FIGHT-ZONE

PFC-2ºD.A.M

16 de junio de 2025

IES Segundo de Chomón

Índice

1. Descripción del proyecto

1.1. Contexto del proyecto

1.1.1. Ámbito y entorno

Este proyecto se basa en las artes marciales y la tecnología. La idea nace al ver que muchos torneos todavía se organizan de forma manual o con herramientas que no están pensadas para eso. El objetivo es modernizar y facilitar todo lo relacionado con la organización y participación en estos eventos, usando una app que funcione como red social y una parte web para los organizadores.

1.1.2. Análisis de la realidad

Hoy en día, muchos torneos de artes marciales no tienen una forma fija para controlar todo: las inscripciones, los combates, los participantes, los resultados, etc. Se usa a menudo WhatsApp, hojas de Excel o papeles sueltos. Algo que hace difícil la organización, los peleadores tampoco tienen un lugar donde ver próximos eventos o ver quienes van a participar.

1.1.3. Solución y justificación de la solución propuesta

Para solucionar esto, propongo una aplicación móvil llamada FIGHT-ZONE, que es parecida a una red social, pero centrada en torneos de artes marciales. Desde la app se pueden ver eventos, inscribirse, mirar los perfiles de otros peleadores y seguir las peleas. También hay una parte web pensada para los organizadores, donde pueden gestionar los participantes, hacer los cuadros de combate y llevar el control del torneo. Con esta herramienta, todo es más fácil, rápido y organizado.

1.1.4. Destinatarios

Los destinatarios son:

- Peleadores que quieren participar en torneos y ver su progreso
- Organizadores que buscan una forma más práctica de gestionar sus eventos
- Entrenadores y academias, que quieren seguir a sus alumnos
- Aficionados que simplemente quieren ver torneos y estar al tanto de todo

1.2. Objetivo del proyecto

El objetivo del proyecto es crear una app móvil y una web para ayudar a organizar y seguir torneos de artes marciales de forma más sencilla. La idea es que sea útil tanto para los que participan como para los que organizan, y así crear una comunidad más conectada y moderna dentro de estos deportes.

1.3. Objetivo del proyecto en la lengua extranjera

Tenemos como objetivo traducir nuestro proyecto a inglés, si es bien recibida en España, para que todos los demás países puedan tener acceso a mi aplicación

2. Documento de Acuerdo del proyecto

2.1. Requisitos funcionales y no funcionales

Requisitos funcionales:

- Registro e inicio de sesión de usuarios
- Ver torneos disponibles y detalles de cada uno
- Inscribirse a un torneo desde la app
- Ver perfiles de otros peleadores
- Visualizar cuadros de combate y resultados
- Sistema de roles para controlar permisos
- Panel de administración para organizadores (solo en la parte web)
- Gestión de torneos, emparejamientos y resultados desde la web

Requisitos no funcionales:

- Interfaz fácil de usar y moderna
- Compatible con Android
- Base de datos SQLServer
- Seguridad en los datos personales (autenticación, validación)
- Escalable, para poder crecer si la usa más gente

2.2. Tareas

Las tareas a realizar son:

- Análisis y diseño del proyecto
- Conexión con la base de datos y lógica de negocio
- Configuración de la base de datos
- Desarrollo de la app (front-end móvil)
- Diseño de interfaces (app y web)
- Desarrollo del panel web para organizadores
- Pruebas funcionales y de usuario
- Documentación del proyecto
- Despliegue y presentación

2.3. Metodologías a seguir para la realización del proyecto

Se seguirá una metodología ágil, tipo SCRUM, dividiendo el trabajo en sprints semanales. Al final de cada sprint se revisa lo desarrollado, se hacen pruebas y se ajustarán tareas para la siguiente semana. Esto ayuda a ir avanzando poco a poco, corrigiendo errores y mejorando el diseño según se desarrolla.

2.4. Planificación temporal de las tareas

Semana	Tareas
1	Análisis del proyecto y definición de requisitos
2	Diseño de la base de datos y primeras pantallas
3	Desarrollo de la API
4	Funcionalidad de inscripción y perfiles
5	Desarrollo de la app móvil (registro, login, ver torneos)
6	Desarrollo del panel web para organizadores
7	Gestión de torneos y cuadros de combate
8	Pruebas, ajustes y mejoras
9	Documentación y preparación de la entrega

2.5. Presupuestos (gastos, ingresos, beneficio)

Concepto	Detalle	Coste aproximado
Dominio web	Para el acceso a la parte web del sistema	10 €/año
Hosting	Alojamiento para la web y la base de datos	30-50 €/año (según proveedor)
SQL Server	Uso de una instancia en la nube (Azure o VPS propio)	0-20 €/año (uso básico en Azure o local con versión gratuita)
Herramientas de desarrollo	Android Studio, Visual Studio Code, SQL Server Management Studio	0 €/año (software gratuito)

