Desarrollo de aplicación híbrida multiplataforma para gestión de planificación deportiva de natación.

Memoria de Proyecto Final de Máster

Máster en Ingeniería Informática

Área Desarrollo de aplicaciones Web

Autor: David Geovanny Béjar Cáceres

Consultor: Joan Giner Miguelez

Profesor: Joan Giner Miguelez

05 de enero de 2021

Abstract

En este trabajo se propone el desarrollo de una app híbrida multiplataforma, que puede ser ejecutada tanto en dispositivos móviles Android/iOS así como en entornos de escritorio en forma de PWA (Progressive Web App) [1], y estará apoyada en un back-end de Firebase [2] que gestionará las colecciones de datos, gestión de cuentas y permitirá sincronizar las cuentas entre varios dispositivos en tiempo real, además compartir fácilmente rutinas de entrenamiento o unir a un grupo común entre varios deportistas.

El presente trabajo se centra en un contexto deportivo, donde muchas veces los entrenadores requieren tener planificación deportiva e ir registrando todo esto en diferentes plataformas o métodos, unos escritos que son susceptibles a pérdidas de documentos, o de tecnologías que la mayoría de veces no son tan específicas en cuanto a detalles que puede tener un entrenamiento de natación como son los diferentes estilos, tamaño de la piscina (olímpica de 50 metros o semiolímpica de 25, u otros en aguas abiertas) y otros particulares que no se consideran la mayoría de veces en otras alternativas.

Abstract (english version)

This work proposes the development of an Hybrid-Web application, targeting multiple platforms, been able to execute in mobile platforms as Android or iOS, but also in desktop computers using the Web as a PWA (Progressive Web App) [1], the application will have a data layer on top of Firebase Cloud Firestore [2] as back-end, allowing the data to be updated in real-time between multiple devices of an user account, and also allowing to share groups to many athletes so they can have reading permissions and check past and future workout sessions.

This project is focused on a sports context, to allow a coach to manage it´s plan for his group of swimmers by adding data in different platforms instead of handwriting material, susceptible to loss of documents, or technologies that most of the time are not so specific in terms of details that a swimming training can have, such as the different swimming strokes, size of the pool (50-meter Olympic or semi-Olympic of 25, or others in open water) or others not considered by alternative software.

Índice

Figuras y tablas

Índice de figuras

Índice de tablas

1. Introducción/Prefacio

Se pretende desarrollar una aplicación multiplataforma, enfocada a la planificación de entrenamiento deportivo de natación, el desarrollo de esta aplicación permitirá gestionar a los entrenadores las sesiones de entrenamiento, guardar registro de las sesiones para los diferentes grupos de deportistas y visualizarlos en sus diferentes etapas de planificación. Los deportistas podrán visualizar toda la planificación de su grupo de entrenamiento e ir marcando el cumplimiento efectivo de las rutinas.

En la actualidad existe varias plataformas para escoger como plataforma de despliegue de software, cada una con su stack propio de desarrollo, desde lenguajes de programación, entornos integrados de desarrollo IDE, hasta arquitecturas de procesador o tamaños de pantalla, sin embargo, cada vez es más común que en equipos pequeños de trabajo o incluso desarrolladores individuales escojan tecnologías que permitan reutilizar el código para varias plataformas, lo que busca es llegar a WORA (Write once, run anywhere) [3], así minimizando la cantidad de trabajo a comparación de dividir el desarrollo en múltiples equipos de desarrollo, o la falta de personas capacitado en desarrollo nativo en cada plataforma, lo que en equipos pequeños de desarrollo se torna imposible mantener varias versiones para múltiples plataformas.

El desarrollo plataforma no es algo nuevo, ya desde hace muchos años encontramos lenguajes de programación que pueden ser compilados para varios sistemas, normalmente C/C++ o Java que corren sobre la JVM (Java Virtual Machine), o algunos más modernos como Kotlin [4] y JavaScript. Pero por lo general el software cliente Front-end ha sido muy dependiente de las tecnologías para crear interfaces gráficas en cada plataforma. En un inicio el desarrollo multiplataforma tenía los problemas de acceso a funcionalidades nativas en cada dispositivo, es algo que ya se ha ido superando para tener una arquitectura nativa o híbrida que permite utilizar APIs del dispositivo o los sensores de estos. Entre las más utilizadas están Flutter de Google [5], React Native de Facebook [6] y Xamarin de Microsoft [7], todas estas tecnologías permiten maximizar el código compartido entre plataformas, con poco código nativo dependiendo de la necesidad.

En este trabajo se propone el desarrollo de una app híbrida multiplataforma, que puede ser ejecutada tanto en dispositivos móviles Android/iOS así como en entornos de escritorio en forma de PWA (Progressive Web App) [1], y estará apoyada en un back-end de Firebase [2] que gestionará las colecciones de datos, gestión de cuentas y permitirá sincronizar las cuentas entre varios dispositivos en tiempo real, además compartir fácilmente rutinas de entrenamiento o unir a un grupo común entre varios deportistas.

1.1 Contexto y justificación del Trabajo

El presente trabajo se centra en un contexto deportivo, donde muchas veces los entrenadores requieren tener planificación deportiva e ir registrando todo esto en diferentes plataformas o métodos, unos escritos que son susceptibles a pérdidas de documentos, o de tecnologías que la mayoría de veces no son tan específicas en cuanto a detalles que puede tener un entrenamiento de natación como son los diferentes estilos, tamaño de la piscina (olímpica de 50 metros o semiolímpica de 25, u otros en aguas abiertas) y otros particulares que no se consideran la mayoría de veces en otras alternativas.

Esta dificultad que muchas veces se encuentran los entrenadores es resuelta de forma nada didáctica al momento de llevar su registro, e incluso más difícil al momento de llevar un seguimiento pasado de los ciclos de entrenamiento con los deportistas, o la visualización de datos en gráficas, cosas que quedan fuera de la capacidad al utilizar medios físicos en papel.

Además, la dificultad con la que muchos entrenamientos suelen llevarse a cabo incluso en remoto, donde un entrenador puede enviar los entrenamientos a un deportista en específico y poder llevar a cabo un seguimiento del cumplimiento.

Muchas de estas dificultades a nivel práctico son acompañadas de otras a nivel técnico, donde tenemos varias plataformas con diferentes tamaños de pantalla, puntos cruciales de experiencia de usuario, donde es más probable que las planificaciones de rutinas diarias sean hechas en pantallas grandes, y dispositivos móviles para la visualización y registro de seguimiento del entrenamiento.

1.2 Objetivos del Trabajo

Brindar una solución multiplataforma para gestionar los entrenamientos de natación en un entorno deportivo, donde se pueda gestionar y visualizar la planificación deportiva por parte de los entrenadores, tomando en cuenta detalles específicos de este deporte.

Funcionalidades de alto nivel:

Entrenador:

- Gestionar grupos de entrenamiento.
- Añadir deportistas a un grupo.

- Crear planificaciones de rutinas de entrenamiento para el grupo.
- Visualizar historial o próximas rutinas de entrenamiento.
- Visualizar estadísticas de rutinas pasadas en cada grupo de entrenamiento.
- Filtros para la visualización de datos de las rutinas.

Deportistas:

- Unirse a un grupo de entrenamiento compartido por el entrenador.
- Visualizar las rutinas pasadas del grupo y futuras.
- Visualizar estadísticas con datos específicos de cada rutina.
- Filtros para la visualización de las rutinas de su grupo de entrenamiento.

Gestión de Cuentas:

- Gestión de entrenadores.
- Gestión de deportistas.

1.3 Enfoque y método seguido

Enfoque:

El enfoque para este trabajo toma en cuenta las dificultades que pueden existir al momento de mantener un registro de la planificación deportiva en la natación, muchas veces las alternativas para deporte son propietarias, o sin haber tenido en mente los detalles específicos particulares de la natación, lo que crea una dificultad para cualquier entrenador o deportista que quiere llevar una planificación o registro de los entrenamientos.

En base a las estas dificultades que pueden existir, se usan medios físicos en papel que no permiten gráficas o aplicar filtros para visualizar el historial de rutinas, además son susceptibles a pérdida, daño o gestión de grandes cantidades de rutinas.

Método:

En cuanto al desarrollo específicamente, se lo lleva basado en casos de uso mediante la implementación de características, el planteamiento de funcionalidades y cumplimiento de los casos de usos se puede evaluar de forma cuantitativa, así podremos saber el éxito o fracaso del proyecto, dificultades, retrasos y limitaciones de tiempo, recursos o técnicos.

Por otro lado, en el apartado de Experiencia de Usuario UX, diseños, colores y calidad de software, es algo que se sigue un método más cualitativo, esto por ser apartados difíciles de medir o cuantificar, muchos de

ellos se pueden tomar como las sensaciones que se pueden resumir en términos de usabilidad como describen Suryn y Seffah (2003) [8]:

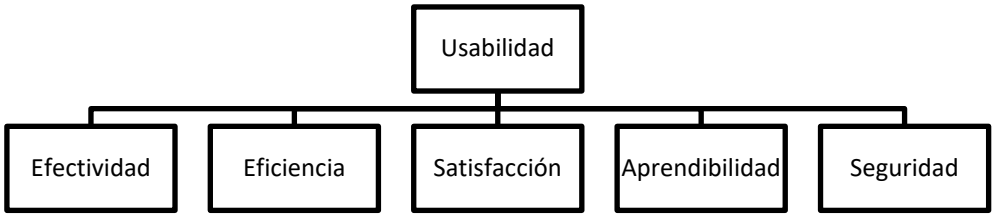


Figura 1: Modelo consolidado de usabilidad [8]

1.4 Planificación del Trabajo

El trabajo se realizará en 4 grandes fases:

- Definición del trabajo y estudio inicial (PEC 1).
- Investigación marco teórico, librerías a usar e inicio del proyecto (PEC 2).
- Fase principal de desarrollo y documentación (PEC 3).
- Entrega final del proyecto, documentación y presentación (PEC 4)

Todas las fases descritas anteriormente corresponden con varias subtareas que se detallan a continuación en el diagrama de Gantt, junto con los tiempos estimados por tarea e interdependencias:

Diagrama de Gantt del proyecto:

ID	Ta Mc	Task Name	Duration	Start	Finish	Predecessors
1		**1.0 PEC1 Definición formal del proyecto y la pauta de trabajo a seguir.**	5.13 days	Wed 23-09-20	Tue 29-09-20	
2		**2.0 PEC 2 Investigación, marco teórico, inicio de proyecto.**	**21.13 days**	**Wed 30-09-20**	**Wed 28-10-20**	**1**
3		2.1 Investigación y marco teórico.	4 days	Wed 30-09-20	Tue 06-10-20	1
4		2.2 Metodología de trabajo.	2 days	Tue 06-10-20	Thu 08-10-20	3
5		2.3 Iniciar documentación.	20 days	Thu 01-10-20	Wed 28-10-2(	1
6		2.4 Herramientas, librerías, frameworks a utilizar.	4 days	Thu 08-10-20	Wed 14-10-20	4
7		2.5 Arrancar el desarrollo del proyecto y alcanzar los primeros hitos.	5 days	Wed 14-10-20	Wed 21-10-20	6
8		2.6 Revisar objetivos y planificación para descubrir errores y aplicar cambios o mejoras.	1 day	Wed 21-10-20	Thu 22-10-20	7
9		2.7 Diagramas UML.	2 days	Thu 22-10-20	Mon 26-10-2(	8
10		2.8 Usabilidad UX.	3 days	Wed 21-10-2(	Mon 26-10-2(	7
11		2.9 Definir Arquitectura, capa cliente, servidor, APIs de terceros.	2 days	Tue 27-10-20	Wed 28-10-20	10
12		**PEC 3 Fase principal de desarrollo, tareas paralelas y primera beta.**	**30.13 days?**	**Thu 29-10-20**	**Wed 09-12-20**	**11**
13		3.1 Desarrollo principal.	**30.13 days?**	**Thu 29-10-20**	**Wed 09-12-2(**	**11**
14		3.1.1 Instalación de paquetes y librerías relevantes.	3 days	Thu 29-10-20	Mon 02-11-20	11
15		3.1.2 Gestionar estado de la aplicación (Redux, Recoil, etc).	3 days	Tue 03-11-20	Thu 05-11-20	14,11
16		3.1.3 Configuración esqueleto de componentes en movil y escritorio.	25.25 days	Tue 03-11-20	Sat 05-12-20	14,11
17		3.1.4 Creación de páginas y componentes a usar.	25.25 days	Tue 03-11-20	Sat 05-12-20	14,11
18		3.1.5 Comunicación entre componentes y estado de la aplicación.	22.25 days	Fri 06-11-20	Sat 05-12-20	15,11
19		3.1.6 Conexión y configuración de Firebase al cliente.	22.25 days	Fri 06-11-20	Mon 07-12-20	15,11
20		3.1.7 Estilos, pequeñas mejoras UX.	2 days	Mon 07-12-20	Tue 08-12-20	15,11,17,18
21		**3.2 Continuar con la documentación.**	30.25 days	Thu 29-10-20	Wed 09-12-20	11
22		**3.3 Beta con video funcionando.**	1 day	Wed 09-12-20	Wed 09-12-20	20
23		**4.0 PEC 4 Entrega final de proyecto, documentación y publicación.**	**19 days**	**Thu 10-12-20**	**Tue 05-01-21**	**13,21,22**
24		**4.1 Finalización del proyecto**	**12 days**	**Thu 10-12-20**	**Fri 25-12-20**	
25		4.1.1 Corrección de Bugs.	12 days	Thu 10-12-20	Fri 25-12-20	22
26		4.1.2 Testing Cypress principales características.	12 days	Thu 10-12-20	Fri 25-12-20	22
27		4.1.1 Estilos y UX.	12 days	Thu 10-12-20	Fri 25-12-20	22
28		4.1 Terminar la documentación.	3 days	Mon 28-12-20	Wed 30-12-20	24
29		4.2 Revisar documentación por citas, errores, ortografía, formato.	2 days	Thu 31-12-20	Fri 01-01-21	28,24
30		4.4 Elaborar presentación escrita-visual.	2 days	Mon 04-01-21	Tue 05-01-21	29

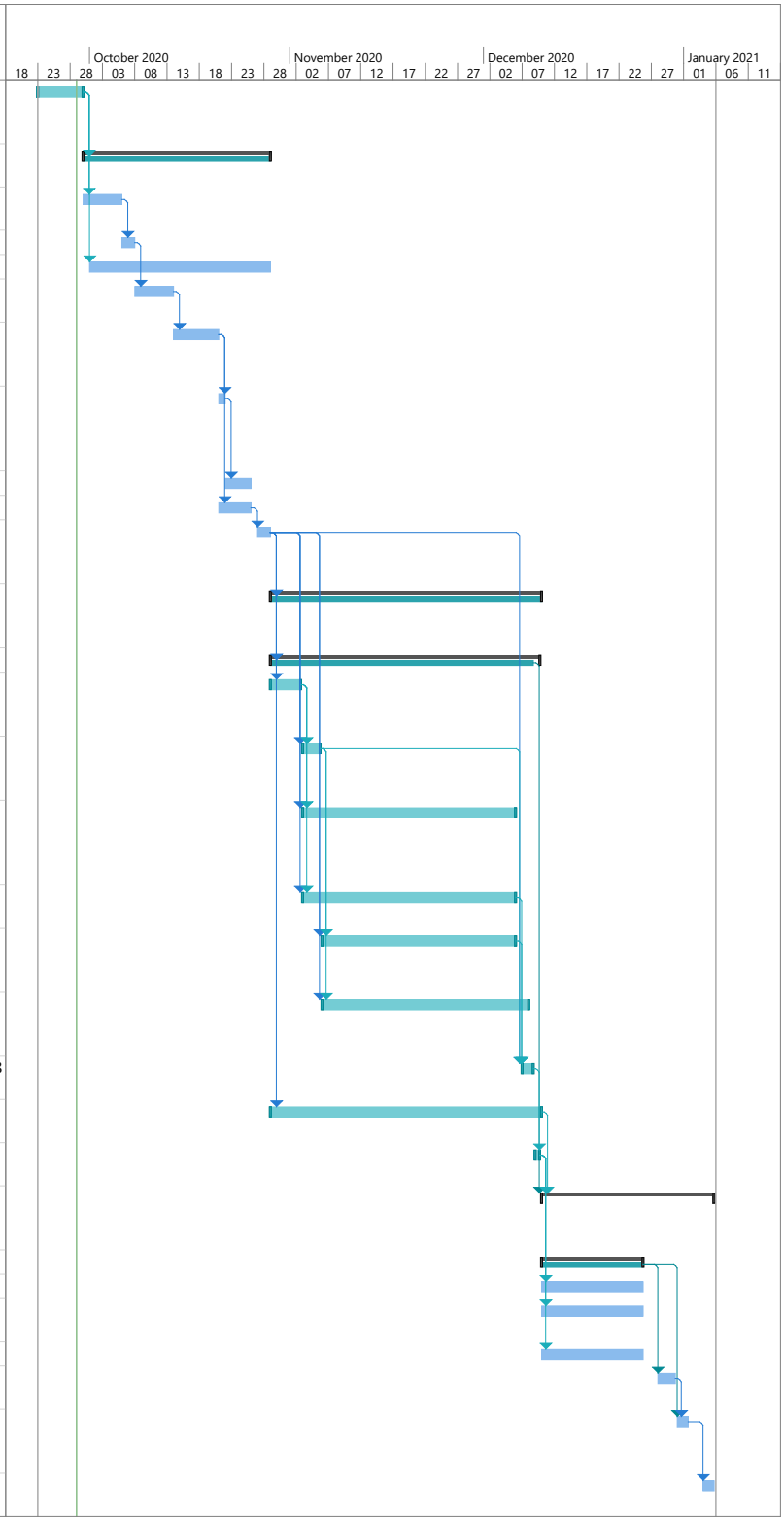


Figura 2: Diagrama de Gantt con la planificación del proyecto.

2. Objetivos

3.1 Principales

- Crear una aplicación híbrida multiplataforma.
- Gestionar de grupos de entrenamiento, deportistas, y sesiones desde la aplicación por parte del entrenador.
- Permitir la suscrición a un grupo de entrenamiento para revisar los entrenamientos en modo lectura.

3.2 Secundarios

- Permitir a entrenadores y nadadores visualizar las sesiones de su grupo de entrenamiento y las estadísticas del grupo.
- Adaptar la aplicación para la interacción de usuario en varias plataformas con sus tamaños de pantalla.

3. Marco teórico

4.1 Antecedentes

4.1.1 Inicios desarrollo multiplataforma

Desde el inicio del software para diferentes plataformas, ha venido limitado por varios motivos: arquitectura de hardware donde existían existían varias arquitecturas del procesador, conocidos también como set de instrucciones, entre ellos las más comunes el IA-32 (Intel Architecture de 32 bits, conocido como i386) [9], y la arquitectura x86-64 [10], ambos capaces de ejecutar varios sistemas operativos en equipos de propósito general, entre otras arquitecturas para propósitos especializados.

Además, a las diferentes plataformas de hardware, existían otras plataformas de software, la proliferación de sistemas operativos para computadores personales, dispositivos móviles, mainframes en empresas o consolas de juegos, etc. Cada una de estas plataformas de software suponían desafíos a la forma de compilar a código binario y ejecutarlo, en los años 70s ya se empezó con los primeros intentos e ideas de facilitar todo el proceso de ejecutar en software en una variedad de plataformas tanto de hardware como de software.

Es a principios de los 70s donde surge el primer intento por parte de la Universidad de California desarrollar el sistema UCSD Pascal utilizando el lenguaje interpretado p-pascal, portable y altamente independiente de sistema operativo[11]. Aunque p-pascal tuvo varias controversias fue inspiración para futuras plataformas y lenguajes de programación, inspirando al Java casi dos décadas más tarde.

En los años 70 y 80 a nivel de lenguajes de programación, donde surge un gran salto en el desarrollo multiplataforma, con el nacimiento del lenguaje C que a futuro daría paso C++, C dio paso al lenguaje de propósito general más popular por décadas, impulsaba el diseño de desarrollo multi-plataforma, al facilitar el desarrollo desde el punto de vista del programador, y tener compiladores disponibles en casi todas las plataformas de software y hardware, daba la posibilidad de generar los binarios para todas estas variedades de plataformas, incluso dando como resultado la estandarización del lenguaje C++ por la Organización Internacional de Estandarización (ISO)[12]. En la actualidad se sigue usando C/C++ en sistemas operativos, software de alto rendimiento, software científico, computación empotrada, desarrollo de videojuegos multiplataforma, e incluso varios sistemas actuales son capaces de ejecutar código C/C++ mediante llamadas e interfaces, permitiendo así compartir software entre varias plataformas.

Es en los años 90 es cuando surge la plataforma de software Java, fue un hito en el mundo de software, la máquina virtual Java era JVM es la encargada de correr el Java Byte Code, mientras la JVM provee de abstracción completa del hardware o sistema operativo, desde memoria hasta procesadores.

La plataforma Java es un conjunto de especificaciones y software Java desarrollado por James Gosling para Sun Microsystems actualmente adquirida por Oracle. Esta plataforma incluye la JVM, un compilador y un set de librerías, no tiene especificación de hardware o sistema operativo, con la visión de correr en todos estos sistemas de forma idéntica. Esta máquina virtual Java ejecuta los programas en Java Bytecode [13]. Originalmente este Java Bytecode se generaba a partir del lenguaje de programación Java, sin embargo, han ido surgiendo otros lenguajes que pueden compilarse para la JVM como Kotlin, Groovy, Scala, etc. EL gran auge de Java como plataforma de desarrollo permitió sentar base para el desarrollo con interfaces gráficas multiplataforma, algo que hasta el momento era muy limitado y difícil de conseguir entre varios sistemas operativos.

