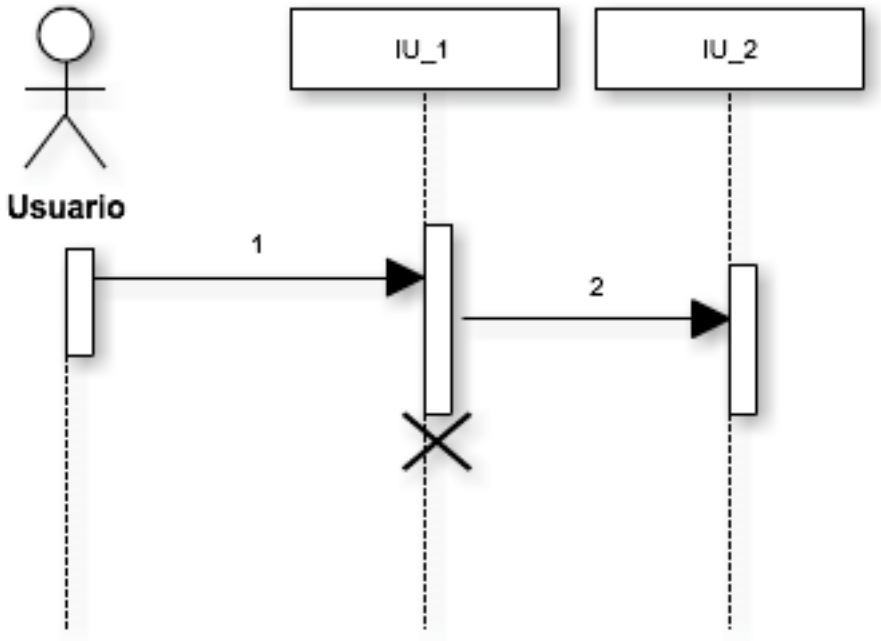


Ilustración 137: SECUENCIA VER INFORMACIÓN JUGADOR

1. El usuario selecciona un equipo de la lista de equipos
2. New(idJugador,nombre,apellidos,dorsal,fechaNac,posición,minutos,amarillas,rojas,asistencias,goles)

6. Ver estadísticas partido

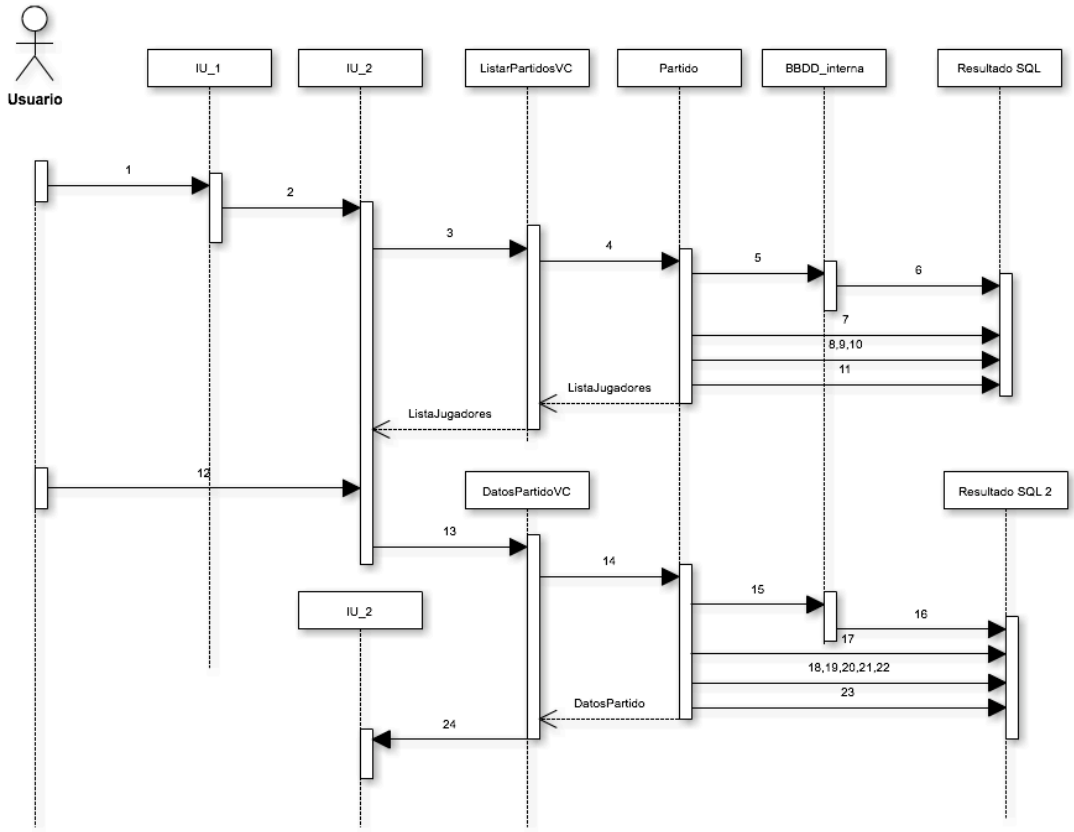


Ilustración 138: SECUENCIA VER ESTADÍSTICAS PARTIDO

1. El usuario selecciona un partido de la lista de partidos
2. new(idPartido,idEquipo)
3. inicializar()
4. cargarJugadores(idEquipo):ListaJugadores
5. execSQL("SELECT * FROM Jugadores WHERE idEquipo=%idEquipo%")
6. new()
7. next()
8. getString("Nombre"):String
9. getString("Apellidos"):String
10. getInt("Dorsal")
11. close()
12. El usuario pulsa sobre el icono de información que hay sobre la celda del jugador
13. new(idPartido,idJugador)
14. obtenerDatosPartido(idPartido,idJugador):DatosPartido
15. execSQL("SELECT * FROM estadísticas WHERE idPartido=%idPartido% AND idJugador=%idJugador%")
16. new()
17. next()
18. getInt("Minutos"):int

19. getInt(“Amarillas”):int
20. getInt(“Rojas”):int
21. getInt(“Asistencias”):int
22. getInt(“Goles”):int
23. close()
24. new(idJugador,idPartido,nombre,apellidos,minutos,amarillas,rojas,asistencias,goles)

7. Crear equipo

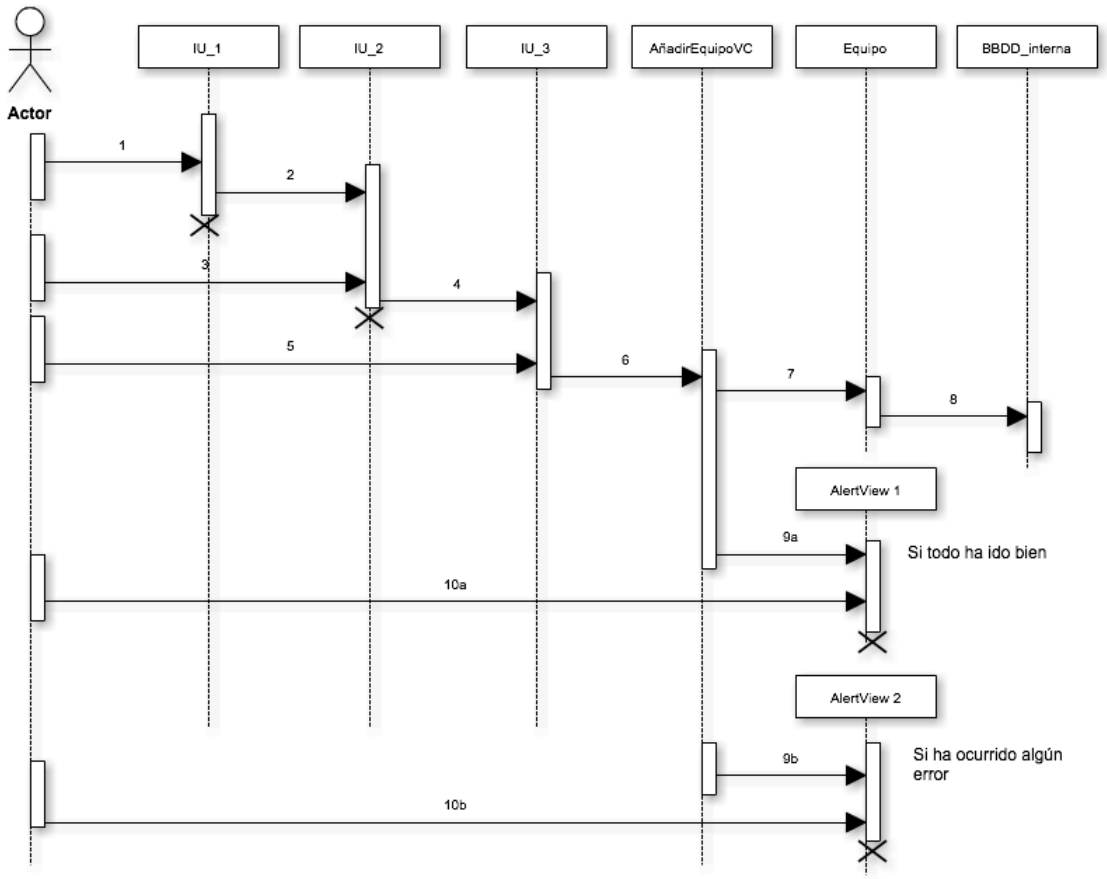


Ilustración 139: SECUENCIA CREAR EQUIPO

1. El usuario selecciona el icono equipos
2. new()
3. El usuario pulsa el icono Crear Equipo
4. new()
5. El usuario introduce los datos del Equipo y pulsa “Aceptar”
6. insertarEquipo(nombre,temporada,categoría,provincia)
7. insertarEquipo(nombre,temporada,categoría,provincia,partidosJugados,partidosGanados,partidosEmpatados,partidosPerdidos)
8. execSQL(“INSERT INTO Equipos (nombre,temporada,categoría,provincia,partidosJugados,partidosGanados,partidosEmpatados,partidosPerdidos) VALUES (%nombre%,%temporada%,%categoría%,%provincia%,0,0,0,0))
   [Si se ha añadido todo correctamente]
   9.a. new()
   10.a Equipo añadido. Aceptar
   [Si ha ocurrido algún error]
   9.b. new()
   10.b. Error. Aceptar