Ingresos: ninguno, al tratarse de un proyecto académico sin uso comercial real por ahora

Beneficio: el objetivo no es económico, sino práctico. El beneficio principal es la adquisición de experiencia profesional y el aprendizaje de herramientas reales

2.6. Contrato/Pliego de condiciones

Este proyecto se desarrolla como parte del módulo de Proyecto del Ciclo Formativo de Grado Superior de Desarrollo de Aplicaciones Multiplataforma (DAM). El trabajo se realiza de forma individual y no cuenta con un cliente o empresa real como destinatario. Aun así, la aplicación debe ajustarse a los requisitos previamente definidos y cumplir con los objetivos establecidos en el diseño. Además, la aplicación debe ser funcional y demostrable de manera práctica.

2.7. Análisis de riesgos

Riesgo	Posible impacto	Solución
Problema con la base de datos	Pérdida de datos	Copias de seguridad y pruebas continuas
Errores en la app o web	Fallos en la entrega	Fase de pruebas y revisión en cada sprint
Falta de tiempo	No completar el proyecto	Buena planificación y priorizar tareas clave
Dificultad técnica (conexión app-web)	Retrasos	Buscar ayuda y documentación oficial

3. Documento de análisis y diseño

3.1. Modelado de datos

El sistema utiliza una base de datos relacional implementada en SQL Server, con varias tablas principales que modelan la información necesaria para gestionar los torneos de artes marciales, los peleadores y sus combates. Las entidades y sus relaciones principales son:

- Users: almacena los datos de los usuarios registrados, incluyendo su rol (usuario normal, peleador, organizador o administrador)
- Fighters: tabla heredada de usuario que es peleador, con atributos específicos como categoría de peso, altura, alcance, y estadísticas de combates (victorias, derrotas, empates)
- Tournaments: almacena los torneos, con datos como nombre, fechas, tipo de deporte, organizador y lugar
- Participantes: relaciona usuarios inscritos a torneos, garantizando que un usuario no se inscriba varias veces al mismo torneo (restricción UNIQUE)
- Fights: representa cada pelea dentro de un torneo, con referencia a los dos peleadores, el estado de la pelea y el ganador si ya se conoce
- FightResults: guarda el resultado de cada pelea, incluyendo el método de victoria y duración

3.2. Análisis y diseño del sistema funcional

El sistema tiene dos módulos funcionales: la app móvil para usuarios, peleadores y organizadores, y la aplicación web exclusiva para organizadores

- App móvil: permite a los usuarios registrarse, ver torneos disponibles, inscribirse en ellos, consultar perfiles de peleadores y seguir el progreso de las peleas
- Web para organizadores: facilita la creación y gestión de torneos, asignación de peleas, seguimiento de resultados y actualización del estado de cada combate

La lógica del sistema está dividida entre el cliente (app móvil y web) y el servidor (backend con SQL Server). La app y la web consumen servicios para consultar y actualizar la base de datos, gestionando inscripciones, resultados y datos de usuarios

3.3. Análisis y diseño de la interfaz de usuario

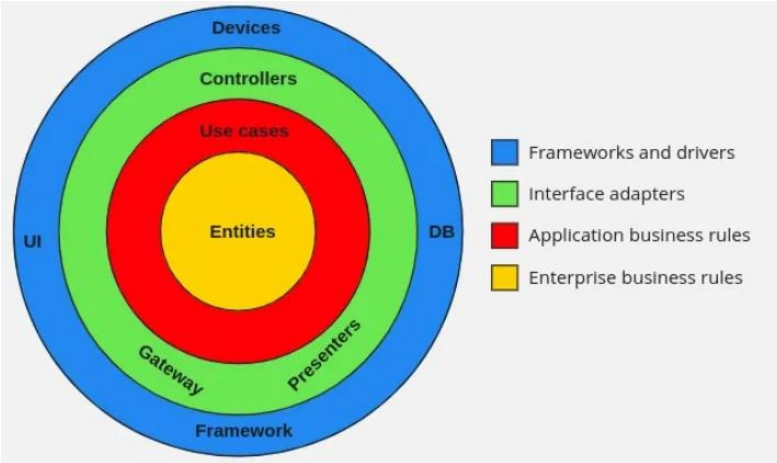
La interfaz de usuario está diseñada para ser intuitiva y accesible:

- En la app, las pantallas principales incluyen: listado de torneos, calendario, lista de peleadores y perfil
- La navegación es sencilla, basada en menús y pestañas para que el usuario pueda acceder rápidamente
- En la web, el organizador tiene un panel de control con acceso a la gestión de torneos, creación de combates, asignación de peleadores y actualización de resultados

3.4. Diseño de la arquitectura de la aplicación

El backend ha sido desarrollado usando la arquitectura de desarrollo Clean Architecture, aprendida durante el periodo de prácticas en Kintech.

