LiveStats

05/06/2025

Carlos Molina Sánchez-Mateos
IES JUAN BOSCO

Visión general

El proyecto aquí expuesto tiene como finalidad el diseño y desarrollo de una aplicación web en la gestión de ligas deportivas de baloncesto, así como la realización de estadísticas en partidos. Esta herramienta nace de la necesidad de contar con un sistema centralizado, eficiente y accesible que permita registrar de forma precisa todas las acciones relevantes que ocurren durante un encuentro, como tiros anotados y fallados, rebotes, asistencias, faltas, tapones, entre otras.

La aplicación está dirigida principalmente a ligas locales, amateurs y semiprofesionales que buscan un medio digital para monitorizar el rendimiento de los jugadores y equipos. A través de una interfaz simple y adaptable a distintos dispositivos, se busca dar la posibilidad a estas personas o clientes objetivos de realizar de manera rápida y eficiente una gestión de sus ligas, facilitando la visualización estadísticas en pocos clics.

Desde el punto de vista técnico, el sistema se basa en una arquitectura cliente-servidor, utilizando tecnologías modernas tanto en el frontend como en el backend, con una base de datos optimizada para el almacenamiento y consulta eficiente de acciones en partidos. Además, se contempla la posibilidad de expandir sus funcionalidades hacia la generación de informes automáticos, notificaciones en vivo y comparativas entre jugadores o equipos a lo largo de una temporada.

Este proyecto no solo responde a una necesidad funcional concreta en el ámbito deportivo, sino que también sienta las bases para una futura evolución hacia soluciones más avanzadas, lo que lo convierte en una propuesta con potencial de crecimiento e impacto real en el entorno del baloncesto amateur y semiprofesional.

Objetivos

1. **Desarrollar una aplicación web intuitiva y accesible** que permita a los usuarios registrar en tiempo real las acciones relevantes de los partidos de baloncesto, tales como tiros, rebotes, asistencias, faltas etc....

2. **Proporcionar un sistema robusto para el almacenamiento y gestión de datos deportivos**, que garantice la integridad, precisión y disponibilidad de la información registrada durante los encuentros.

3. **Facilitar el acceso a estadísticas individuales y colectivas** mediante informes claros y visualizaciones gráficas, para apoyar la toma de decisiones de entrenadores, árbitros y demás responsables deportivos.

4. **Optimizar el proceso tradicional de registro manual** mediante una solución digital que reduzca errores y agilice el acceso a los datos, mejorando la eficiencia en la gestión de partidos.

5. **Diseñar una arquitectura escalable y modular** que permita futuras ampliaciones funcionales, como integración con sistemas de visualización en vivo, generación automática de resúmenes y comparativas históricas.

6. **Garantizar la compatibilidad y adaptabilidad del sistema en distintos dispositivos**, incluyendo ordenadores, tablets y móviles, para facilitar su uso en distintos contextos deportivos.

Justificación del proyecto

En el ámbito deportivo, y específicamente en el baloncesto, el uso de herramientas para gestión de competiciones y realización de estadísticas es una herramienta cada día más habitual y ampliamente extendida. Sin embargo, en muchas ligas amateurs y semiprofesionales, el registro de estas estadísticas no suele realizarse debido a la falta de una herramienta asequible .

La creación de una aplicación web especializada en la gestión de estadísticas de partidos de baloncesto responde a la necesidad de modernizar este proceso, facilitando el trabajo de los agentes involucrados. La digitalización de los registros permitirá no solo una mayor precisión en los datos, sino también un acceso inmediato a análisis detallados que contribuyan a la planificación y mejora continua de la gestión de sus competiciones.

Además, la existencia de una herramienta centralizada y fácil de usar fomenta la profesionalización de la gestión deportiva a niveles donde antes no era accesible debido a limitaciones tecnológicas o económicas. Esto representa un valor agregado para clubes y ligas que buscan potenciar sus competencias y ofrecer mejores condiciones a sus deportistas.

En resumen, este proyecto justifica su desarrollo por su capacidad para optimizar procesos, mejorar la calidad y rapidez del análisis estadístico y aportar un recurso tecnológico valioso para la comunidad del baloncesto en sus diversas categorías.

Identificación de necesidades y requerimientos

Para que la aplicación cumpla con su propósito, es esencial identificar tanto las necesidades funcionales como las no funcionales del sistema. Estas necesidades han sido detectadas tras analizar el entorno de uso de ligas y clubes de baloncesto a nivel amateur y semiprofesional.

Necesidades funcionales:

1. Registro de partidos y configuración de datos básicos: fecha, equipos participantes, estadista, etc.
2. Registro en tiempo real de acciones como:
    - Tiros anotados y fallados (de 2, 3 y libres)
    - Rebotes (ofensivos y defensivos)
    - Asistencias
    - Faltas (cometidas y recibidas)
    - Tapones realizados y recibidos
    - Expulsiones
3. Visualización de estadísticas por jugador, por equipo y por partido.
4. Gestión de usuarios con distintos roles (administrador, estadista, usuario).
5. Sistema de autenticación y seguridad para proteger los datos registrados.

Necesidades no funcionales:

1. Interfaz web responsiva, accesible desde ordenadores, tablets y dispositivos móviles.

2. Almacenamiento seguro de los datos en una base de datos estructurada.

3. Fluidez y actualización de datos sin recargar páginas como la de hacer estadísticas o gestión de datos.

4. Escalabilidad del sistema para permitir su crecimiento con nuevas funcionalidades.

5. Diseño modular que facilite el mantenimiento y la evolución del sistema.

6. Cumplimiento de buenas prácticas de usabilidad y accesibilidad.

Entorno tecnológico propuesto

Para el desarrollo del sistema, se plantea una arquitectura basada en tecnologías web modernas, orientada a facilitar la escalabilidad, la mantenibilidad del código y la flexibilidad en su despliegue. La elección de las herramientas responde tanto a criterios de eficiencia como de compatibilidad con entornos de desarrollo ampliamente utilizados en el ámbito profesional y académico.

Lenguajes y frameworks

- **Backend:** PHP con el framework Symfony, que permite estructurar el código de forma modular y mantener buenas prácticas de desarrollo.

- **Frontend:** HTML, CSS, JavaScript (utilizando Ajax y jQuery para la comunicación asincrónica y la manipulación del DOM), y Twig como motor de plantillas integrado en Symfony.

- **Base de datos:** MySQL, en su versión compatible con MariaDB, para la gestión estructurada de datos relacionales.

- **ORM:** Doctrine, como sistema de mapeo objeto-relacional, lo que facilita el acceso y manipulación de la base de datos desde el código PHP.

- **Control de versiones:** Git, con repositorio alojado en GitHub para el seguimiento de cambios y trabajo colaborativo.

Herramientas y entornos de desarrollo

- **IDE:** PHPStorm como entorno principal para el desarrollo backend, y Visual Studio Code para la edición y maquetación del frontend.

- **Gestión de dependencias:** Composer, herramienta estándar para la instalación y actualización de bibliotecas en proyectos PHP.

- **Gestión de base de datos:** phpMyAdmin, como herramienta visual para la administración y consulta de la base de datos.

- **Entorno de ejecución:** XAMPP, que proporciona un stack completo (Apache, MySQL, PHP) para pruebas y desarrollo local.

Otros elementos

- **Docker:** Para contenerizar la aplicación y facilitar su despliegue en diferentes entornos con configuración consistente.

- **Despliegue:** Se contempla la utilización de un servidor virtual mediante Google Compute Engine, lo cual permitirá hospedar la aplicación y la base de datos en un entorno accesible y escalable en la nube.

Identificación del público

La aplicación está dirigida a diversos perfiles dentro del ámbito del baloncesto, especialmente en contextos amateurs, semi profesionales y educativos. A continuación se detallan los principales grupos de usuarios a los que va orientada la herramienta:

1. **Organizadores de torneos y ligas:**
   Requieren funcionalidades para gestionar múltiples equipos, partidos y categorías. Buscan una solución que les permita centralizar la información de forma organizada y mantener un historial completo de los encuentros.

2. **Jugadores:**
   Aunque con un perfil de uso más consultivo, pueden acceder a sus propias estadísticas y evolución a lo largo de la temporada, lo cual puede resultar motivador y útil para su desarrollo deportivo.

3. **Academias deportivas:**
   Instituciones que promueven el baloncesto como actividad formativa y competitiva. La aplicación puede ser utilizada como recurso didáctico y herramienta de seguimiento del progreso de sus integrantes.

Cada uno de estos públicos tiene necesidades específicas que han sido consideradas durante la fase de diseño del sistema, con el objetivo de ofrecer una experiencia de usuario adaptada, efectiva y coherente con el rol que desempeñan en el entorno deportivo.

Estudio de mercado y competencia

Actualmente, el mercado de las aplicaciones deportivas enfocadas en la recopilación de estadísticas de baloncesto presenta varias soluciones disponibles, especialmente dirigidas a clubes profesionales o federaciones. Sin embargo, muchas de estas herramientas están orientadas a usuarios con alto presupuesto o requieren dispositivos específicos, lo que limita su adopción en ligas amateurs, centros educativos o clubes semiprofesionales.

Análisis de herramientas existentes:

1. **StatCrew / Genius Sports / Hudl Assist**
   Estas plataformas son ampliamente utilizadas en competiciones de alto nivel. Ofrecen potentes motores de análisis y generación de vídeos asociados a las estadísticas. No obstante, presentan costes elevados, una curva de aprendizaje pronunciada y una dependencia de recursos tecnológicos avanzados.

2. **Apps móviles como "Basketball Stats Keeper", "My Basketball Stats" o "Scorebook Live":**
   Están más enfocadas en el uso individual o en partidos informales. Aunque ofrecen funcionalidades básicas de registro, suelen carecer de capacidad multiusuario, control de roles, almacenamiento en la nube o acceso simultáneo desde diferentes dispositivos.

Oportunidades detectadas:

- Existe una carencia evidente de soluciones intermedias que combinen potencia funcional, facilidad de uso y bajo coste, especialmente diseñadas para:
    - Ligas locales y regionales.
    - Colegios y academias deportivas.
    - Clubes con estructura semiprofesional o amateur.

Ventaja competitiva del proyecto:

Este proyecto busca posicionarse como una solución accesible, flexible y adaptada a las verdaderas necesidades de quienes no cuentan con grandes presupuestos ni infraestructura técnica avanzada, pero desean llevar un control preciso de las estadísticas. Al enfocarse en la simplicidad de uso, escalabilidad y enfoque colaborativo entre usuarios (árbitros, entrenadores, jugadores), la aplicación se sitúa en un nicho con gran demanda no completamente cubierta por la competencia actual.

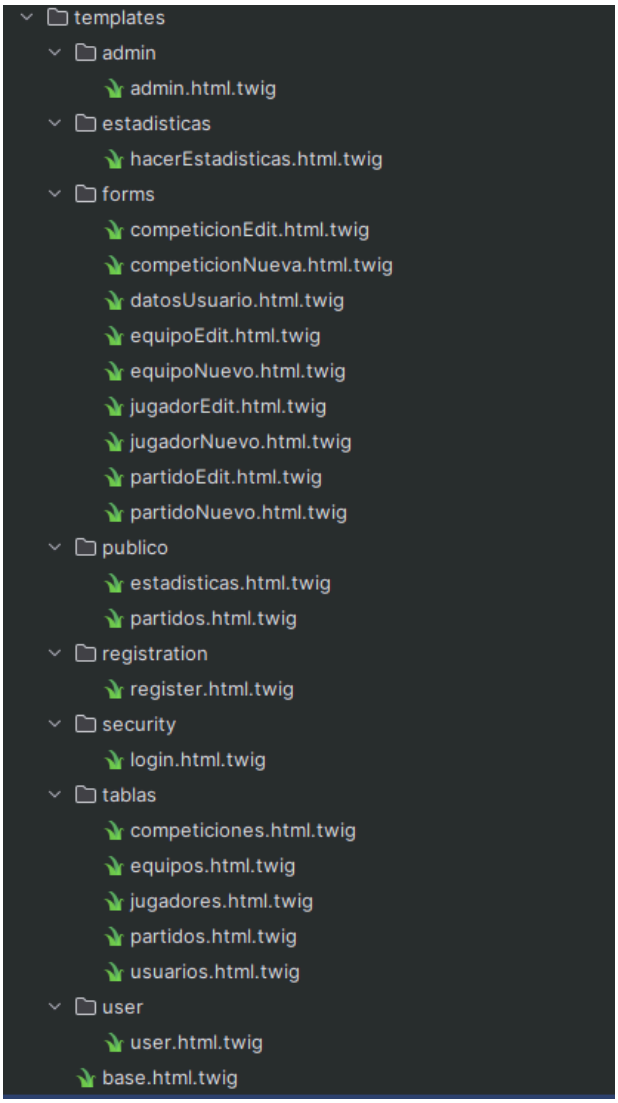
Diseño y planificación

a. Definición de la arquitectura del proyecto

El proyecto se basa en una arquitectura cliente-servidor siguiendo el modelo clásico de aplicaciones web. El sistema se divide en tres capas principales:

1. **Capa de presentación (frontend):**
   Implementada mediante tecnologías web estándar (HTML, CSS, Bootstrap), y utilizando **Twig** como motor de plantillas integrado en Symfony. Esta capa es responsable de la interacción con el usuario final, mostrando datos y capturando entradas.

   a. **Twig:** La estructura del sistema de plantillas,permite una estructura jerárquica de plantillas, en la que se define una plantilla base (layout general) y se reutilizan componentes comunes (cabecera, pie de página, menús, etc.) en todas las vistas.



b. **Paleta de colores:** debido a la temática de baloncesto se busco una paleta de colores reconocible y relacionada con este sector, usando los colores, #ea7b40 #2c2c2c como colores “principales” y una segunda paleta para fondos, y detalles en formularios, etc...