8. Crear Jugador

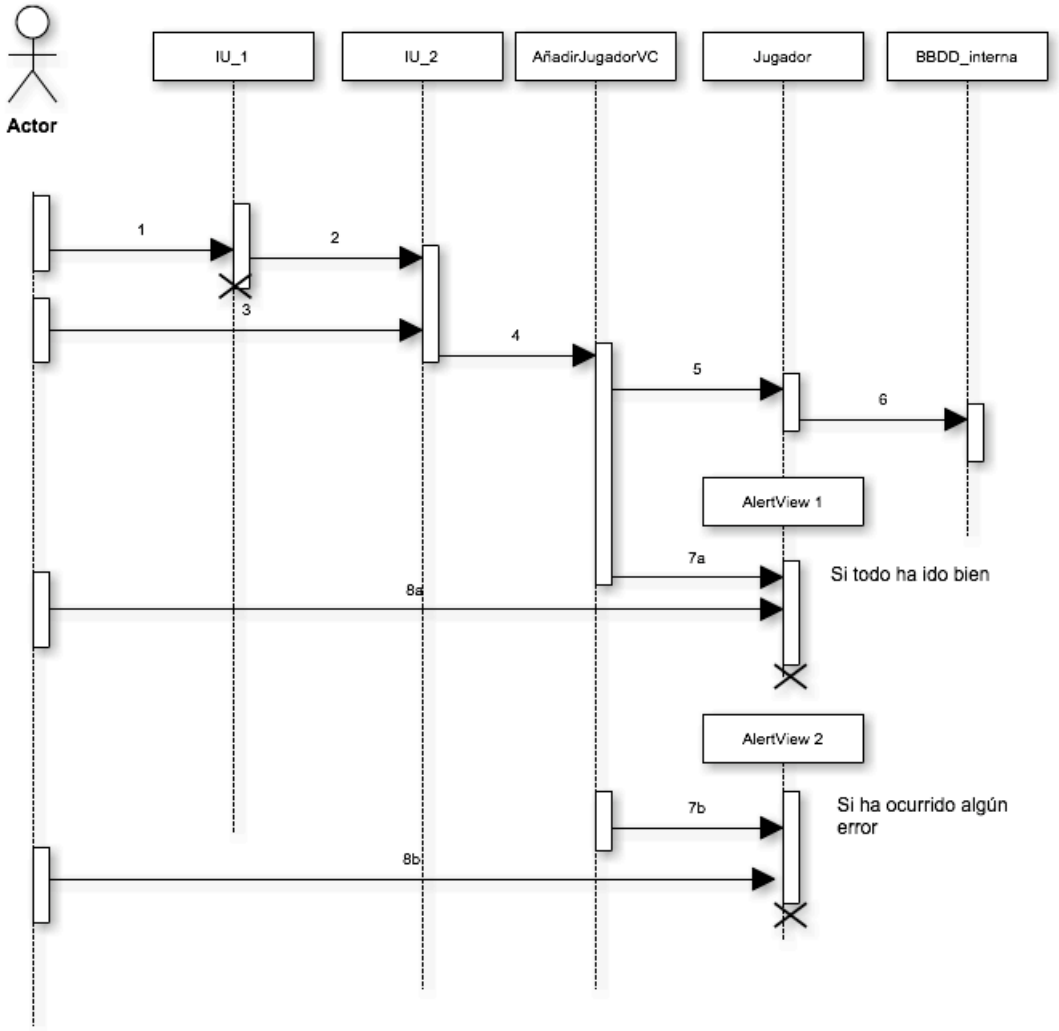


Ilustración 140: SECUENCIA CREAR JUGADOR

1. El usuario pulsa el botón "+" situado en la parte superior derecha de la lista de jugadores
2. New(idEquipo)
3. El usuario introduce los datos del Jugador y pulsa "Aceptar"
4. insertarJugador(nombre,apellidos,fechaNac,posición,dorsal,idEquipo)
5. insertarJugador(nombre,apellidos,fechaNac,posición,dorsal,idEquipo,minutos,amarillas,rojas,asistencias,goles)
6. execSQL("INSERT INTO jugadores (nombre,apellidos,fechaNac,posición,dorsal,idEquipo,minutos,amarillas,rojas,asistencias,goles) VALUES (%nombre%,%apellidos%,%fechaNac%,%posición%,%dorsal%,%idEquipo%,0,0,0,0,0)
   [Si se ha añadido todo correctamente]
   7.a. new()
   8.a Jugador añadido. Aceptar
   [Si ha ocurrido algún error]
   7.b. new()

8.b. Error. Aceptar

9. Crear partido

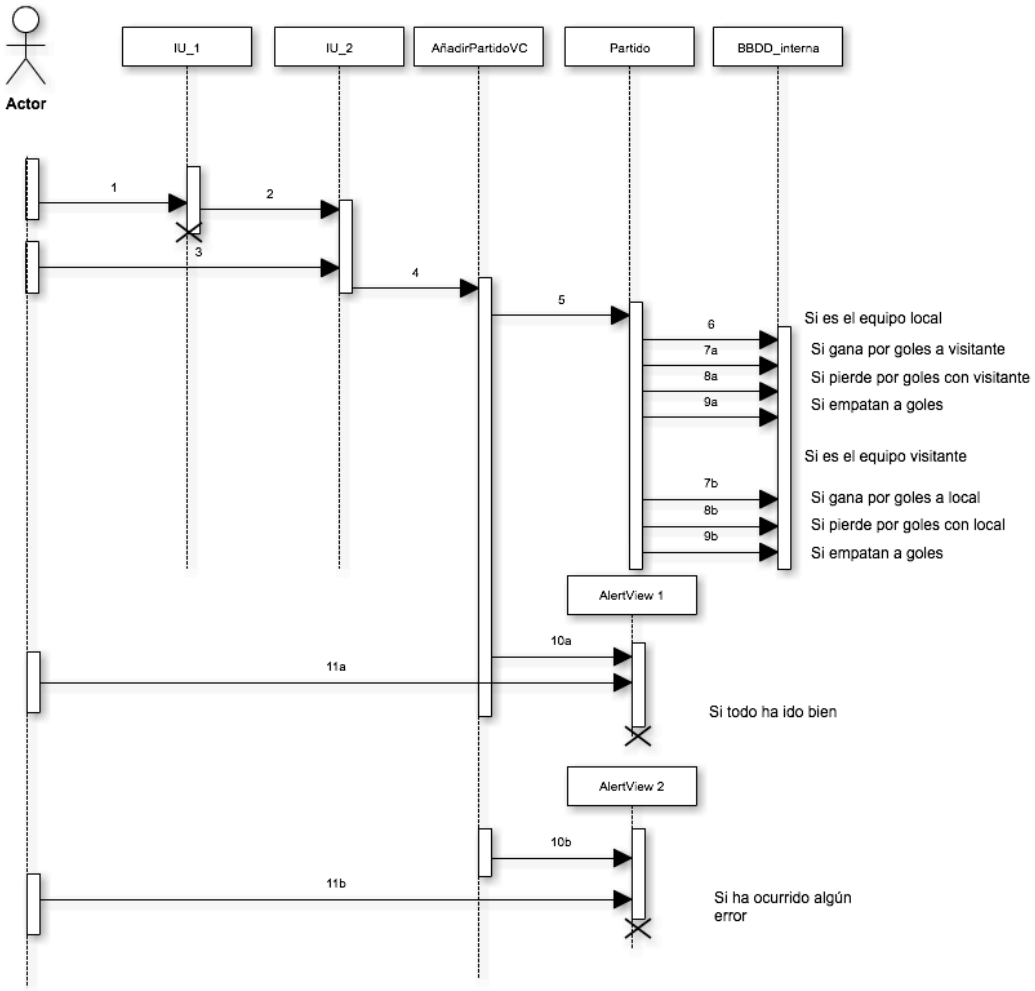


Ilustración 141: SECUENCIA CREAR PARTIDO

1. El usuario pulsa el botón "+" situado en la parte superior derecha de la lista de partidos
2. new(idEquipo,nombreEquipo)
3. El usuario introduce los datos del Partido y pulsa "Aceptar"
4. agregarPartido(equipoLocal,golesLocal,equipoVisitante,golesVisitante,fecha Partido,idEquipo)
5. agregarPartido(equipoLocal,golesLocal,equipoVisitante,golesVisitante,fecha Partido,idEquipo)
6. execSQL("INSERT INTO partidos (equipoLocal,golesLocal,equipoVisitante,golesVisitante,fechaPartido,idEquipo) VALUES (%equipoLocal%,%golesLocal%,%equipoVisitante%,%golesVisitante%,%fechaPartido%,%idEquipo%))
   [Si es el equipo local]
      [Si gana por goles al visitante]
      7.a. execSQL("UPDATE equipos SET partidosJugados= partidosJugados + 1, partidosGanados=partidosGanados +1 WHERE idEquipo=%idEquipo%")