Clean Architecture es un patrón de diseño de software que busca organizar el código de forma que sea fácil de mantener, probar y escalar. Su principal objetivo es separar las responsabilidades en capas, de modo que el núcleo de la aplicación (la lógica de negocio) esté aislado del exterior como de la interfaz de usuario o la base de datos.



3.4.1. Tecnologías/Herramientas usadas y descripción de las mismas

Las tecnologías utilizadas fueron:

- App móvil: desarrollada en Dart usando Flutter
- Web: implementada con Razor
- Base de datos: SQL Server, alojada en un servidor propio o en la nube
- Backend: API REST que comunica la app y la web con la base de datos, gestionando la lógica de negocio y seguridad hecha en .NET 8 con C#
- Herramientas adicionales: Visual Studio para desarrollo web y la API, SQL Server Management Studio para la gestión de base de datos y Visual Studio Code para la app

3.4.2. Arquitectura de componentes de la aplicación

La arquitectura sigue un modelo cliente-servidor:

- Clientes: app móvil (Android) y aplicación web, que consumen la API REST
- Servidor: API REST que procesa peticiones, aplica lógica de negocio y accede a la base de datos SQL Server
- Base de datos: almacena toda la información estructurada y garantiza integridad y seguridad

4. Documento de implementación e implantación del sistema

4.1. Implementación

En esta fase se desarrolla el código fuente de la aplicación, implementando todas las funcionalidades definidas en el análisis y diseño. Se crea tanto la aplicación móvil/web para usuarios y organizadores, como la base de datos en SQL Server.

4.2. Pruebas

Se realizan pruebas funcionales para verificar que todas las funcionalidades cumplen con los requisitos. También pruebas de usabilidad para garantizar que la interfaz sea intuitiva y fácil de usar, así como pruebas de rendimiento para evaluar la capacidad de respuesta. Finalmente, se corrigen errores detectados y se repiten las pruebas hasta obtener un sistema estable y fiable para su salida

5. Documento de cierre

5.1. Documento de instalación y configuración

Paso 1. Bajarse el repositorio de Github e instalar las herramientas de los siguientes enlaces (SQL Server Management Studio, SQL Server Express, Visual Studio):

https://github.com/MarioPerez125/TFC.git

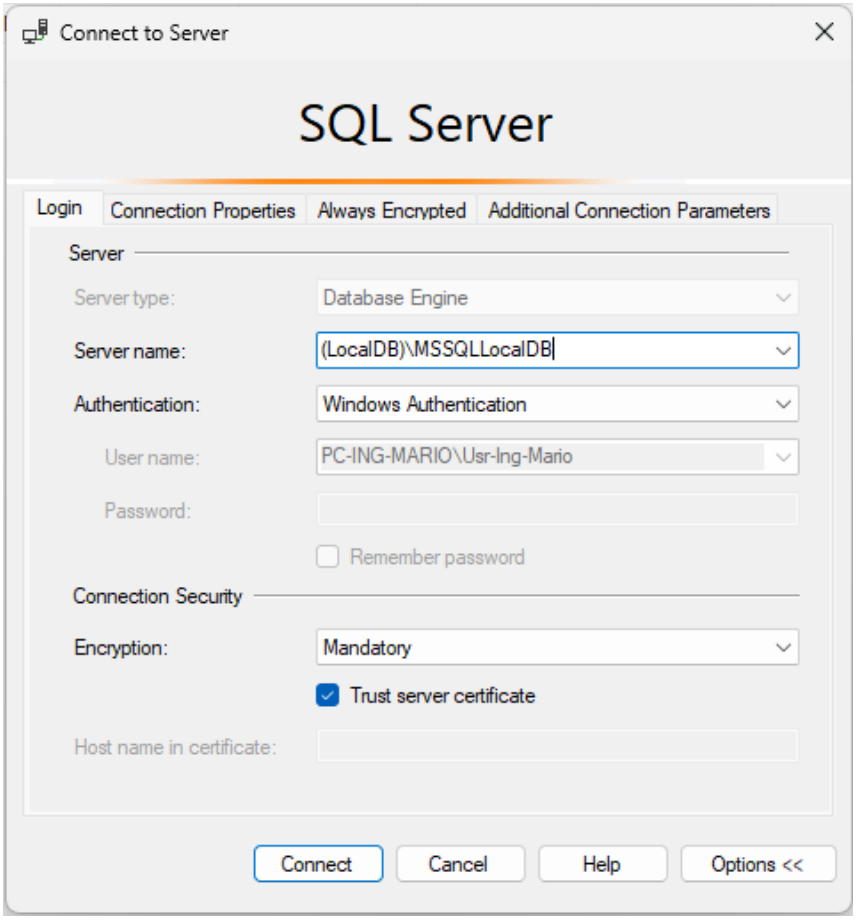
https://go.microsoft.com/fwlink/p/?linkid=2216019&clcid=0x40A&culture=es-es&country=es

https://go.microsoft.com/fwlink/?linkid=2313753

https://visualstudio.microsoft.com/es/thank-you-downloading-visual-studio/?sku=Community&channel=Release&version=VS2022&source=VSLandingPage&cid=2030&passive=false