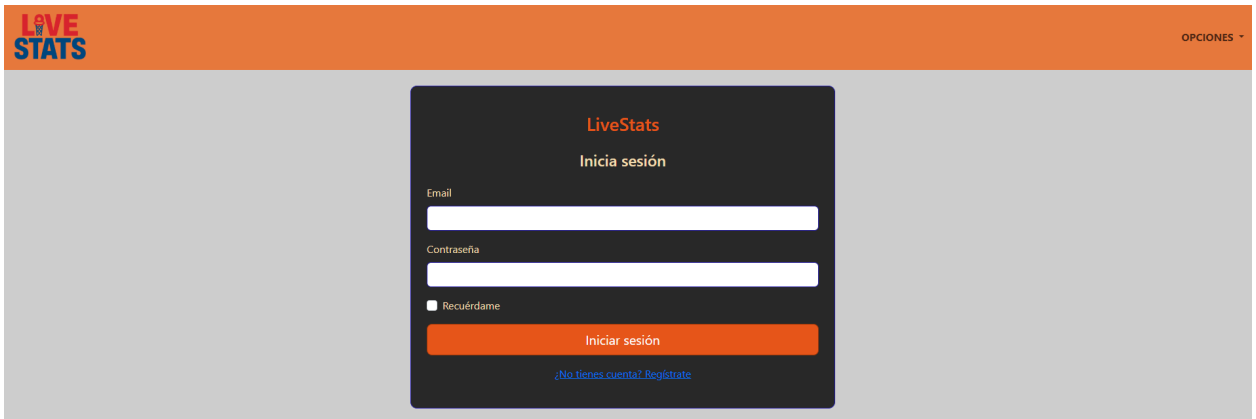
Color	Hex	RGB
	#ea7b40	(234,123,64)
	#2c2c2c	(44,44,44)

Color	Hex	RGB
	#e75719	**(231,87,25)**
	#9c380d	**(156,56,13)**
	#241aa5	**(36,26,165)**
	#08052f	**(8,5,47)**
	#d0d0d0	**(208,208,208)**

c. **Bootstrap:** Además, se hace uso del framework CSS Bootstrap, que aporta componentes listos para usar como botones, formularios, alertas, iconografía (via Bootstrap Icons), y rejillas flexibles para organizar el contenido. facilitando además una interfaz es responsiva desde el inicio, y aprovechando sus clases para dar colores a la interfaz de realización de estadísticas, haciendo que sea más intuitivo y amigable

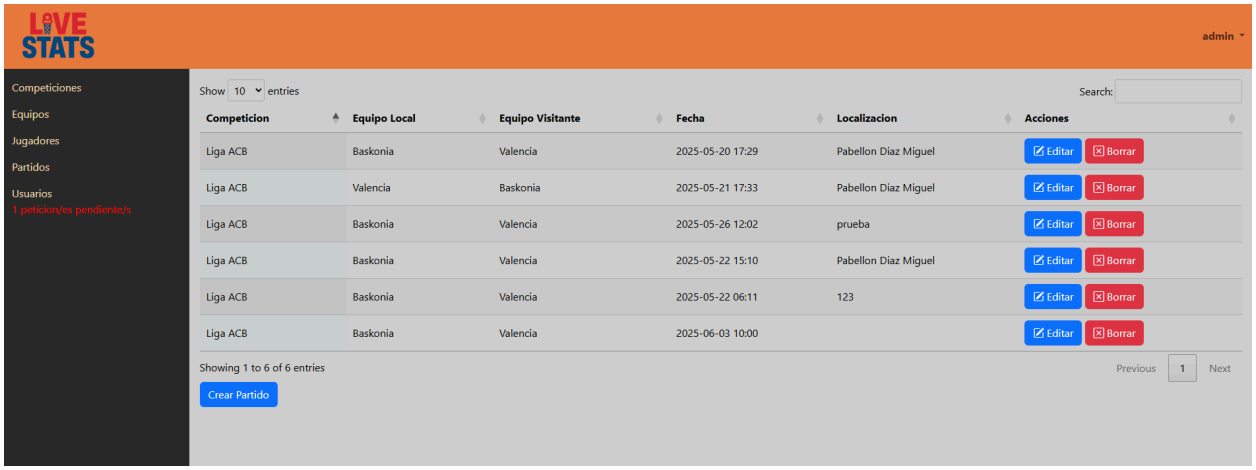
d. **Diseño de la interfaz de usuario:**

Pagina inicio: Empieza con el login, con un barra de navegación común a casi todas las vistas, optando por un estilo simple e intuitivo, con una card para el formulario.

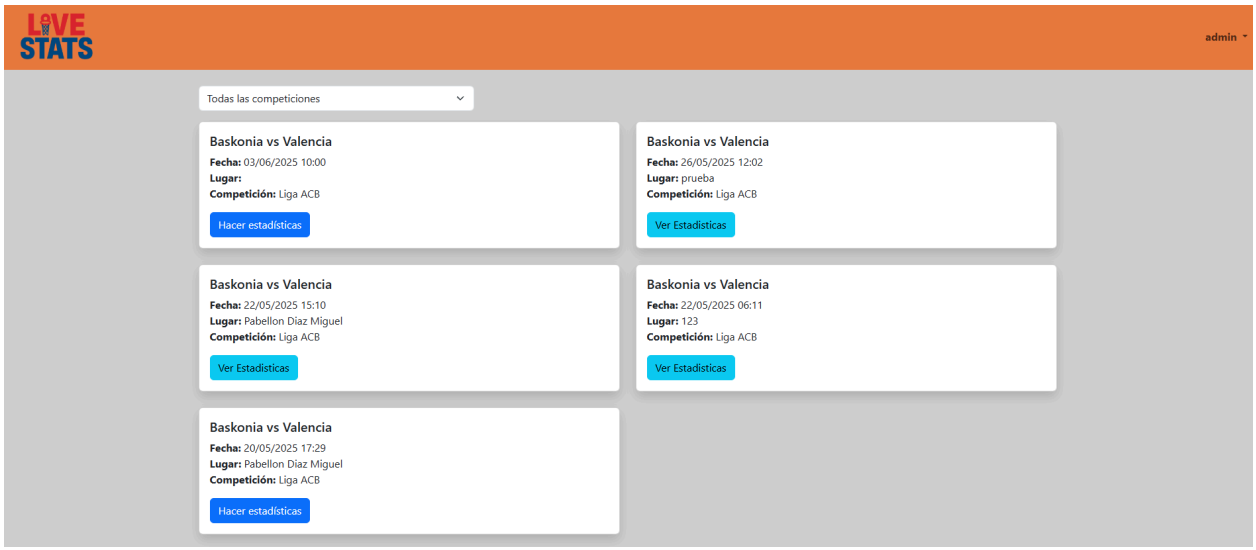


Página de registro: mantiene la misma idea que la página de login

Página de gestión publica y privada: Añade una barra lateral donde carga dinámicamente las tablas para poder gestionar los datos



Página de partidos: Pagina con cards de los partidos, que muestra botones dependiendo de si se han terminado o no, o si tienes permisos o no para hacer estadísticas, con un filtro de competicion



Vista de estadísticas: Vista sencilla para mostrar las estadísticas del partido, con 1 tabla para cada equipo

Estadísticas del Partido

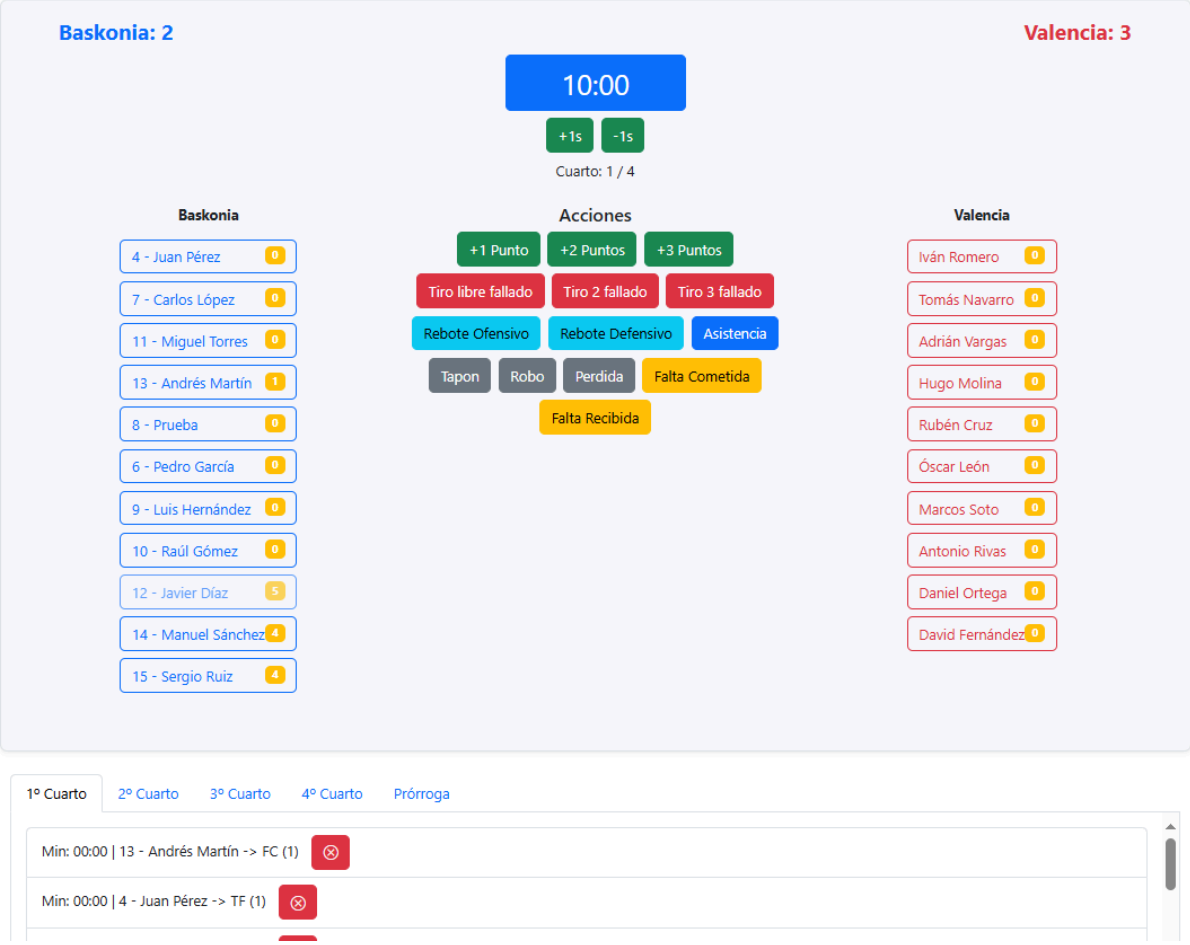
Baskonia 22

Valencia 44

Estadísticas Baskonia

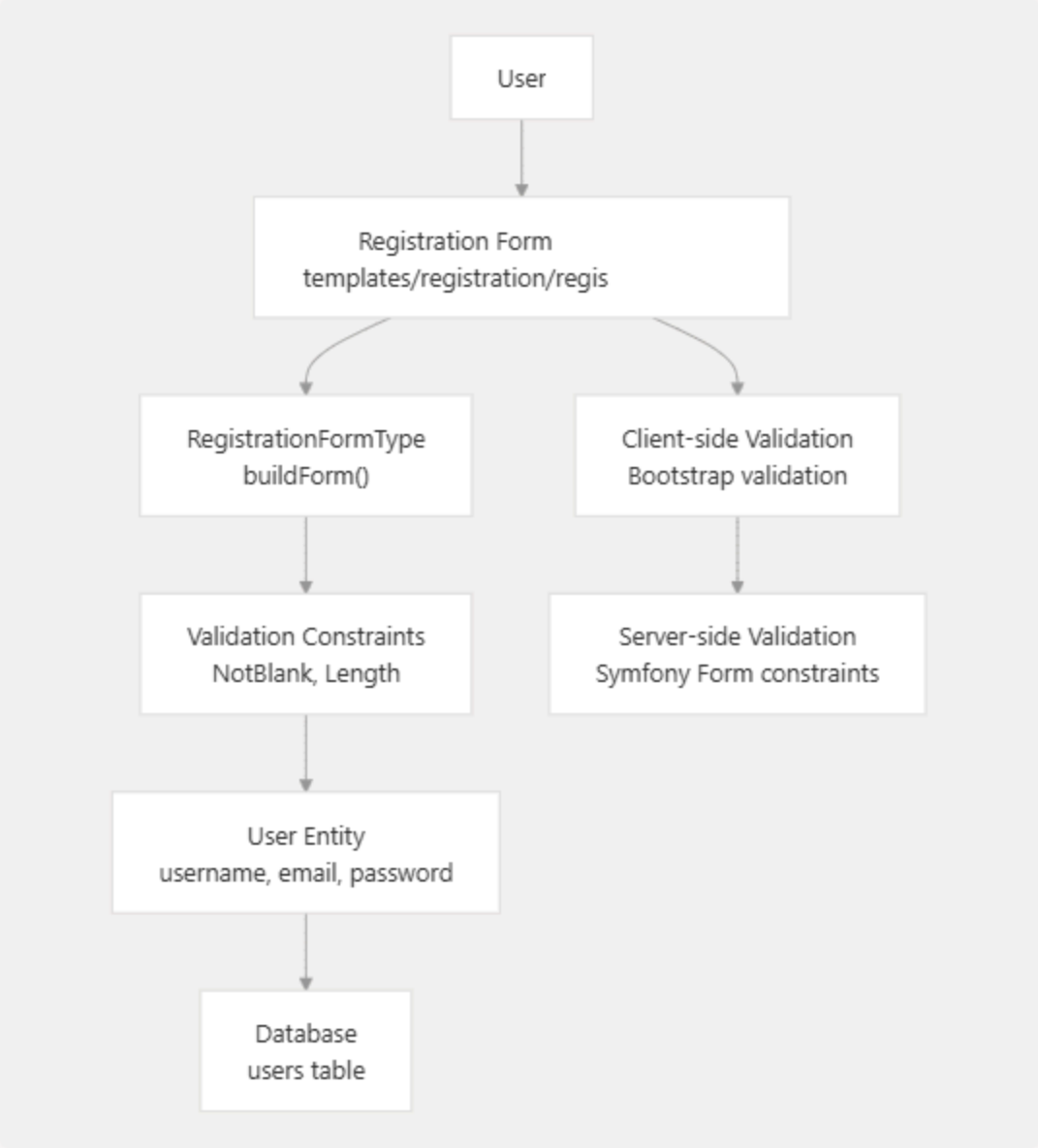
Dorsal	Jugador	PTS	2P	%2	3P	%3	TL	%1	REB OF	REB DEF	AST	TAP	ROB	PER	F.COM	F.REC
4	Juan Pérez	5	2/2	100.0%	0/0	0%	1/1	100.0%	0	0	0	0	0	0	0	0
7	Carlos López	8	1/1	100.0%	2/2	100.0%	0/0	0%	1	0	0	0	0	0	0	0
11	Miguel Torres	3	0/0	0%	1/1	100.0%	0/0	0%	0	0	0	0	0	0	0	0
13	Andrés Martín	0	0/0	0%	0/0	0%	0/0	0%	1	0	0	0	0	0	0	0
8	Prueba	0	0/0	0%	0/0	0%	0/1	0.0%	1	0	0	0	0	0	8	0
6	Pedro García	0	0/0	0%	0/0	0%	0/0	0%	1	0	0	3	0	0	4	0
9	Luis Hernández	0	0/0	0%	0/0	0%	0/0	0%	0	0	0	0	0	0	5	0
10	Raúl Gómez	0	0/0	0%	0/0	0%	0/0	0%	0	0	0	0	0	0	0	0
12	Javier Díaz	0	0/0	0%	0/0	0%	0/0	0%	0	0	0	0	0	0	5	0
14	Manuel Sánchez	0	0/0	0%	0/0	0%	0/0	0%	0	0	0	0	0	0	7	0
15	Sergio Ruiz	0	0/0	0%	0/0	0%	0/0	0%	0	0	0	0	0	0	5	0
	Totales	16	3/3	100.0%	3/3	100.0%	1/2	50.0%	4	0	0	3	0	0	34	0

Página de realización de estadísticas: vista que muestra la lista de jugadores de ambos, equipos, resultados, faltas de jugadores, con la zona central para gestionar las acciones del partido, con una zona inferior que va cargando dinámicamente las acciones de los partidos.