[Si pierde por goles con el visitante
8.a. execSQL("UPDATE equipos SET partidosJugados= partidosJugados + 1, partidosPerdidos=partidosPerdidos +1 WHERE idEquipo=%idEquipo%")
[Si el número de goles es igual]
9.a. execSQL("UPDATE equipos SET partidosJugados= partidosJugados + 1, partidosEmpatados=partidosEmpatados +1 WHERE idEquipo=%idEquipo%")
[Si es el equipo visitante]
[Si gana por goles al local]
7.b. execSQL("UPDATE equipos SET partidosJugados= partidosJugados + 1, partidosGanados=partidosGanados +1 WHERE idEquipo=%idEquipo%")
[Si pierde por goles con el local]
8.b. execSQL("UPDATE equipos SET partidosJugados= partidosJugados + 1, partidosPerdidos=partidosPerdidos +1 WHERE idEquipo=%idEquipo%")
[Si el número de goles es igual]
9.b. execSQL("UPDATE equipos SET partidosJugados= partidosJugados + 1, partidosEmpatados=partidosEmpatados +1 WHERE idEquipo=%idEquipo%")

[Si se ha añadido todo correctamente]
10.a. new()
11.a Partido añadido. Aceptar
[Si ha ocurrido algún error]
10.b. new()
11.b. Error. Aceptar

10. Crear estadísticas partido

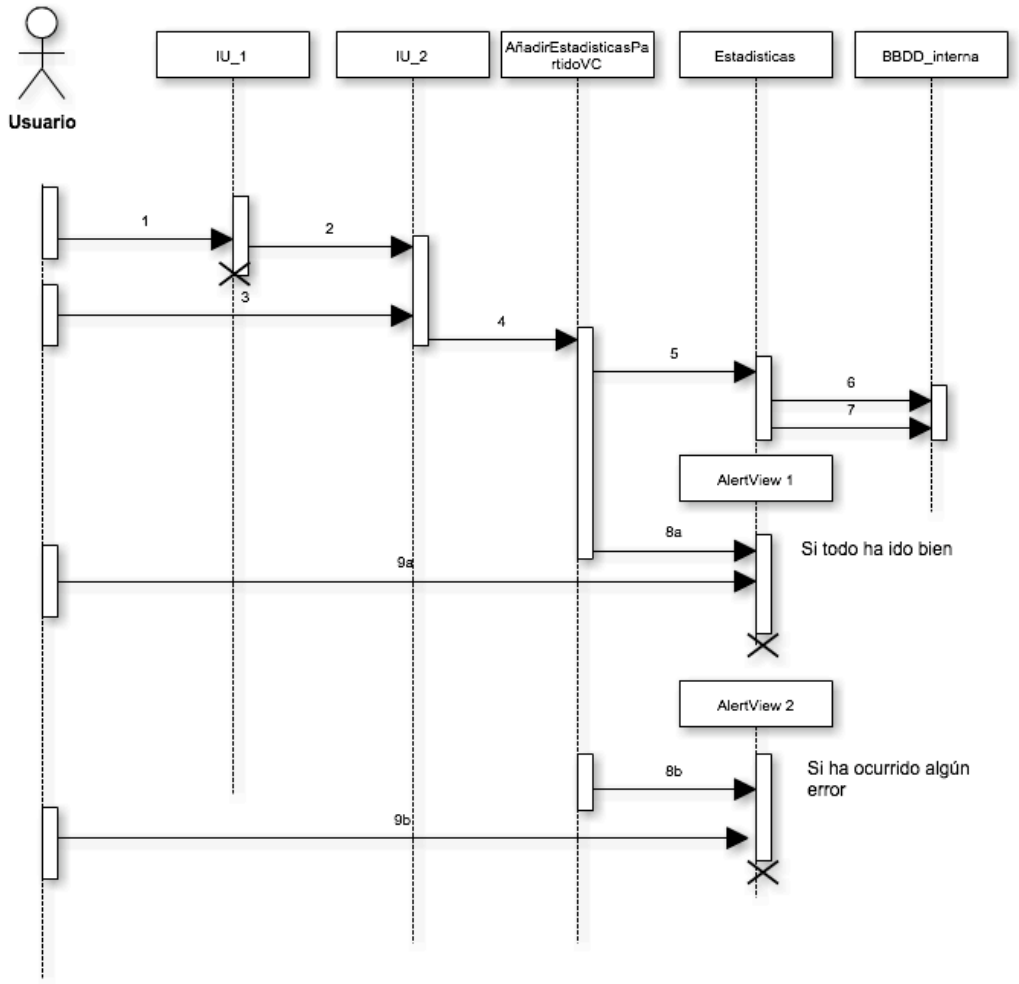


Ilustración 142: SECUENCIA CREAR ESTADÍSTICAS PARTIDO

1. El usuario pulsa sobre la celda de un jugador
2. new(idJugador,idPartido,idEquipo)
3. El usuario introduce los datos sobre las estadísticas del partido y pulsa "Aceptar"
4. agregarEstadisticas(minutos,amarillas,rojas,asistencias,goles,idJugador,idPartido,idEquipo)
5. agregarEstadisticas(minutos,amarillas,rojas,asistencias,goles,idJugador,idPartido,idEquipo)
6. execSQL("INSERT INTO estadisticas(minutos,amarillas,rojas,asistencias,goles,idJugador,idPartido,idEquipo) VALUES (%minutos%,%amarillas%,%rojas%,%asistencias%,%goles%, %idJugador%,%idPartido%,%idEquipo%)")
7. execSQL("UPDATE jugadores SET minutos = minutos + %minutos%, amarillas = amarillas + %amarillas%, rojas = rojas + %rojas%, asistencias = asistencias + %asistencias%, goles = goles + %goles% WHERE idJugador = %idJugador%")

[Si se ha añadido todo correctamente]
8.a. new()
9.a Estadísticas añadidas. Aceptar
[Si ha ocurrido algún error]
8.b. new()
9.b. Error. Aceptar

11. Pizarra

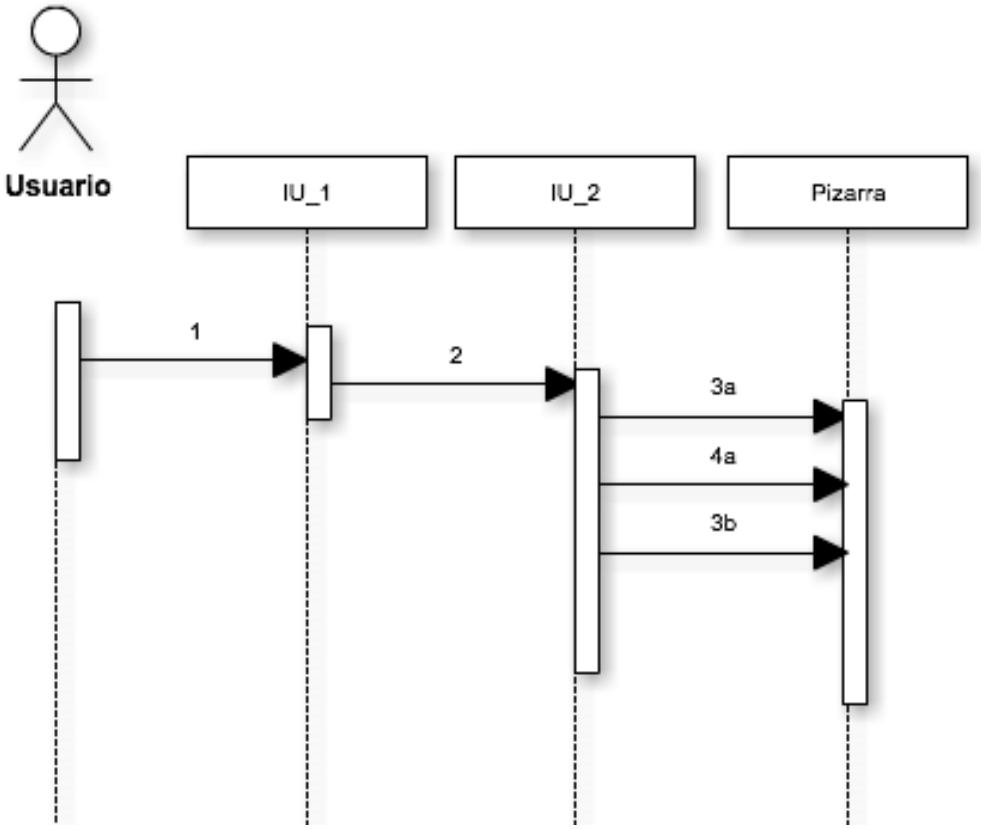


Ilustración 143: SECUENCIA PIZARRA

1. El usuario pulsa el icono “Pizarra” del Menú principal
2. New()
   [si pulsa sobre un color y desliza el dedo para dibujar]
   3a. touchesBegan()
   4a. touchesMoved()
   [si desea borrar el contenido pulsando el botón borrar]
   3b. borrar()