Se recomienda la instalación por defecto de SQL Server Express (básica), así como la instalación de Visual Studio Community junto con ASP.NET Core y web.

Paso 2. Ejecutar el SQL Server y configurarlo como se muestra en la imagen para comunicarse con el servidor local. Poner en el apartado de Server name la siguiente cadena “(LocalDB)\MSSQLLocalDB”.

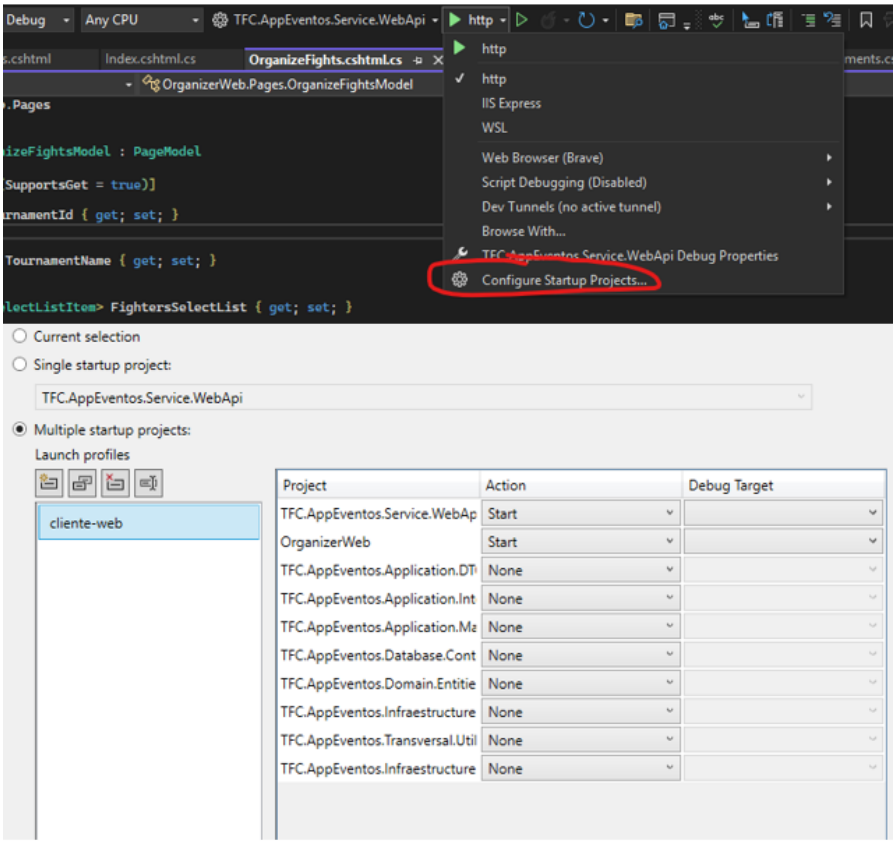


Una vez configurado, abrir en SQL el archivo “TFC-Database.sql” y conectarse. Luego hay que cambiar la línea de comandos que se muestra en la figura y sustituir la parte señalada por la ruta de su ordenador (cambiar el “\PC\” por su usuario “\[usuario]\”). Finalmente, se ejecuta el comando.

```
CREATE DATABASE [TFC-DB2]
 CONTAINMENT = NONE
 ON  PRIMARY
( NAME = N'TFC-DB2', FILENAME = N'C:\Users\PC\TFC-DB2.mdf' , SIZE = 3264KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB )
 LOG ON
( NAME = N'TFC-DB2_log', FILENAME = N'C:\Users\PC\TFC-DB2_log.ldf' , SIZE = 832KB , MAXSIZE = 2048GB , FILEGROWTH = 10%)
GO
```

Paso 3. Abrir en Visual Studio el archivo que se encuentra en la ruta “tfc\TFC\TFC-AppEventos\TFC-AppEventos.sln”

Paso 4. Asignar como proyectos de inicio el WebApi y el OrganizerWeb, así ya estarán listas para usarse.