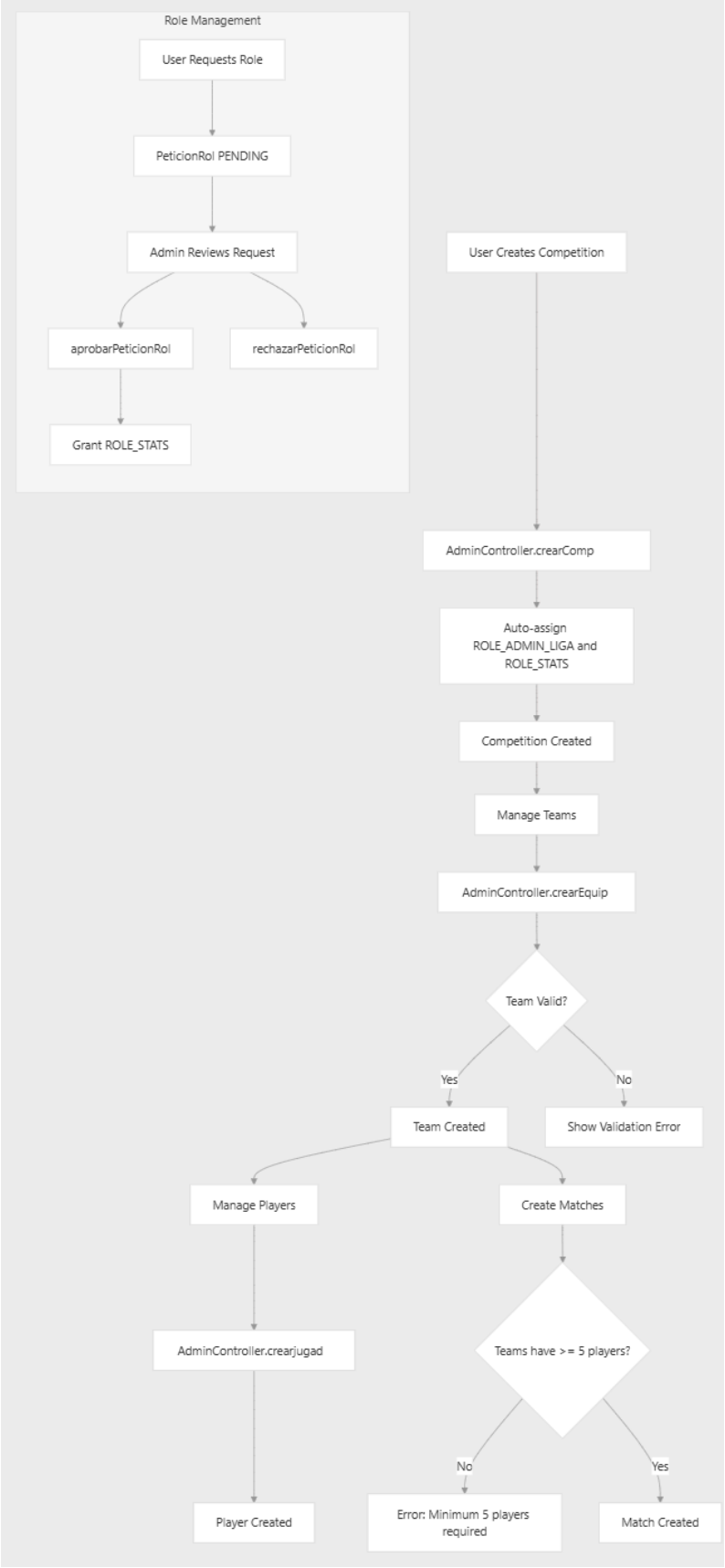


2. **Capa de lógica de negocio (backend):**
   Desarrollada en **PHP con Symfony**, esta capa gestiona la lógica de la aplicación, el control de acceso a usuarios, el procesamiento de estadísticas y la validación de datos. Se estructura siguiendo el patrón MVC (Modelo-Vista-Controlador), lo que facilita la modularidad del sistema.
   a. **Registro**:

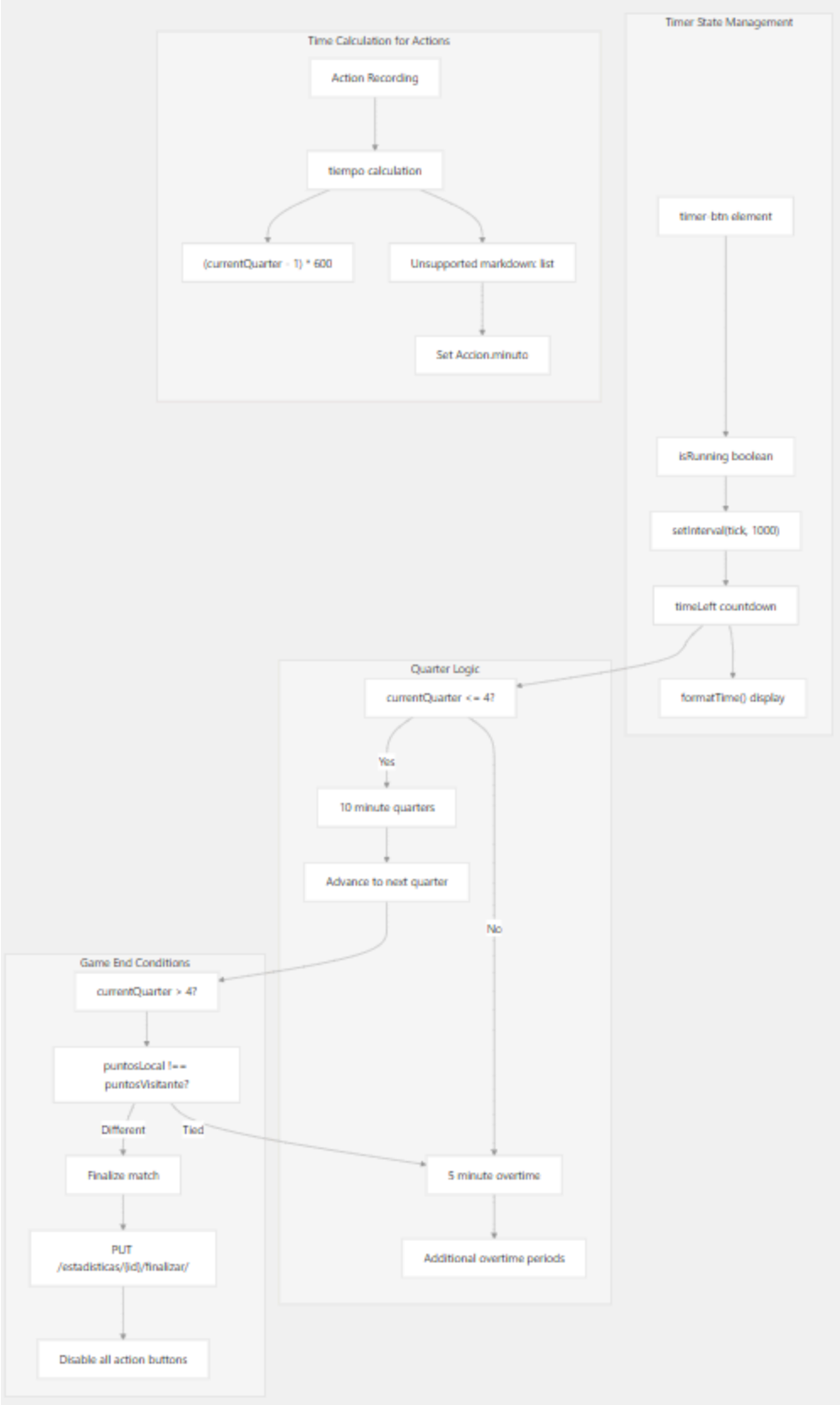
- **Nombre de usuario** : mínimo 3 caracteres, máximo 180 caracteres
- **Correo electrónico** : Validación de correo electrónico estándar
- **Contraseña** : Mínimo 6 caracteres, máximo 4096 caracteres



b. Panel de Admin



c. Lógica de partido



La implementación del sistema se realizó de manera progresiva, siguiendo la planificación establecida. Se comenzó por las funcionalidades esenciales, como el sistema de autenticación y gestión de usuarios, y se avanzó hacia módulos más complejos como la gestión de partidos, registro de estadísticas y visualización de datos.

Las funcionalidades clave desarrolladas fueron:

- Autenticación y registro de usuarios, con control de sesiones y validación de credenciales.
- Panel de administración, accesible tras el login, desde donde se pueden gestionar equipos, jugadores y partidos.
- Registro de eventos en tiempo real, como puntos, rebotes, faltas y demás estadísticas asociadas a los jugadores durante los partidos.
- Visualización pública de partidos y estadísticas, con vistas dinámicas, estructuradas y filtrables.
- Distinción de roles y accesos, garantizando que solo los usuarios autorizados puedan realizar ciertas acciones.

Cada uno de estos módulos se integró con el sistema de base de datos, aplicando relaciones lógicas adecuadas y utilizando Doctrine como ORM para facilitar las operaciones CRUD.

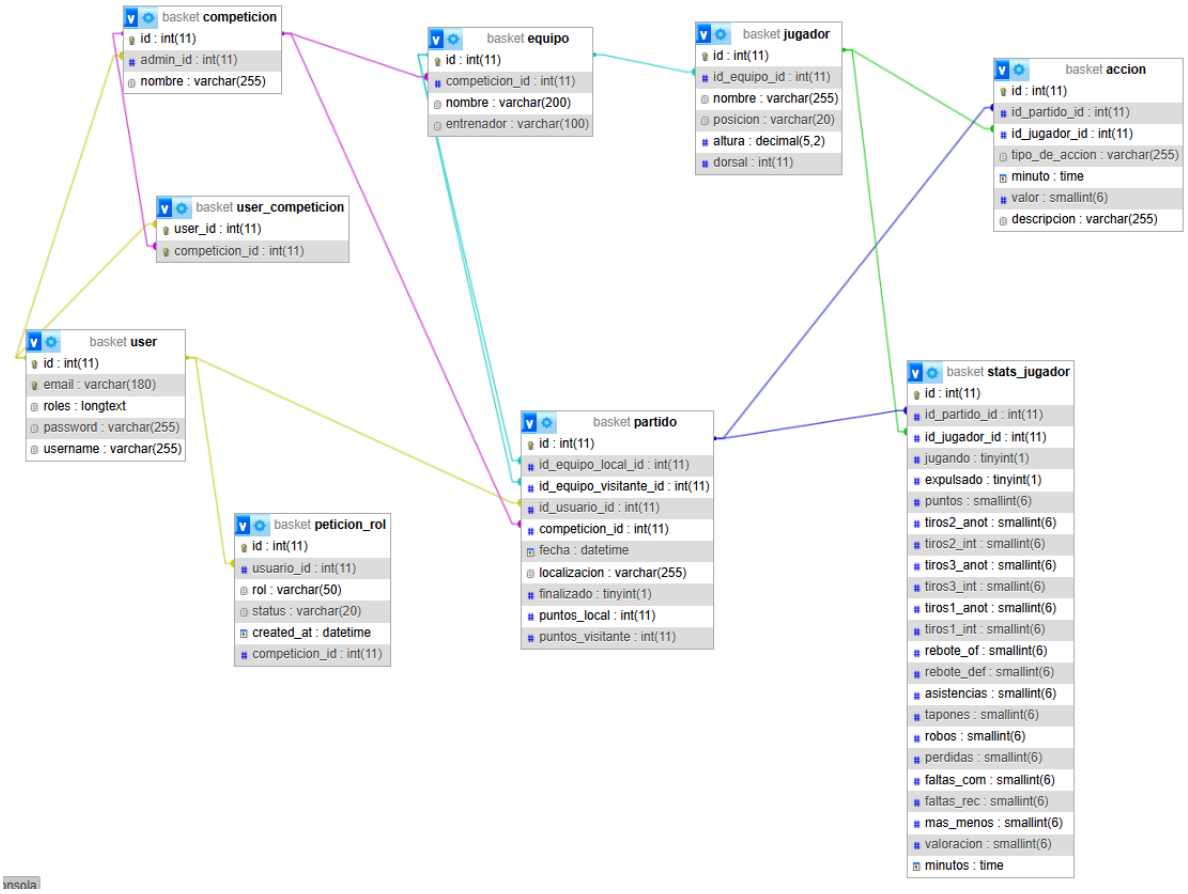
Corrección de errores y optimización del rendimiento

Durante el proceso de implementación, se realizaron ajustes continuos para solucionar errores detectados en las pruebas manuales. Además, se aplicaron algunas medidas básicas de optimización como:

- Reducción de consultas innecesarias a la base de datos.
- Uso adecuado de AJAX para evitar recargas completas de página, mejorando la experiencia de usuario.
- Optimización del uso de recursos estáticos (CSS, JS) para mejorar tiempos de carga.
- Simplificación de estructuras condicionales y repetitivas en los controladores y vistas.