4.2 Frameworks multi-plataforma

Para el desarrollo de aplicaciones multi-plataforma existen actualmente varias alternativas, cada una con sus ventajas y desventajas al escoger alguna de ellas, usualmente se selecciona una de ellas en base a los siguientes criterios:

- Plataformas para la que está pensado el software correr.
- Experiencia del equipo de desarrolladores con el Framework.
- Experiencia del equipo de desarrolladores con el lenguaje.
- Soporte y estabilidad.

4.2.1 Flutter

Flutter es un framework creado por Google para el desarrollo de aplicaciones que compila a código nativo a múltiples plataformas, pensado en el desarrollo ágil y alto rendimiento de la aplicación. Flutter renderiza el contenido conocido como “widgets” directamente mediante código C/C++ con su motor de renderizado 2D, que corre por debajo, se encarga la composición, gestos, animaciones[5].

Flutter utiliza el lenguaje de programación Dart[14], un lenguaje que corre tanto en el Front-end como en el back-end, con capacidades de compilación AOT (ahead-of-time)[15].

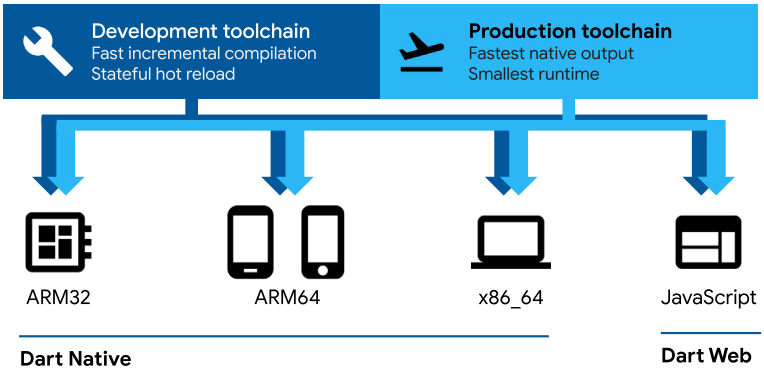


Figura 3: Dart compila de forma nativa a varias plataformas[15].

Flutter permite ejecutar en los siguientes sistemas operativos:

- Android
- iOS
- Mac OS
- Web (sin versión estable)
- Linux (sin versión estable)
- Windows (sin versión estable)
- Google Fuchsia (sin versión estable)

4.2.2 React Native

React Native es una librería de desarrollo híbrido creado por Facebook basado en tecnologías web con el paradigma “learn once, write anywhere”. Proporciona un conjunto básico de componentes nativos independientes de la plataforma, como *View*, *Text* e *Image, TextInput, ScrollView, StyleSheet*, que se asignan directamente a los componentes básicos de la interfaz de usuario nativa de la plataforma, además de muchos otros componentes específicos de la plataforma[6], [16].

Una de las desventajas de React Native al momento de utilizarlo es la personalización de los componentes, debido a que no corre sobre un entorno de navegador, CSS, HTML y DOM estándar no pueden aplicarse, React Native brinda la posibilidad de usar una herramienta parecida a CSS. Además no todas las librerías Web y React son compatibles, es decir muchas tendrán que ser portadas a esta librería con grandes cambios para su compatibilidad[17].

Figura 4: Componentes de React Native en la aplicación[16].

React Native se puede ejecutar en las siguientes plataformas:

- Android
- iOS
- Web (no soporte oficial)

4.2.3 Xamarin

Xamarin es una plataforma de aplicaciones de código abierto de Microsoft para crear aplicaciones iOS y Android modernas y de alto rendimiento con C # y .NET. .NET nos provee varios componentes entre ellos el lenguaje de programación C#, librerías para funciones básicas para trabajar con datos, archivos I/O, además editores y herramientas de compilación para Windows, Linux, MacOS y Docker. [18]. Xamarin extienda las funcionalidades de .NET con las siguientes características:

- Framework base para acceder a características nativas de dispositivos.
- Lenguaje XAML como lenguaje de diseño para construir apps dinámicas con C#.
- Librerías para plataformas específicas, que incluyen acceso a APIs de Google, Apple, Facebook y más.
- Extensiones para los editores de texto, para proporcionar resaltado de sintaxis, finalización de código, diseñadores y otras funcionalidades específicamente para desarrollar páginas móviles.

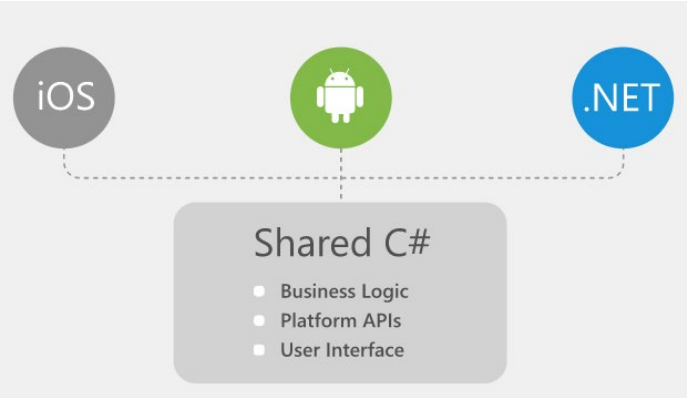


Figura 5: Desarrollo multi-plataforma con Xamarin y C#[18].

Xamarin también dispone de compilación AOT (ahead-of-time) para que reduzca el tiempo de inicio, aumente el uso compartido de memoria y mejore el rendimiento.

Xamarin mediante Xamarin Forms permiten crear interfaces de usuarios ricas en contenido utilizando el modelo Model-View-View-Model (MVVM). Xamarin.Forms se integra en páginas, diseños y controles para crear y diseñar aplicaciones móviles desde una única API que es altamente extensible. Se pueden extender la clase de cualquier control para personalizar su comportamiento o definir sus propios controles, diseños, páginas y celdas[19].

Xamarin puede correr en las siguientes plataformas:

- Android
- iOS
- Windows Apps (no Windows .exe).

Una de las grandes desventajas al usar esta plataforma es la cantidad de código que se comparte en plataformas no es necesariamente el 100%, se requiere que el grupo de desarrolladores tengan conocimiento y experiencia específica en cada una de las plataformas, Microsoft promete que al menos el 75 % de código será multi-plataforma escrito en C#[19], es decir que puede requerir hasta un 25% de código específico en cada plataforma.
Otra gran desventaja es el soporte limitado para Windows, sólo se pueden construir aplicaciones conocidas como Windows UWP, que pueden ser subidas a la tienda de aplicaciones de Microsoft, sin embargo, queda fuera de alcance la compilación como un programa de Windows tradicional con un instalador en su forma .exe o .msi para plataformas x86 o x64. Microsoft tiene el control completo de las distribuciones para Windows en sus tipos de paquetes disponibles App Package (.msix y .appx), App Bundle (.msixbundle y .appxbundle) o App Package Upload (.msixupload y .appxupload) [20].

También es la alternativa con menor popularidad entre las demás, según el interés de búsquedas en el motor de búsqueda de Google, se puede apreciar que Xamarin siempre ha estado en un interés muy inferior a comparación con las demás, y en declive en estos últimos 5 años:

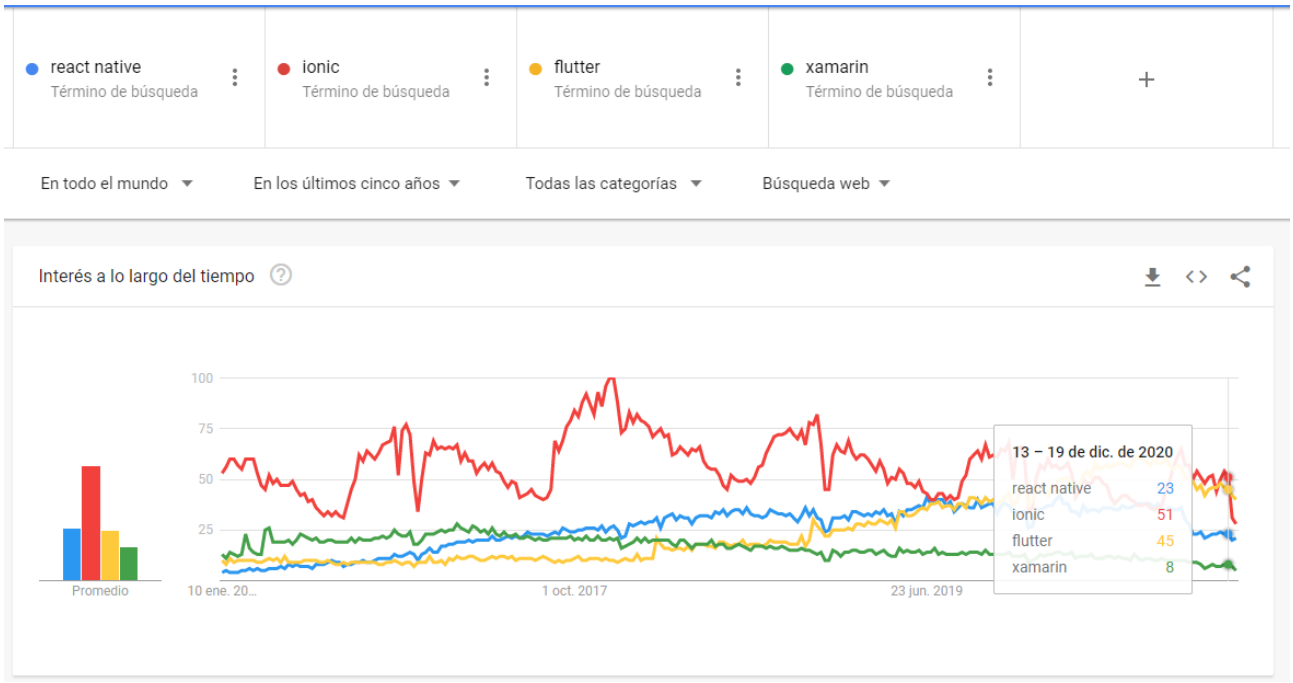


Figura 6: Interés de búsqueda en Google Trends durante los últimos 5 años.

4.2.4 Ionic React/Angular/Vue + Capacitor

Es un kit de herramientas de interfaz de usuario móvil de código abierto para crear experiencias de aplicaciones web y nativas multiplataforma de alta calidad. Mediante un mismo código, que se ejecuta en todas partes con JavaScript y tecnologías web[21]. Ionic dispone de componentes pre diseñados para varias plataformas, se puede entender como una librería de componentes agnósticos del framework o librería de desarrollo Web, es decir, el desarrollador podrá desarrollar en Angular, React o Vue como normalmente lo haría, con la ventaja del catálogo de herramientas de Ionic.

Gracias a Ionic se puede desarrollar para varias mediante el paradigma “write once, run anywhere”, el mismo código correrá sobre todas las plataformas dentro de un navegador incrustado[17]. Esto da una gran ventaja a pequeñas Startups en el mundo de la tecnología, permite un desarrollo ágil aprovechando la experiencia de cualquier equipo de desarrollo web, sin requerir conocimiento del desarrollo nativo de cada plataforma, todo se construye mediante JavaScript/TypeScript, HTML y CSS.

Ionic corre en las siguientes plataformas:

- Android
- iOS
- Electron
- Progressive Web App
- Web

Sin embargo, para acceder a las funcionalidades nativas de cada dispositivo requiere de un puente que lo conecte con el desarrollo web, es aquí donde entra Capacitor a escena.
Capacitor es un tiempo de ejecución nativo multiplataforma que facilita la creación de aplicaciones web modernas que se ejecutan de forma nativa en iOS, Android y la Web. Capacitor, que representa la próxima evolución de las aplicaciones híbridas, crea aplicaciones nativas web, proporcionando un enfoque moderno de contenedor nativo para los equipos que desean crear primero la web sin sacrificar el acceso completo a los SDK nativos cuando lo necesitan[22].

Capacitor se considera un sucesor de Apache Cordova [23] y PhoneGap [24], para acceder a las funcionalidades nativas de cada plataforma mediante el uso de Plugins (Complementos) para Swift en iOS, Java/Kotlin en Android y JavaScript para la Web. Además es retro compatible con muchos de los Plugins existentes de Cordova[25].

Además, Capacitor nace de la misma compañía a cargo de Ionic, por lo que su compatibilidad siempre dará confianza al equipo de desarrolladores, sin embargo, Capacitor es agnóstico, por lo que se puede incluir en cualquier otro framework o librería web.

Entre las varias funcionalidades nativas con Plugins de Capacitor tenemos:

- API de procesos en segundo plano[26].
- Cámara[27].
- Acceso al portapapeles[28].
- Información del dispositivo [29].
- Acceso al sistema de archivos del dispositivo[30].
- Geolocalización [31].
- API teclado virtual [32].
- Notificaciones [33].
- Revisión de permisos [34].
- Splash Screen al iniciar la app [35].
- Almacenamiento nativo clave/valor en cada plataforma [36].

4.3 Ventajas del paradigma “write once, run everywhere” en Ionic

Ionic React hace uso de este paradigma, permite tener la misma interfaz de usuario entre las diferentes plataformas, utilizando el diseño web responsivo, CSS, y herramientas de detección de plataforma para adaptar la interfaz de usuario a Android o iOS.

El uso de tecnologías web permite personalizar y reutilizar miles de librerías en nuestras aplicaciones, a diferencia del paradigma “learn once, write anywhere”, donde React Native requiere que el desarrollador aprenda un grupo de conceptos, y luego construir para plataformas específicas en Android e iOS, añadiendo complejidad y tiempo requerido para el tiempo de desarrollo, diferenciando entre híbrido-web (Ionic) e Híbrido-nativo (React Native)[17].

Entre algunas ventajas al utilizar Ionic tenemos [17]:

- Consistencia del diseño en todas las plataformas.
- Portabilidad, mismo código HTML, Javascript, CSS puede ser usado por otros Frameworks y plataformas como Electron.
- Personalización, en React con Ionic se puede acceder al DOM (virtual-DOM) y todas las funcionalidades del estándar Web.
- Estabilidad, gracias a los estándares web durante años, la plataforma en la que se basa es muy sólida, gran cantidad de documentación, asegurando estabilidad y garantizando que el código escrito corra sin problemas a futuro.

4. Metodología

4.1 Metodología de desarrollo SCRUM:

Metodologías ágiles por sprints, de modo que cada sprint contiene una Release donde se lanzan mejoras del proyecto. Esto produce una mejora constante del producto. El proyecto con SCRUM se adapta al problema en cuestión, se define y con frecuencia cambia en tiempo real.

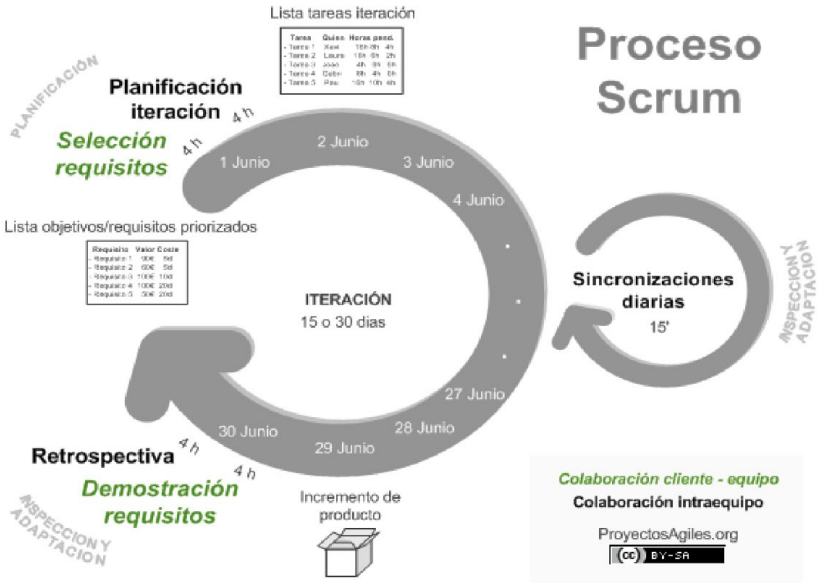


Figura 7: Metodología SCRUM por iteraciones [37].

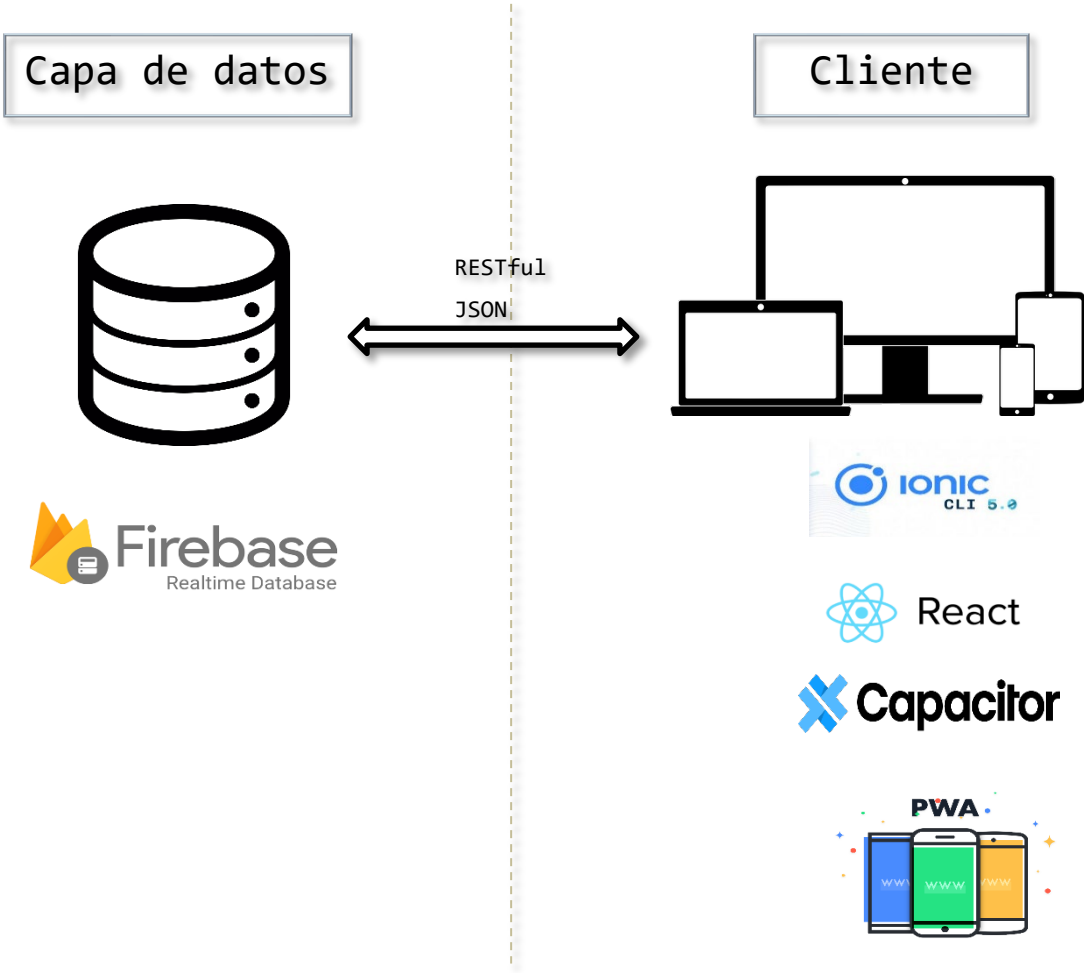
El ciclo de desarrollo empieza por la planificación de la iteración, en este caso consta de dos pasos:

- Selección de requisitos: se planea los requisitos y la prioridad de los requisitos de la aplicación.
- Planificación de la iteración: se seleccionan los requisitos y se crean tareas para empezar el desarrollo, en este caso las tareas creadas en Jira ayudaban con el proceso de desarrollo, se describe con más detalle las tareas Jira en el capítulo 6.

Luego de resolver la etapa de desarrollo en la iteración, se realizan ajustes pequeños a los requisitos y si es necesario se puede cambiar hasta en un 15% a 20% la estimación de la iteración y las tareas.
Una vez terminado se entra en la etapa retrospectiva, se revisa el cumplimiento de los requisitos, problemas encontrados que puedan dar a otras tareas para nuevas iteraciones y lecciones aprendidas [37].

5. Arquitectura de la aplicación

5.1 Diagrama de arquitectura de la aplicación



5.2 Cliente Front-end

El cliente donde se centra la aplicación hace uso del framework de desarrollo híbrido React Native [6], que junto a la librería Ionic 5[38] permiten gran facilidad para el desarrollo de aplicaciones híbridas y posterior despliegue como PWA.

Como ya se ha mencionado en capítulos anteriores, el desarrollo híbrido no permite aprovechar todas las capacidades del dispositivo, en este caso al hacer uso de tecnologías web para el desarrollo, se requiere conectar a características de los diferentes dispositivos, para ello se hace uso de Capacitator, una API que funciona a medio de conector nativo con la capa del dispositivo, con esto se puede aprovechar de las

funcionalidades nativas de plataformas Android, iOS, PWA, Electron, etc[39]. En la siguiente imagen se puede apreciar cómo se integran las capas en el cliente:

Figura 8: Layers en el cliente híbrido con capacidades nativas mediante Capacitor [40].

5.3 Back-end Firebase

En Firebase se tiene en cuenta la estructura de colecciones y documentos, para que permitan acceder de forma ordenada a cada entrenador, el procesamiento y obtención de estadísticas son realizadas en local, esto debido a que no presentan una carga grande de procesamiento en el Front-end.
En la estructura de los datos dentro de Firebase no tiene una única solución, al tratarse de una base de datos no relacional, también conocida como No-SQL, dependerá de varios factores, no necesariamente la normalización de la base de datos [41]. Se propone el siguiente diseño:

- Colección usuarios, donde cada documento representa un usuario, puede ser de tipo entrenador o nadador, depende el tipo de usuario tendrá permisos para editar, ver o gestionar las otras colecciones desde el cliente.
- Colección nadadores, donde cada documento corresponde a un nadador, estos nadadores están vinculados a un grupo y un entrenador.
- Colección entrenamientos, donde cada documento corresponde a una sesión de entrenamiento,
- Colección grupos, documentos cada grupo de entrenamiento.