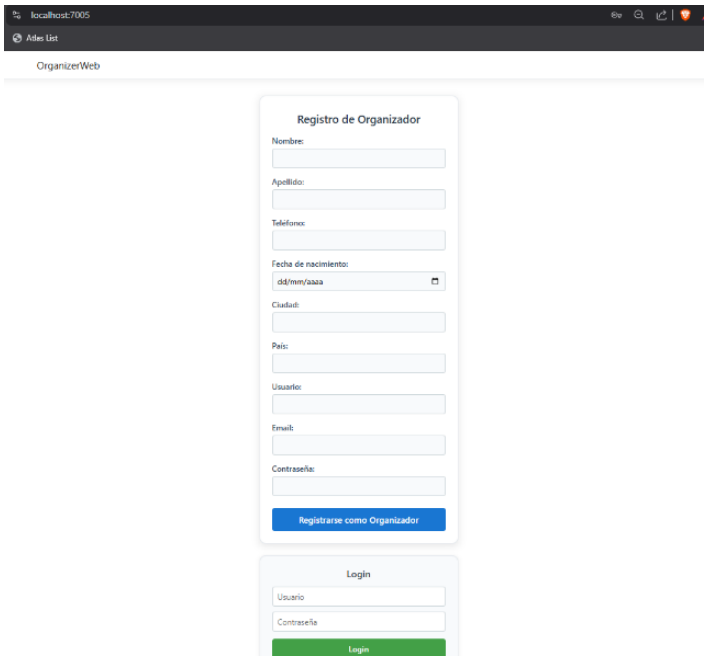


Paso 5. Ejecutar la app desde Flutter

5.2. Manual de usuario

Manual Web

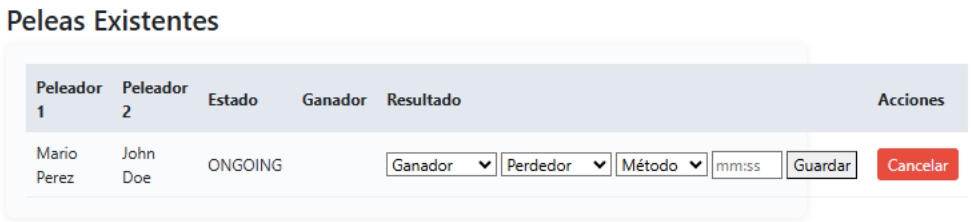
Al ejecutar el servicio se abrirá una ventana para registrase, como se muestra en la figura



Una vez registrado/iniciada la sesión, aparecerá un formulario para crear un torneo, tal y como muestra la imagen de la izquierda. Al organizar un torneo, aparecerá registrado tal y como se observa en la figura de la derecha.

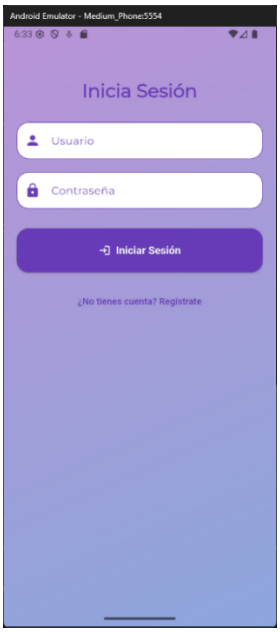
Para organizar los combates, hay que clicar en el torneo creado que aparecerá en el apartado “Mis torneos” (creando un torneo previamente). Se abrirá un menú para organizar cada combate, tal y como se muestra en la imagen.

Los peleadores deben registrarse desde la app para que aparezcan como peleadores inscritos en el torneo. Además, cuando se ha terminado un combate, se puede registrar el resultado (ganador y perdedor) clicando en el combate corresponde desde el apartado “Peleas existentes”. El menú para registrar el resultado se muestra en la siguiente imagen.

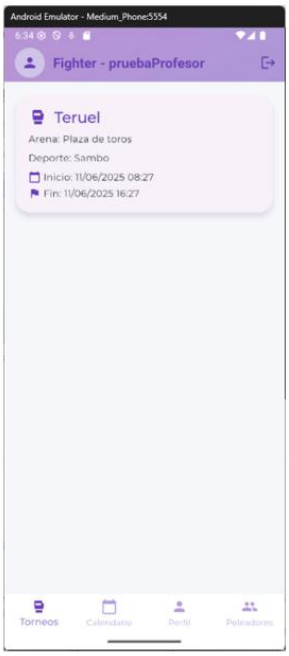


Manual App

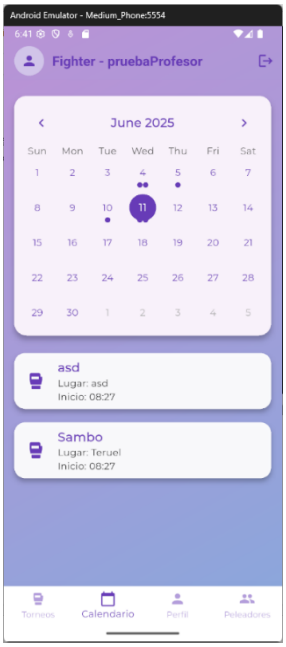
Al abrir la app aparecerá el siguiente menú para iniciar sesión o registrarse en caso de no tener una cuenta.