Gracias a este enfoque iterativo, el sistema se ha mantenido funcional y coherente a lo largo del desarrollo, permitiendo una entrega estable y ajustada a los objetivos planteados.

3. **Capa de persistencia (base de datos):**
   Utiliza **MySQL/MariaDB** y está gestionada mediante **Doctrine**, el ORM de Symfony, que permite la abstracción de las operaciones con la base de datos. En ella se almacenan los datos relacionados con usuarios, partidos, estadísticas, equipos, jugadores, etc.



Relaciones:

1. `competicion (admin_id) => user (id)`: una competición tiene un administrador (user)

2. `equipo (competition_id) => competicion (id)`: un equipo pertenece a una competición

3. `jugador (id_equipo_id) => equipo (id)`: un jugador pertenece a un equipo

4. `partido (id_equipo_local_id) => equipo (id)`: el equipo local de un partido es un equipo

5. `partido (id_equipo_visitante_id) => equipo (id)`: el equipo visitante de un partido es un equipo

6. `partido (competicion_id) => competicion (id)`: un partido pertenece a una competición

7. `partido (id_usuario_id) => user (id)`: un partido fue creado o gestionado por un usuario

8. `accion (id_partido_id) => partido (id)`: una acción ocurrió en un partido

9. `accion (id_jugador_id) => jugador (id)`: una acción fue realizada por un jugador

10. `stats_jugador (id_partido_id) => partido (id)`: las estadísticas pertenecen a un partido

11. `stats_jugador (id_jugador_id) => jugador (id)`: las estadísticas pertenecen a un jugador

12. `user_competicion (user_id) => user (id)`: un usuario participa en una competición

13. `user_competicion (competicion_id) => competicion (id)`: una competición tiene usuarios asignados

14. `peticion_rol (usuario_id) => user (id)`: una petición de rol fue hecha por un usuario

15. `peticion_rol (competicion_id) => competicion (id)`: una petición de rol está asociada a una competición

Para la creación de las estadísticas de manera automática realicé **5 triggers de BBDD** para la creación automática de los campos de estadísticas para un partido, así también para la creación, edición y borrado de acciones de estos

Para crear campos de estadísticas del partido

```
BEGIN
    -- Insertar jugadores del equipo local
    INSERT INTO stats_jugador (
      id_partido_id, id_jugador_id, jugando, expulsado, puntos, tiros2_anot, tiros2_int,
      tiros3_anot, tiros3_int, tiros1_anot, tiros1_int,
      rebote_of, rebote_def, asistencias, tapones, robos, perdidas,
      faltas_com, faltas_rec, mas_menos, valoracion, minutos
    )
    SELECT
      NEW.id, j.id, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, '00:00:00'
    FROM jugador j
    WHERE j.id_equipo_id = NEW.id_equipo_local_id;

    -- Insertar jugadores del equipo visitante
    INSERT INTO stats_jugador (
      id_partido_id, id_jugador_id, jugando, expulsado, puntos, tiros2_anot, tiros2_int,
      tiros3_anot, tiros3_int, tiros1_anot, tiros1_int,
      rebote_of, rebote_def, asistencias, tapones, robos, perdidas,
      faltas_com, faltas_rec, mas_menos, valoracion, minutos
    )
    SELECT
      NEW.id, j.id, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, '00:00:00'
    FROM jugador j
    WHERE j.id_equipo_id = NEW.id_equipo_visitante_id;
  END
```

TRAS INSERTAR ACCIÓN (PARA BORRAR Y EDITAR SON PRACTICAMENTE IGUALES)

```
BEGIN
    -- Actualizar estadísticas de tiros anotados y fallados
    IF NEW.tipo_de_accion = 'TA' THEN
      -- Actualizar puntos y tiros según el valor del tiro
      IF NEW.valor = 2 THEN
        UPDATE stats_jugador
        SET puntos = puntos + 2,
          tiros2_anot = tiros2_anot + 1,
          tiros2_int = tiros2_int + 1
        WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
      ELSEIF NEW.valor = 3 THEN
        UPDATE stats_jugador
        SET puntos = puntos + 3,
          tiros3_anot = tiros3_anot + 1,
          tiros3_int = tiros3_int + 1
        WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
      ELSEIF NEW.valor = 1 THEN
        UPDATE stats_jugador
        SET puntos = puntos + 1,
          tiros1_anot = tiros1_anot + 1,
          tiros1_int = tiros1_int + 1
        WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
      END IF;
    ELSEIF NEW.tipo_de_accion = 'TF' THEN
      -- Actualizar tiros intentados según el valor del tiro fallado
      IF NEW.valor = 2 THEN
        UPDATE stats_jugador
        SET tiros2_int = tiros2_int + 1
        WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
      ELSEIF NEW.valor = 3 THEN
        UPDATE stats_jugador
        SET tiros3_int = tiros3_int + 1
        WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
      ELSEIF NEW.valor = 1 THEN
        UPDATE stats_jugador
```

```
        SET tiros1_int = tiros1_int + 1
        WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
    END IF;
END IF;

-- Actualizar rebotes
IF NEW.tipo_de_accion = 'RO' THEN
    UPDATE stats_jugador
    SET rebote_of = rebote_of + 1
    WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
ELSEIF NEW.tipo_de_accion = 'RD' THEN
    UPDATE stats_jugador
    SET rebote_def = rebote_def + 1
    WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
END IF;

-- Actualizar asistencias
IF NEW.tipo_de_accion = 'AS' THEN
    UPDATE stats_jugador
    SET asistencias = asistencias + 1
    WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
END IF;

-- Actualizar tapones
IF NEW.tipo_de_accion = 'BL' THEN
    UPDATE stats_jugador
    SET tapones = tapones + 1
    WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
END IF;

-- Actualizar pérdidas
IF NEW.tipo_de_accion = 'TO' THEN
    UPDATE stats_jugador
    SET perdidas = perdidas + 1
    WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
END IF;
```

```
    -- Actualizar robos
    IF NEW.tipo_de_accion = 'ST' THEN
      UPDATE stats_jugador
      SET robos = robos + 1
      WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
    END IF;

    -- Actualizar faltas
    IF NEW.tipo_de_accion = 'FC' THEN
      UPDATE stats_jugador
      SET faltas_com = faltas_com + 1
      WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;

      -- Expulsar al jugador si alcanza 5 faltas
      IF (SELECT faltas_com FROM stats_jugador
        WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id) = 5
THEN
        UPDATE stats_jugador
        SET expulsado = 1
        WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
      END IF;
    ELSEIF NEW.tipo_de_accion = 'FR' THEN
      UPDATE stats_jugador
      SET faltas_rec = faltas_rec + 1
      WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
    END IF;

    -- Actualizar expulsión directa
    IF NEW.tipo_de_accion = 'EX' THEN
      UPDATE stats_jugador
      SET expulsado = 1
      WHERE id_partido_id = NEW.id_partido_id AND id_jugador_id = NEW.id_jugador_id;
    END IF;
  END
```

Planificación de las tareas y los recursos necesarios

La planificación del desarrollo se ha basado en un **orden lógico y funcional propio**, priorizando los módulos fundamentales para asegurar un flujo de trabajo continuo y efectivo. El proyecto ha sido abordado de forma progresiva, permitiendo construir sobre funcionalidades ya completadas y testadas, lo que me permitió darme cuenta de carencias o necesidades a cubrir conforme iba avanzando el desarrollo.

La planificación general ha seguido las siguientes etapas:

1. **Diseño e implementación del sistema de autenticación**
   Se comenzó por crear el sistema de **login y registro de usuarios**, elemento clave para diferenciar accesos entre la zona pública y la privada del sistema. Este módulo incluyó la validación de datos, gestión de sesiones y control de accesos, roles, etc....

2. **Desarrollo de la interfaz de gestión (zona privada)**
   Una vez implementado el sistema de acceso, se procedió a diseñar y construir la **interfaz de gestión privada**, que permite a los usuarios autorizados gestionar competiciones, equipos, jugadores, partidos, etc...

3. **Diseño de la parte pública del sistema**
   Posteriormente, se desarrolló la **zona pública**, accesible sin autenticación, donde se muestran los partidos disponibles, sus resultados y estadísticas asociadas. Aquí también se contemplaron filtros, navegación y una presentación clara y organizada de la información.

4. **Implementación del módulo de registro de estadísticas**
   Esta etapa incluyó tanto la **lógica de funcionamiento para registrar eventos durante un partido (puntos, rebotes, faltas, etc.)**, como su visualización en tiempo real o posterior mediante la vista de estadísticas.

5. **Revisión y pruebas del sistema completo**
   Finalmente, se realizaron pruebas funcionales y de integración sobre los distintos módulos para asegurar su correcto funcionamiento en conjunto. También se llevaron a cabo correcciones y ajustes.

a. **Recursos utilizados:**

- **Herramientas de desarrollo**: PHPStorm
- **Servidor local**: XAMPP
- **Control de versiones**: Git (repositorio en GitHub)
- **Gestión de dependencias**: Composer
- **Base de datos**: MariaDB, gestionada mediante PhpMyAdmin
- **Frameworks y tecnologías**: Symfony, Twig, Bootstrap, jQuery, Ajax

Documentación técnica

La documentación técnica del proyecto está orientada a desarrolladores y personas encargadas del mantenimiento o evolución del sistema. Incluye la descripción de la arquitectura del software, los lenguajes y herramientas utilizadas, y la estructura general del código.

Aspectos destacados:

Lenguaje y framework principal

- El proyecto ha sido desarrollado usando el framework **Symfony**, que proporciona una estructura organizada y robusta para aplicaciones web en PHP. Se han empleado varios lenguajes y tecnologías complementarias, cada uno con un propósito específico dentro del sistema:
    - **PHP:** Lenguaje principal del backend, utilizado en la lógica de negocio a través de controladores, entidades y repositorios.
    - **Twig:** Motor de plantillas de Symfony, usado para generar las vistas de forma dinámica, separando el código de presentación del backend.
    - **JavaScript (con jQuery y AJAX)**: Utilizado para mejorar la interactividad de la interfaz, permitiendo acciones sin recarga de página.
    - **HTML y CSS:** Para la estructura y diseño de la interfaz de usuario.
    - **Bootstrap:** Framework de diseño que facilita la creación de interfaces responsivas y aporta componentes visuales como botones, tablas, formularios e iconos.

-
- **Composer:** También se han requerido de ciertas dependencias de composer para poder realizar todo el proyecto tal y como se ha hecho:
  - **`symfony/framework-bundle`**
    Núcleo del framework Symfony, que proporciona la infraestructura básica del proyecto.
  - **`symfony/orm-pack`**
    Conjunto de paquetes para trabajar con Doctrine ORM y bases de datos relacionales.
  - **`doctrine/annotations`**
    Permite usar anotaciones en entidades y controladores.
  - **`doctrine/doctrine-bundle`**
    Integración de Doctrine en Symfony.
  - **`symfony/security-bundle`**
    Gestión de usuarios, autenticación y control de acceso.
  - **`symfony/maker-bundle`** (solo en desarrollo)
    Herramienta para generar código automáticamente (controladores, entidades, formularios...).
  - **`symfony/asset`**
    Para la gestión de recursos front-end como imágenes, scripts o estilos.

- **`twig/twig`**
  Motor de plantillas utilizado para generar las vistas del frontend.

- **`symfony/twig-bundle`**
  Integración de Twig en Symfony.

- **`symfony/validator`**
  Sistema de validación de datos usado en formularios o entidades.

- **`symfony/form`**
  Permite crear y gestionar formularios de manera estructurada y segura.
- **`symfony/routing`**
  Para el sistema de rutas y controladores del proyecto.

- El proyecto ha seguido el **patrón Modelo-Vista-Controlador (MVC)**, el cual encaja perfectamente con la estructura propuesta por Symfony..
  - **Modelo (Model)**: Representado principalmente por las **entidades** ubicadas en `src/Entity`. Estas clases definen la estructura de los datos que se almacenan en la base de datos. Se complementan con los **repositorios** (`src/Repository`), que gestionan las operaciones de consulta y manipulación de datos mediante el ORM Doctrine.

  - **Vista (View)**: Ubicada en `templates/`, donde se encuentran los archivos `.twig`. Estas plantillas generan el HTML final que se envía al navegador. La estructura Twig permite reutilizar fragmentos de código (`partials`, `layouts`) y facilita una presentación limpia y coherente.

- **Controlador (Controller)**: Ubicado en `src/Controller/`, contiene las clases que gestionan las rutas y las peticiones del usuario. Cada controlador se encarga de recibir la solicitud, procesar la lógica necesaria (incluyendo validaciones, consultas y procesos de negocio) y devolver una respuesta (generalmente una vista renderizada o una redirección).

Además se han aprovechado la posibilidad de realizar la mayoría de estos archivos mediantes comandos:

- **php bin/console make:entity** (entidades y repos)
- **php bin/console make:auth** (login)
- **php bin/console make:form** (registro)
- **php bin/console make:controller** (controladores)

- La base de datos utilizada es **MariaDB**, y su gestión se ha realizado a través de **Doctrine ORM**, integrado con Symfony. Esto permite mapear las tablas como entidades PHP y definir relaciones de forma sencilla.

  Para mantener la base de datos sincronizada con las entidades, se han usado **migraciones**, generadas y aplicadas con los siguientes comandos:

  - **`php bin/console doctrine:migrations:diff`** → Genera la migración automáticamente

  - **`php bin/console doctrine:migrations:migrate`** → Aplica los cambios a la base de datos

- **Control de versiones:** Git, con repositorio alojado en GitHub y manejado íntegramente dentro de PHPStorm gracias a la funcionalidad que ofrece el programa al respecto.