Es importante destacar las asociaciones o referencias que existen entre los diferentes campos, pues las sesiones de entrenamiento estarán asociadas a grupos, estos grupos a ciertos entrenadores, y los nadadores estarán asociados a los grupos y entrenadores. En la siguiente figura se aprecia las colecciones en la plataforma de Firebase:

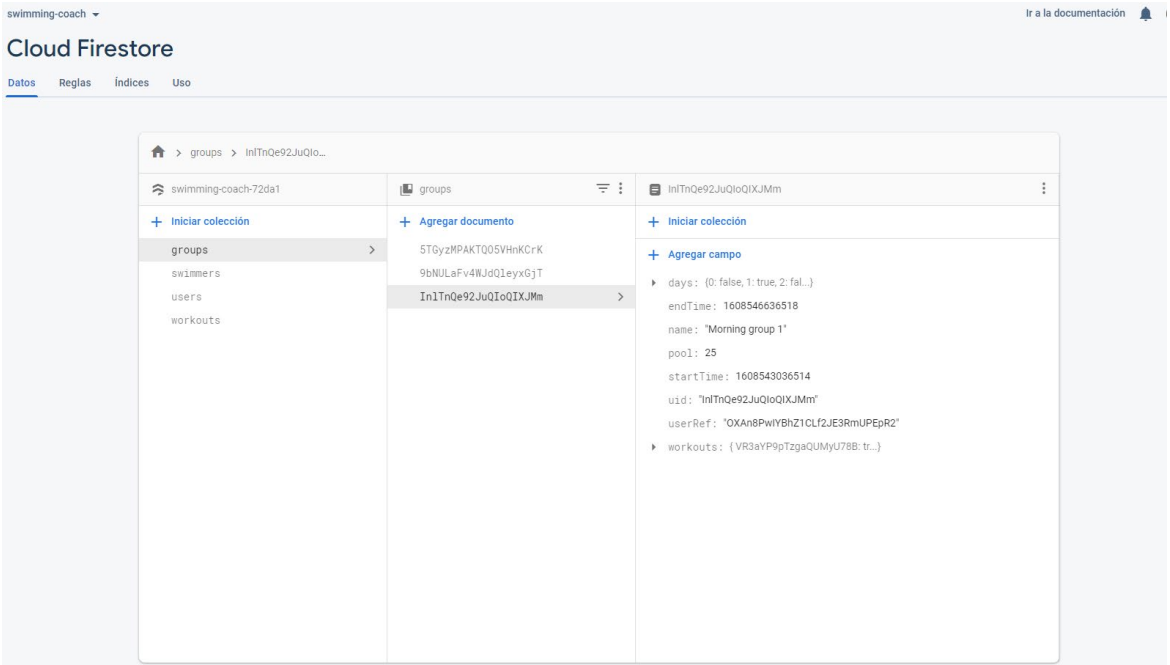


Figura 9: Colecciones y documentos en la plataforma de Firebase.

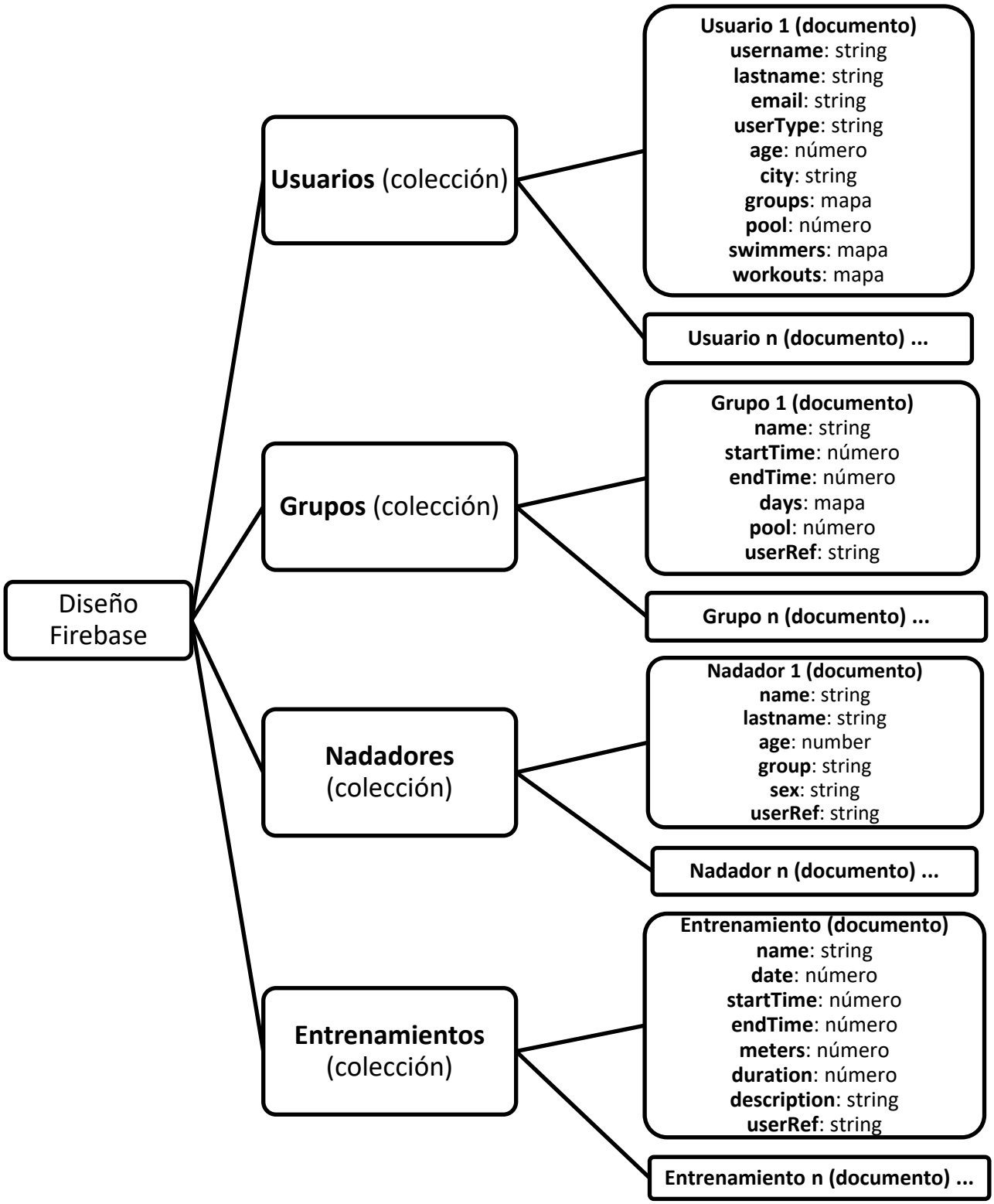


Figura 10: Diseño base de datos en Firebase, colecciones y documentos con sus campos.

6. Planificación y proceso de desarrollo

La planificación del trabajo se lo realizó en 4 grandes fases que corresponden a cada una de las entregas en las 4 PEC. El detalle completo se puede apreciar en la Figura 2: Diagrama de Gantt con la planificación del proyecto. Que se mostró anteriormente en este documento. A nivel de avance del proyecto se pueden definir las siguientes fases:

Tabla 1: Planificación del proyecto por fases e hitos.

Fase	Hitos
Definición del proyecto	• Tema y nombre del proyecto definidos. • Objetivos generales. • Motivación.
Investigación y definición de tecnologías a usar	• Definidas las librerías y herramientas a usar en el desarrollo. • Definir la arquitectura de la aplicación. • Definir la estructura de la base de datos. • Definir los roles de usuarios y permisos.
Desarrollo principal	• Inicio del proyecto Ionic-React. • Añadir librería de estado global de la aplicación Recoil[42]. • Estructura principal de la aplicación, páginas y navegación definida. • Formularios de la aplicación con validación mediante librería de formularios. • Autenticación de la aplicación a componentes web restringidos mediante Firebase. • Diseño e interfaz terminada para dispositivos móviles. • Diseño e interfaz terminada para computadores de escritorio. • Primera beta. • Conexión a la base de datos de Firebase.
Corrección de bugs, test y entrega	• Tests de integración para las funcionalidades principales de la aplicación mediante Cypress. • Bugs encontrados y corregidos en las plataformas de prueba Android OS y Web.

	• Finalización de la documentación.

6.1 Desarrollo por tareas:

El proceso de desarrollo se realiza utilizando SCRUM, en cada sesión de trabajo se dedicó unos minutos a revisar las tareas pendientes, plantear tareas por hacer. Para la gestión de las tareas se ha utilizado Jira [43], gracias a este se pueden crear tarea de forma más organizada, detallar la tarea, tiempo empleado, comentarios, etc. La siguiente figura muestra la lista de tareas en la web del proyecto de Jira:

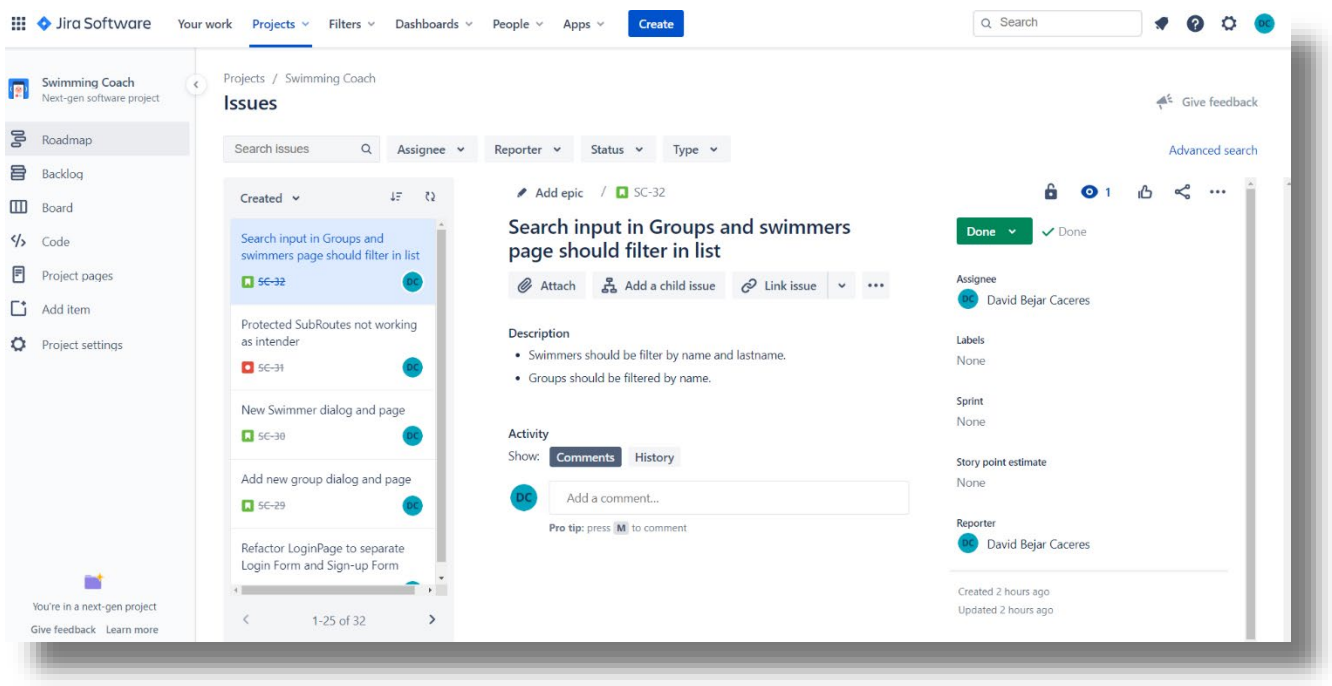


Figura 11: Tareas del proyecto en la web de Jira para este proyecto.

Estas tareas se sincronizan desde el GitKraken. Desde el GitKraken se puede gestionar las tareas, pero también crear ramas desde develop para cada una de las tareas, para las tareas en el proyecto se crea una rama a partir de develop, se realizan los cambios que cumplan los requisitos, una vez terminado se lo prueba y se realiza el “merge” de regreso a develop. En la siguiente figura se aprecia las tareas del proyecto además las ramas que se han ido creando durante el desarrollo:

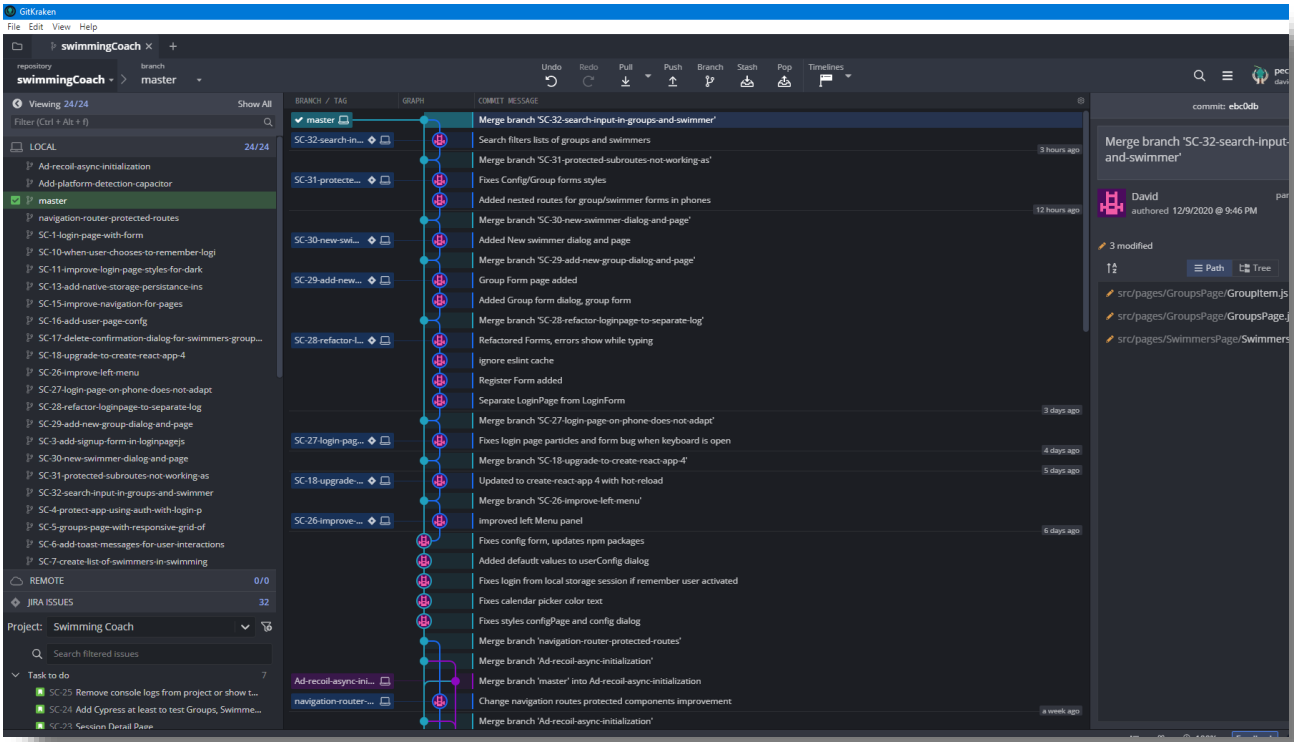


Figura 12: Tareas en GitKraken y desarrollo de las diferentes ramas para cada tarea.

Luego de varias tareas resueltas, en la siguiente sesión de trabajo se revisa el funcionamiento, se detectan bus, y finalmente se lo compila para Android y revisar que los últimos cambios no rompan la versión anteriormente compilada en Android.

Al final de este proyecto se realizaron 48 tareas, cada una de ellas ha sido resuelta, a excepción de la tarea “SC-48”, que queda abierta para limpieza de código y pruebas de mejora continua en caso de detectar bugs a futuro. Estas tareas han sido exportadas desde la plataforma de Jira y están detallas al final de este documento en el Anexo 3.

6.2 Requisitos funcionales de la aplicación:

- Multiplataforma con soporte para Web y Android.
- Autenticación de la aplicación para el acceso a la aplicación.
- El usuario podrá escoger entre dos perfiles, entrenador (COACH) y nadador (SWIMMER).
- Entrenador podrá crear grupos de entrenamiento con datos básicos como tamaño de piscina y horas de entrenamiento.
- Entrenador podrá crear sesiones de entrenamiento para cada grupo de entrenamiento.
- Entrenador podrá añadir nadadores a su lista de nadadores con datos como nombre y edad.

- Filtrar las listas de nadadores, grupos y sesiones de entrenamientos.
- Filtros avanzados para las sesiones de entrenamiento por distancia, duración y últimos semanas o meses.
- Entrenador podrá compartir código del grupo para que un nadador se suscriba al grupo y vea todos los grupos de entrenamiento.
- Nadador podrá suscribirse simplemente con el código del entrenador con permisos de lectura, pero no de escritura ni modificación, sólo podrá terminar su suscripción al grupo.

7. Librerías, APIs y acceso a funcionalidades nativas.

En proyecto híbrido hace uso de múltiples paquetes NPM de terceros, con los cuales se han creado componentes que componen la interfaz. Existen varias otras sub-dependencias dentro de los paquetes y librerías utilizadas que no se detallan en este trabajo. A continuación, están los paquetes NPM y librerías utilizadas en este proyecto:

7.1 Librerías y APIs utilizadas:

Tabla 2: Librerías NPM y APIs utilizadas en el proyecto.

	Nombre	Versión	Breve descripción
1	@ionic/cli[44]	6.12.0	La interfaz de línea de comandos (CLI) de Ionic es la herramienta de referencia para desarrollar aplicaciones Ionic.
2	@capacitor/core @capacitor/android @capacitor/cli [22]	2.4.2	Capacitor es un tiempo de ejecución nativo multiplataforma que facilita la creación de aplicaciones basadas en web que se ejecutan de forma nativa en iOS, Android y la Web.
3	@capacitor-community/react-hooks[45]	0.0.11	Un conjunto de hooks para ayudar a los desarrolladores de Capacitor a utilizar las API nativas de Capacitor.
4	@ionic/pwa-elements [46]	3.0.1	Colección de experiencias de IU predefinidas para API web, como cámara / video, enfocadas en la creación de aplicaciones web progresivas (PWA) que cumplen y superan las experiencias de aplicaciones móviles nativas.
5	@ionic/react @ionic/react-router [47]	5.5.2	Framework de aplicaciones móviles de código abierto que facilita la creación de aplicaciones web nativas y PWA de alta calidad con tecnologías web.
6	@ionic/react-hooks [48]	0.0.8	Soporte de Hooks en proyectos Ionic React.
7	classnames [49]	2.2.6	Una sencilla utilidad de JavaScript para unir condicionalmente classNames.
8	i18next [50]	19.8.3	Framework de internacionalización, extensible y multiplataforma.
9	react-i18next [51]	11.7.3	Implementación de i18next para React y React Native, con soporte para Hooks en componentes funcionales. Esta librería está encargada de la internacionalización dentro del proyecto.
10	ionicons [52]	5.2.3	Set de 1300 íconos en SVG utilizados parte de Ionic Framework.
11	primeicons [53]	4.1.0	Set de íconos para las librerías UI de PrimeFaces y Primereact.
12	primereact [54]	5.0.2	PrimeReact es un rico conjunto de componentes de interfaz de usuario de código abierto para React. En el Proyecto, la mayoría de componentes en formularios son compuestos por PrimeReact gracias a la gran personalización y compatibilidad con React.

13	react, react-dom [55]	16.13.0	Una librería de JavaScript para crear interfaces de usuario. Esta es el framework con el que se ha realizado todo el código del proyecto.
14	react-error-boundary [56]	3.1.0	Proporciona un wapper que puede usar para envolver componentes. Cualquier error de representación en la jerarquía de sus componentes se puede manejar con elegancia. Se usa para mostrar una pantalla de error personalizada y evitar la pantalla blanca con el error por defecto de Reac.
15	react-hook-form [57]	6.13.1	Librería para formularios basado en Hooks, enfocado en el alto rendimiento y, flexible y uso de Hooks en componentes funcionales. Esta librería se ha usado en los formularios del proyecto, se encarga de la validación, errores, y se usan componentes controlados.
16	react-particles-webgl	1.0.10	Una biblioteca de partículas 2D / 3D construida con React, Three.js y WebGL. Se lo utiliza para la animación en la pantalla de Login. Al hacer uso de webgl con la GPU no afecta en el rendimiento.
17	react-router, react-router-dom [58]	5.2.0	Colección de componentes de navegación que se componen de forma declarativa. Se usa para la navegación entre páginas, rutas protegidas a componentes y carga dinámica con lazy loading de páginas.
18	react-scripts	3.4.0	Scripts para el build, arranque y test de proyectos React.
19	react-use [59]	15.3.4	Colección de Hooks para React, muy utilizado a lo largo del proyecto, como para detectar teclas presionadas para disparar acciones, observar tamaños de pantalla para Desktop o dispositivos móviles, temporizadores, etc.
20	recoil	0.0.13	Librería para la gestión de estado de la aplicación, similar a Redux o Saggas. Optimizada para React y Hooks en componentes funcionales.
21	three	0.123.0	Librería 3D con un renderizador WebGL predeterminado. Requerido para las animaciones de la pantalla de Login.
22	@dutchconcepts/capacitor-barcode-scanner[60]	1.1.1	Plugin Capacitor para escanear códigos de barras y códigos QR mediante la cámara del dispositivo.
23	react-qr-code[61]	1.0.5	Genera códigos QR a partir de texto. Utilizado para compartir el código de grupo por parte de un entrenador.
24	firebase[62]	8.2.0	Para acceder a las funcionalidades desde la app cliente.
25	react-firebase-hooks[63, p.]	2.2.0	Librería de Hooks con la cual se realizan las consultas, dando acceso a las colecciones y documentos de la base de datos en Firebase, además en forma de Hooks se encarga de devolver variables con el valor de la base de datos, otro booleano para saber si está cargando el contenido o existen errores de carga.