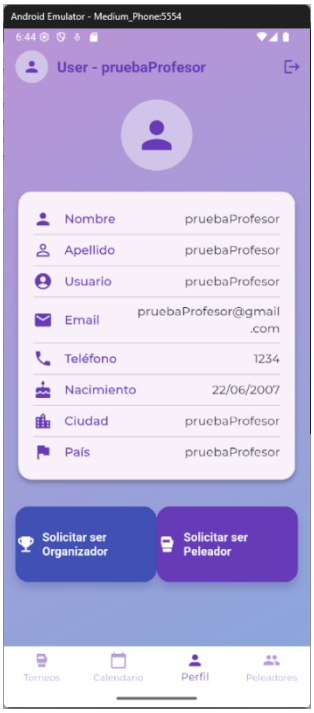
Después aparecerá una ventana que es el scroll panel de los torneos disponibles en este momento, aquellos que no han terminado. Si se clica se pueden ver los combates que se están realizando (como se ve en la siguiente figura), desde la misma ventana se puede realizar la inscripción a los torneos.



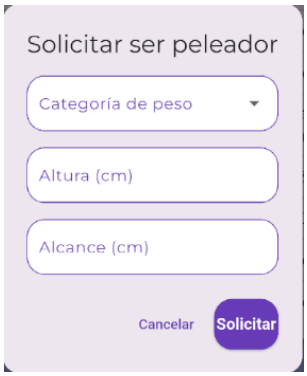
La segunda pestaña de la app es un calendario en donde aparecen los diferentes torneos, al clicar en un torneo aparece el menú anterior.



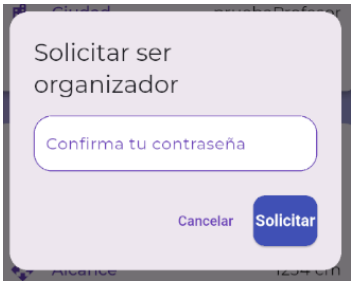
La tercera pestaña corresponde al perfil de cada usuario. Al crear un perfil, se asignará el rol “User”, pero se puede cambiar a “Organizador” y a “Peleador”.



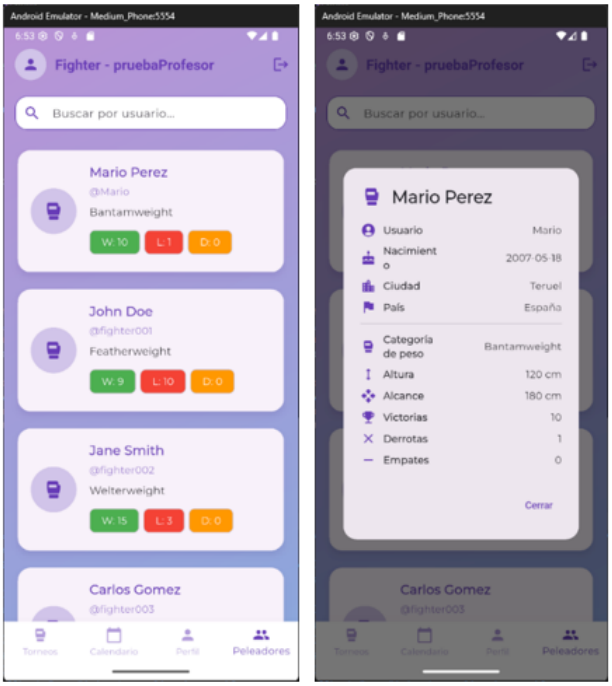
Al registrarse como “Peleador”, se tendrá que rellenar un formulario con los diferentes datos del peleador (categoría de peso, altura y alcance), apareciendo en el menú de usuario otra card con los detalles del peleador, también podrá inscribirse en los torneos.