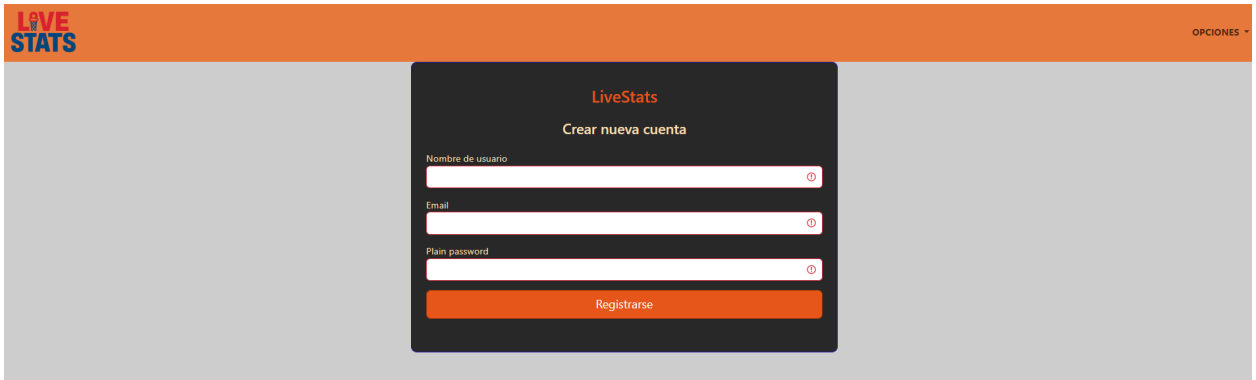
- **Rutas y controladores:** Cada funcionalidad del sistema está asociada a un controlador en Symfony
    - AdminController.php
    - AuthController.php
    - PublicController.php
    - RegistrationController.php
    - StatsController.php
    - TablaController.php
    - UserController.php

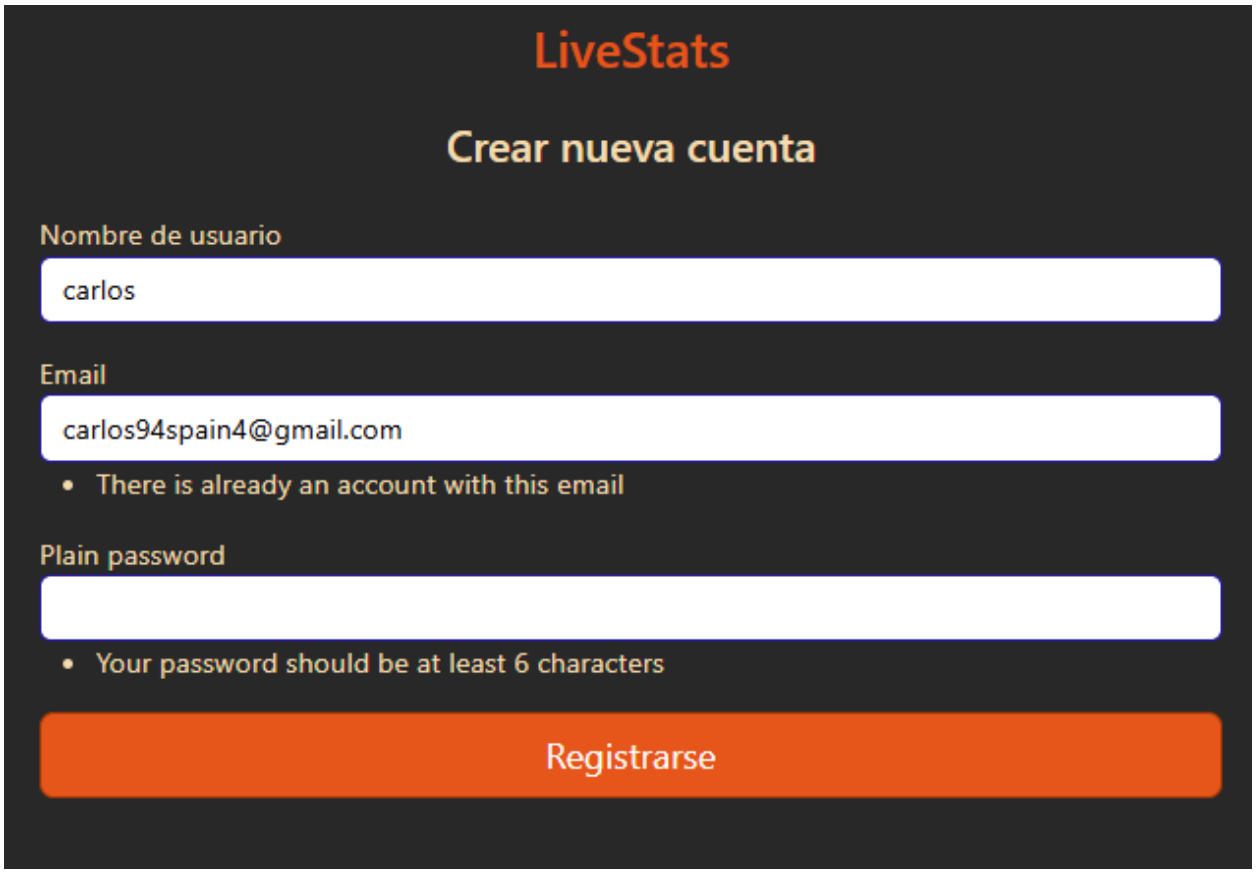
- **Entidades:** El sistema cuenta con entidades como
    - `Accion.php`
    - `Competicion.php`
    - `Equipo.php`
    - `Jugador.php`
    - `Partido.php`
    - `PeticionRol.php`
    - `StatsJugador.php`
    - `User.php`

- **Seguridad:** Symfony Security para autenticación y roles mediante el archivo **config/packages/secyurity.yaml**

Documentación de Usuario

Un usuario que entré en nuestra web podrá registrarse en la vista, para ello deberá darle al enlace que redirige desde el login “no tiene cuenta? registrarse” y pasar parámetros válidos



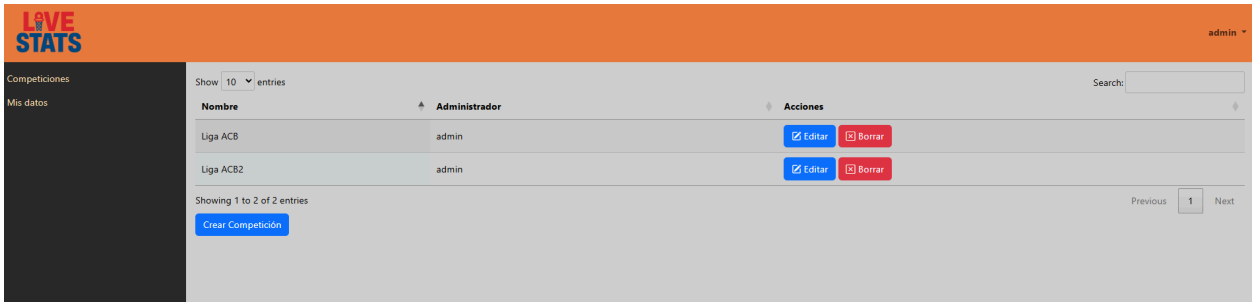


Una vez registrado podrá iniciar sesión con su nueva cuenta, con los datos introducidos en el registro

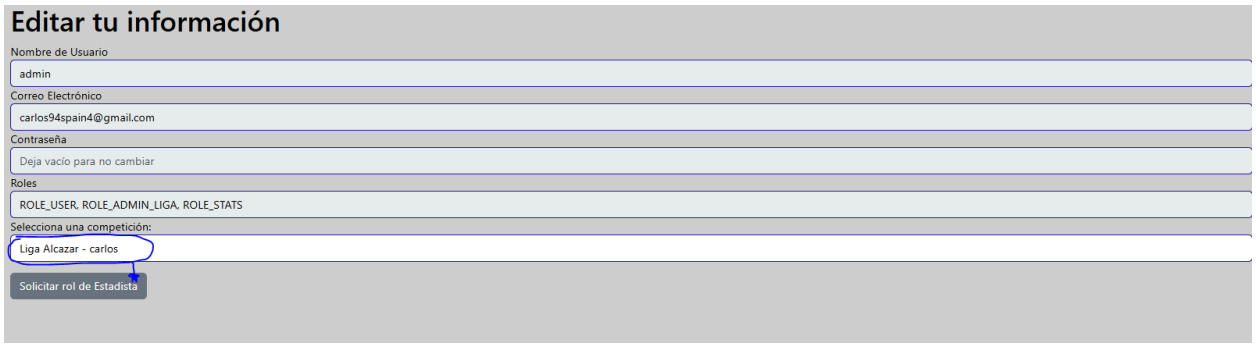
Tras iniciar en la barra de navegación cambiará el menú de opciones, mostrando tu usuario y un menú desplegable

Panel de usuario

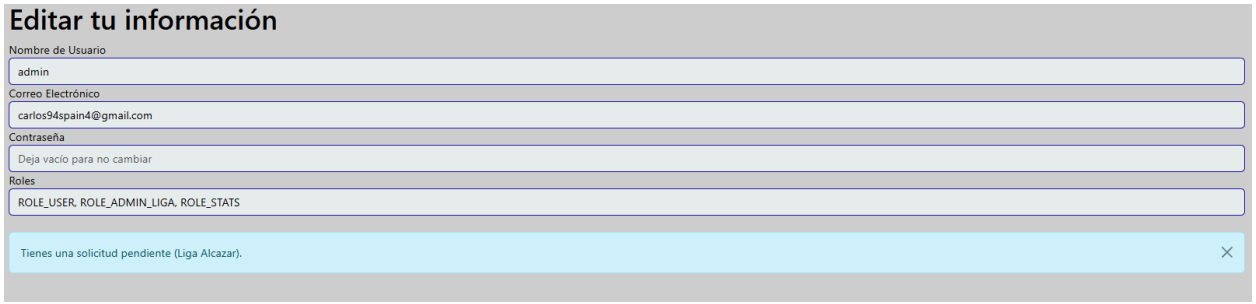
En esta vista tendrá un menú lateral, aqui podrá crear sus propias competiciones si lo necesitase o quisiese, si no, tiene la opción de ver sus datos y solicitar ser estadista de una competición creada por otro usuario



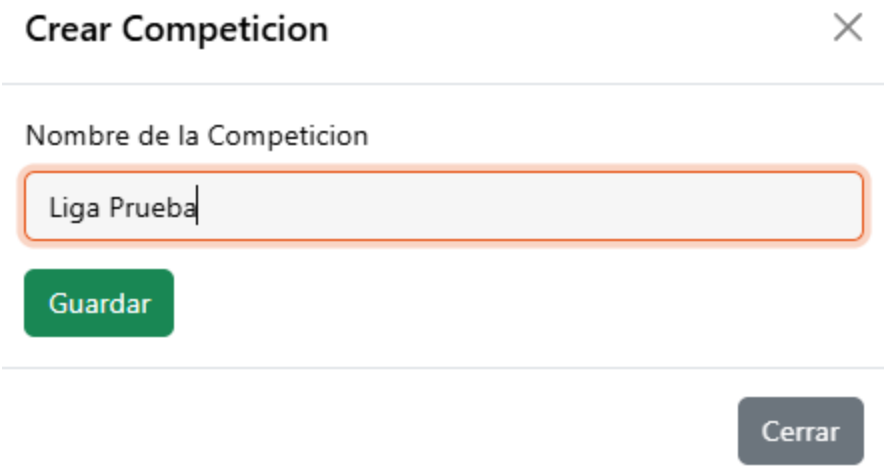
Solicitar ser estadista de una competición eligiendo desde el desplegable, y pulsando el botón “solicitar rol de estadista”.



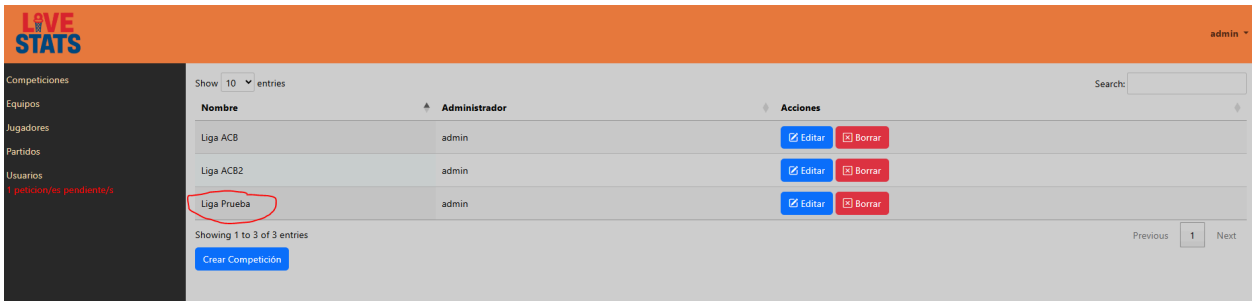
Tras esto se mostrará un mensaje con el estado de la peticion



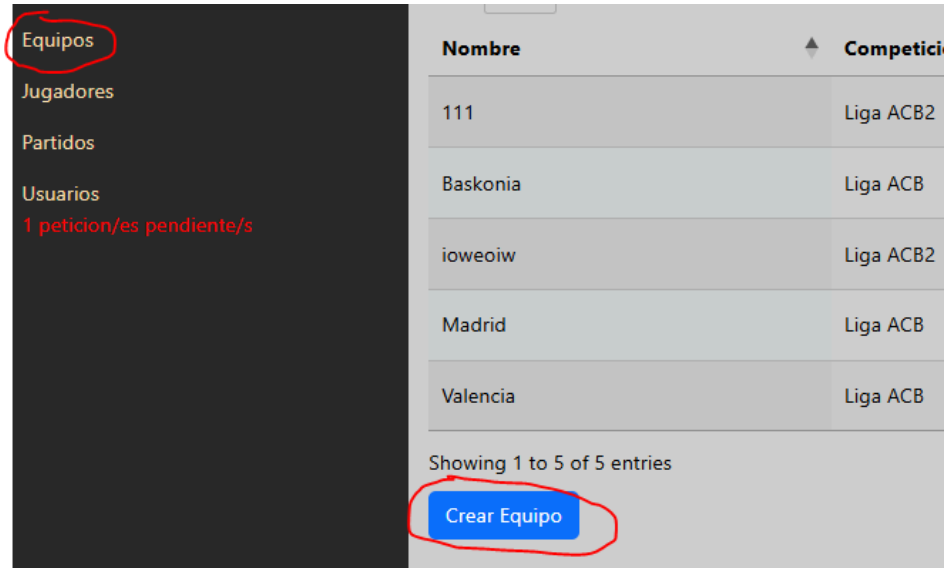
Para crear una competición solo hay que darle al boton “crear competición” en la vista de competiciones, lo que abrirá un modal para poder introducir el nombre