12. Registrarse

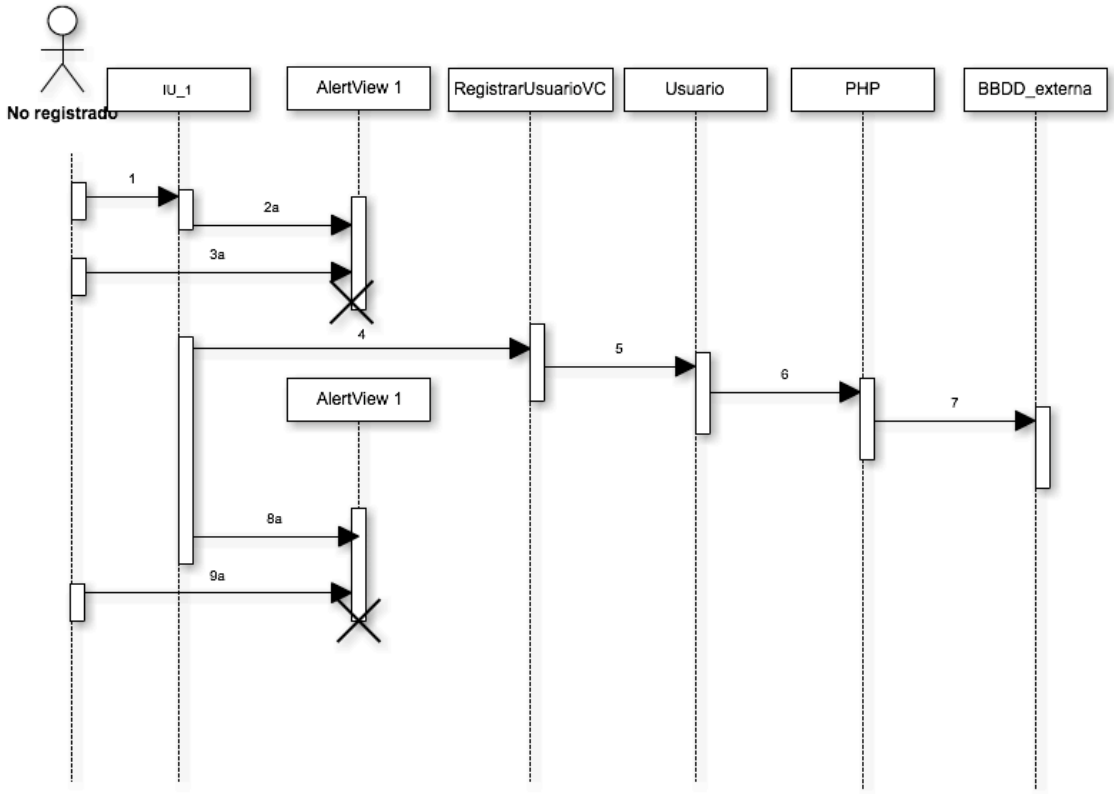


Ilustración 144: SECUENCIA REGISTRARSE

1. El usuario rellena los campos y pulsa el botón aceptar
   [si hay algún campo sin rellenar]
   2a. new()
   3a. Pulsar "Aceptar"
   [Si todos los datos han sido rellenados correctamente y el usuario no existe en el sistema]
4. Registrarse(nombre, apellidos, nombreUsuario, password, mail):void
5. Registrarse(nombre, apellidos, nombreUsuario, password, mail):void
6. Registrarse.php
7. execSQL("INSERT INTO Usuarios (nombre,apellidos,nombreUsuario,mail,password) VALUES (%nombre%,%apellidos%,%nombreUsuario%,%mail%,%password%))
   [Si todo ha ido bien]
   8a. new()
   9a. Pulsar "Aceptar"

13. Iniciar Sesión

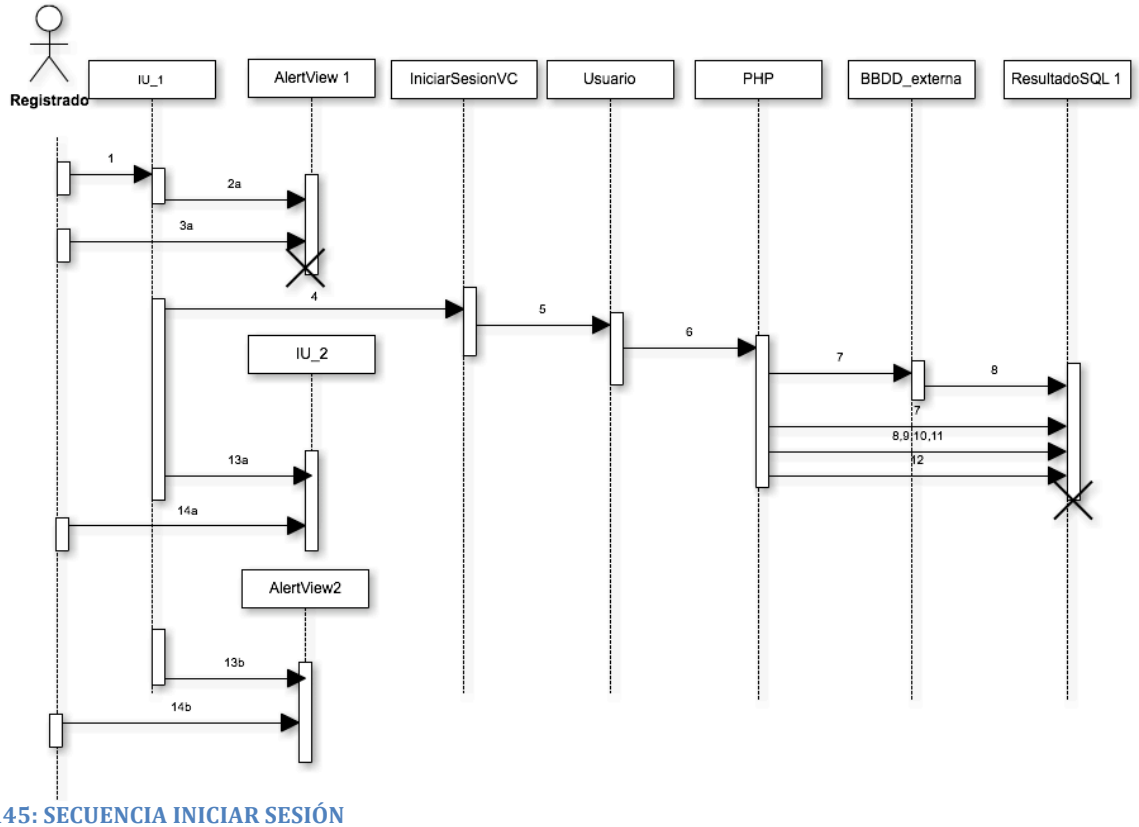


Ilustración 145: SECUENCIA INICIAR SESIÓN

1. El usuario rellena los campos y pulsa el botón identificarse

[si hay algún campo sin rellenar]

2a. new()

3a. Pulsar "Aceptar"

[si todo los datos se han rellenado correctamente]

4. Identificarse(nombreUsuario,password)
5. Identificarse(nombreUsuario,password)
6. inicioSesion.php
7. execSQL("SELECT password FROM usuarios WHERE nombreUsuario=%nombreUsuario% and password=%password%)
8. new()
9. next()
10. getString("nombreUsuario"):String
11. getString("password"):String
12. close

[si la identificación ha sido correcta]

13a. new()

14a. Bienvenido. Pulsar "Aceptar"

[so no lo ha sido]

13b. new()

14b. Error. Pulsar "Aceptar"

14. Lista Mensajes

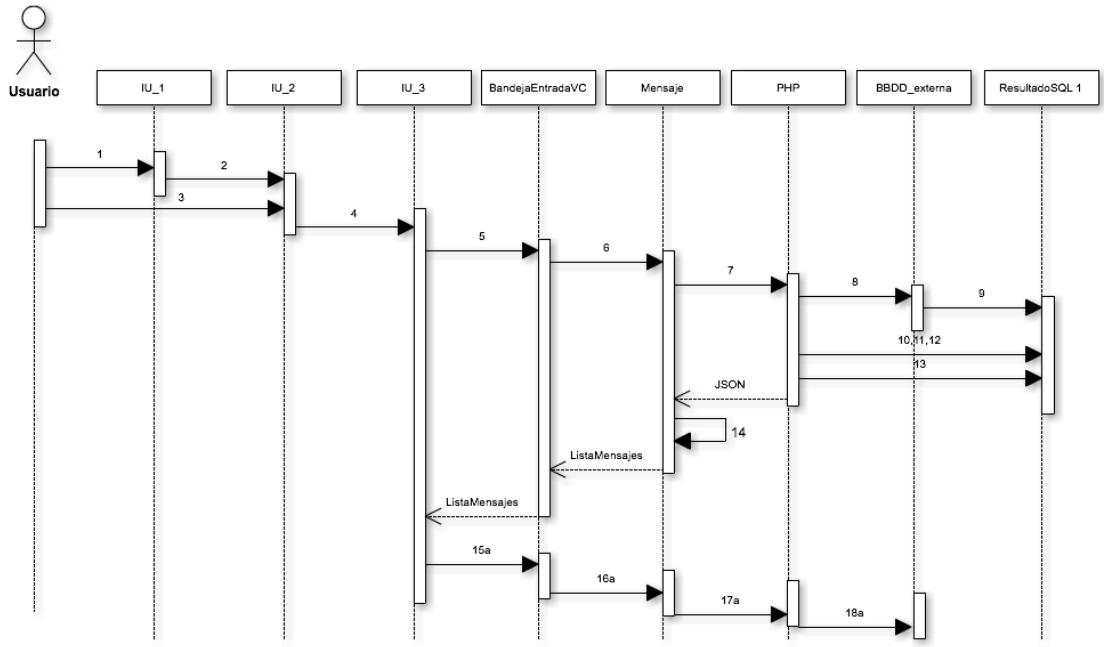


Ilustración 146: SECUENCIA LISTA MENSAJES

1. El usuario pulsa en el icono “Mensajes” del Menú principal
2. New()
3. El usuario pulsa el botón “Bandeja de Entrada” del Menú de mensajes
4. new()
5. Inicializar()
6. cargarMensajes(nombreUsuario):ListaMensajes
7. buscarMensajes.php
8. execSQL(“SELECT * FROM mensajes WHERE nombreUsuario=%usuarioDestino%”)
9. new()
10. next()
11. getString(“usuarioOrigen”):String
12. getString(“asunto”):String
13. close()
14. Deserializar JSON