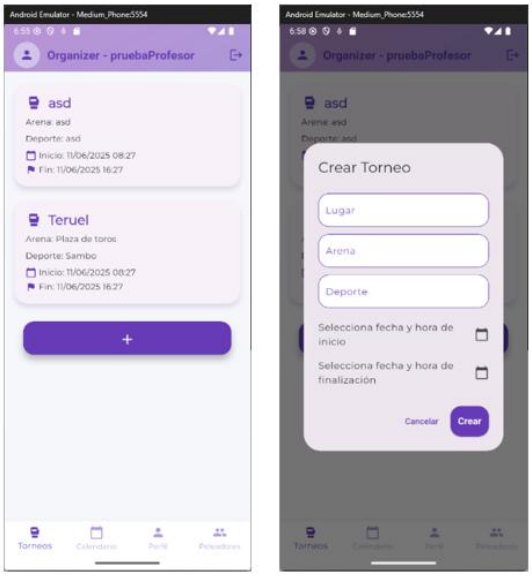
Si se registra como “Organizador”, aparecerá una ventana en la cual se debe introducir la contraseña de usuario, con este rol se pueden organizar torneos.



La cuarta pestaña corresponde a la lista de todos los peleadores de la base de datos (todos los que se hayan registrado en la app). En esta ventana se pueden visualizar todos los peleadores y sus estadísticas (combates ganados y perdidos, así como sus detalles de peleador), tal y como se muestra en la figura.



Por último, cabe destacar que, desde la App, los “Organizadores” únicamente tienen acceso a crear torneos, tal y como se muestra en la figura. En cambio, desde la Web sí que pueden organizar las peleas de cada torneo, así como crear torneos.



5.3. Resultados obtenidos y conclusiones

Se ha desarrollado una app funcional, rápida y estética que cumple con todas las funciones para organizar eventos de artes marciales. Es una aplicación muy escalable, debido a que actualmente está diseñada para realizar un torneo de cualquier arte marcial o deporte orientado a los combates. Esto conlleva a una problemática actual de la app y es que cada disciplina marcial tiene unas reglas, categorías de peso y normativa deportiva diferente, pudiendo apuntarse peleadores de diferentes disciplinas (por ejemplo, Judo y Muay Thai) al mismo torneo. Esto se debe a que actualmente no existe una opción en la app para indicar el deporte que realiza cada usuario, por lo que una vía de mejora sería añadir formularios específicos por disciplina a la hora de inscribirse en los torneos (como categoría de peso), eliminando los propios datos de peleador. Esto evitaría la incompatibilidad entre disciplinas, así como una gestión precisa de los torneos.

A futuro, se plantea añadir información detallada de los combates (número de asaltos, las tarjetas de puntos de los jueces, etc.), lo cual enriquecería la experiencia de los organizadores y participantes.

En conclusión, la aplicación cumple con su cometido inicial y presenta una arquitectura que facilita su escalabilidad. Con las mejoras propuestas, podría convertirse en una herramienta para la gestión profesional de torneos de artes marciales y deportes de combate.

5.4. Diario de bitácora

Durante los primeros días me enfoqué en diseñar la estructura y definir las entidades básicas de la aplicación. A medida que avanzaba, comenzaron a surgir problemas, especialmente relacionados con la base de datos. Uno de los principales problemas fue que, al modificar elementos en SQL Server, Visual Studio no siempre detectaba correctamente los cambios. Esto me llevó a pasar varios días resolviendo una relación que no funcionaba como debía.

Intenté implementar JWT para la autenticación, ya que al principio parecía una tarea sencilla. Sin embargo, con el tiempo descubrí que generaba más complicaciones que beneficios en esta etapa del desarrollo. Por lo que decidí dejarlo sin usar por el momento, aunque conservé el código por si decido implementarlo más adelante.

Con el paso del tiempo se me fueron ocurriendo nuevas ideas para la aplicación, pero muchas de ellas produjeron más errores. También surgieron problemas al buscar nombres de usuario en la web, fallos en los botones, en la autenticación.

Al realizar pruebas con la página web, noté que estaba llamando directamente a los métodos internos de la API, lo cual impedía desacoplarla adecuadamente del resto del sistema. Esto me obligó a rehacer varias partes para que la web se comunicara correctamente mediante peticiones HTTP a la API, como debería haber sido desde el principio.

Poco a poco, y a medida que el desarrollo avanzaba, los problemas comenzaron a reducirse. Me di cuenta de que muchos errores provenían del hecho de que mis clases contenían solo la información estrictamente necesaria para guardarse en la base de datos. Esto dificultaba la implementación de interfaces, ya que, por ejemplo, obtener el nombre de un usuario implicaba navegar por varias clases. Finalmente, opté por añadir información a los DTO, lo cual simplificó mucho el desarrollo y mejoró la organización del código.