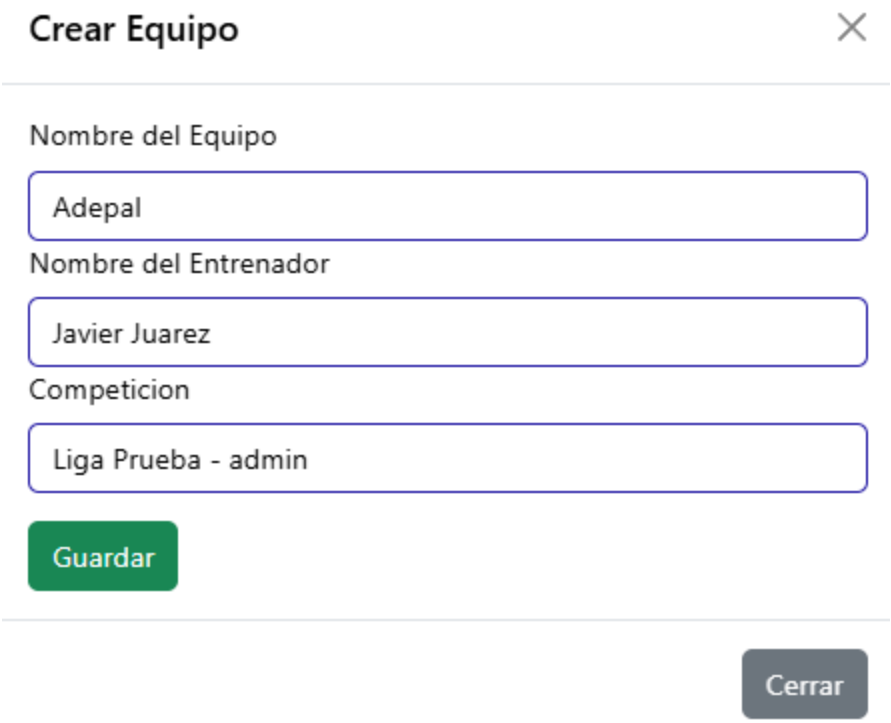
tras hacerlo, se le redirige a un nuevo panel administrativo, el de su competición, desde el que podrá gestionar, equipos, jugadores, partidos, y las peticiones de usuarios que otros hagan sobre su competición



Por ejemplo para crear un equipo, seleccionas la opción de equipos en el menú lateral, y luego en el botón de crear equipos

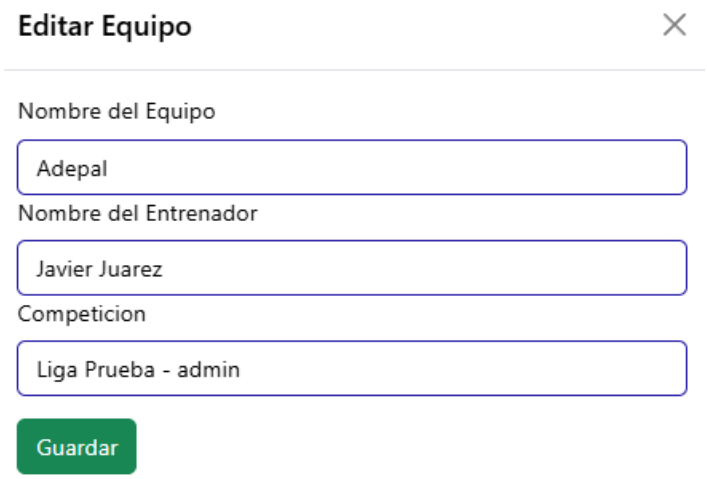


Esto abre un desplegable donde introducir los datos del equipo, a destacar que tienes que elegir una competición de las que eres adminsitrador

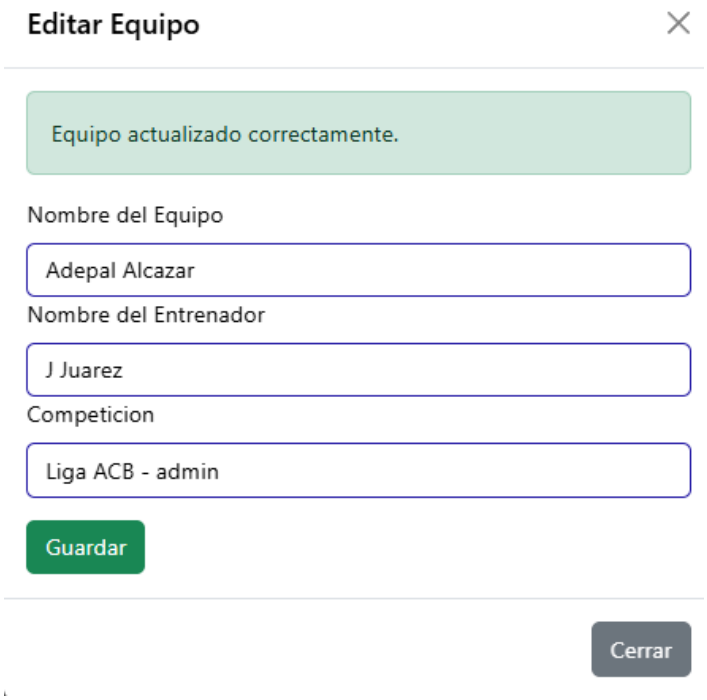


Tras guardar podrás borrar o editar los datos del equipo

Al editar saldrá un formulario igual al anterior pero con los datos del equipo ya cargado

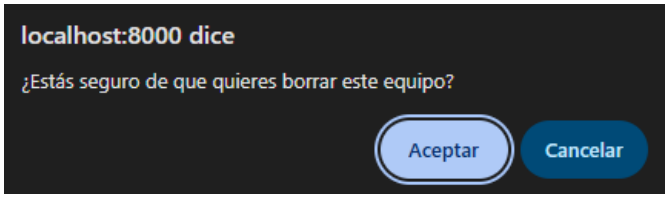


Al guardarlo recargará la tabla mostrará un mensaje confirmando la correcta actualización

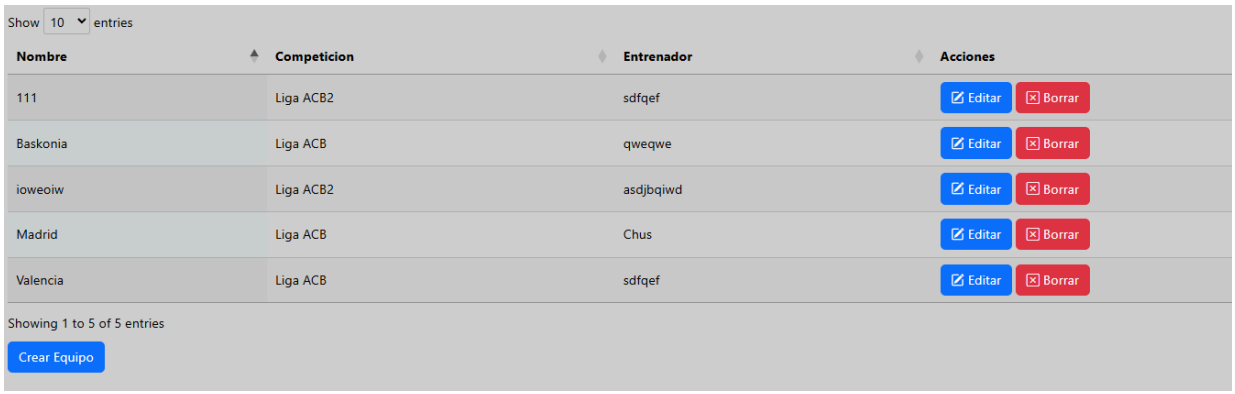


y recargará la tabla ya mostrando los nuevos datos

Para borrar saltará un mensaje de confirmación antes de hacerlo

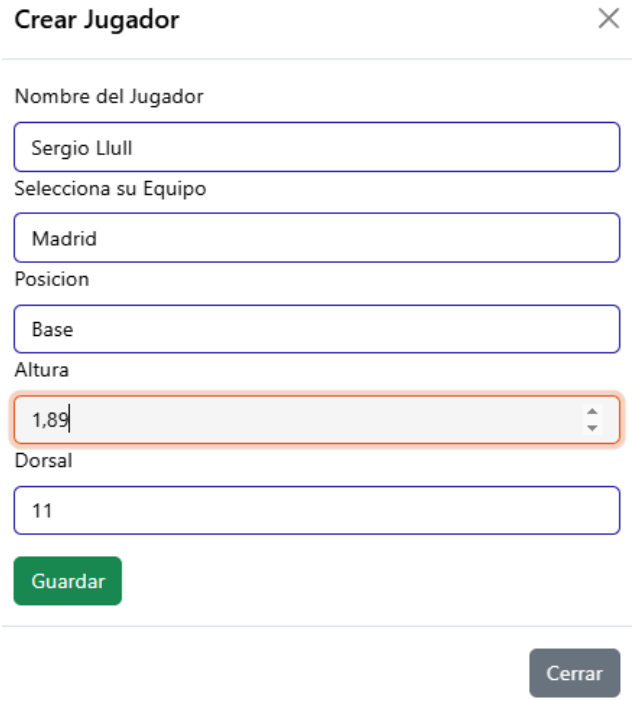


Tras lo cual regará la tabla esta vez ya sin el equipo creado



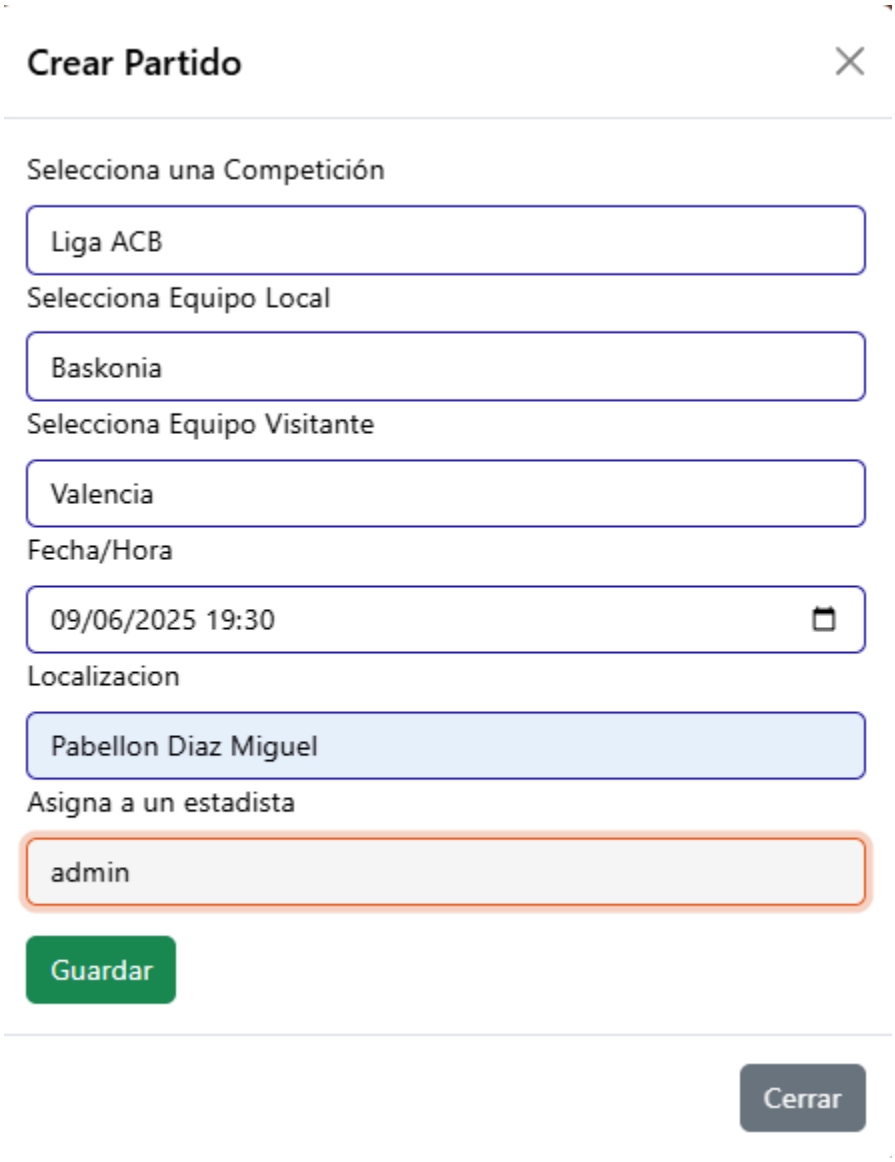
LA MISMA LÓGICA Y PROCEDIMIENTOS SIRVEN PARA JUGADORES Y PARTIDOS, DESTACANDO:

JUGADORES, tienes que elegir entre tus equipos para añadirlo a ese equipo



Partidos: para crear partido primero eliges una competición, tras ello, se eligen un equipo local y otro visitante, deshabilitando los elegidos para la siguiente opción, y lo otro importante es elegir a un estadista para realizarla, si no tienes a nadie más siempre puedes elegirte a ti mismo para esa labor