    [si quiere eliminar un mensaje]

    15a. eliminarMensaje(idMensaje)

    16a. eliminarMensaje(idMensaje)

    17a. eliminarMensaje.php

    18a. execSQL(“DELETE FROM mensajes WHERE idMensaje=%idMensaje%)

15. Buscar usuarios

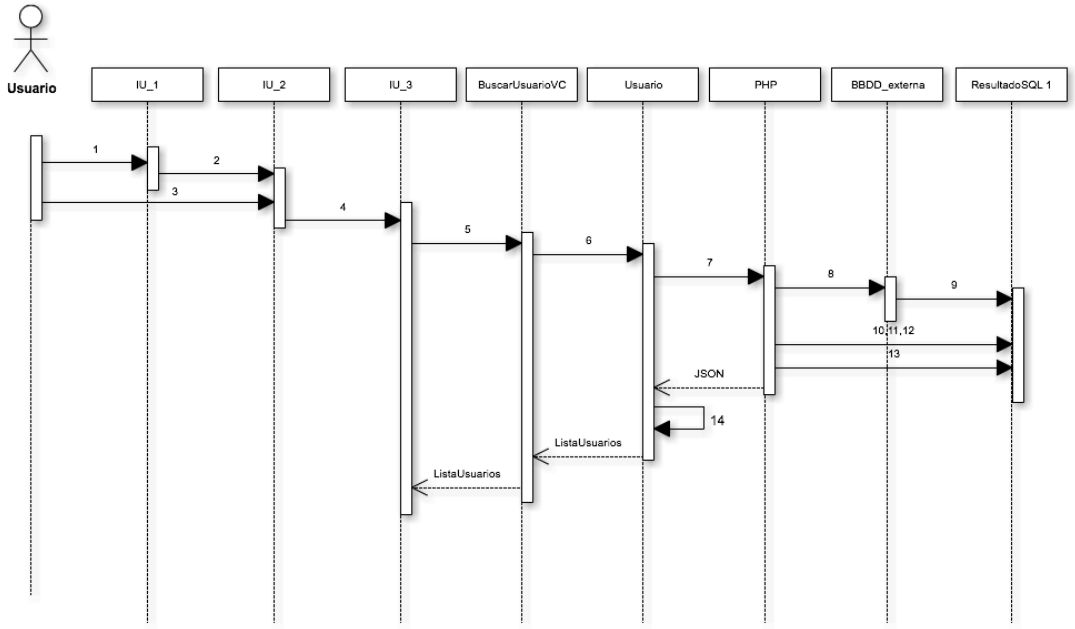


Ilustración 147: SECUENCIA BUSCAR USUARIOS

1. El usuario pulsa en el icono “Mensajes” del Menú principal
2. New()
3. El usuario pulsa el botón “Buscar usuario” del Menú de mensajes
4. new()
5. Inicializar()
6. obtenerUsuarios():ListaUsuarios
7. buscarUsuarios.php
8. execSQL(“SELECT * FROM usuarios)
9. new()
10. next()
11. getString(“nombreUsuario”):String
12. getString(“mail”):String
13. close()
14. Deserializar JSON

16. Enviar mensaje

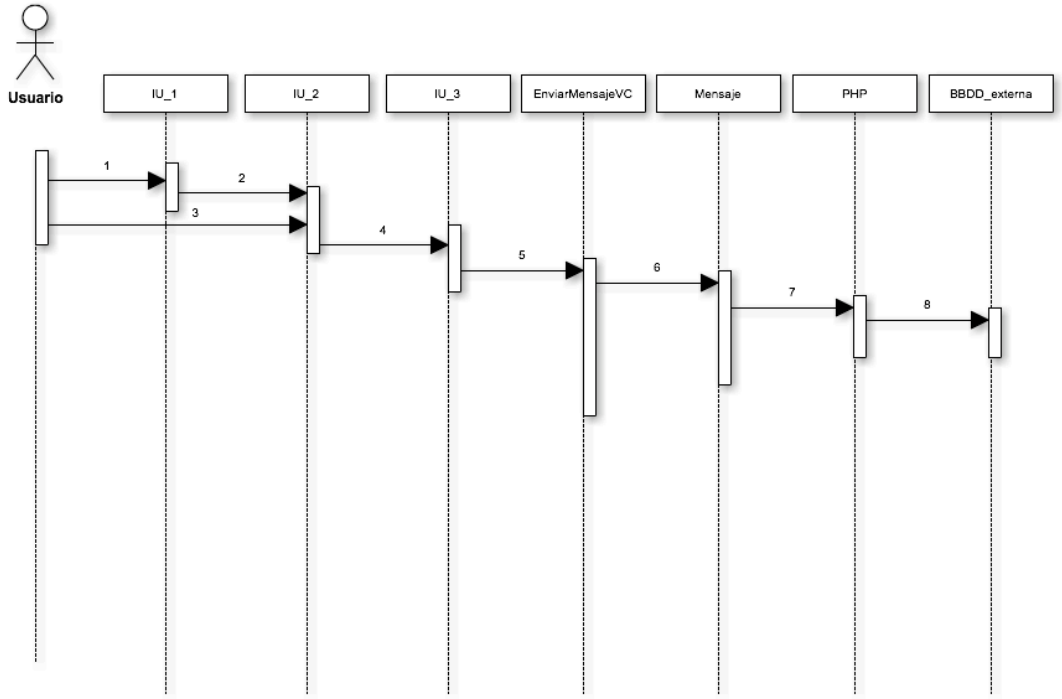


Ilustración 148: SECUENCIA ENVIAR MENSAJE

1. El usuario pulsa en el icono “Mensajes” del Menú principal
2. New()
3. El usuario pulsa el botón “Enviar Mensaje” del Menú de mensajes
4. new()
5. enviarMensaje(usuarioOrigen,usuarioDestino,asunto,texto):void
6. enviarMensaje(usuarioOrigen,usuarioDestino,asunto,texto):void
7. enviarMensaje.php
8. execSQL(“INSERT INTO usuarios (usuarioOrigen,usuarioDestino,asunto,texto) VALUES (%usuarioOrigen%,%usuarioDestino%,%asunto%,%texto%)

17. Identificarse liga

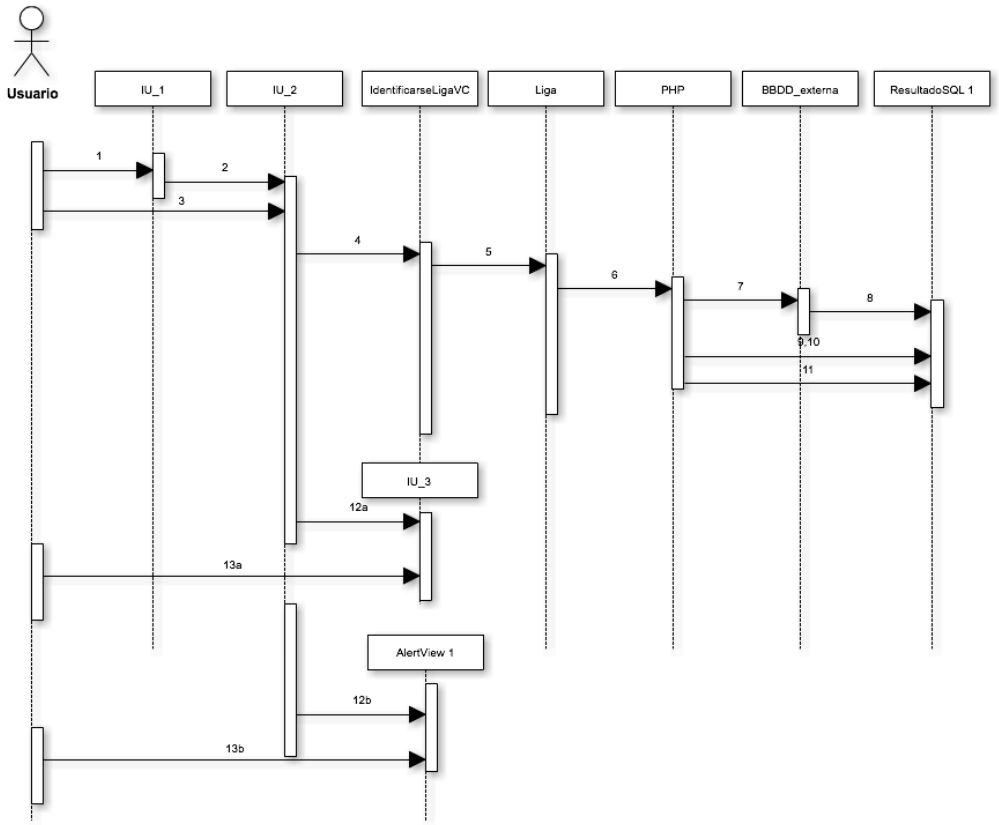


Ilustración 149: SECUENCIA IDENTIFICARSE LIGA

1. El usuario selecciona una liga de la lista de ligas
2. New()
3. El usuario introduce la contraseña de la liga
4. Identificarse(idLiga,password)
5. Identificarse(idLiga,password)
6. identificarseLiga.php
7. execSQL("SELECT password FROM ligas WHERE idLiga=%idLiga%)
8. new()
9. next()
10. getString("password")String
11. close()

    [Si el password coincide con el de la liga]

    12a. New()

    13a. Bienvenido. Pulsar "Aceptar"

    [Si no coincide]

    12b. new()

    13b. Error. Pulsar "Aceptar"