7.1 Funcionalidades nativas del dispositivo:

Una de las partes más importantes del desarrollo híbrido es la posibilidad de acceder a funcionalidades nativas del dispositivo desde nuestro código, en este caso poder acceder desde JavaScript, gracias a Capacitor y sus múltiples plugins se puede acceder a muchas de estas funcionalidades. Para ello se han utilizado las siguientes características:

7.1.1 Estado del teclado virtual

El estado del teclado virtual mediante el plugin de Capacitor*@capacitor-community/react-hooks*[45] nos devuelve dos variables: isOpen y keyboardHeight. En la aplicación se utiliza para calcular la altura del teclado virtual y renderizar nuevamente el componente de la página de inicio de sesión, gracias a esto se evita que el contenido de la página quede debajo del teclado, y el usuario podrá hacer "Scroll" para visualizar todos los campos del formulario.

El siguiente código muestra el uso del Hook useKeyboardState(), que nos devuelve la altura del teclado y la usa para calcular la altura de la página y renderizar el componente:

swimmingCoach - LoginPage.js

```
1  const {keyboardHeight} = useKeyboardState();
2  const {height} = useWindowSize();
3  const pageFullHeight = useMemo(() => height - keyboardHeight, [height, keyboardHeight]);
4
```

7.1.2 Acceso a cámara

El acceso a la cámara se requiere para escanear los códigos QR que genera el entrenador para compartir con sus nadadores un grupo de entrenamiento, el plugin de Capacitor encargado de esto es @dutchconcepts/capacitor-barcode-scanner[60]. Una vez el código ha sido escaneado, el nadador puede aceptar el grupo compartido para suscribirse y podrá ver los entrenamientos y datos del grupo que va compartiendo el entrenador.

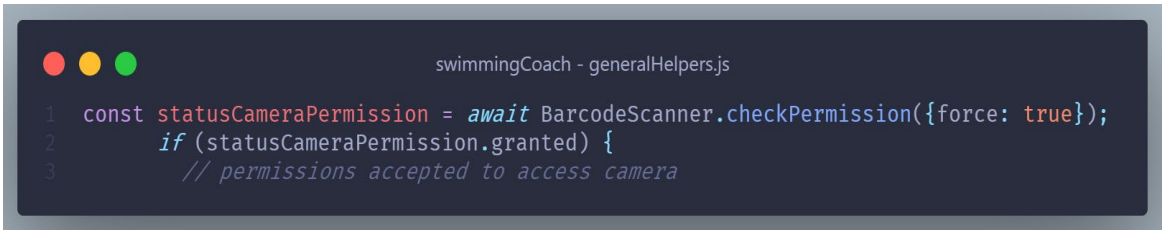
En el siguiente extracto de código podemos ver los siguientes puntos importantes:

- Revisa que la funcionalidad nativa esté disponible en la plataforma en la línea 3.

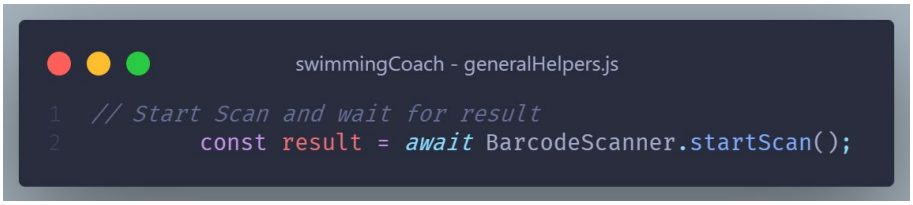
swimmingCoach - generalHelpers.js

```
1  if (Capacitor.isPluginAvailable('BarcodeScanner')) {
2        const {BarcodeScanner} = Plugins;
3        // BarcodeScanner supported in this device
```

- Revisa que los permisos de acceso a cámara hayan sido aceptados por el usuario en la línea 7.



- Finalmente abre la cámara y espera a tener un código QR para leer en la línea 17.



7.1.3 Acceso al almacenamiento nativo del dispositivo

El almacenamiento en el dispositivo se guardan algunos datos al cargar la aplicación, preferencias de usuario, preferencia de idioma, tema de la aplicación. En este caso es importante guardar esta información en el almacenamiento, sin embargo, por defecto lo guarda en el almacenamiento “localStorage” para clave valor, que estamos acostumbrados en la Web, y este almacenamiento en dispositivos móviles es borrado constantemente por el sistema operativo, en Android suele borrarse periódicamente, pero en iOS el comportamiento es mucho más agresivo al momento de borrar este almacenamiento. Este plugin da Capacitor nos permite almacenar en un almacenamiento permanente dentro de los dispositivos móviles, en el caso de Android, usará SharedPreferences y en iOS UserDefault [36]. Capacitor se encarga de seleccionar el almacenamiento adecuado en cada plataforma, dejando al desarrollador simplemente el acceso a la API para almacenar los objetos clave/valor.

A continuación, se aprecia un fragmento de código para recuperar y eliminar del almacenamiento un objeto:

8. Diagramas UML y perfiles de usuario

8.1 Actores:

Un actor especifica un rol que adopta una entidad externa (usuario, hardware externo u otro sistema) que interacciona directamente con el sistema, cada uno de ellos tiene un nombre y uno o varios roles en concreto, estos actores pueden ser primarios o secundarios[64].
En esta aplicación como actores tenemos a:

- Entrenador: Encargado de crear grupos de entrenamiento, registrar sesiones de entrenamiento, planificar sesiones futuras, añadir deportistas a grupos de entrenamiento, editar o eliminar grupos de entrenamiento, eliminar deportistas de grupos de entrenamiento, y será capaz de ver las estadísticas de cada grupo de entrenamiento.
- Deportistas: serán capaces de ver las sesiones dentro de su grupo de entrenamiento una vez el entrenador los comparta el grupo al que corresponden, también podrán ver sesiones pasadas de entrenamiento y gráficos de cada una.

8.2 Diagrama UML casos de uso:

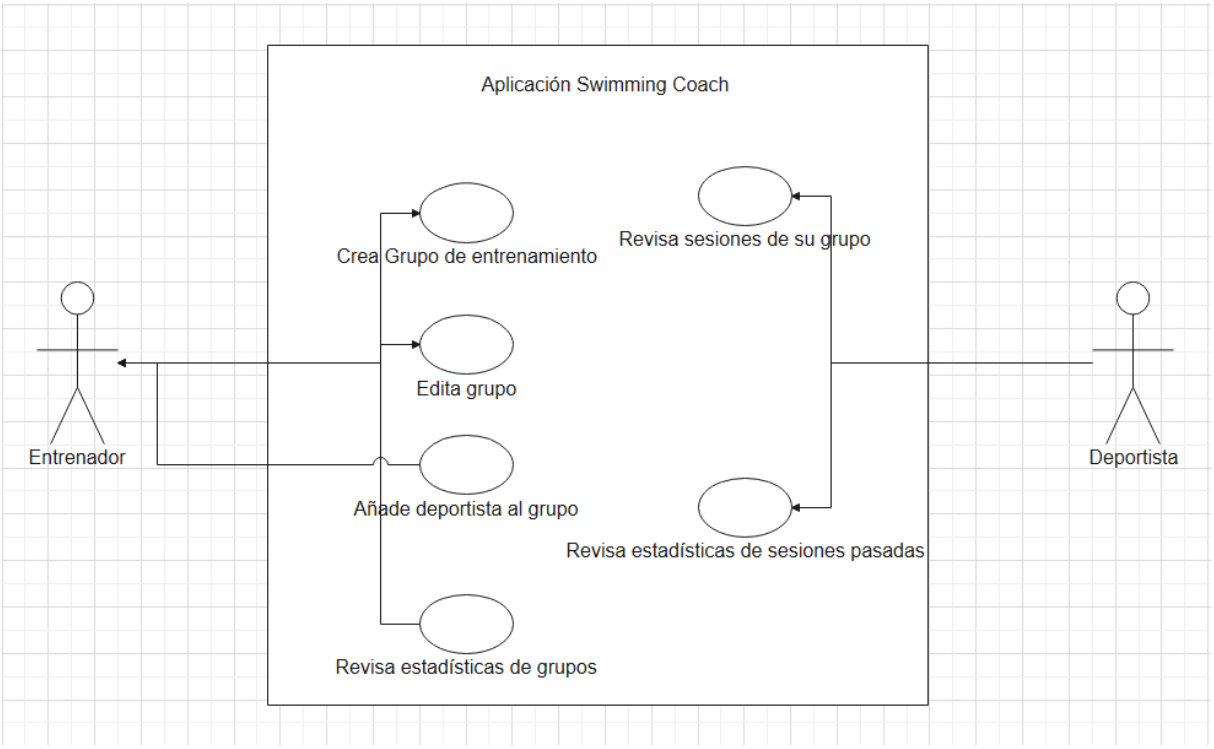


Figura 13: Casos de uso en la aplicación. Creado con Wondershare EdrawMax [65] .

8.3 Acciones de los usuarios:

Los perfiles dentro de la aplicación son los siguientes:

Tabla 3: Perfiles en la aplicación

	Perfil	Acciones
1	Entrenador	• Crea, lee, edita, borra (CRUD) grupos de entrenamiento. • Crea, lee, edita, borra (CRUD) nadadores. • Asignar nadador a grupo. • Desasignar nadador de grupo. • Editar nadador. • Crea, lee, edita, borra (CRUD) las sesiones de entrenamiento. • Ver estadísticas de grupo de entrenamiento.
2	Nadador	• Editar perfil de usuario. • Ver sesiones de grupo • Ver estadísticas de grupo de entrenamiento.

Un nadador no podrá entrar al grupo de entrenamiento sin que el entrenador comparta el grupo al que pertenece.

9. Usabilidad/UX

Para el desarrollo de la experiencia de usuario y diseño de la aplicación se utiliza principalmente el software Figma[66] y Zeplin[67], cuidando de los estándares modernos en cuanto a usabilidad y accesibilidad. Todo esto abarca colores, tipografías, íconos, contraste, etc. En la siguiente figura se aprecia el diseño y planificación de la interfaz con Figma:

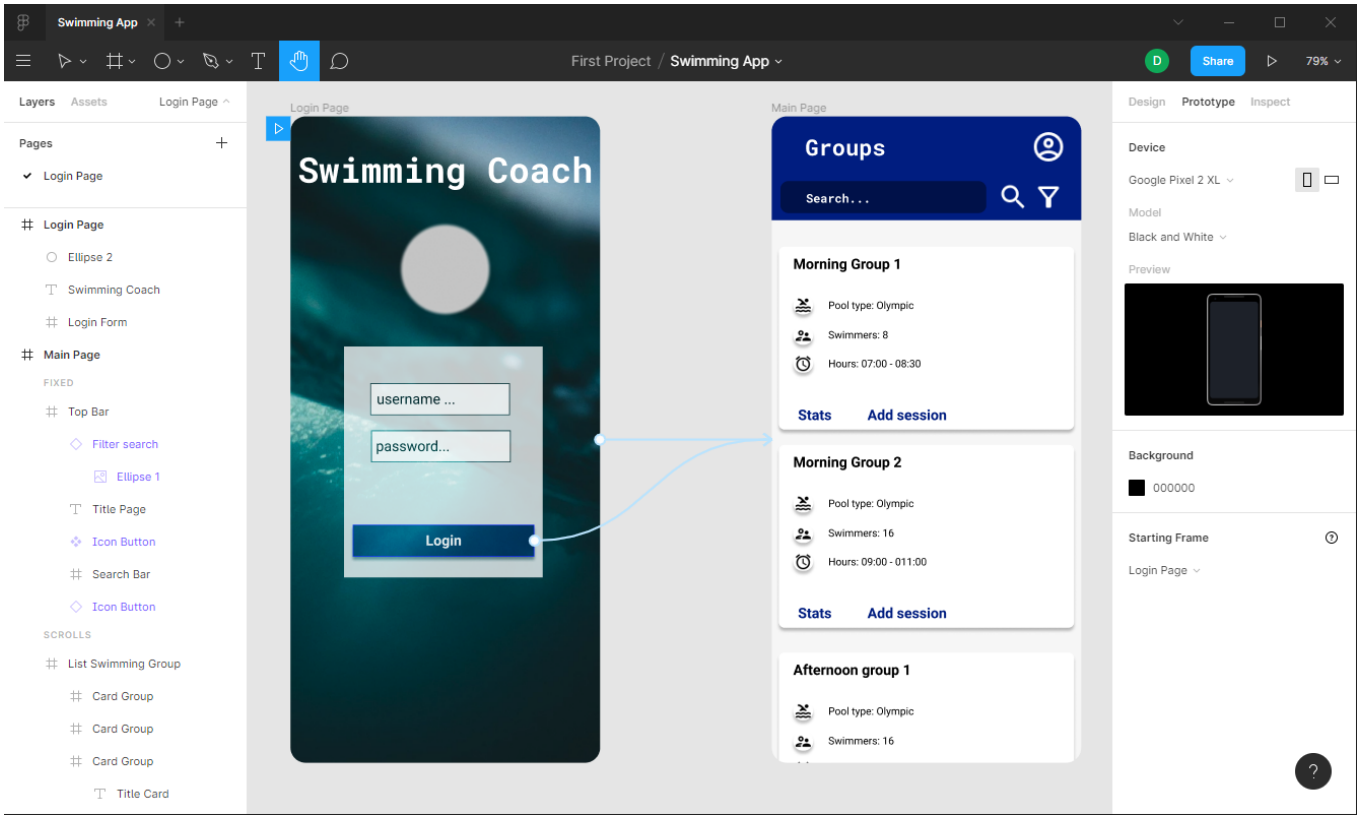


Figura 14: Diseño/UX de la aplicación con Figma.

Estos estilos sirven de guía para el desarrollo original del proyecto, sobre todo para los componentes que tengan mayor complejidad al momento de presentarse en la interfaz. Una de las grandes ventajas de Figma es la capacidad de exportar la maqueta y poderla utilizar tal y como se lo ha diseñado a nivel de estilos y componentes. Por ejemplo, en la Figura 7, se representan las interfaces propuestas en dispositivos móviles para el Login y la lista de grupos de entrenamiento, que una vez en el proyecto ha quedado de la siguiente forma tanto para la paleta de colores en tema oscuro (por defecto, como tema claro)

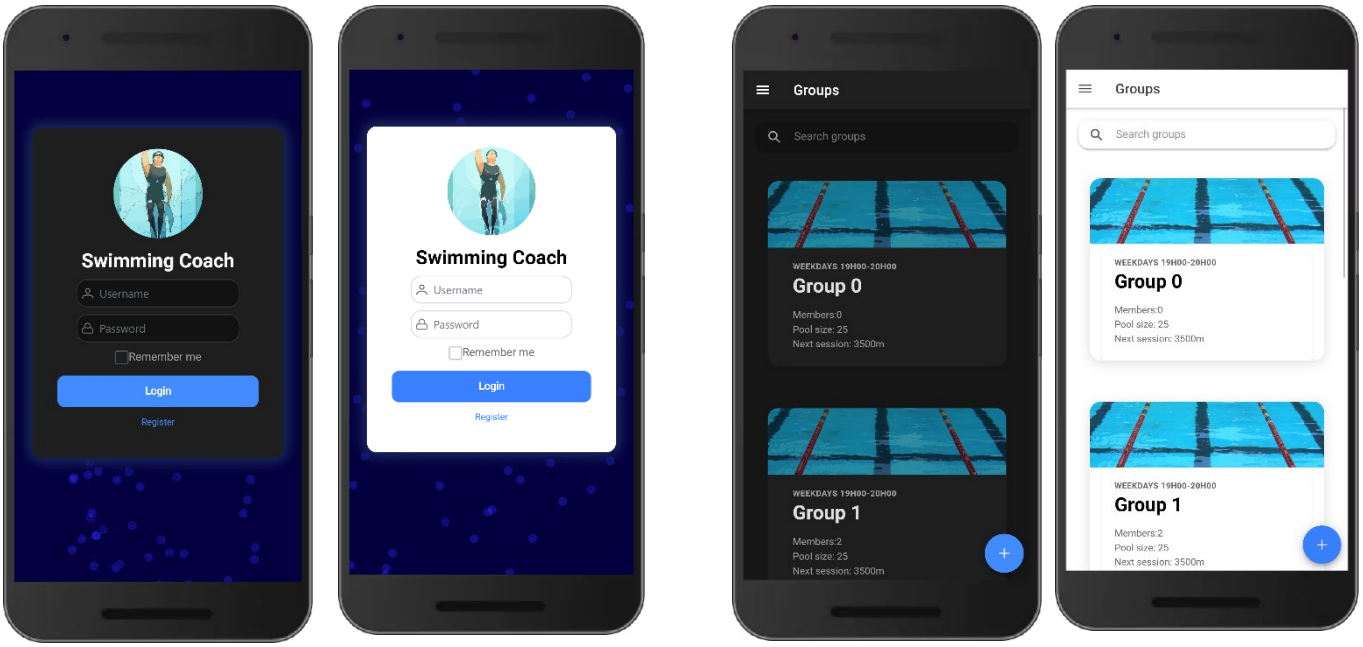


Figura 15: Diseños páginas Inicio de sesión (izquierda) y Grupos (derecha) en proyecto real.

9.1 Múltiples tamaños de pantalla:

Uno de los grandes desafíos de las PWA, es la adaptabilidad que tienen al ser ejecutados en diferentes dispositivos, desde las clásicas máquinas de escritorio (Desktop, Laptop) con sus dispositivos I/O como el teclado, ratón y puntero. Sin embargo, existen muchos otros dispositivos que tiene un tamaño de pantalla menor como el de las tablets, o de los dispositivos móviles en general, donde una interfaz de usuario sin cambios a la de escritorio sería inutilizable, además estos dispositivos móviles por lo general están enfocados para una navegación e interacción mediante pantalla táctil, gestos y botones de navegación.

Es por ello que en el proyecto se adapta a diferentes tamaños de dispositivos, la forma más efectiva en este caso ha sido utilizar media-queries de CSS, que permiten detectar tamaños de pantalla, cambios orientación y resoluciones, sin afectar el rendimiento o reducción de FPS.

9.1.1 Página inicio de sesión

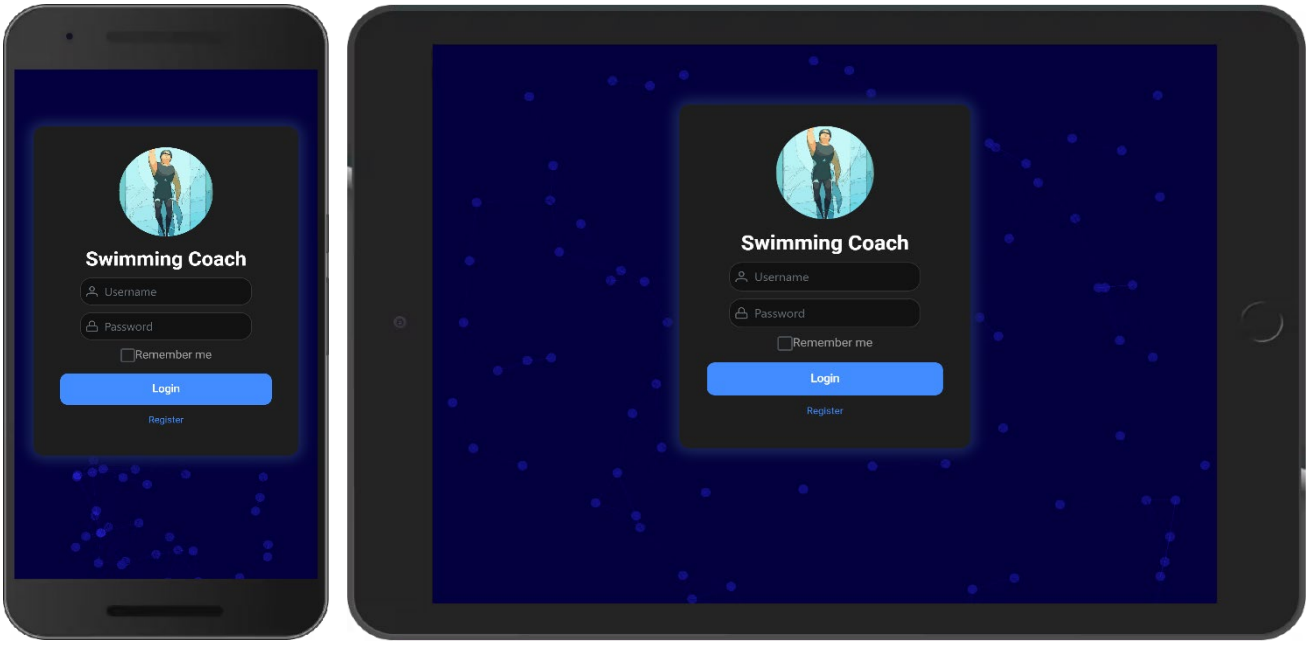


Figura 16: Página de inicio de sesión en pantalla pequeña (izquierda) y grande (derecha).

9.1.2 Menú navegación y páginas.

Posiblemente uno de los lugares más evidentes donde se diferencia entre la adaptación en diferentes tamaños de pantallas, es en el menú de navegación, una pantalla grande permite tener visible todo el tiempo este menú de navegación, sin perder el espacio principal para las diferentes páginas. En el caso de dispositivos móviles con tamaños de pantalla menores, se utiliza un menú desplegable, siempre y cuando el usuario lo requiera para cambiar entre las diferentes Páginas de la aplicación.