6. Bibliografía

SQL Server. https://www.microsoft.com/es-es/sql-server

C#. https://learn.microsoft.com/es-es/dotnet/csharp

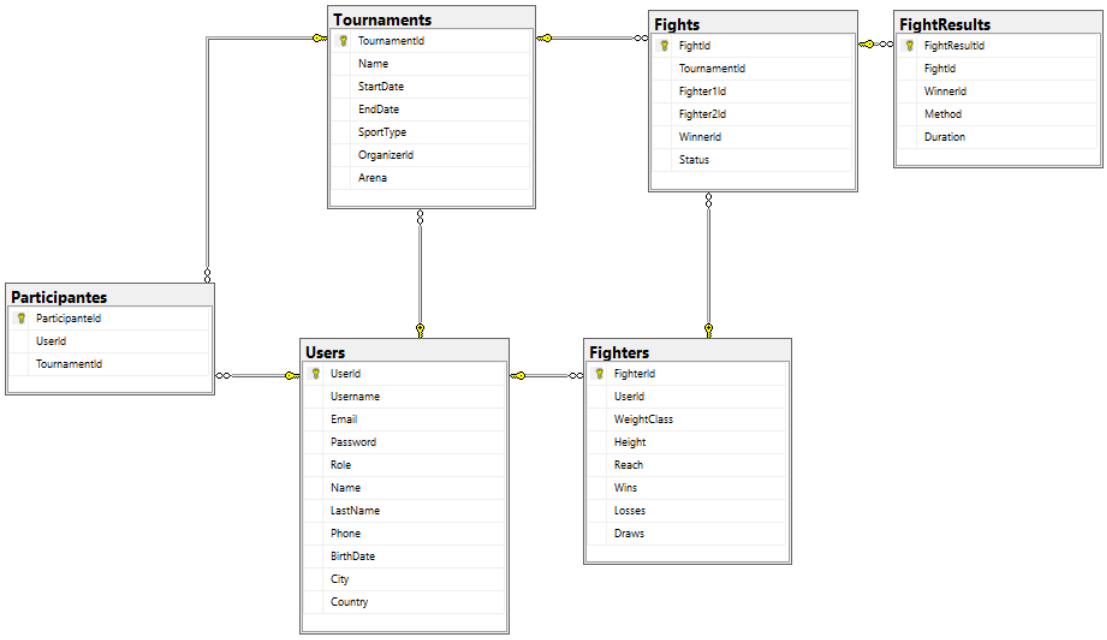
ASP.NET Core. https://learn.microsoft.com/en-us/aspnet/core/?view=aspnetcore-8.0

Flutter. https://docs.flutter.dev/

Razor. https://learn.microsoft.com/en-us/aspnet/core/razor-pages/?view=aspnetcore-9.0&tabs=visual-studio

7. Anexos

Anexo I. Diagrama entidad relación



Anexo II. AppSettings y LaunchSettings de la API

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",

  "ConnectionStrings": {
    "SQLServerConnection": "Server=(LocalDB)\\MSSQLLocalDB;Database=TFC-DB2;Integrated Security=True;TrustServerCertificate=True;"
  },
  "Jwt": {
    "Key": "a-string-secret-at-least-256-bits-long",
    "Issuer": "http://localhost:5263",
    "Audience": "http://localhost:5263",
    "ExpireMinutes": "43200"
  }
}
```

```
{
  "$schema": "http://json.schemastore.org/launchsettings.json",
  "iisSettings": {
    "windowsAuthentication": false,
    "anonymousAuthentication": true,
    "iisExpress": {
      "applicationUrl": "http://localhost:5263",
      "sslPort": 0
    }
  },
  "profiles": {
    "http": {
      "commandName": "Project",
      "dotnetRunMessages": true,
      "launchBrowser": false,
      "launchUrl": "swagger",
      "applicationUrl": "http://localhost:5263",
      "environmentVariables": {
        "ASPNETCORE_ENVIRONMENT": "Development"
      }
    }
  }
}
```

Anexo III. AppSettings y LaunchSettings de la Web

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "ConnectionStrings": {
    "SQLServerConnection": "Server=(LocalDB)\\MSSQLLocalDB;Database=TFC-DB2;Integrated Security=True;TrustServerCertificate=True;"
  },
  "AllowedHosts": "*"
}
```

```
{
  "$schema": "http://json.schemastore.org/launchsettings.json",
  "iisSettings": {
    "windowsAuthentication": false,
    "anonymousAuthentication": true,
    "iisExpress": {
      "applicationUrl": "http://localhost:54988",
      "sslPort": 44371
    }
  },
  "profiles": {
    "http": {
      "commandName": "Project",
      "dotnetRunMessages": true,
      "launchBrowser": true,
      "applicationUrl": "http://*:5157",
      "environmentVariables": {
        "ASPNETCORE_ENVIRONMENT": "Development"
      }
    }
  }
}
```

Anexo IV. Configuración Ruta Web para llamar a la API en Program.cs

```
builder.Services.AddHttpClient("Api", client =>
{
    client.BaseAddress = new Uri("http://localhost:5263/");
});
```