18. Lista jornadas

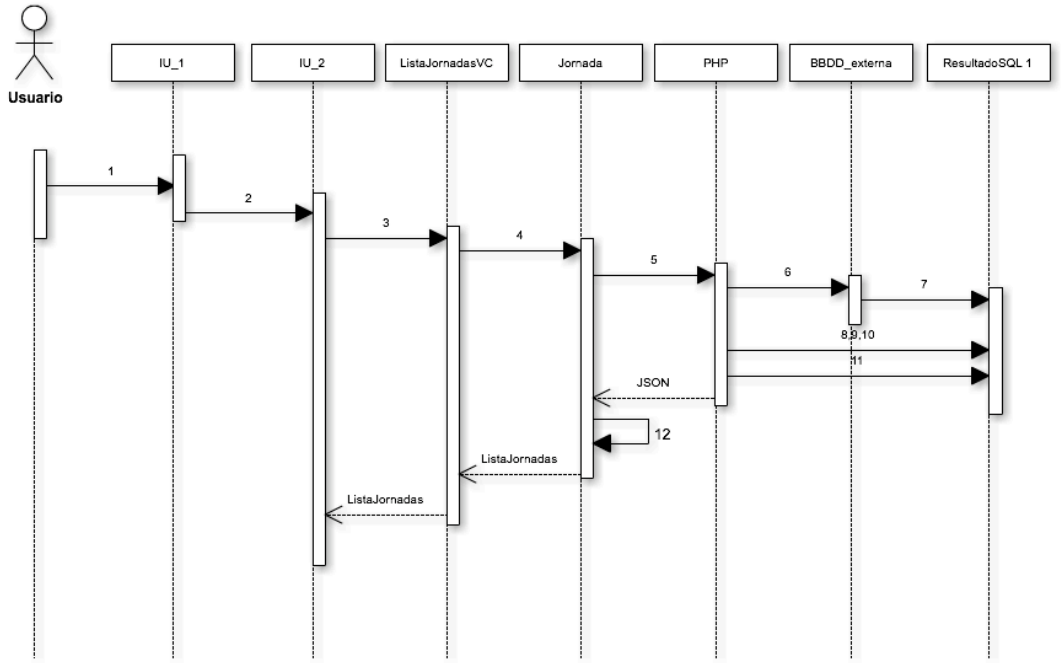


Ilustración 150: SECUENCIA LISTA JORNADAS

1. El usuario pulsa el botón “Jornadas” del Menú de liga
2. new()
3. Inicializar()
4. cargarJornadas(idLiga):ListaJornadas
5. listaJornadas.php
6. execSQL(“SELECT * FROM jornadas WHERE idLiga=%idLiga%”)
7. new()
8. next()
9. getString(“jornada”):String
10. getString(“fecha”):String
11. close()
12. Deserializar JSON

19. Lista partidos

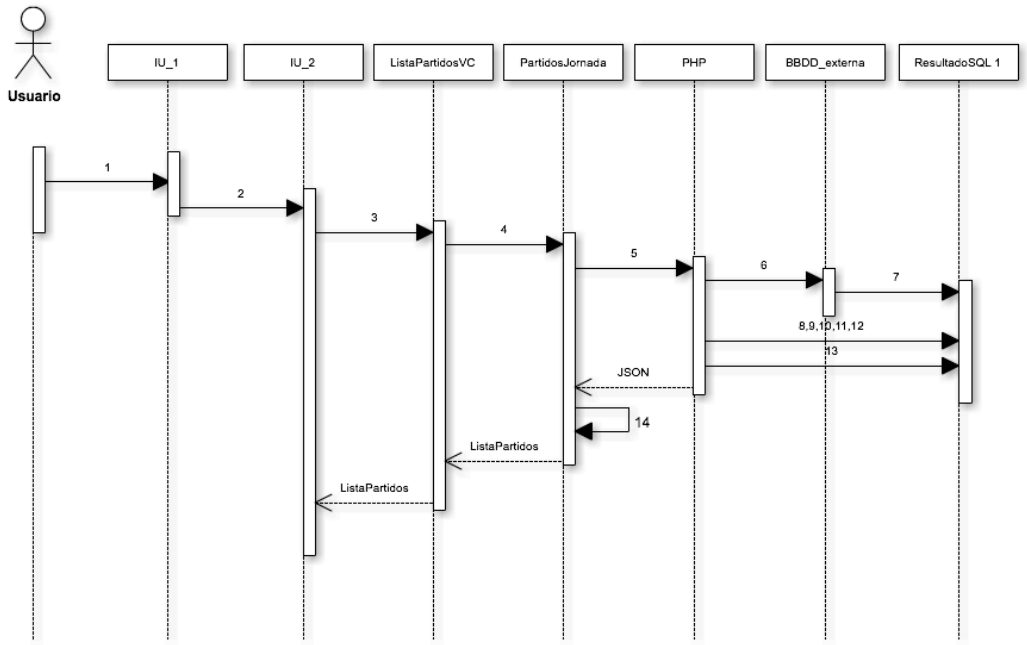


Ilustración 151: SECUENCIA LISTA PARTIDOS

1. El usuario selecciona una jornada de la lista de Jornadas
2. new()
3. Inicializar()
4. cargarPartidos(idLiga,idJornada):ListaPartidos
5. listaPartidos.php
6. execSQL("SELECT * FROM partidos WHERE idLiga=%idLiga% and idJornada=%idJornada%")
7. new()
8. next()
9. getString("local"):String
10. getString("visitante"):String
11. getInt("golesLocal"):int
12. getInt("golesVisitante")int
13. close()
14. Deserializar JSON

20. Clasificación

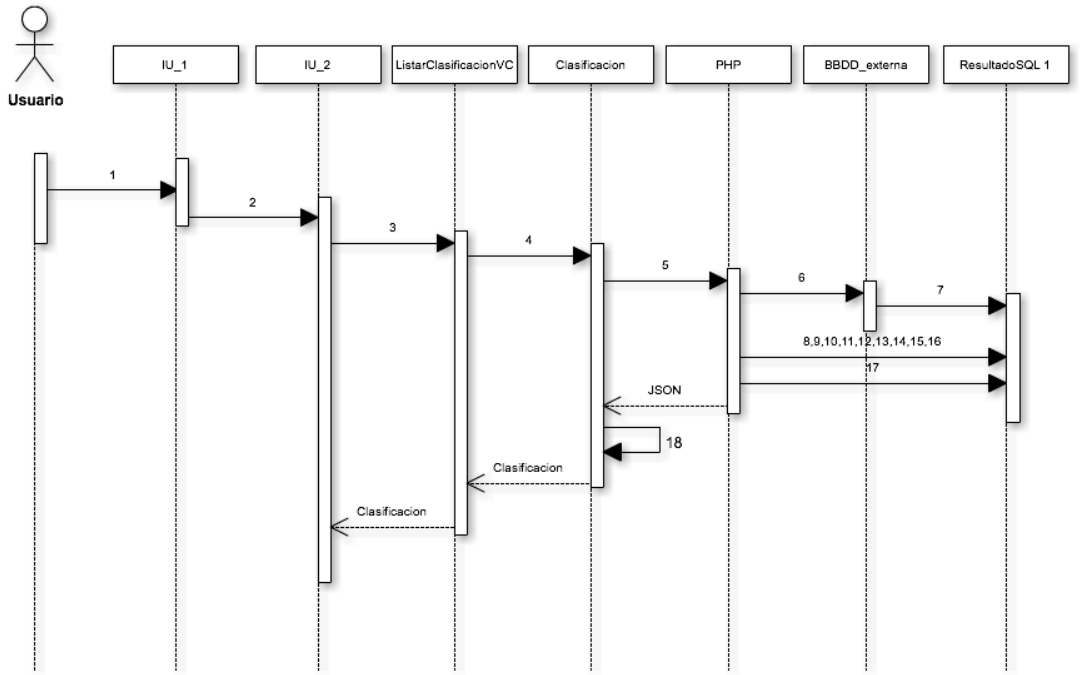


Ilustración 152: SECUENCIA CLASIFICACIÓN

1. El usuario pulsa en el botón “Clasificación” del Menú de liga
2. new()
3. Inicializar()
4. cargarClasificacion(idLiga):Clasificacion
5. listaClasificacion.php
6. execSQL(“SELECT * FROM clasificacion WHERE idLiga=%idLiga)
7. new()
8. next()
9. getString(“nombre”):String
10. getInt(“puntos”):int
11. getInt(“jugados”)int
12. getInt(“ganados”)int
13. getInt(“empatados”)int
14. getInt(“perdidos”)int
15. getInt(“favor”)int
16. getInt(“contra”)int
17. close()
18. Deserializar JSON