En tamaños de pantalla convencionales de escritorio, se pueden mostrar varios botones para diferentes acciones en la barra de cabecera al mismo nivel que el título. En cambio, para dispositivos móviles con menores tamaños de pantalla, se pueden utilizar botones flotantes, y así dejar limpia la cabecera debido al espacio reducido. En la siguiente figura se aprecia la versión en pantalla pequeña con un botón flotante en la esquina derecha inferior, mientras que en la versión de escritorio (pantalla grande), se lo muestran los botones en la cabecera:

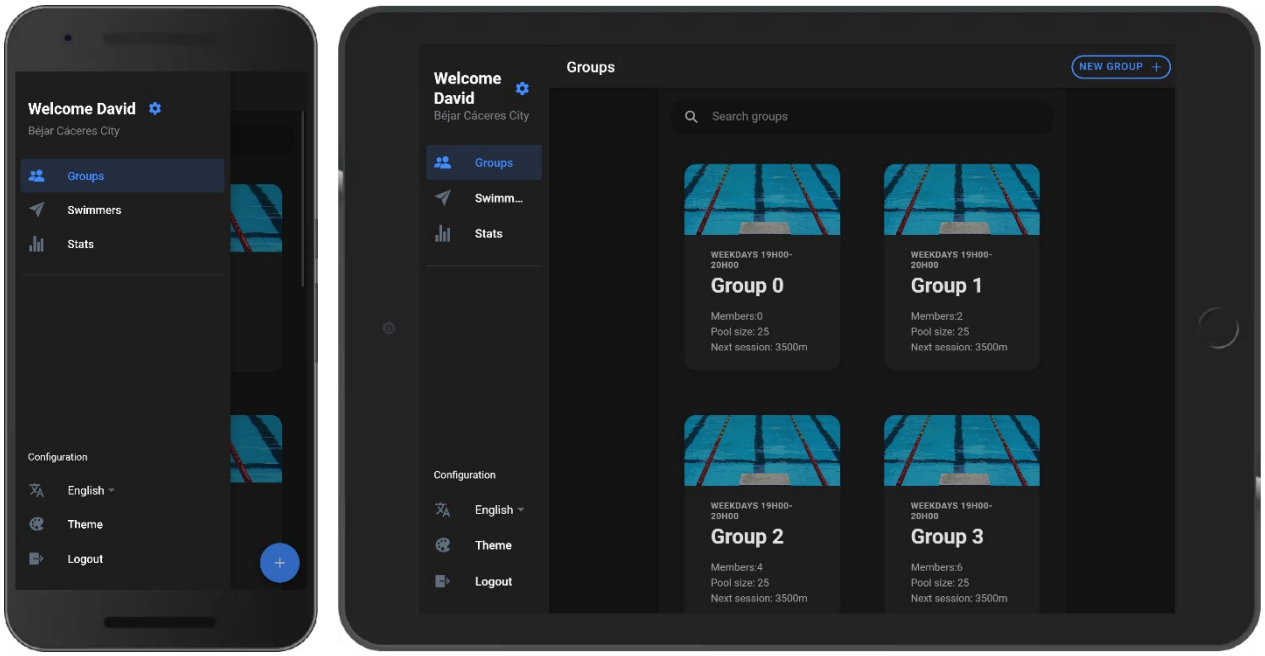


Figura 17: Menú de navegación desplegable en pantallas pequeñas (izquierda), en pantallas grandes (derecha). Botones flotantes para las páginas en dispositivos móviles (izquierda), y botones en cabecera para pantallas grandes (derecha).

9.1.3 Diálogos o páginas

Otro asunto a considerar en este proyecto es la utilización de diálogos en pantallas grandes, mientras que en dispositivos móviles con pantalla pequeña, puede ser imposible meter diálogos junto con el resto del contenido en pantalla, en pantallas grandes es mejor reducir la cantidad de páginas o redireccionamientos para varias componentes. A continuación, se puede apreciar el formulario de usuario para editar sus datos, mientras que las pantallas grandes permiten uso de diálogos con “modals”, en dispositivos de pantalla pequeña abre directamente una nueva página que contiene al formulario:

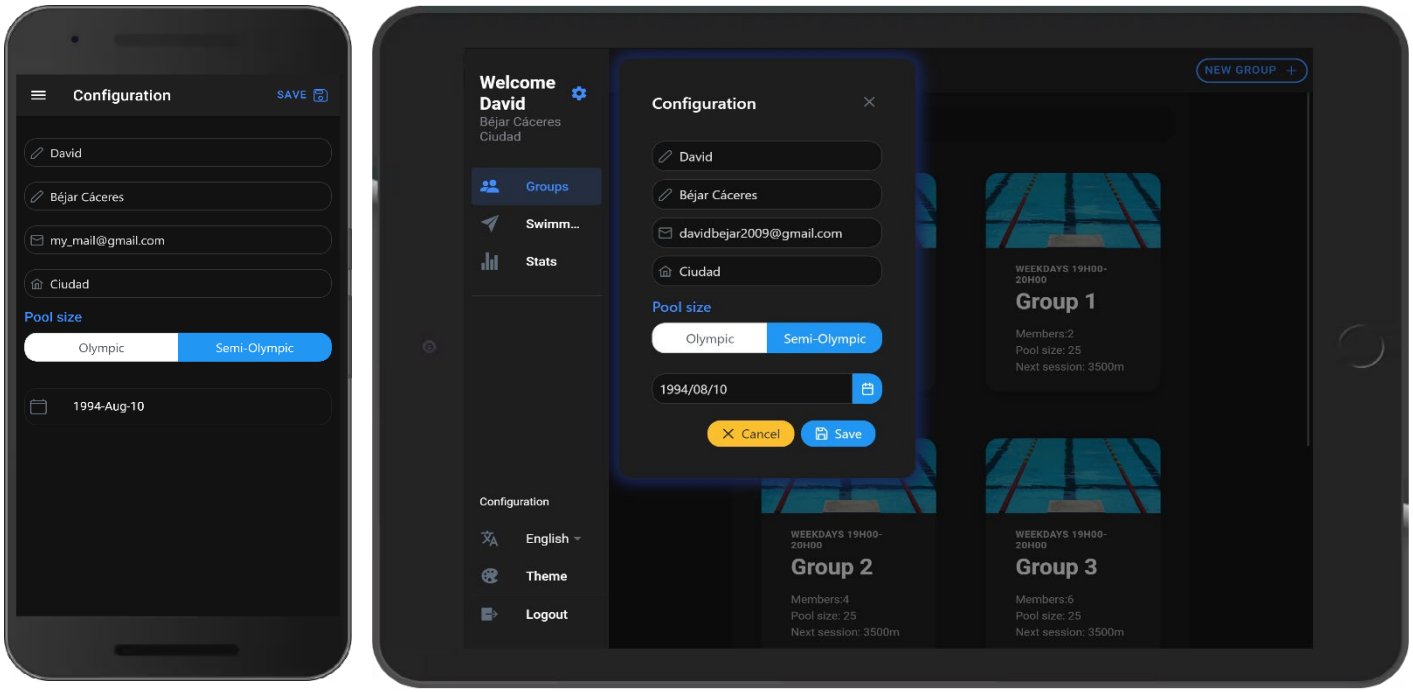


Figura 18: Use de páginas para formularios en pantallas pequeñas (izquierda) y diálogos en pantallas grandes (derecha).

9.1.4 Interacciones puntero/táctil

Los dispositivos de pantalla táctil son muy diferentes a los tradicionales teclado y puntero que tenemos en los computadores tradicionales, cualquiera de estos métodos son completamente válidos a la hora de entregar una experiencia de usuario satisfactoria, sin embargo, se puede caer en problemas de usabilidad al tener solamente una pantalla táctil cuando le interfaz y las interacciones fueron diseñadas para computadores de escritorio, y los mismos problemas de usabilidad se tendrían al momento de usar puntero y teclado para usar interfaces diseñadas para dispositivos con pantallas táctiles basadas en gestos. En este proyecto se han resuelto estos problemas, pensando en estos dos modos de interactuar.

Por ejemplo, los elementos de las listas responderán a Click derecho para abrir un menú contextual con opciones para cada elemento, pero además permite iniciar acciones con gestos al deslizar derecha o izquierda estos elementos. En la siguiente figura se aprecia claramente las acciones sobre los elementos de la lista:

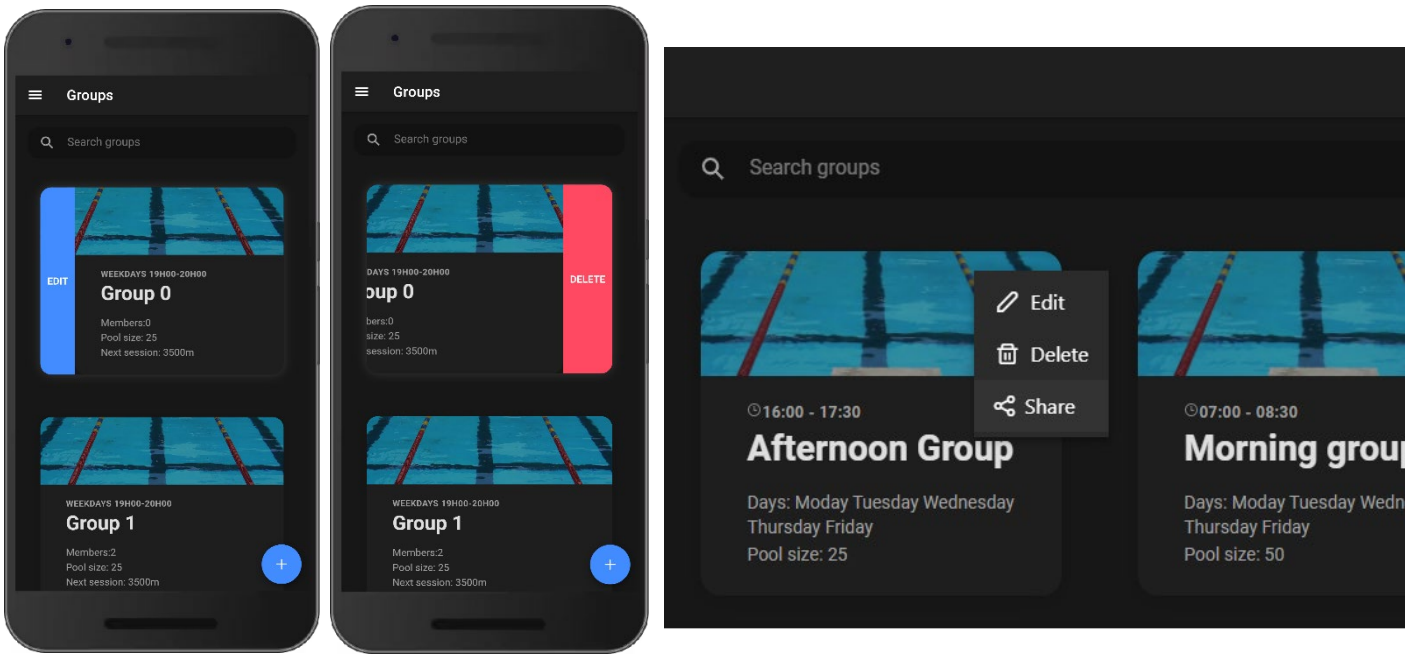


Figura 19: Interacciones gestos en pantallas pequeñas táctiles (izquierda) e interacciones con puntero usando menú contextual (derecha).

Otra adaptación para mejorar la experiencia de usuario es tener un comportamiento diferente de algunos componentes en los formularios, aunque un típico calendario puede resultar común en pantallas de escritorio, puede resultar muy extraño en plataformas de dispositivos móviles, en claro ejemplo son los componentes de fecha y hora que existen en los formularios, en este caso se ha decidido utilizar componentes que replican el comportamiento de elementos nativos en cada plataforma, a continuación se aprecia el componente de selección de fecha para dispositivos móviles con su selector replicando el comportamiento nativo de la plataforma, y el normal de un selector de fecha en pantallas grandes.

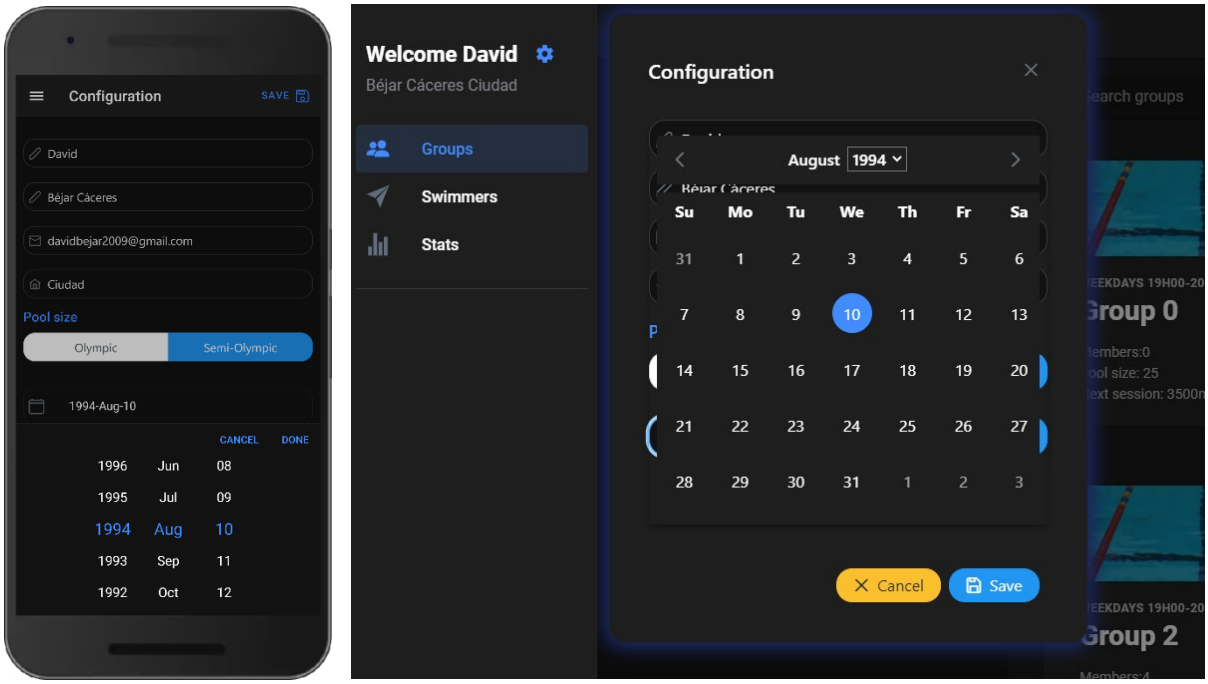


Figura 20: Componentes de formularios replicando a componentes nativos (izquierda) y otros pensados para punteros en pantallas grandes (derecha).

9.2 Internacionalización

Otra parte importante dentro del desarrollo de aplicaciones, es la internacionalización, usualmente al inicio de un proyecto no parece tener tanta relevancia la internacionalización, sin embargo, al expandirse el proyecto para usuarios más variados se torna indispensable proveer de varios idiomas en la aplicación. En este proyecto se utilizó la librería i18 para las traducciones. El usuario por defecto inicia en inglés (“en”), y desde el menú principal puede cambiar de idioma, automáticamente todas las cadenas de texto que tengan referencia a la función translate t() de i18 cambiarán automáticamente al nuevo idioma.

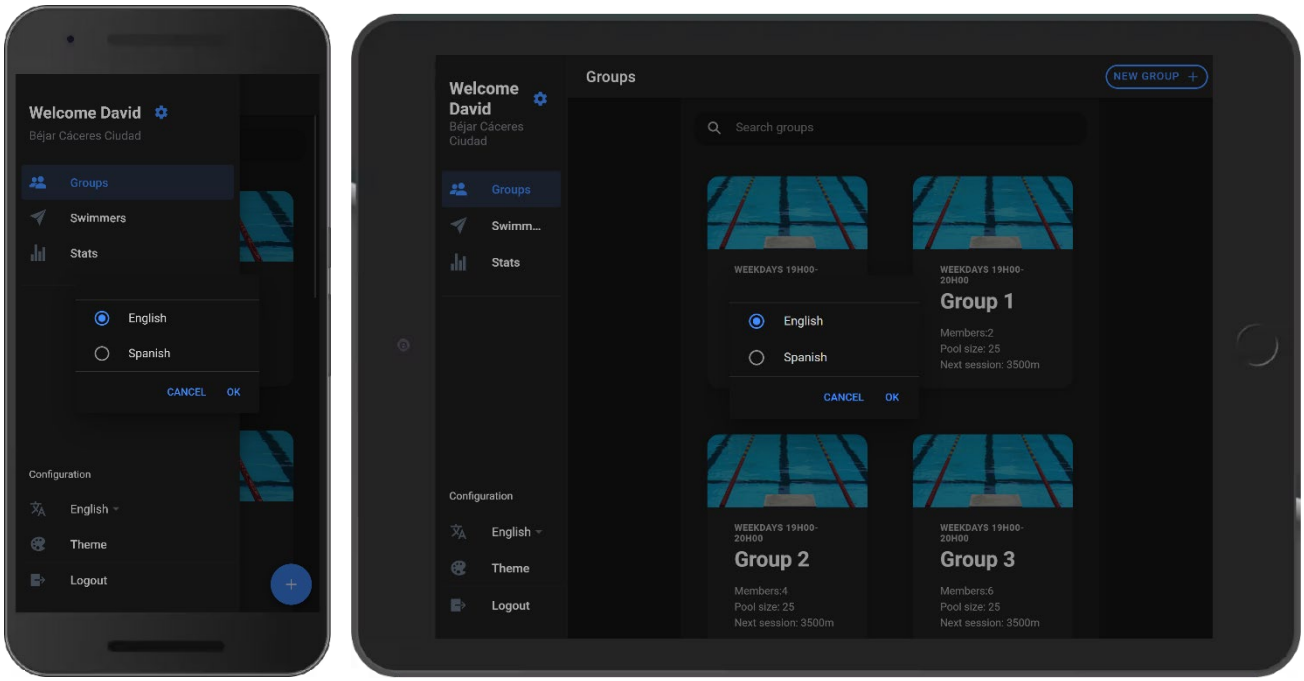


Figura 21:Selección de idioma para toda la aplicación desde el menú principal.

Con esta librería i18, es bastante sencillo añadir múltiples idiomas a la aplicación, luego del inicio y configuración para el proyecto, puede ser ejecutado desde cualquier componente mediante Hooks especiales de React. Los idiomas se podrán añadir en archivos “idioma.json”. En caso de no tener declarada la traducción con el identificador indicado, tomará un valor por defecto. A continuación, se puede ver un pequeño extracto de las traducciones usadas para este proyecto:

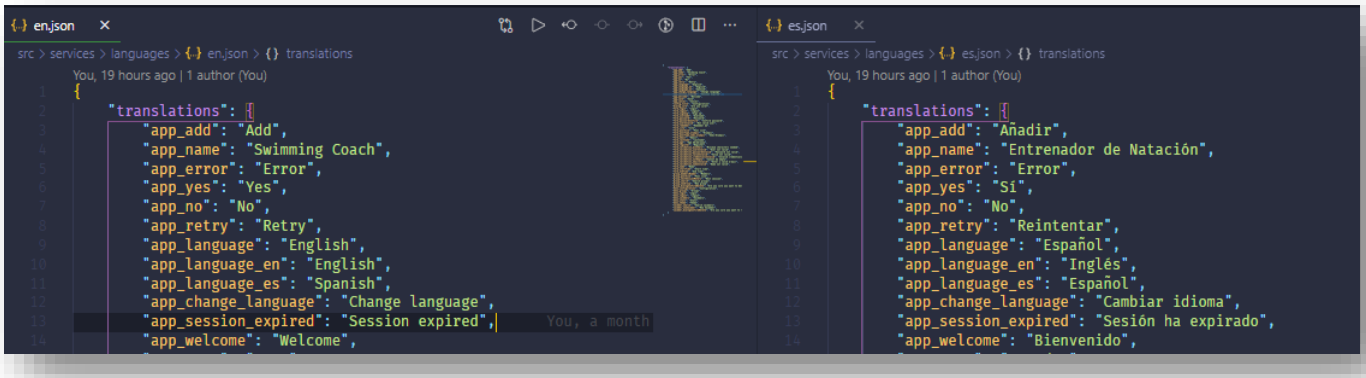


Figura 22:Archivos json con las traducciones a inglés (izquierda) y español (derecha).

9.3 Mejoras en rendimiento

Posiblemente uno de los aspectos más críticos en el software tiene que ver con el rendimiento y la aparente "fluidez" con la que se mueve el usuario por la interfaz. En los pequeños detalles se puede evidenciar la buena experiencia de usuario, caídas de FPS al realizar acciones, desplazar listas, abrir nuevas páginas o el tiempo de carga inicial inciden en gran medida para la experiencia. No sólo basta con realizar un Build en producción del proyecto, existen varias técnicas que pueden implementarse en React para ganar rendimiento de nuestra aplicación.
En el siguiente apartado se detallan algunas técnicas usadas para mejorar el rendimiento en este proyecto:

- Memoizar componentes con React.memo [68], para evitar el renderizado innecesario de componentes se utiliza el HOC React.memo(), con esta técnica se puede evitar entre uno a varios renderizados, y aunque en componentes funcionales la comparación sea no tan avanzada, nos posibilita pasar funciones de comparación de las Props que recibe el componente.
- Memoizar funciones o variables con React.useCallback() y React.useMemo() [68], con estas funciones se pueden memoizar funciones o variables dentro de cada componente, de esta forma no se generarán nuevas instancias, sólo cambiarán las funciones o variables si una de sus dependencias ha cambiado.
- Separar componentes, al tener componentes grandes, cada cambio que exista será calculado por React en el Shadow DOM, esto no causará ningún problema hasta que el componente crezca demasiado, en tal caso para actualizar hasta lo más mínimo tendrá que renderizar todo el componente. En este caso lo mejor es separar en componentes y así solamente renderizar los cambios en las partes específicas del Shadow DOM que deben cambiar. Un ejemplo de ello son las listas y los elementos de las listas, o también las páginas con los formularios dentro de aquella página. Posiblemente esta sea una de las formas más efectivas para mejorar el rendimiento.
- Lazy Loading o también conocido como División de código[69], esto posiblemente no muestre una clara señal de fluidez en el sistema, sin embargo, al añadir React.lazy() para la carga de componentes, serán importados tan pronto sean requeridos, esto brinda grandes ventajas al abrir el proyecto en la primera página, no será necesario que llegue del servidor todo el proyecto en un gran "chunck.js", sino que se dividirá en varios bajo demanda, así la carga inicial de la primera página en pantalla puede ser más rápido sobre todo para usuarios con conexión lenta.
- Optimizar el estado de la aplicación (redux, recoil, etc), esto también es muy importante, tanto los componentes que tengan acceso directo a selectores dentro de la aplicación como la mayoría de los componentes hijos serán renderizados nuevamente al cambiar el estado de la aplicación donde se lo necesite, en este caso es muy importante momoizar estos componentes con HOC o utilizar los selectores específicamente en los componentes que los usará. En este caso al usar Recoil como librería de estado global de la aplicación, se optimiza la actualización de estado a los componentes

gracias a la arquitectura de esta librería pensada para React con componentes funcionales, junto con las técnicas de “memoización” anteriormente mencionadas.