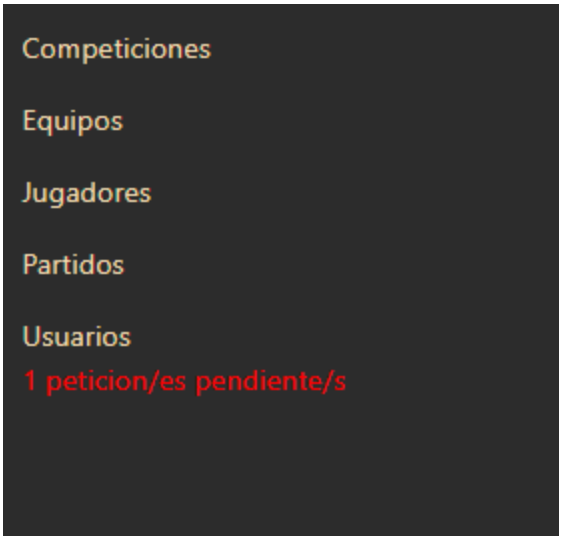
(IMPORTANTE, LOS EQUIPOS DEBEN TENER AL MENOS 5 JUGADORES)



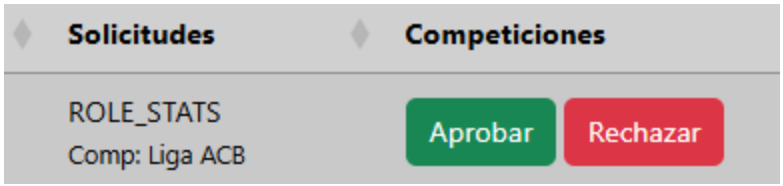
Usuarios: en esta sección podrás gestionar que usuarios (si lo solicitan) tienen permisos para ser estadista de tu competición

Ademas también podrás revocar ese permiso a usuarios ya asignados

Si existe alguna solicitud pendiente aparecerá resaltado en el menú lateral



Para aprobar o rechazar solo habrá que pulsar a uno de estos 2 botones

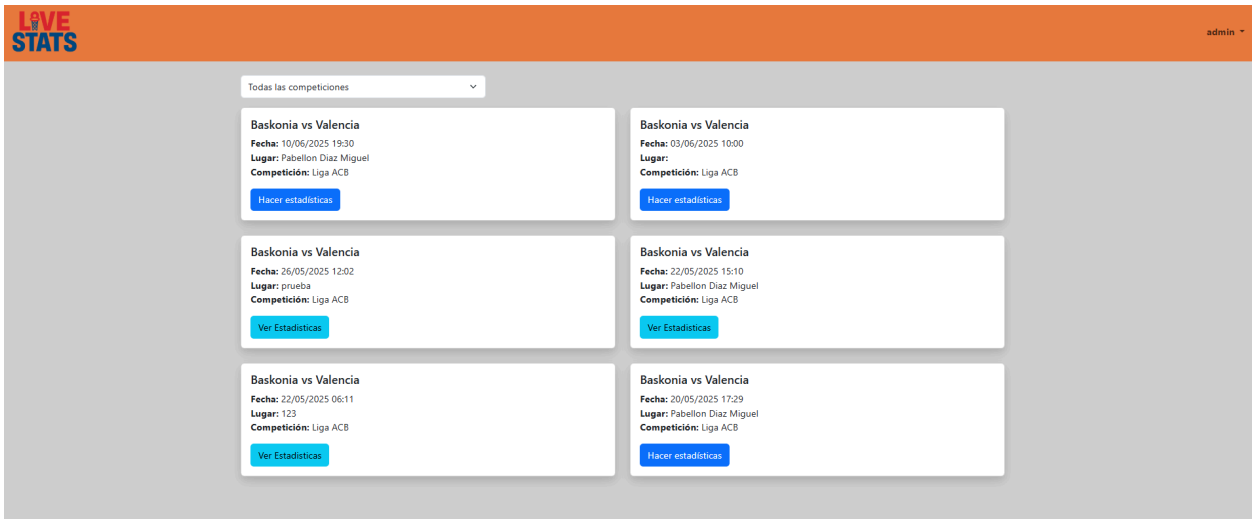


Una vez aprobado se mostrará como un usuario con un desplegable para elegir la competición de la que se le borrará en caso de ser necesario

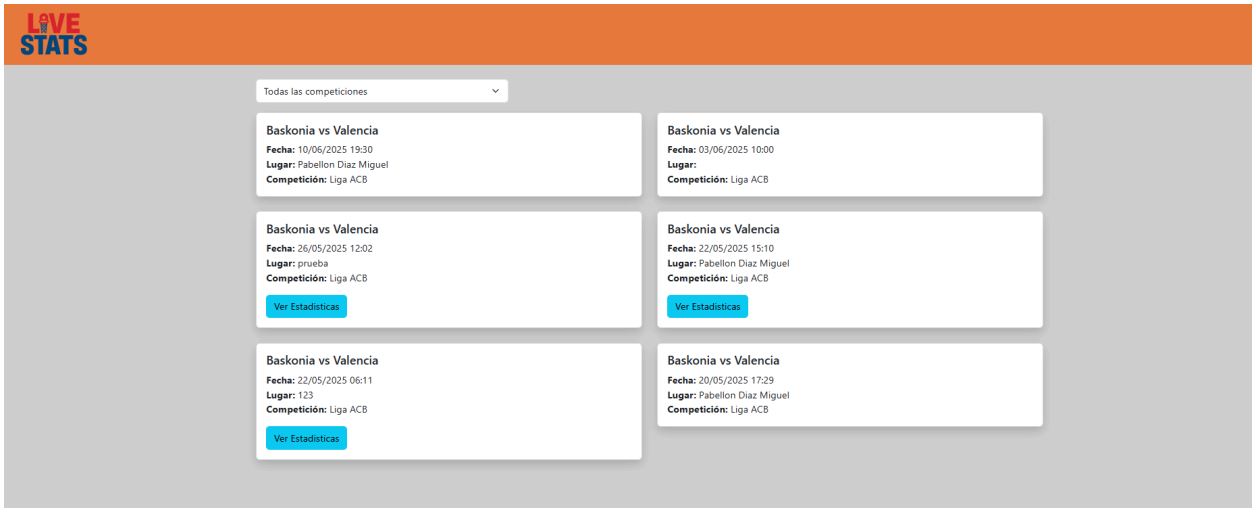
Si se le decide borrar, saldrá un confirm para asegurarse y luego se recargará la tabla ya sin el usuario, o si tiene varias competiciones asignadas, sin esa competición el desplegable

Vista partidos:

En esta vista se mostrarán la lista de los partidos, con un filtro de competiciones, si el partido no está finalizado, y el usuario logueado es el estadista del partido se le mostrará un botón que le llevará a la vista para hacer las estadísticas, si el partido ha terminado mostrará otro botón que redirigirá a la vista de estadísticas.



Vista de otro usuario



Vista estadistica

Se visualizará el resultado del partido y las estadísticas de los jugadores de ambos equipos y sus totales

Estadísticas del Partido

Baskonia 37

Valencia 16

Estadísticas Baskonia

Dorsal	Jugador	PTS	2P	%2	3P	%3	TL	%1	REB OF	REB DEF	AST	TAP	ROB	PER	F.COM	F.REC
4	Juan Pérez	1	0/0	0%	0/0	0%	1/1	100.0%	0	0	0	0	0	0	5	0
7	Carlos López	4	0/0	0%	1/1	100.0%	1/1	100.0%	0	0	0	0	0	0	5	0
11	Miguel Torres	5	1/1	100.0%	1/1	100.0%	0/0	0%	0	0	0	0	0	0	5	0
13	Andrés Martín	3	1/1	100.0%	0/0	0%	1/1	100.0%	0	0	0	0	0	0	0	0
8	Prueba	4	0/0	0%	1/1	100.0%	1/1	100.0%	0	1	0	0	0	0	0	0
6	Pedro García	3	0/0	0%	1/1	100.0%	0/0	0%	0	0	0	0	0	0	0	0
9	Luis Hernández	3	0/0	0%	1/1	100.0%	0/0	0%	0	0	0	0	0	0	0	0
10	Raúl Gómez	8	1/1	100.0%	2/2	100.0%	0/0	0%	1	0	0	1	0	0	1	0
12	Javier Díaz	2	1/2	50.0%	0/0	0%	0/0	0%	0	1	0	1	2	0	0	0
14	Manuel Sánchez	3	0/0	0%	1/1	100.0%	0/0	0%	0	1	0	1	0	0	0	0
15	Sergio Ruiz	0	0/0	0%	0/0	0%	0/0	0%	0	0	1	0	0	0	0	0
	Totales	36	4/5	80.0%	8/8	100.0%	4/4	100.0%	1	3	1	3	2	0	16	0

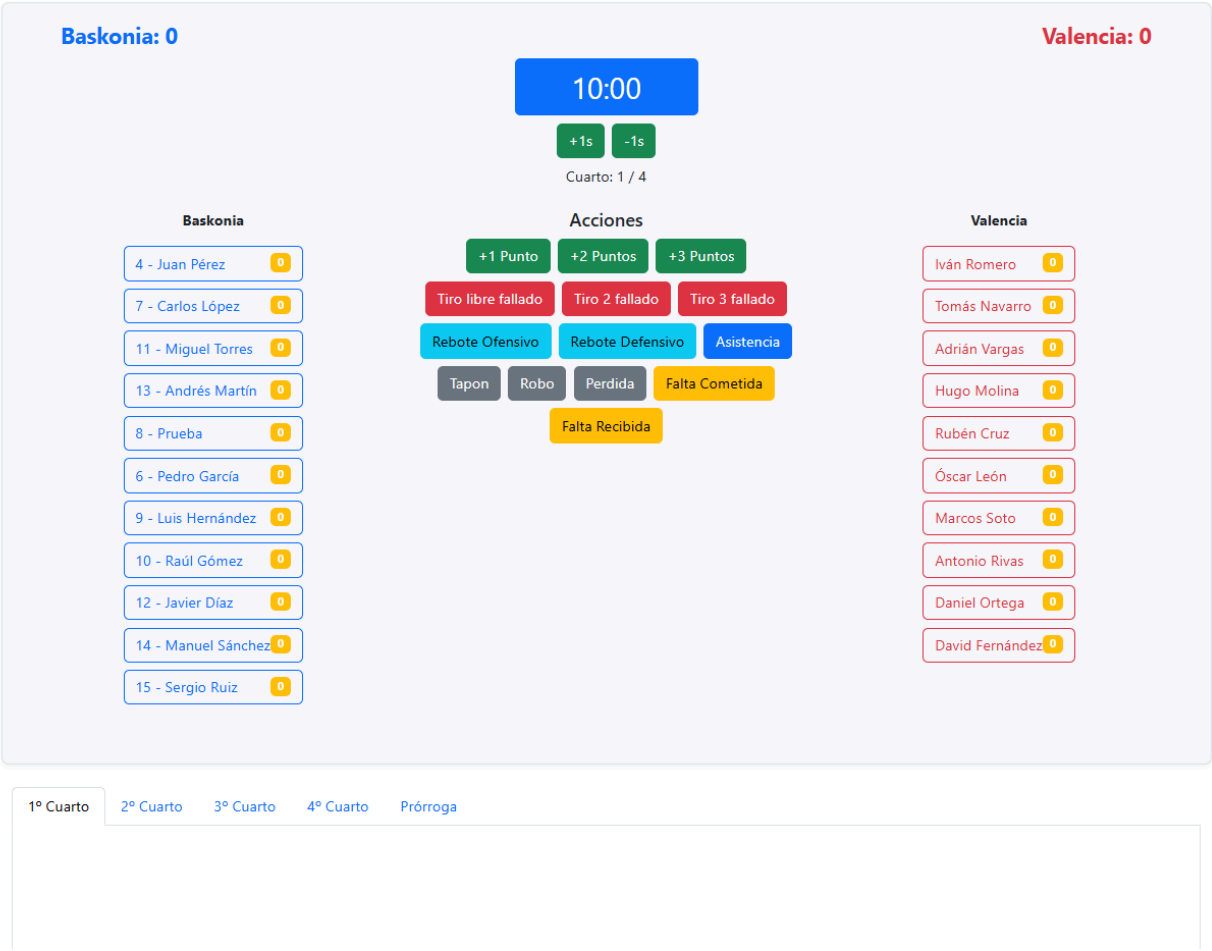
Estadísticas Valencia

Dorsal	Jugador	PTS	2P	%2	3P	%3	TL	%1	REB OF	REB DEF	AST	TAP	ROB	PER	F.COM	F.REC
8	Iván Romero	0	0/0	0%	0/0	0%	0/0	0%	0	0	0	0	0	0	0	0
10	Tomás Navarro	9	1/1	100.0%	2/2	100.0%	1/2	50.0%	0	0	0	0	0	0	0	1
13	Adrián Vargas	3	0/0	0%	1/1	100.0%	0/0	0%	0	0	0	0	0	0	0	3