21. Crear liga

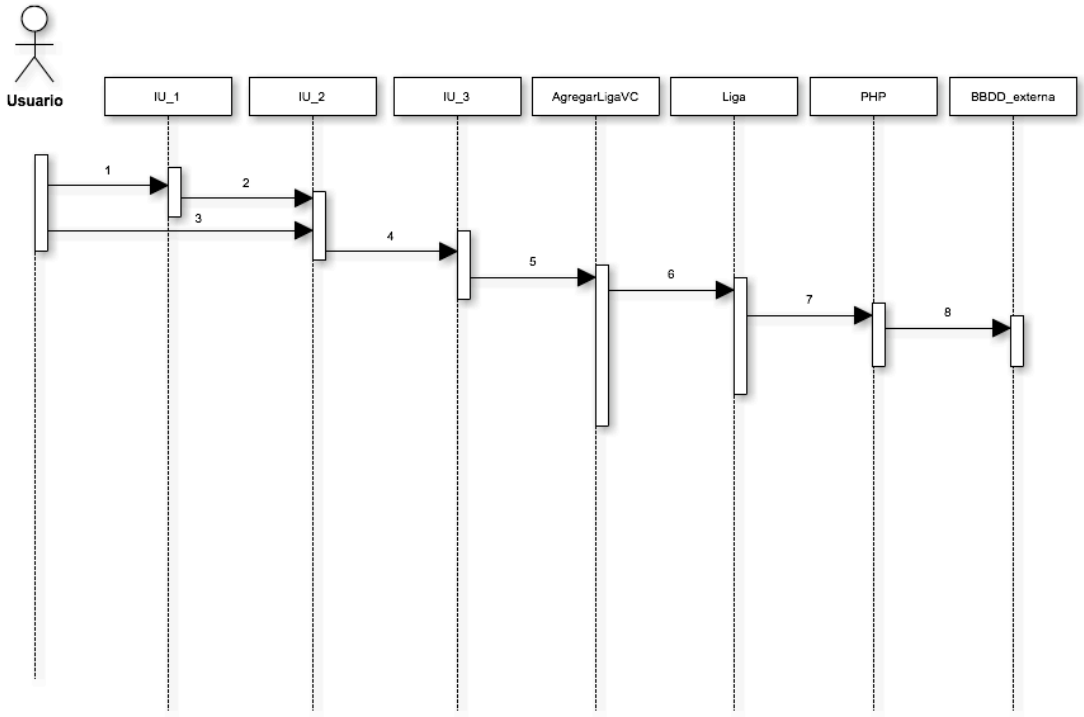


Ilustración 153: SECUENCIA CREAR LIGA

1. El usuario pulsa el icono “Ligas” del Menú principal
2. New()
3. El usuario pulsa el icono “Crear Liga”
4. New()
5. nuevaLiga(nombre,temporada,password,idUsuario)
6. nuevaLiga(nombre,temporada,password,idUsuario)
7. ligas.php
8. execSQL(“INSERT INTO ligas (nombre,temporada,password,idUsuario) VALUES (%nombre%,%temporada%,%password%,%idUsuario%))

22. Crear equipo liga

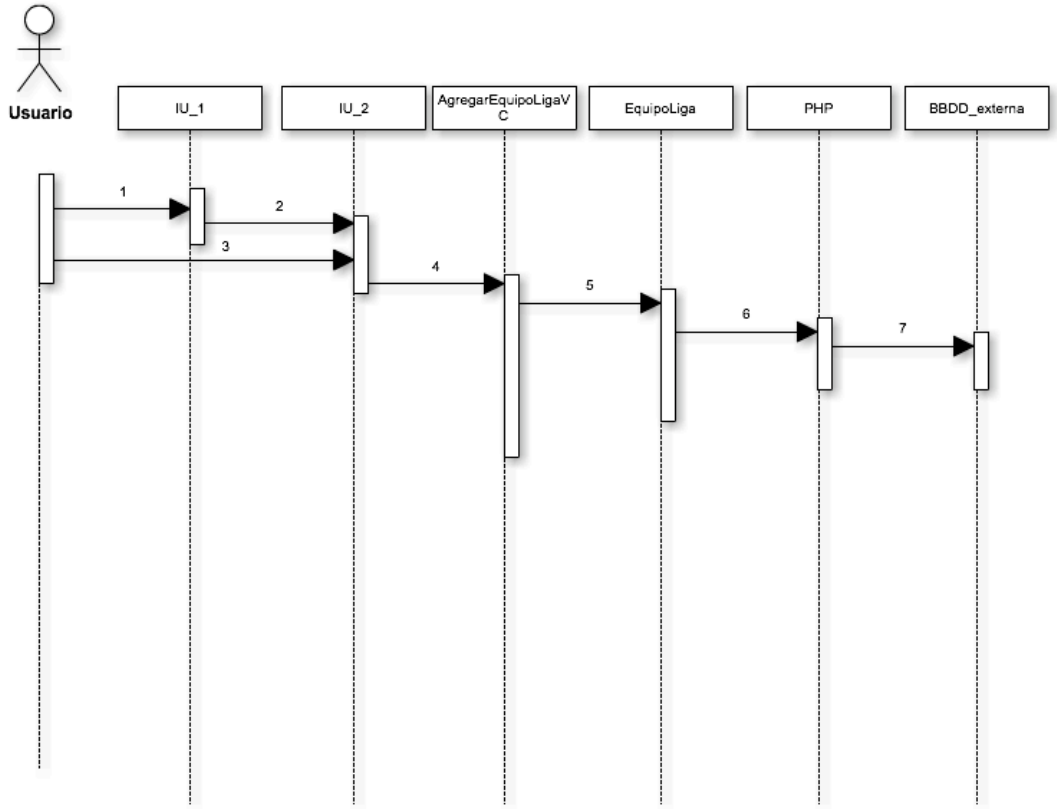


Ilustración 154: SECUENCIA CREAR EQUIPO LIGA

1. El usuario pulsa el botón “Nuevo equipo” del Menú de la liga
2. New()
3. El usuario introduce el nombre del nuevo equipo
4. agregarEquipoLiga(nombre, idLiga)
5. agregarEquipoLiga(nombre, idLiga)
6. equiposLiga.php
7. execSQL(“INSERT INTO equiposLiga (nombre,idLiga,puntos,jugados,ganados,empatados,perdidos,favor,contra)VALUES((%nombre%,%idLiga%,0,0,0,0,0,0,0)”)

23. Crear partido jornada

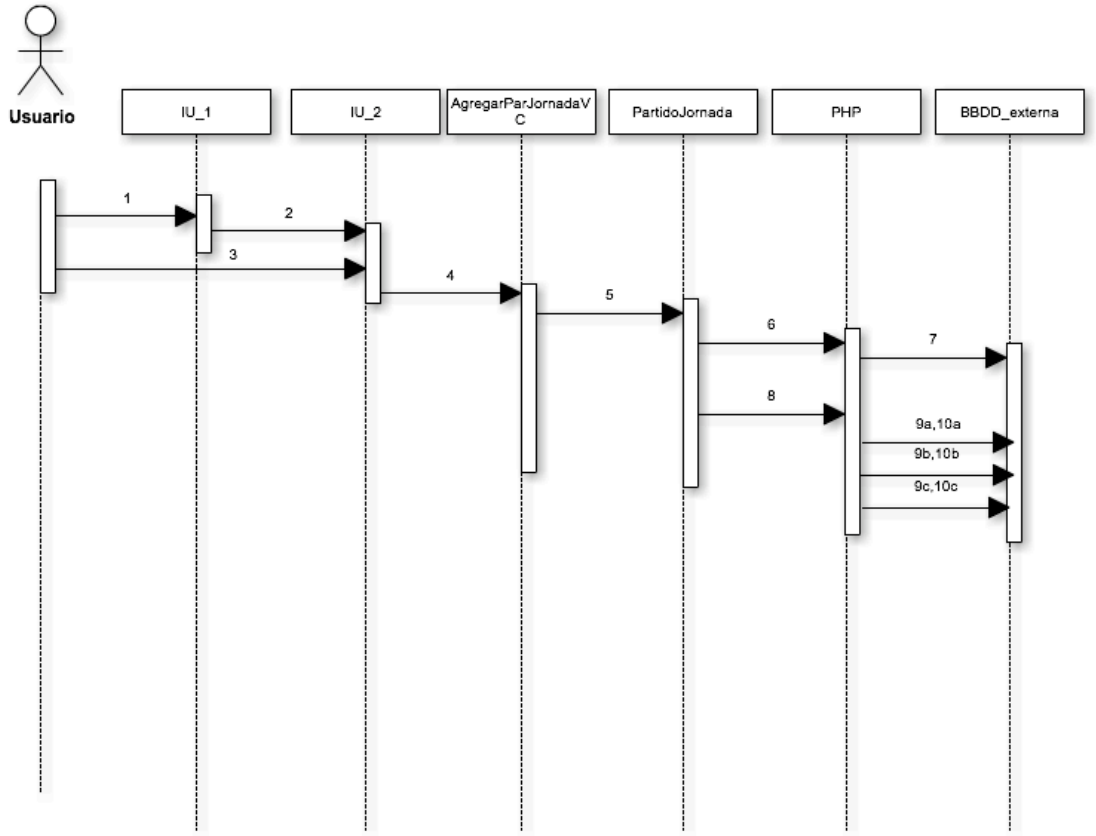


Ilustración 155: SECUENCIA CREAR PARTIDO JORNADA

1. El usuario pulsa el botón "+" situado en la parte superior derecha de la lista de partidos
2. New()
3. El usuario introduce los datos del partido y pulsa "Aceptar"
4. agregarPartido(equipoLocal,equipoVisitante,golesLocal,golesVisitante,idJornada,idLiga)
5. agregarPartido(equipoLocal,equipoVisitante,golesLocal,golesVisitante,idJornada,idLiga)
6. partidos.php
7. execSQL("INSERT INTO partidos (equipoLocal,equipoVisitante,golesLocal,golesVisitante,idJornada,idLiga)VALUES(%equipoLocal%,%equipoVisitante%,%golesLocal%,%golesVisitante%,%idJornada%,%idLiga%)")
8. actualizarClasificacion.php
   [si goles local > goles visitante]
   9a. execSQL("update equiposLiga SET jugados = jugados + 1,ganados=ganados + 1,puntos=puntos + 3,favor=favor + golesLocal,contra=contra + golesVisitante WHERE equipoLocal=%equipoLocal% and idLiga=%idLiga%")
   10a. execSQL("update equiposLiga SET jugados = jugados + 1,perdidos=perdidos + 1, favor=favor +

golesLocal,contra=contra + golesVisitante WHERE equipoVisitante=%equipoVisitante% and idLiga=%idLiga%")

[si goles local<goles visitante]