9.4 Android SlashScreen

Otra característica que tenemos en Android es una pantalla de entrada al lanzar la app, conocida como SplashScreen, permite que toda la app siga el mismo esquema de colores, logo y corresponda con el logo de la aplicación. En la siguiente figura se aprecia en el dispositivo Android al iniciar la aplicación:

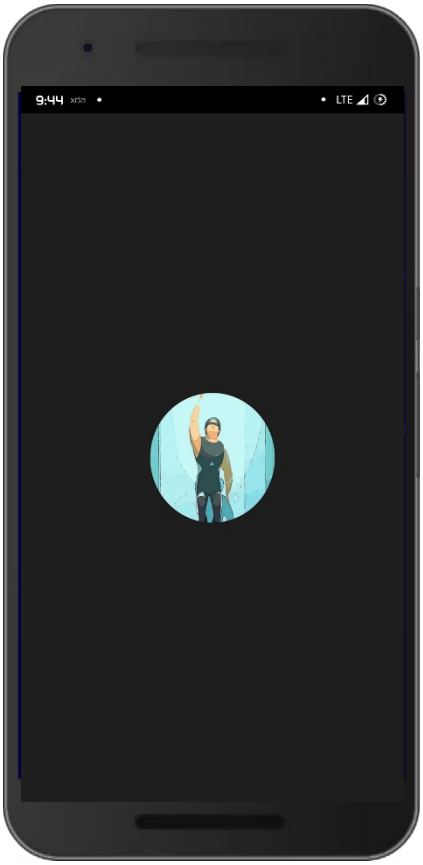


Figura 23: SplashScreen Android al lanzar la aplicación.

9.6 Feedback en la interfaz gráfica de errores

9.6.1 Mensajes de error Toast

Otro punto importante para brindar una experiencia de usuario es el Feedback que obtiene de varias acciones, entre ellas están las notificaciones Toast que existen en caso existan errores, en la siguiente pantalla se aprecia un mensaje de error en la parte inferior de la pantalla del dispositivo al ingresar las credenciales de inicio de sesión incorrectas, además el cambio de color sutil del fondo animado a un color rojo:

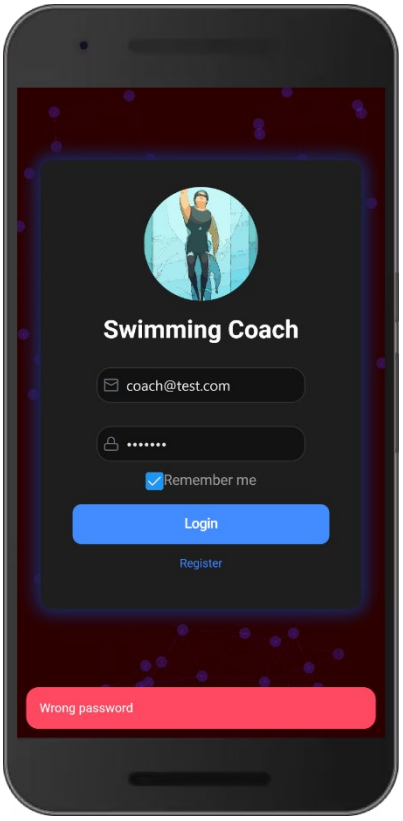


Figura 24: Feedback en caso de error mediante mensajes Toast.

9.7 Validación de formularios y campos

Otra forma de brindar Feedback al usuario dentro de la aplicación es mediante la validación de formularios, por debajo de los formularios de la aplicación se hacen varias validaciones para los campos, algunos son requeridos antes de enviar los formularios con los datos, esto gracias a la librería React Hook Form se puede integrar con varias librerías de componentes. Por ejemplo, la siguiente figura se aprecia la validación de formularios, al tener un campo en error, se pinta de color rojo y añade un ícono de alerta en el lado derecho del campo, que al dar Click lanzará un mensaje Toast con detalles del error:

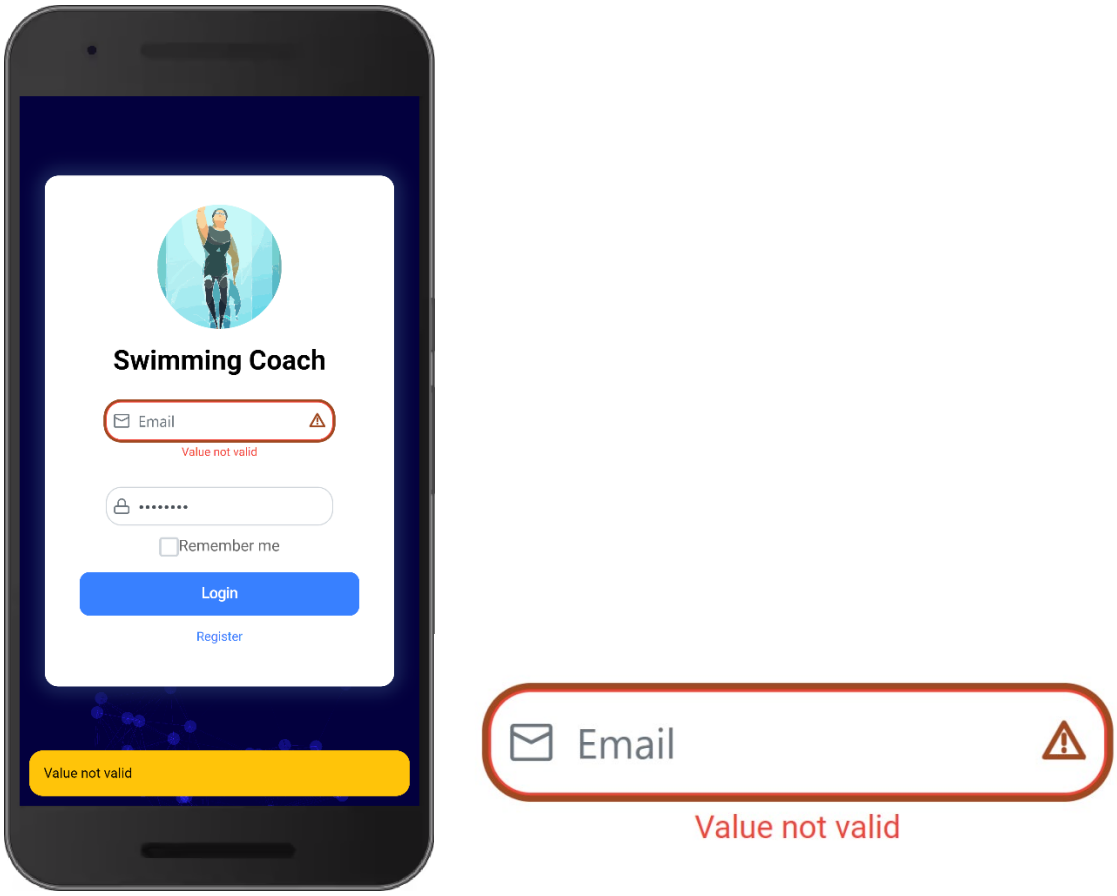


Figura 25: Validación de formularios, errores visibles en campos y Toast.

10. Test de integración Cypress

Los test realizados en este proyecto son realizados con Cypress [70], es una librería de test para Javascript, posiblemente de las mejores que existen hoy en día, son enfocadas sobre todo para test de interfaces de usuario, además permiten simular llamadas al servidor, disparar eventos para los componentes en pantalla, etc. Permiten recopilar datos extra de los tests, capturas de pantalla e incluso grabar en formato de vídeo los tests, todo esto de forma automática, y compatible con los proveedores de servicios principales para Integración Continua como Docker, Travis, Jenkings, etc.

Una vez instalado y configurado Cypress, se procede a abrir desde la consola con el comando

```
$ npm run cypress
```

Y la consola de test se abrirá lista para seleccionar el test que queremos iniciar:

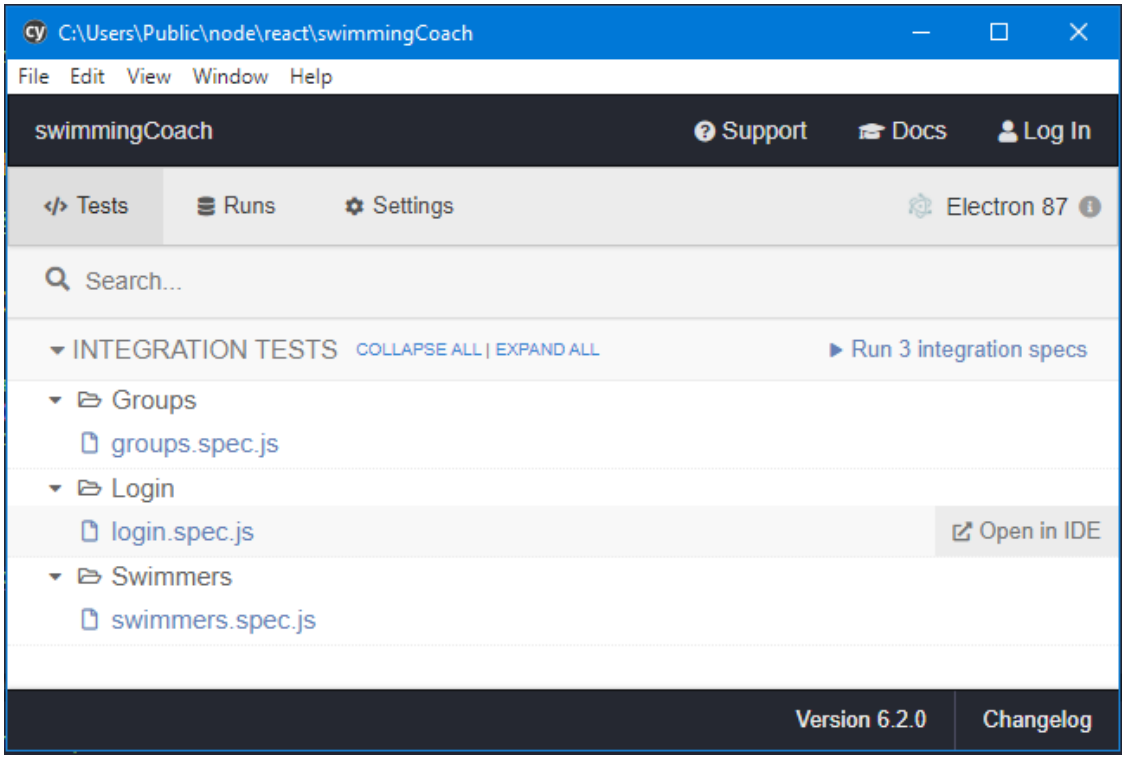


Figura 26: Consola de test de Cypress para el proyecto.

Los siguientes tests se han creado mediante Cypress dentro de la carpeta /cypress/integration:

10.1 Test página de inicio de sesión:

Ubicado en el archivo */cypress/integration/Login/login.spec.js*. Este test se encarga de comprobar lo siguiente:

- Que los elementos tanto del formulario de inicio de sesión se encuentren visibles en el DOM.
- Que los elementos del formulario de registro de nuevo usuario sean visibles en el DOM
- Que desde el formulario inicio de sesión se puede redirigir al registro de nuevo usuario y viceversa.
- Que al ingresar valores en los inputs de texto están realizando las validaciones de los formularios para cada campo.
- Al disparar un inicio de sesión con usuario o contraseña incorrecto, se muestra al usuario el error y cambia los colores por 2 segundos a color rojo el fondo.
- Finalmente iniciará la sesión con usuario y contraseña, comprobará que se ha redirigido a la página principal luego de la autenticación.

El código específico encargado para el inicio de sesión es el siguiente:

swimmingCoach - login.spec.js

```
describe('Login with test user', () => {
  before(() => {
    visitLoginPage();
    cy.waitForReact();
  });

  it('Login test user', () => {
    cy.get('#input-login-email').type(testUser.email).should('have.value', testUser.email);
    cy.get('#input-login-password').type(testUser.password).should('have.value', testUser.password);
    cy.get('.p-checkbox-box').click().should('have.class', 'p-highlight').should('have.attr', 'aria-checked', 'true');
    cy.get('#button-login').click().wait(2000);
    cy.url().should('include', '/groups');
  });
});
```

Figura 27: Ejemplo test inicio sesión Cypress.

Una vez completado se podrá visualizar que todos los test pasan satisfactoriamente:

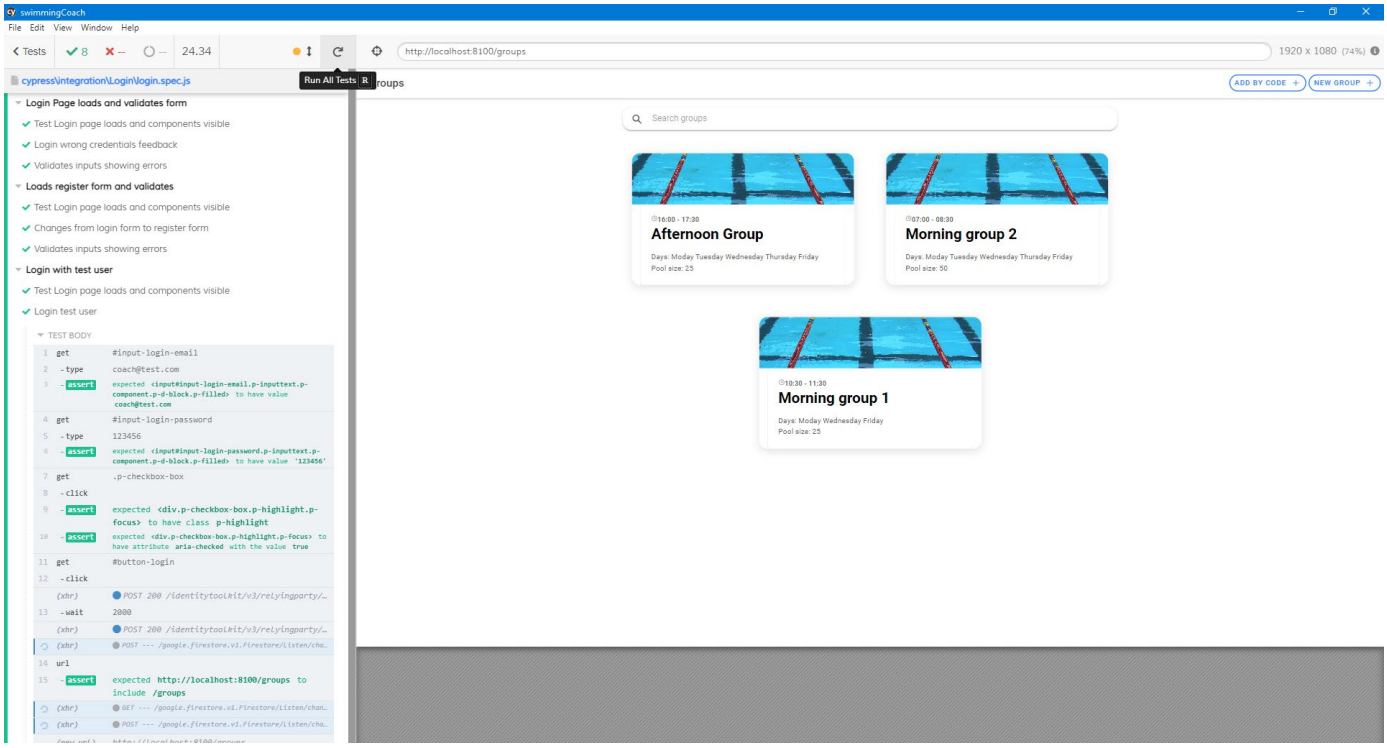


Figura 28: Test Cypress login.spec.js.

10.2 Test página grupos de entrenamiento y menú principal:

Ubicado en el archivo */cypress/integration/Groups/groups.spec.js*. Este test se encarga de comprobar lo siguiente:

- Luego de iniciar sesión el usuario es redirigido a la página principal con los grupos de entrenamiento.
- Revisa que los componentes no tengan problemas al renderizar, y que se encuentren visibles en el DOM.
- Que el diálogo con el formulario de información de usuario se abre al dar click en el botón que se encuentra en el menú lateral izquierdo.
- Que el diálogo de configuración de usuario tenga el formulario y todos sus campos visibles en el DOM.
- Que los campos del formulario de configuración de usuario validen correctamente y muestren los íconos de campo incorrecto.
- Cambio de tema desde el menú lateral.
- Items de cambiar tema, idioma, y fin de sesión.
- Abre el diálogo de nuevo grupo con el formulario de grupo.
- Que el formulario tiene todos los campos en el DOM y valida correctamente.

- Que los campos con error de validación tengan la clase de error adecuada y esté el ícono de error.
- Crea un nuevo grupo.
- Que el menú contextual se abra en el grupo creado.
- Que el diálogo de compartir grupo sea visible y se vea el código.
- Borra el grupo de test creado anteriormente.

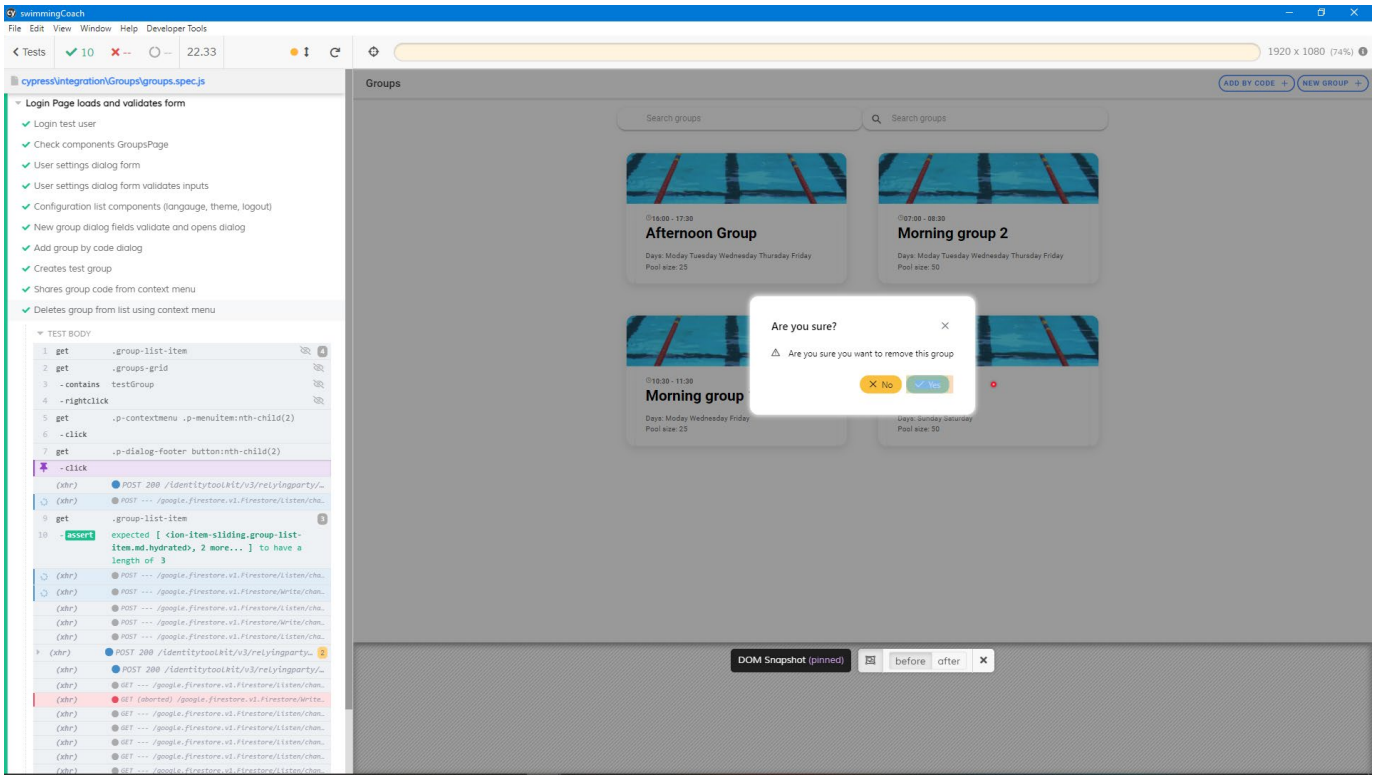


Figura 29: Test Cypress groups.spec.js.

10.3 Test página lista de nadadores y menú principal:

Ubicado en el archivo */cypress/integration/Swimmers/swimmers.spec.js*. Este test se encarga de comprobar lo siguiente:

- Al visitar la página con la lista de nadadores, requiera autenticación primero, es decir, que el componente esté protegido.
- Comprueba que, al autenticar correctamente, el usuario es redireccionado a la página con la lista de nadadores, y no a la página por defecto con la lista de grupos de entrenamiento.
- Comprueba que el menú lateral izquierdo tenga todos los componentes visibles en el DOM.
- Comprueba que los componentes de la página de nadadores estén visibles.
- Que el diálogo de crear/editar un nuevo nadador se pueda abrir o cerrar.
- Que el formulario dentro del diálogo crear/editar nadador funcione correctamente con la validación de cada campo y las clases para pintar por CSS estén presentes.