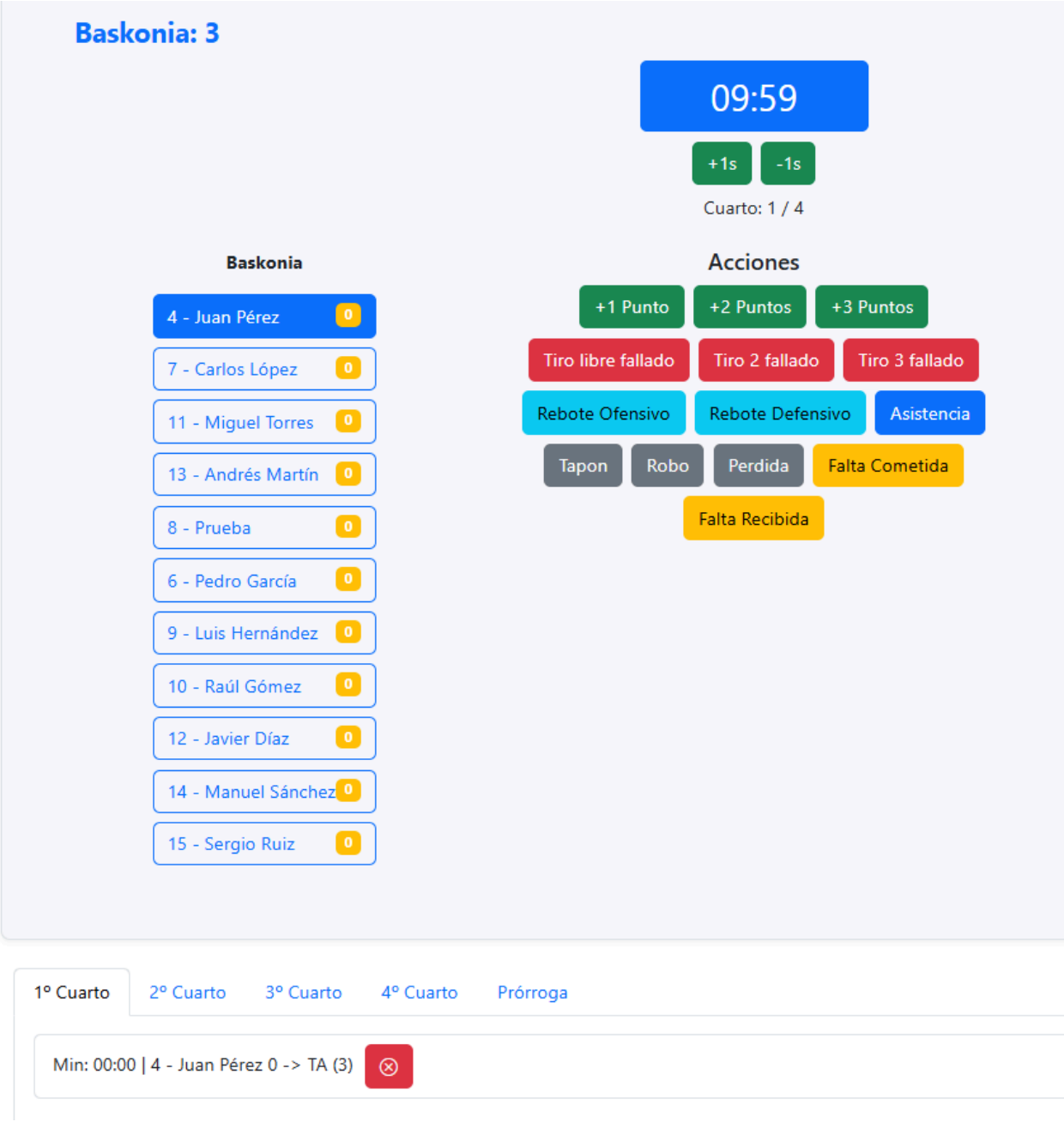
En el fondo de la página habrá un botón para volver a la vista de partidos

Realizar Estadísticas

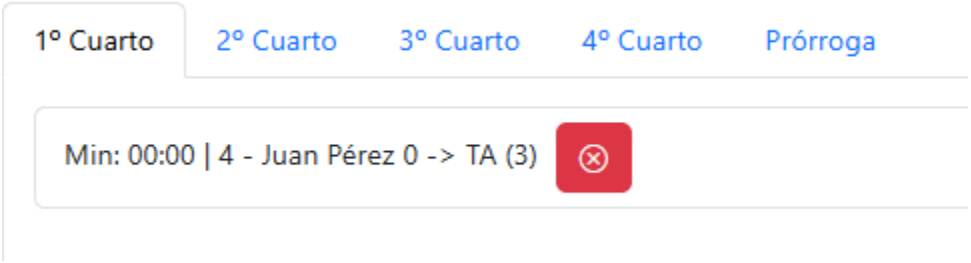
La vista se compondrá de el marcador del equipo local y del visitante, botones de lista de los jugadores de ambos equipos y el número de faltas de cada jugador(necesarias para poder “expulsar” a un jugador), una zona central donde están los botones para controlar las acciones del partido y en la zona inferior un registro de las acciones



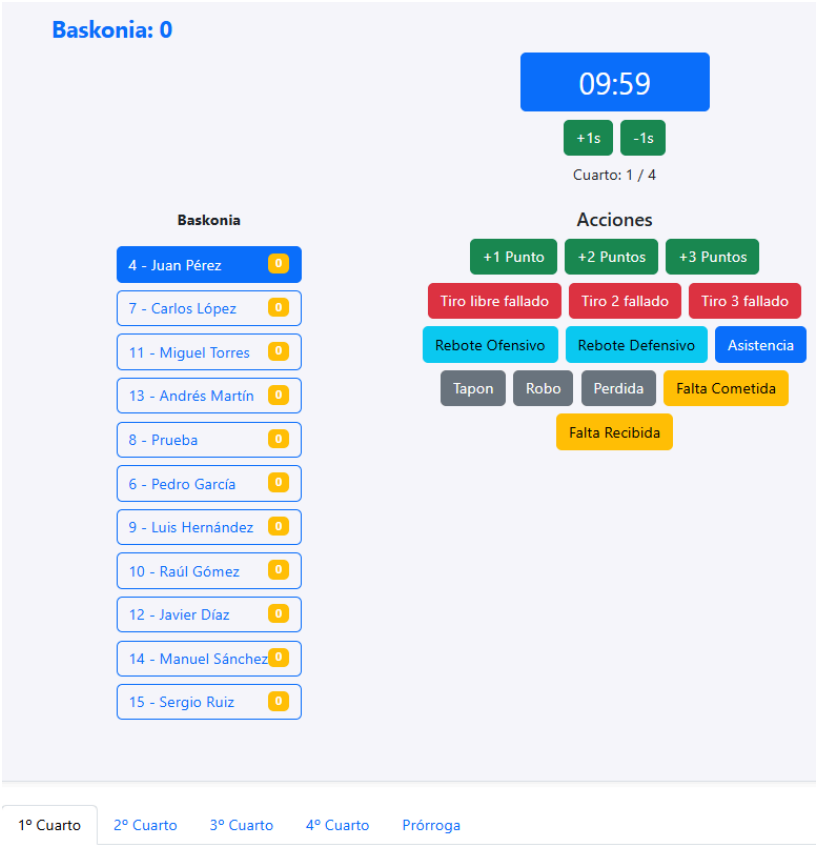
Para registrar acciones se puede hacer tanto con el reloj parado como con el reloj en marcha, y el sistema es se elige a un jugador de uno de los 2 equipos y después el tipo de acción a registrar.



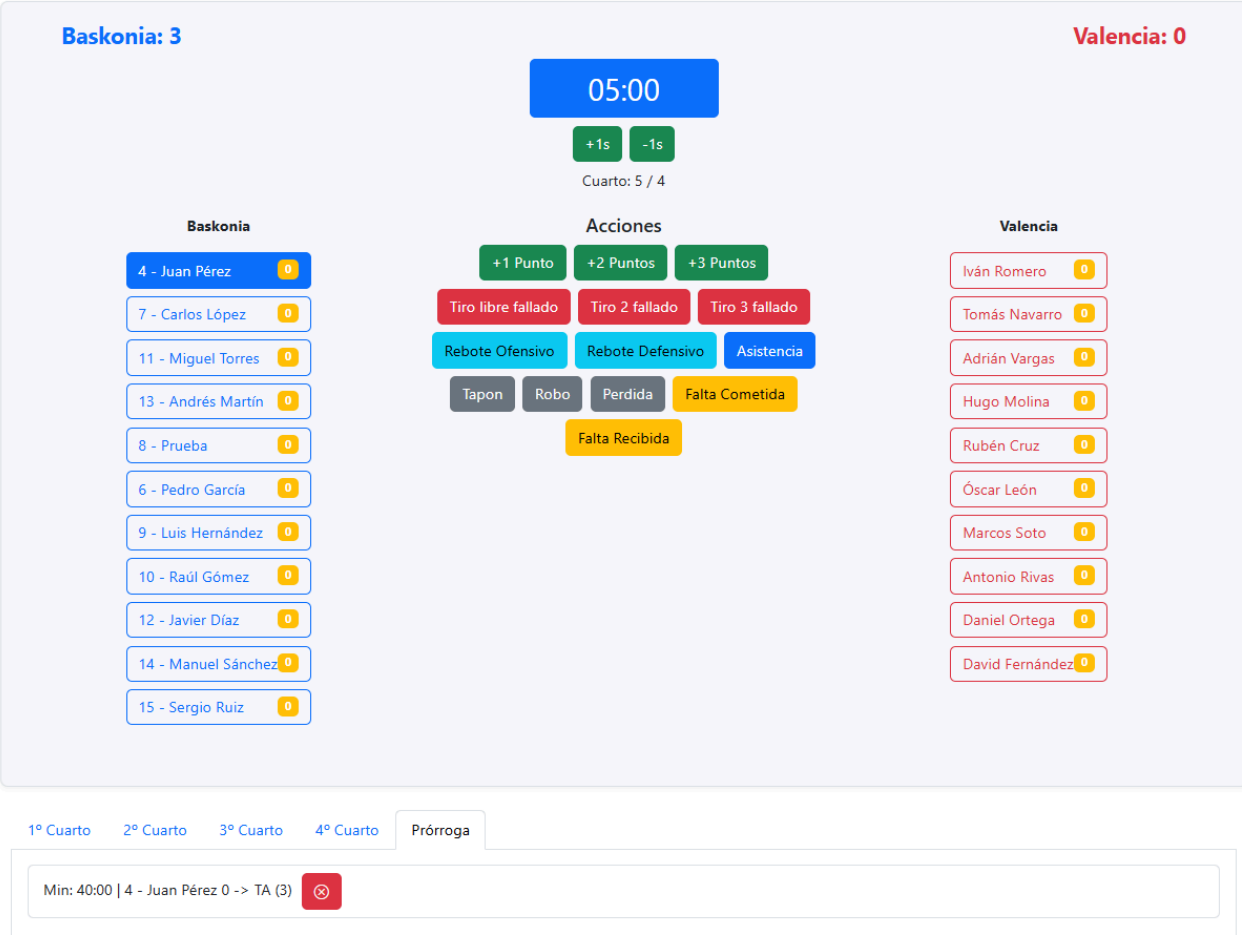
También se pueden borrar acciones en caso de habernos equivocado, solo habrá que pulsar el botón rojo junto a la acción más un modal de confirmación



Al borrarse, si son tiros anotados el marcador se actualizará, si es una falta cometida el registro de faltas también lo hará



La mecánica de partido es simple, el tiempo y el contador de cuartos tienen que llegar a 0:00 y a 4/4 siempre y cuando uno de los 2 equipos tenga más puntos que el otro, en caso contrarío se iniciará un 5/4 quinto cuarto de 5:00 minutos que actuará de prorroga, así hasta que uno de los 2 haya ganado



Una vez terminado el partido, los botones se deshabilitarán y el temporizador cambiará su contenido a un enlace que redirige a ver las estadísticas

Manual de instalación y configuración

El proyecto se ejecuta mediante **Docker**, lo que simplifica su despliegue y evita problemas de configuración.

Requisitos

- Tener instalado **Docker** y **Docker Compose**.

Instrucciones

Crear un archivo `docker-compose.yaml` en una carpeta vacía con el siguiente contenido:

```
version: "3.8"

services:

  app:

    image: camosama94/livestats-app:latest

    ports:

      - "8080:8080"

    depends_on:

      - db

    environment:

      APP_ENV: prod


      DATABASE_URL:"mysql://symfony:symfony@db:3306/basket?serverVersion
      =mariadb-10.4.28&charset=utf8mb4"



  db:

    image: mariadb:10.4.28

    restart: always
```

```
    environment:
      MYSQL_DATABASE: basket
      MYSQL_USER: symfony
      MYSQL_PASSWORD: symfony
      MYSQL_ROOT_PASSWORD: root
    ports:
      - "3306:3306"
```

Ejecutar comando:

docker compose up

Abre el navegador y visita:

http://localhost:8080

Mantenimiento y evolución

Plan de mantenimiento y soporte

Una vez finalizado el desarrollo inicial del sistema, se establece un plan básico de mantenimiento con los siguientes objetivos:

- **Corrección de errores detectados en producción.**
- **Actualización de dependencias (Symfony, PHP, paquetes de Composer).**
- **Monitoreo del rendimiento y uso de recursos.**
- **Respaldo periódico de la base de datos** y mantenimiento del contenedor Docker.

El mantenimiento será realizado de forma puntual según las necesidades y no se contempla, de momento, un equipo de soporte permanente, ya que el sistema está orientado a un entorno controlado.

---

b. Identificación de posibles mejoras y evolución del proyecto

Durante el desarrollo y uso de la aplicación se han detectado varias líneas de mejora para versiones futuras:

- **Actualización de algunas vistas por terminar y ejoras en el diseño responsive y accesibilidad para dispositivos móviles.**
- **Realización de vista para poder seguir partido en vivo**
- **Estructuración completa de temporadas, jornadas, clasificación, etc.**
- **Respaldo periódico de la base de datos** y mantenimiento del contenedor Docker.
- **Mayor control a la hora de hacer estadísticas, relacionando tipos de acciones con otras.**
- **Controlar la habilitación de inicio de partidos en base a hora programada**

Evaluación del proyecto

El proyecto cumple satisfactoriamente con el objetivo inicial de permitir la gestión de partidos y estadísticas de baloncesto en tiempo real. La arquitectura elegida (Symfony + MVC + Docker) ha demostrado ser adecuada para construir un sistema modular, mantenible y escalable.

La lógica del negocio se ha estructurado de forma clara, las vistas son usables y el uso de Docker facilita la instalación y despliegue del entorno.

Cumplimiento de objetivos y requisitos

- Registro e inicio de sesión de usuarios funcional
- Gestión completa de competiciones, equipos, jugadores y partidos, etc...
- Registro de acciones en tiempo real
- Visualización de estadísticas por jugador y partido
- Aplicación contenida y ejecutable con Docker

Bibliografía y referencias

Fuentes utilizadas en el proyecto

- **Symfony Documentation**
  Symfony SAS
  URL: https://symfony.com/doc

- **Doctrine ORM**
  Doctrine Project
  URL: https://www.doctrine-project.org

- **Twig Template Engine**
  Symfony SAS
  URL: https://twig.symfony.com/

- **Bootstrap Icons**
  Bootstrap Team
  URL: https://icons.getbootstrap.com/

- **Stack Overflow**
  Stack Exchange Inc.
  URL: https://stackoverflow.com

- **ChatGPT**
  OpenAI
  URL: https://chatgpt.com/

- **Color HEX**
  Color-Hex.com
  URL: https://www.color-hex.com/