9b.execSQL("update equiposLiga SET jugados = jugados + 1,perdidos=perdidos + 1,favor=favor + golesLocal,contra=contra + golesVisitante WHERE equipoLocal=%equipoLocal% and idLiga=%idLiga%")

10b.execSQL("update equiposLiga SET jugados = jugados + 1,ganados=ganados + 1,puntos=puntos + 3,favor=favor + golesLocal,contra=contra + golesVisitante WHERE equipoVisitante=%equipoVisitante% and idLiga=%idLiga%")

[si goles local = goles visitante]

9c. execSQL("update equiposLiga SET jugados = jugados + 1,empatados=empatados + 1,puntos=puntos + 1,favor=favor + golesLocal,contra=contra + golesVisitante WHERE equipoLocal=%equipoLocal% and idLiga=%idLiga%")

10c. execSQL("update equiposLiga SET jugados = jugados + 1,empatados=empatados + 1,puntos=puntos + 1,favor=favor + golesLocal,contra=contra + golesVisitante WHERE equipoVisitante=%equipoVisitante% and idLiga=%idLiga%")

24. Lista ligas

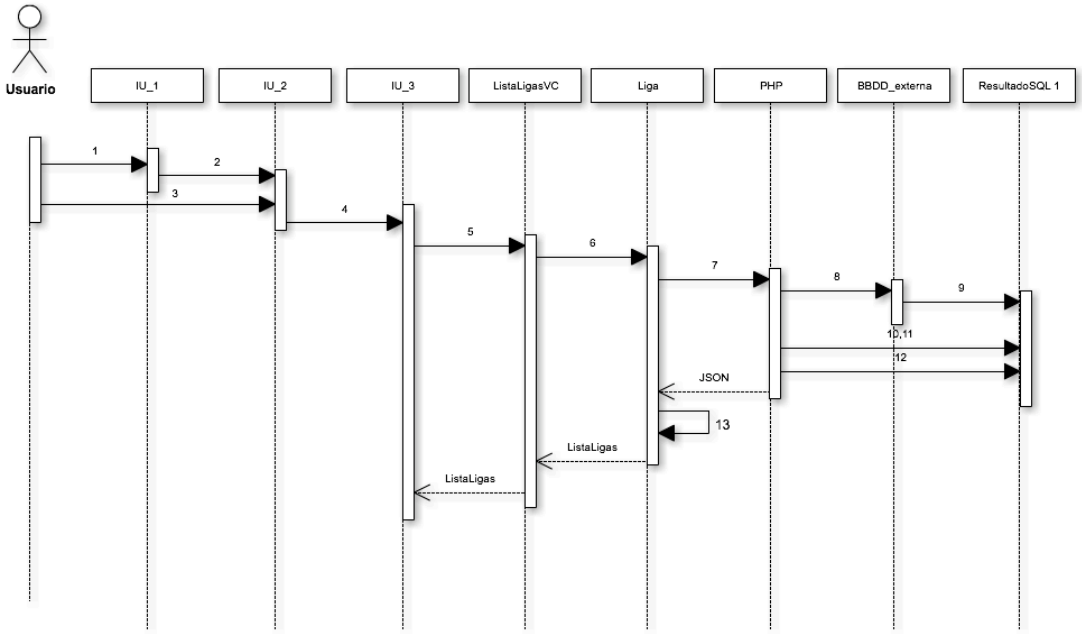


Ilustración 156: SECUENCIA LISTA LIGAS

1. El usuario pulsa en el icono “Ligas” del Menú principal
2. New()
3. El usuario pulsa el botón “Lista ligas” del Menú de la liga
4. new()
5. Inicializar()
6. cargarLigas():ListaLigas
7. listaLigas.php
8. execSQL(“SELECT * FROM ligas)
9. new()
10. next()
11. getString(“nombre”):String
12. close()
13. Deserializar JSON

25. Ver Perfil

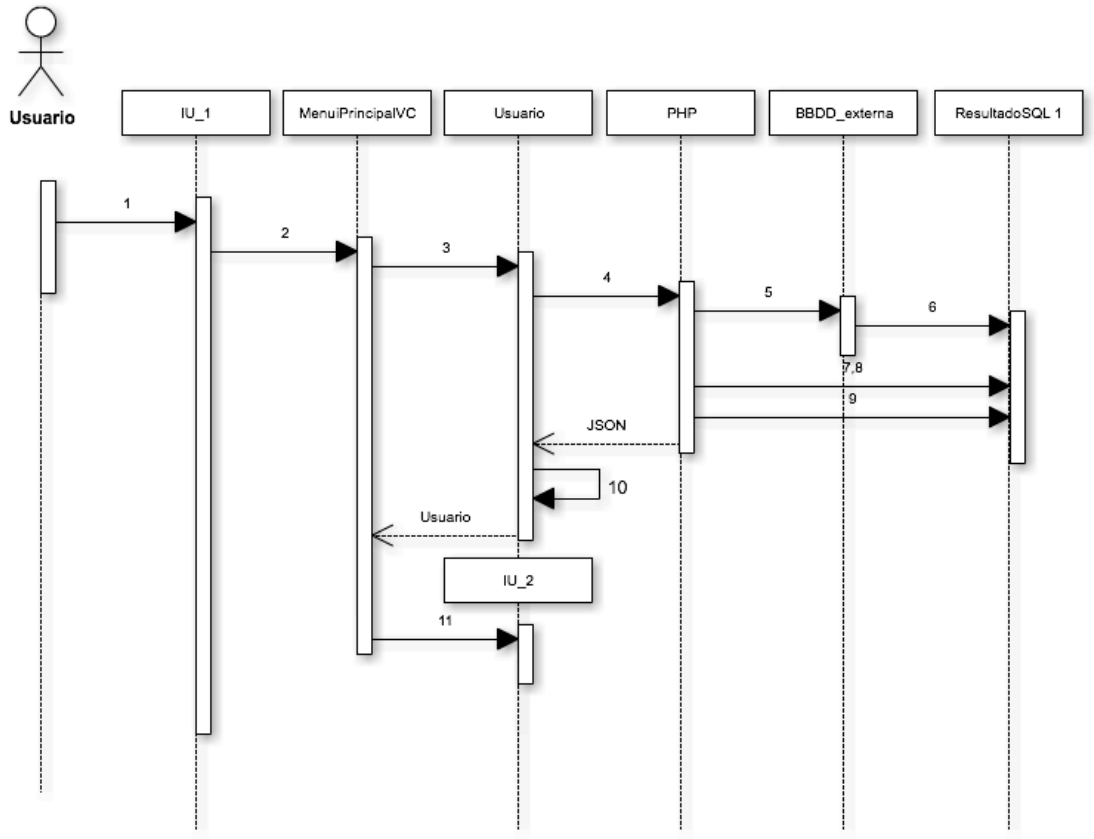


Ilustración 157: SECUENCIA VER PERFIL

1. El usuario pulsa el icono “Perfil” del Menú principal
2. obtenerDatosUsuario(nombreUsuario)
3. obtenerDatosUsuario(nombreUsuario)
4. usuario.php
5. execSQL(“SELECT * FROM usuarios WHERE nombreUsuario=%nombreUsuario%”)
6. new()
7. next()
8. getDatos(“Datos”):DatosUsuario
9. close()
10. Deserializar JSON
11. new(DatosUsuario)

26. Eliminar usuario

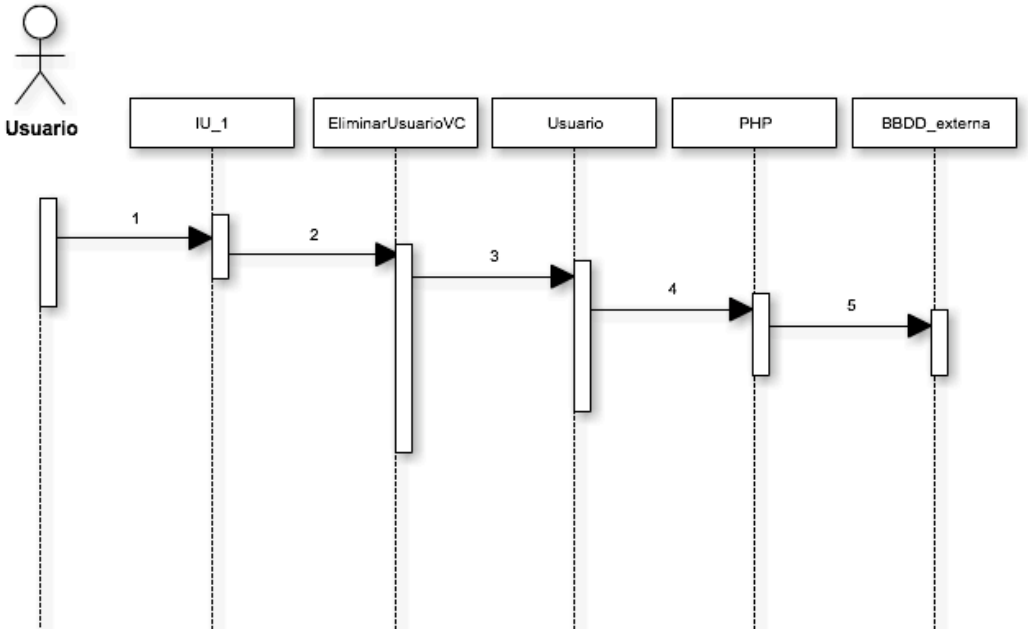


Ilustración 158: SECUENCIA ELIMINAR USUARIO

1. El usuario pulsa el botón eliminar usuario de la pantalla del perifil
2. eliminarUsuario(idUsuario)
3. eliminarUsuario(idUsuario)
4. eliminarUsuario.php
5. execSQL("DELETE FROM usuarios WHERE idUsuario=%idUsuario%)