- Que la funcionalidad de crear un nuevo nadador funcione correctamente y el nuevo nadador de prueba se muestre en la lista.
- Remover un nadador mediante el menú contextual sobre el usuario de prueba.
- Que el diálogo de confirmación de nadador sea visible y accione la eliminación del usuario.

Por ejemplo, en el siguiente fragmento de código se busca el nuevo nadador de prueba añadido durante el test, se simula un click derecho sobre el elemento para abrir el menú contextual, donde al simular el click sobre la opción de eliminar se debe eliminar el nadador y los elementos de la lista debe ser igual al inicial menos uno:

swimmingCoach - groups.spec.js

```
it('Deletes group from list using context menu', () => {
  cy.get('.group-list-item').then(itemsArray => {
    const startAmount = itemsArray.length;

    cy.get('.groups-grid').contains(groupCreated.name).rightclick();
    cy.get('.p-contextmenu .p-menuitem:nth-child(2)').click();
    cy.get('.p-dialog-footer button:nth-child(2)').click();
    cy.get('.group-list-item').should('have.length', startAmount - 1);
  });
});
```

Figura 30: Test Cypress para abrir menú contextual y eliminar un elemento de lista.

Una vez terminado todo el test, en la barra lateral izquierda, Cypress nos mostrará el estado de todos los test, exitosamente superados:

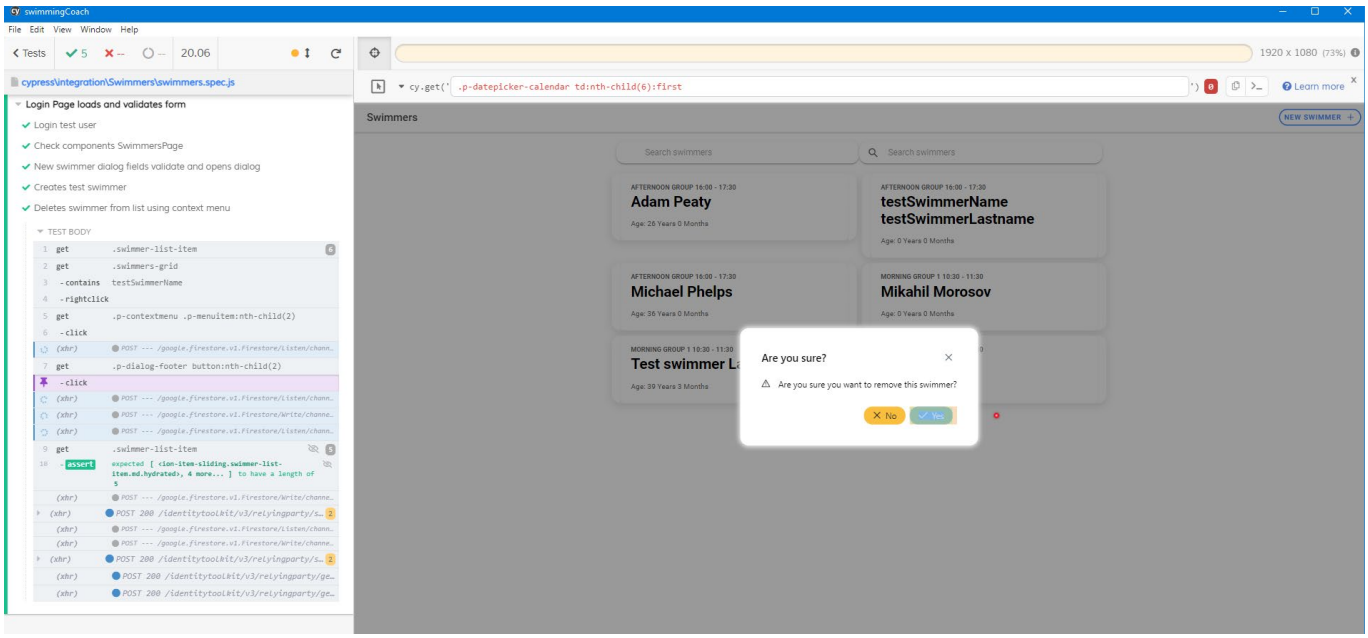


Figura 31: Fin test superado swimmers.spec.js.

11. Requisitos e instrucciones de instalación

11.1 Instalación previa de requisitos básicos

Para iniciar la aplicación PWA se requiere configurar el equipo con el siguiente software:

- Nodejs LTS.
- Ionic CLI (npm install -g @ionic/cli)

11.2 Instalación de paquetes npm

Antes de iniciar el “Build” se requieren instalar todas las dependencias del proyecto, una vez situado en el root del proyecto ejecutar el siguiente comando:

```
$ npm i
```

En caso que de errores o warnings se puede ejecutar los siguientes comandos

```
$ npm Audit fix
$ npm fund
```

11.3 Arrancar el proyecto en modo de desarrollo

Al ejecutar el siguiente comando, Ionic CLI iniciará la compilación del proyecto y lo abrirá automáticamente en el navegador. Por defecto http://localhost:8100/ a la pantalla de inicio de sesión

```
$ ionic serve
```

11.4 Build en modo de producción como PWA

Una vez con los paquetes de las dependencias instaladas, se puede iniciar el Build del proyecto en modo de producción, una vez terminado, se localizará en "/build" toda la PWA para ser desplegada en un servidor:

```
$ ionic build --prod
```

Para arrancarlo en local se lo puede utilizar un servidor con los siguientes comandos, y por defecto lo arrancará en "http://localhost:5000":

```
$ npm install -g serve
$ serve -s build
```

11.5 Compilación y ejecución Android

Para la compilación de la PWA en otras plataformas Android/iOS/Electron es necesario Capacitor, con los siguientes comandos se lo añade al proyecto, se realiza el build en modo de producción y se lo inicia en Android Studio para compilar un instalador en formato .apk que puede instalarse en cualquier dispositivo Android. Para esto es necesario los siguientes requisitos previos:

- Android Studio IDE (compila el proyecto en un .apk).
- Dispositivo Android o Emulador Android.
- Android Debugging Bridge (ADB) en caso de conectar el dispositivo por cable al equipo.

```
$ ionic capacitor add android
$ npx cap copy
$ npx cap open android
```

Al tener ejecutado el código anterior, ya tendremos el Android Studio abierto, desde allí ya dependerá si se quiere arrancar la aplicación en el emulador o en el dispositivo físico, o simplemente hacer el build del proyecto para obtener un .apk. Por defecto el target SDK es de 29.0 y compilador para todas las arquitecturas (ABI), por lo que garantiza compatibilidad con la mayoría de variantes en hardware y software del ecosistema Android.

12. Bugs

- React-router tiene un bug en su versión que ocasiona un bucle infinito al redireccionar luego de terminar el proceso de autenticación. Se ha encontrado una solución no completa hasta que exista un bugfix en la librería.
- React-hook-forms no es compatible con todas las librerías de componentes, simplemente se garantiza el funcionamiento de los formularios con componentes planos HTML, en este proyecto se utilizan tanto componentes de Ionic como de PrimeReact, que no son completamente compatibles con React-hook-forms. Se ha encontrado la solución parcial al utilizar al tener un Wrapper para cada componente, estos componentes de los formularios son “Componentes Controlados”.
- Ionic Router, algunas animaciones en dispositivos móviles tienen un pequeño parpadeo al cambiar de rutas, queda a la espera del arreglo.
- El plugin de Capacitor para leer códigos QR desde la cámara se muestra debajo del WebView de la app, este es un problema mencionado en la página oficial de la librería @dutchconcepts/capacitor-barcode-scanner [60]. Para solucionar este problema se ha tenido que ocultar el contenido del <body> de la página mientras espera la finalización de la lectura con la cámara de forma asíncrona.

Tabla 4: Bugs y soluciones encontradas.

	Bug	Descripción	Estado	Solución
1	React-router bucle	React-router tiene un bug en su versión que ocasiona un bucle infinito al redireccionar luego de terminar el proceso de autenticación.	Solucionado	Se añade estado a la ruta de <Redirect /> en prevLocation: path. Luego en LoginPage.js, si se detecta que debe redireccionar a la página con la ruta protegida.
2	React-hook-forms validación formularios	React-hook-forms no es compatible con todas las librerías de componentes para la validación de formularios.	Solucionado	Wrapper para cada componente, estos componentes de los formularios son “Componentes Controlados”. https://react-hook-form.com/api/#Controller
3	Ionic Router parpadeo navegación.	Algunas animaciones en dispositivos móviles tienen un pequeño parpadeo al cambiar de rutas, queda a la espera del arreglo.	Pendiente	Aún no encontrada

4	Imagen de cámara detrás del WebView	El plugin de Capacitor para leer códigos QR desde la cámara se muestra debajo del WebView de la app, este es un problema mencionado en la página oficial de la librería @dutchconcepts/capacitor-barcode-scanner [60]	Solucionado	Se ha tenido que ocultar el contenido del <body> de la página mientras espera la finalización de la lectura con la cámara de forma asíncrona.

13. Proyección y proyectos a futuro

- Expandir las funcionalidades para mantener el registro también del rendimiento/resultados de eventos competitivos por cada nadador.
- Registro de rendimiento/resultados para los grupos de entrenamiento en los eventos competitivos.
- Mejorar el diseño en la base de datos para reducir el acceso a Firebase, Firebase contiene muchas funciones en la nube que se pueden explorar.
- Despliegue del proyecto como una verdadera Progressive Web App en la Web con un certificado SSL, uno de los requisitos para que una Progressive Web App esté completa es la compra de certificados e integrarlos con su dominio propio.

14. Conclusión/-es

- React junto con Ionic+Capacitor, permiten crear aplicaciones híbridas que pueden ser compiladas y desplegadas en múltiples plataformas.
- La experiencia de usuario UX está en los detalles, no solamente el rendimiento es una medida, el “Feedback” al usuario en la interfaz gráfica son otras piezas claves, mensajes emergentes, colores de alerta, etc.
- La interacción en dispositivos de pantalla pequeña o táctil, puede no ser viable para pantallas grandes en equipos de escritorio, hay que adaptar el comportamiento de los componentes para las distintas plataformas.
- Los usuarios pueden suscribirse como nadadores a los grupos de un entrenador, los nadadores pueden darse de alta en la suscripción para ver los entrenamientos y revisar las sesiones pasadas.
- El desarrollo híbrido-Web multi plataforma brinda grandes ventajas para equipos pequeños o proyectos de un solo desarrollador.

Anexo 1. Bibliografía

[1] «Progressive web apps (PWAs)», *MDN Web Docs*. https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps (accedido sep. 29, 2020).
[2] «Firebase». https://firebase.google.com/ (accedido sep. 29, 2020).
[3] «Write once, run anywhere - Wikipedia». https://en.wikipedia.org/wiki/Write_once,_run_anywhere (accedido sep. 29, 2020).
[4] «Kotlin/Native - Kotlin Programming Language». https://kotlinlang.org/docs/reference/native-overview.html (accedido sep. 29, 2020).
[5] «Flutter - Beautiful native apps in record time». https://flutter.dev/ (accedido sep. 29, 2020).
[6] «React Native · A framework for building native apps using React». https://reactnative.dev/ (accedido sep. 29, 2020).
[7] «Xamarin | Open-source mobile app platform for .NET». https://dotnet.microsoft.com/apps/xamarin (accedido sep. 29, 2020).
[8] A. Abran, A. Khelifi, W. Suryn, y A. Seffah, «Consolidating the ISO usability models», en *Proceedings of 11th international software quality management conference*, 2003, vol. 2003, pp. 23-25.
[9] I. Darwin *et al.*, «file–determine file type, BSD General Commands Manual, file (1)», *BSD, Jan*, vol. 2007, 1973.
[10] «x86-64 Code Model». https://developer.apple.com/library/archive/documentation/DeveloperTools/Conceptual/MachOTopics/1-Articles/x86_64_code.html (accedido oct. 26, 2020).
[11] «UCSD PASCAL - Available technology for licensing from the University of California, San Diego». https://techtransfer.universityofcalifornia.edu/NCD/19327.html (accedido oct. 26, 2020).
[12] 14:00-17:00, «ISO/IEC 14882:2017», *ISO*. https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/06/85/68564.html (accedido oct. 26, 2020).
[13] «Free Online Chapters of Inside the Java Virtual Machine by Bill Venners». https://www.artima.com/insidejvm/ed2/index.html (accedido oct. 27, 2020).
[14] «Dart programming language». https://dart.dev/ (accedido ene. 05, 2021).
[15] «Platforms». https://dart.dev/platforms.html (accedido ene. 05, 2021).
[16] «Core Components and APIs · React Native». https://reactnative.dev/docs/components-and-apis (accedido ene. 05, 2021).
[17] «Ionic Article: Ionic React vs React Native», *Ionic Framework*, feb. 18, 2020. https://ionicframework.com/resources/articles/ionic-react-vs-react-native (accedido ene. 05, 2021).
[18] «What is Xamarin? | .NET», *Microsoft*. https://dotnet.microsoft.com/learn/xamarin/what-is-xamarin (accedido ene. 05, 2021).
[19] «Xamarin.Forms | .NET», *Microsoft*. https://dotnet.microsoft.com/apps/xamarin/xamarin-forms (accedido ene. 05, 2021).
[20] dianmsft, «Packaging MSIX apps - MSIX». https://docs.microsoft.com/en-us/windows/msix/package/packaging-uwp-apps (accedido ene. 05, 2021).
[21] Ionic, «Ionic - Cross-Platform Mobile App Development», *Ionic Framework*. https://ionicframework.com/ (accedido ene. 05, 2021).
[22] «Cross-platform Native Runtime for Web Apps - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs (accedido dic. 01, 2020).
[23] «Apache Cordova». https://cordova.apache.org/ (accedido ene. 05, 2021).
[24] «PhoneGap». https://phonegap.com/ (accedido ene. 05, 2021).
[25] «Plugin Search - Apache Cordova». https://cordova.apache.org/plugins/ (accedido ene. 05, 2021).
[26] «Background Tasks - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs/apis/background-task (accedido ene. 05, 2021).
[27] «Camera - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs/apis/camera (accedido ene. 05, 2021).
[28] «Clipboard - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs/apis/clipboard (accedido ene. 05, 2021).

[29] «Device - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs/apis/device (accedido ene. 05, 2021).
[30] «Filesystem - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs/apis/filesystem (accedido ene. 05, 2021).
[31] «Geolocation - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs/apis/geolocation (accedido ene. 05, 2021).
[32] «Keyboard - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs/apis/keyboard (accedido ene. 05, 2021).
[33] «Local Notifications - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs/apis/local-notifications (accedido ene. 05, 2021).
[34] «Permissions - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs/apis/permissions (accedido ene. 05, 2021).
[35] «Splash Screen - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs/apis/splash-screen (accedido ene. 05, 2021).
[36] «Storage - Capacitor», *Capacitor: Cross-platform native runtime for web apps*. https://capacitorjs.com/docs/apis/storage (accedido ene. 05, 2021).
[37] «Qué es SCRUM», *Proyectos Ágiles*, ago. 04, 2008. https://proyectosagiles.org/que-es-scrum/ (accedido ene. 05, 2021).
[38] Ionic, «Ionic Framework - Ionic Documentation», *Ionic Docs*. https://ionicframework.com/docs/undefined (accedido sep. 29, 2020).
[39] M. Netkow, «Announcing Capacitor 2.0», *Ionic Blog*, abr. 08, 2020. https://ionicframework.com/blog/announcing-capacitor-2-0/ (accedido oct. 28, 2020).
[40] M. Lynch, «Announcing Capacitor 1.0», *Ionic Blog*, may 22, 2019. https://ionicframework.com/blog/announcing-capacitor-1-0/ (accedido oct. 28, 2020).
[41] «Structure Your Database | Firebase Realtime Database», *Firebase*. https://firebase.google.com/docs/database/web/structure-data (accedido ene. 04, 2021).
[42] «Recoil - The Asynchronous way to manage state [Part 1] - DEV». https://dev.to/shubhamk/recoil-the-asynchronous-way-to-manage-state-part-1-18kk (accedido nov. 25, 2020).
[43] «Jira | Issue & Project Tracking Software | Atlassian». https://www.atlassian.com/software/jira (accedido dic. 09, 2020).
[44] Ionic, «Ionic CLI - Ionic Documentation», *Ionic Docs*. https://ionicframework.com/docs/cli (accedido dic. 01, 2020).
[45] *capacitor-community/react-hooks*. Capacitor Community, 2020.
[46] *ionic-team/pwa-elements*. Ionic, 2020.
[47] *ionic-team/ionic-framework*. Ionic, 2020.
[48] E. Lucas, «Announcing Ionic React Hooks», *Ionic Blog*, dic. 03, 2019. https://ionicframework.com/blog/announcing-ionic-react-hooks/ (accedido dic. 02, 2020).
[49] J. Watson, *JedWatson/classnames*. 2020.
[50] «Getting started i18next». https://www.i18next.com/overview/getting-started (accedido dic. 02, 2020).
[51] *i18next/react-i18next*. i18next, 2020.
[52] Ionic, «Ionicons: The premium icon pack for Ionic Framework». https://ionicons.com/ (accedido dic. 02, 2020).
[53] *primefaces/primeicons*. PrimeFaces, 2020.
[54] *primefaces/primereact*. PrimeFaces, 2020.
[55] «React – A JavaScript library for building user interfaces». https://reactjs.org/ (accedido dic. 02, 2020).
[56] B. Vaughn, *bvaughn/react-error-boundary*. 2020.
[57] «Home | React Hook Form - Simple React forms validation». https://www.react-hook-form.com/ (accedido dic. 02, 2020).
[58] «React Router: Declarative Routing for React.js». https://reactrouter.com/ (accedido dic. 02, 2020).
[59] V. Dalecky, *streamich/react-use*. 2020.
[60] *DutchConcepts/capacitor-barcode-scanner*. Dutch Concepts, 2021.
[61] «react-qr-code», *npm*. https://www.npmjs.com/package/react-qr-code (accedido ene. 05, 2021).
[62] «firebase», *npm*. https://www.npmjs.com/package/firebase (accedido ene. 05, 2021).
[63] «react-firebase-hooks», *npm*. https://www.npmjs.com/package/react-firebase-hooks (accedido ene. 05, 2021).

[64] «Transparencias — OCW - UC3M». http://ocw.uc3m.es/ingenieria-informatica/herramientas-de-la-inteligencia-artificial/contenidos/transparencias/ (accedido abr. 07, 2016).
[65] «All-In-One Cross-Platform Diagram Software for Flowchart, Org Chart and Mind Map». https://www.edrawsoft.com/ (accedido oct. 28, 2020).
[66] «Figma: the collaborative interface design tool.» https://www.figma.com/ (accedido oct. 28, 2020).
[67] «Zeplin—Collaboration and handoff for product teams». https://zeplin.io/ (accedido oct. 28, 2020).
[68] «React Top-Level API – React». https://reactjs.org/docs/react-api.html (accedido dic. 09, 2020).
[69] «Code-Splitting – React». https://reactjs.org/docs/code-splitting.html (accedido dic. 09, 2020).
[70] «JavaScript End to End Testing Framework», *JavaScript End to End Testing Framework | cypress.io*. https://www.cypress.io/ (accedido dic. 09, 2020).
[71] «react-particles-webgl», *npm*. https://www.npmjs.com/package/react-particles-webgl (accedido ene. 05, 2021).

Anexo 2. Entregables del proyecto

1. **Código del proyecto en un comprimido**: “PEC_FINAL_prj_Bejar_Caceres_David.zip”. Incluye todas las imágenes y diseños usados dentro del proyecto.
2. **Presentación del proyecto en PowerPoint**: “PEC_FINAL_prs_Bejar_Caceres_David.pptx”.
3. **Video de la presentación**: Plataforma Present@ del aula virtual.

Anexo 3. Código fuente (extractos)

Selección de partes relevantes del código fuente del servicio/aplicación creado, descripción y comentarios.

Anexo 4. Tareas Jira del proyecto

[SC-1] Login page with form Created: 13/Nov/20 Updated: 16/Nov/20 Resolved: 16/Nov/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	2 days, 4 hours		
Original Estimate:	Not Specified		

Attachments:	image-20201113-130926.png
Sprint:	
Rank:	0|hzzzzz:

Description

- Login Page.
- Form to login with: Username Input, password Input, Login Button.
- Background implement particleJs effect, you have to test with following libraries:
  1) https://timellenberger.com/particles
  2) https://vincentgarreau.com/particles.js/#bubble
  3) https://rpj.bembi.org/#simple

Comments

Comment by David Bejar Caceres [13/Nov/20]

[SC-2] Login Form add validation Created: 14/Nov/20 Updated: 02/Jan/21 Resolved: 16/Nov/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	3 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i00007:

Description

- Select the best fitting Form validation library NPM package.
- Validate Login form for username and password fields.
- password min lenght should be 6.
- username field min lenght 4.
- Add error messages for fields validation.
- If posible make a suttle backgriund color animation if error in login.

[SC-3] Add SignUp form in LoginPage.js Created: 14/Nov/20 Updated: 02/Jan/21 Resolved: 09/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	6 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0000f:

Description

- Add another form for sign up new user.
- This form should appear insted of Login form after user clicks sign up button belows login button.
- Add basi validation for sign up form.

Mandatory fields:

- Username
- Password
- Repeat password
- Name
- Lastname
- City
- Age
- default pool size , 25 or 50 meters (by defult 25 meters pool)
- mail

[SC-4] Protect app using auth with login page, if user not loged in, redirects to LoginPage.

Created: 14/Nov/20 Updated: 02/Jan/21 Resolved: 16/Nov/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	6 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0000n:

Description

Protect all components if no user.
All routes must be protected except for the Error Boundary.
When Login page is present, left Menu component should be hidden.

[SC-5] Groups Page with responsive grid of cards for groups of swimmers Created: 14/Nov/20 Updated: 02/Jan/21 Resolved: 02/Jan/21

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	4 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0000v:

Description

- Groups are present for each coach.
- Each group is a Card with: groupName, poolSize (25/50 meters), workoutDays, workoutHours, swimmersCount, nextSessionMeters.
- Floating button for adding a new Group.

[SC-6] Add toast messages for user interactions

Created: 14/Nov/20 **Updated:** 02/Jan/21 **Resolved:** 16/Nov/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	6 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i00013:

Description

If user cliks in warning icon of a not valid field, toast should be shown for the error message.

Try woth this two libraries and decide the best one:
https://ionicframework.com/docs/api/toast

or

https://fkhadra.github.io/react-toastify/introduction/

[SC-7] Create list of swimmers in swimming-coach/Swimmers

Created: 15/Nov/20 **Updated:** 02/Jan/21 **Resolved:** 16/Nov/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 day, 7 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0001b:

Description

List of swimmers in all groups of the coach.

In case of Desktop screen we need a grid of users using 2 rows

[SC-8] Plan Firebase collections and structure

Created: 15/Nov/20 **Updated:** 02/Jan/21 **Resolved:** 19/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	2 days		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0001j:

Description

Plan Collections for Coach, groups, swimmers, sessions, relations betweenthem.

Comments

Comment by David Bejar Caceres [19/Dec/20]

- Users collection
- Groups collection
- Sessions collection

[SC-9] Connect to Firebase to auth Created: 15/Nov/20 Updated: 02/Jan/21 Resolved: 17/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	2 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0001r:

Description

Use Firebase to Auth login and sign up

[SC-10] When user chooses to remember login, disable auto logout from app

Created: 15/Nov/20 **Updated:** 02/Jan/21 **Resolved:** 16/Nov/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	30 minutes		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0001z:

Description

By default REFRESH_USER_TIME is set to 10 minutes, if user chooses to remember account in Login, disable auto logout.

[SC-11] Improve Login page styles for Dark theme and minor visuals

Created: 15/Nov/20 **Updated:** 02/Jan/21 **Resolved:** 24/Nov/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	5 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i00027:

[SC-12] Add dark mode Created: 15/Nov/20 Updated: 02/Jan/21 Resolved: 16/Nov/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0002f:

Description

Dark mode should be selectable from the left panel in configurations.

[SC-13] Add native storage persistance instead of window.localStorage

Created: 16/Nov/20 **Updated:** 02/Jan/21 **Resolved:** 02/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	4 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0002n:

Description

window.localstorage can be recycled by the OS, in android can do it from third party actions, iOS is more agressive, it can be deleted at any moment.

[SC-14] Add user session persistance in case user wants to remember session Created: 16/Nov/20 Updated: 02/Jan/21 Resolved: 16/Nov/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0002v:

Description

this have to disable the auto logout after the standard REFRESH_USER_TIME.

[SC-15] Improve navigation for GroupsPage and SwimmersPage to have own url path and custom toolbar

Created: 21/Nov/20 **Updated:** 02/Jan/21 **Resolved:** 24/Nov/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	2 minutes		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i00033:

[SC-16] Add user page confg Created: 22/Nov/20 Updated: 02/Jan/21 Resolved: 24/Nov/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	7 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0003b:

Description

Config should be opened after click in the config on left Menu Icon Button

[SC-17] Delete Confirmation dialog for swimmers and groups using modal Created: 24/Nov/20 Updated: 02/Jan/21 Resolved: 25/Nov/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	30 minutes		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0003j:

[SC-18] Upgrade to create-react-app 4 Created: 25/Nov/20 Updated: 02/Jan/21 Resolved: 02/Jan/21

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	3 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0003r:

Description

https://blog.logrocket.com/whats-new-in-create-react-app-4-0-0/

Comments

Comment by David Bejar Caceres [02/Dec/20]

https://blog.logrocket.com/whats-new-in-create-react-app-4-0-0/

[SC-19] Edit swimmer form Created: 02/Dec/20 Updated: 02/Jan/21 Resolved: 09/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	6 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0003z:

Description

1) Edit swimmer form.
2) For desktop is better if we use a Dialog, for phones better a new Page.

Follow example as User config page/dialog.

[SC-20] Group edit form Created: 02/Dec/20 Updated: 02/Jan/21 Resolved: 09/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 day, 1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i00047:

Description

1) Edit group form.
2) For desktop is better if we use a Dialog, for phones better a new Page.

Follow example as User config page/dialog.

[SC-21] Group detail Page Created: 02/Dec/20 Updated: 02/Jan/21 Resolved: 26/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 day		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0004f:

Description

Group detail page. This will tisplay list of sessions in group.
After user clicks on a Group, a Group detail page is opened.
Group page needs to show a button to edit it.
Edit Buttton on desktop has to be in top bar right, in phones we can use a Floating buttom in right bottom of screen.

[SC-22] List of workout sessions Component

Created: 02/Dec/20 Updated: 02/Jan/21 Resolved: 26/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	4 minutes		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0004n:

Description

We need to create a list of sessions, this can be used to display sessions in a group or a swimmer.
In sessions list we need a new button to create a new session.

[SC-23] Session Detail Page Created: 02/Dec/20 Updated: 02/Jan/21 Resolved: 26/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	3 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0004v:

Description

Session Detail page should be opened after user selects a session from list of sessions. displays all the info of the session.

[SC-24] Add Cypress Groups Swimmers Login and Register Pages

Created: 02/Dec/20 **Updated:** 02/Jan/21 **Resolved:** 29/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 day		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i00053:

Description

https://www.cypress.io/

[SC-25] Remove console logs from project or show them only in dev mode, not production

Created: 02/Dec/20 Updated: 02/Jan/21 Resolved: 02/Jan/21

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	30 minutes		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0005b:

Description

```
if (!process.env.NODE_ENV || process.env.NODE_ENV === 'development') {
// dev code
} else {
// production code
}
```

[SC-26] Improve left Menu Created: 03/Dec/20 Updated: 02/Jan/21 Resolved: 09/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0005j:

Description

- We need the menu to be scrollable in phones during landscape, each list should be scrollable.
- If button config page in desktop is clicked, Config dialog should start from left to right animation, and the final position has to be to the left of the panel for a better UX.

[SC-27] Login page on phone, does not adapts to heigh when keyboard is opened. Created: 04/Dec/20 Updated: 02/Jan/21 Resolved: 09/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Bug	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour, 1 minute		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0005r:

Description

- Some very small screen devices will suffer when keyboard is opened.
- No scrollbar is shown when keyboard is opened and elements are behind keyboard.

[SC-28] Refactor LoginPage to separate Login Form and Sign-up Form

Created: 08/Dec/20 **Updated:** 02/Jan/21 **Resolved:** 08/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	7 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0005z:

Description

- Login form should be a separate component.
- Errors, validation, submit should be controlled from LoginPage component.
- Add Sign up component with all the login from LognPage to control erros, toast, validation.

[SC-29] Add new group dialog and page Created: 08/Dec/20 Updated: 02/Jan/21 Resolved: 09/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	2 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i00067:

Description

- New group dialog for deskptop
- New Group page for phones.

[SC-30] New Swimmer dialog and page Created: 09/Dec/20 Updated: 02/Jan/21 Resolved: 09/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	3 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0006f:

Description

- Dialog for desktop.
- Page for phones.

[SC-31] Protected SubRoutes not working as intender Created: 09/Dec/20 Updated: 02/Jan/21 Resolved: 09/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Bug	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	3 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0006n:

Description

Fix sub page navigation like:
/swimmers/swimmer-form
instead of /swimmer-form

[SC-32] Search input in Groups and swimmers page should filter in list

Created: 09/Dec/20 **Updated:** 02/Jan/21 **Resolved:** 09/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	3 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0006v:

Description

- Swimmers should be filter by name and lastname.
- Groups should be filtered by name.

[SC-33] Add PWA page title and icon in browser, android app name

Created: 13/Dec/20 **Updated:** 02/Jan/21 **Resolved:** 15/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i00073:

Description

- title should be visible in browser tab.
- page title should be translated to locale.
- Icon should be visible in browser tab.
- Add name to the android app.
- Add Icon to the android app.

[SC-34] Encrypt password Created: 17/Dec/20 Updated: 02/Jan/21 Resolved: 17/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	2 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0007b:

Description

User saved to firebase should encrypt password, currently in plain text.

[SC-35] Add register form field about user type coach or swimmer

Created: 17/Dec/20 **Updated:** 02/Jan/21 **Resolved:** 17/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0007j:

Description

- user type field should be added to register form
- coach or swimmer.

[SC-36] Create groups Created: 17/Dec/20 Updated: 02/Jan/21 Resolved: 17/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0007r:

Description

- Save info of groups from GroupForm.

[SC-37] Display firebase groups in groups page Created: 17/Dec/20 Updated: 02/Jan/21 Resolved: 19/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0007z:

Description

Groups in firebase should be listed in Groups Page.

Probably we need to add HOC to Groups page to handle state and data.

[SC-38] Firebase swimmers for user

Created: 19/Dec/20 Updated: 02/Jan/21 Resolved: 23/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	3 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i00087:

Description

- Show coach swimmers from firebase (CRUD).

[SC-39] Create global toast Created: 19/Dec/20 Updated: 02/Jan/21 Resolved: 19/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0008f:

Description

- create a global toast component.
- Function should be able to show from every component.

[SC-40] Add share group code Created: 19/Dec/20 Updated: 02/Jan/21 Resolved: 20/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	2 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0008n:

Description

- Add share group code so swimmers can add group and see all sessions.
- Group UID should be on a dialog after context menu is selected.

[SC-41] Test capacitor qr code scanner and generator

Created: 20/Dec/20 **Updated:** 02/Jan/21 **Resolved:** 21/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	3 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0008v:

[SC-42] Bug when loading Swimmers page toast is shown

Created: 24/Dec/20 **Updated:** 02/Jan/21 **Resolved:** 24/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Bug	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i00093:

[SC-43] Workout page form for phones Created: 26/Dec/20 Updated: 02/Jan/21 Resolved: 26/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	2 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0009b:

Description

If phones we must use a new page for the workout form, currently only uses a dialog for deskptop.

[SC-44] Add Date for workout form so user can plan future workouts or past workouts **Created:** 26/Dec/20 **Updated:** 02/Jan/21 **Resolved:** 27/Dec/20	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0009j:

Description

Workout form needs a Date field, so user will be able to add future or past workouts sessions.

Don´t forget to parse to milliseconds before saving to firebase.

[SC-45] Workouts filter fo date range and show stats from group based on filtered

Created: 27/Dec/20 **Updated:** 02/Jan/21 **Resolved:** 27/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	3 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0009r:

Description

- Added a filter to select range of dates to filter workouts.
- Info card of group should total distance and total workouts based on filtered elements.

[SC-46] Disables actions for SWIMMER user type, Context Edit and delete

Created: 29/Dec/20 **Updated:** 02/Jan/21 **Resolved:** 29/Dec/20

Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	6 hours		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i0009z:

Description

disable context menu.
disable swipe left or right items to edit or delete.
Swimmer should be only allowed to view not write.
disable swimmers page
disables new group dialog.

[SC-47] Improve app for a PWA Created: 30/Dec/20 Updated: 02/Jan/21 Resolved: 02/Jan/21	
Status:	Done
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Done	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	1 hour		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i000a7:

Description

We have to make a few modifications before building the PWA, favicon, service worker and check the /public/manifest.json. This will provide a better score in Lightroom, currently we have this values:

83 Performance
77 Accessibility
100 Best Practices
73 SEO

[SC-48] Clean code for final presentation Created: 05/Jan/21 Updated: 05/Jan/21	
Status:	In Progress
Project:	Swimming Coach
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Story	**Priority:**	Medium
Reporter:	David Bejar Caceres	**Assignee:**	David Bejar Caceres
Resolution:	Unresolved	**Votes:**	0
Labels:	None		
Remaining Estimate:	0 minutes		
Time Spent:	30 minutes		
Original Estimate:	Not Specified		

Sprint:	
Rank:	0|i000af:

Description

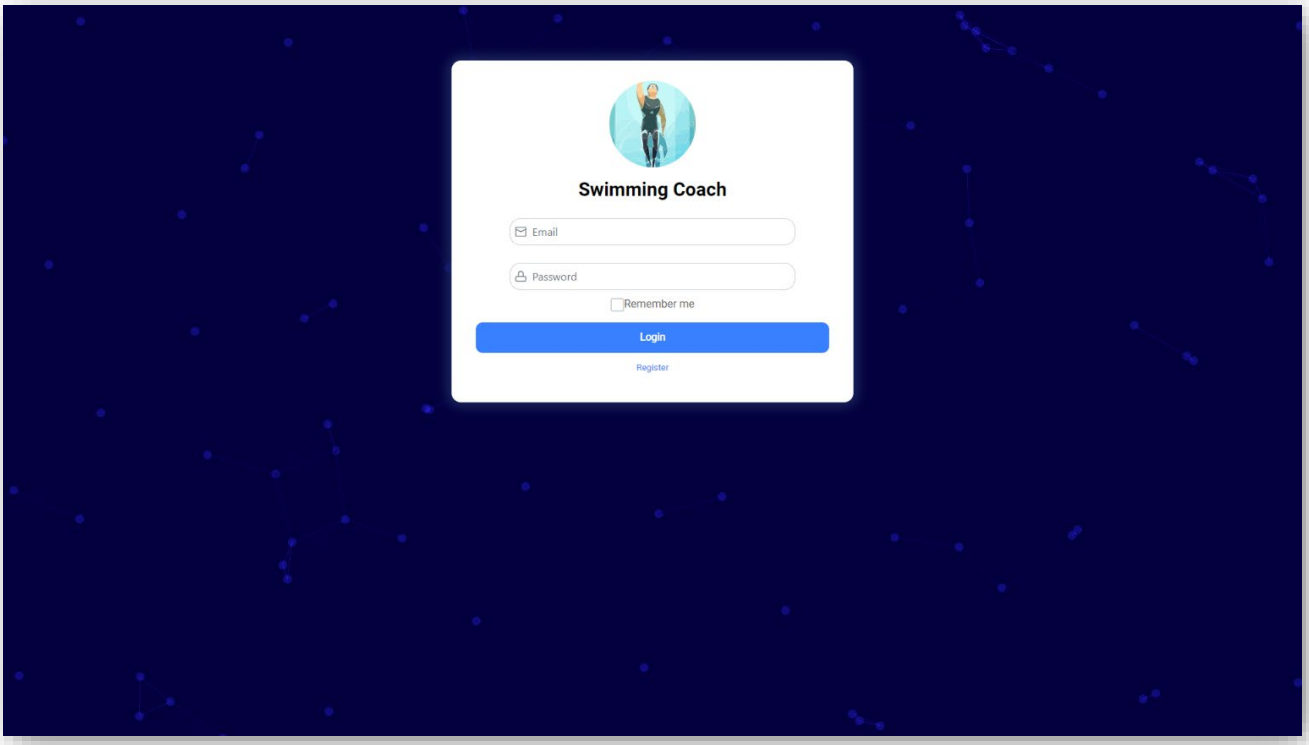
- Remove Cypress example test.
- Remove a few console.logs
- Remove unused imports
- Make sure all test are passing.

Generated at Tue Jan 05 11:36:23 UTC 2021 by David Bejar Caceres using Jira 1001.0.0-SNAPSHOT#100153-sha1:c1bea39d3b9857c5dc7106b720dbab2879f27d97.

Anexo 5. Capturas de pantalla

5.1 Computador de escritorio temas claro/oscuro (resolución 1920x1080):

5.1.1 Pantalla inicio de sesió



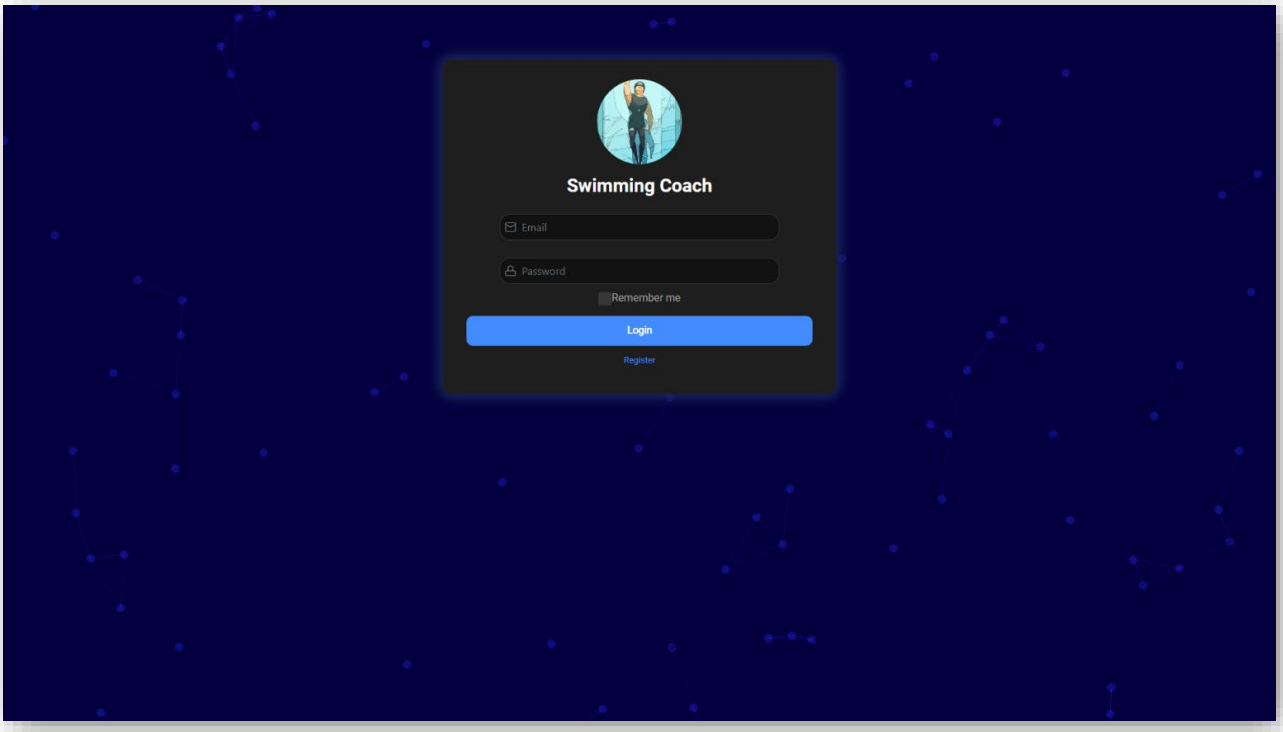
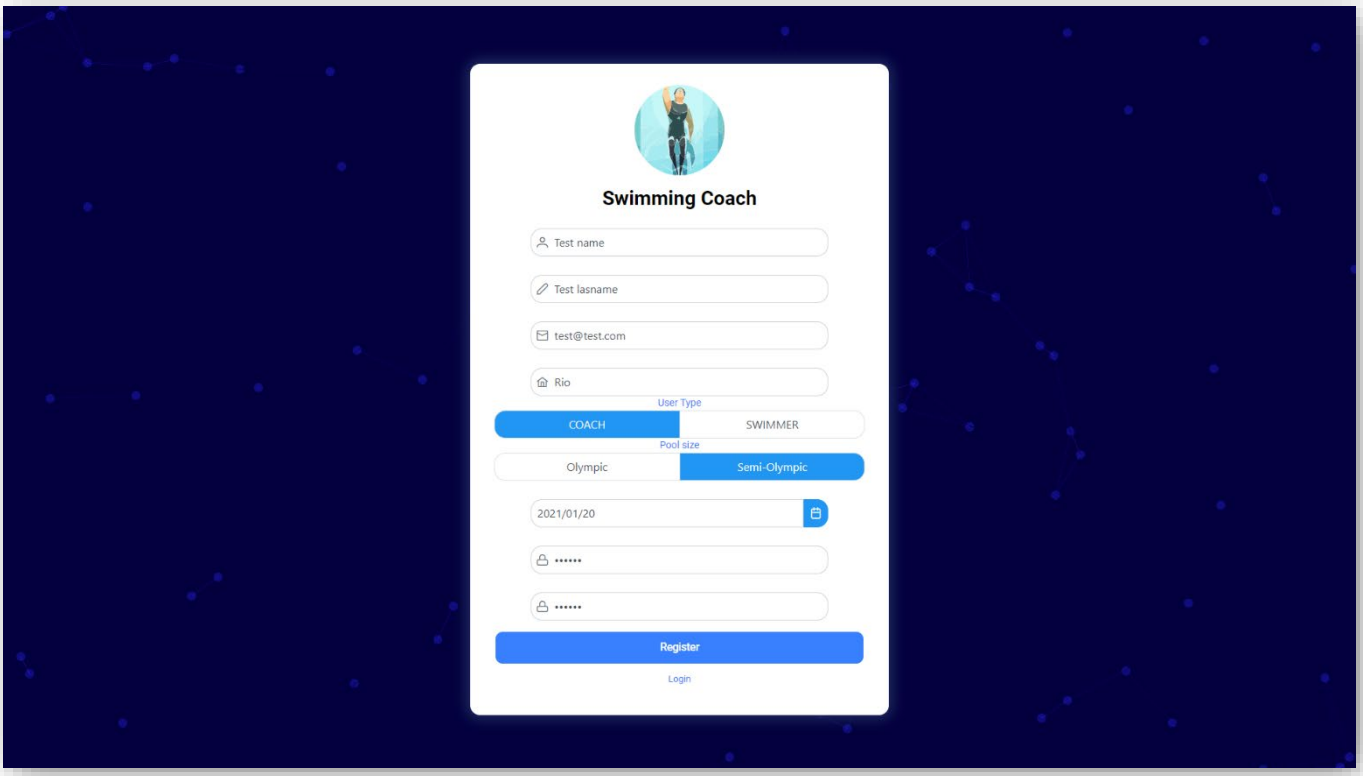


Figura 32: Pantalla inicio de sesión, escritorio.

5.1.2 Pantalla inicio sesión formulario registro de usuario:



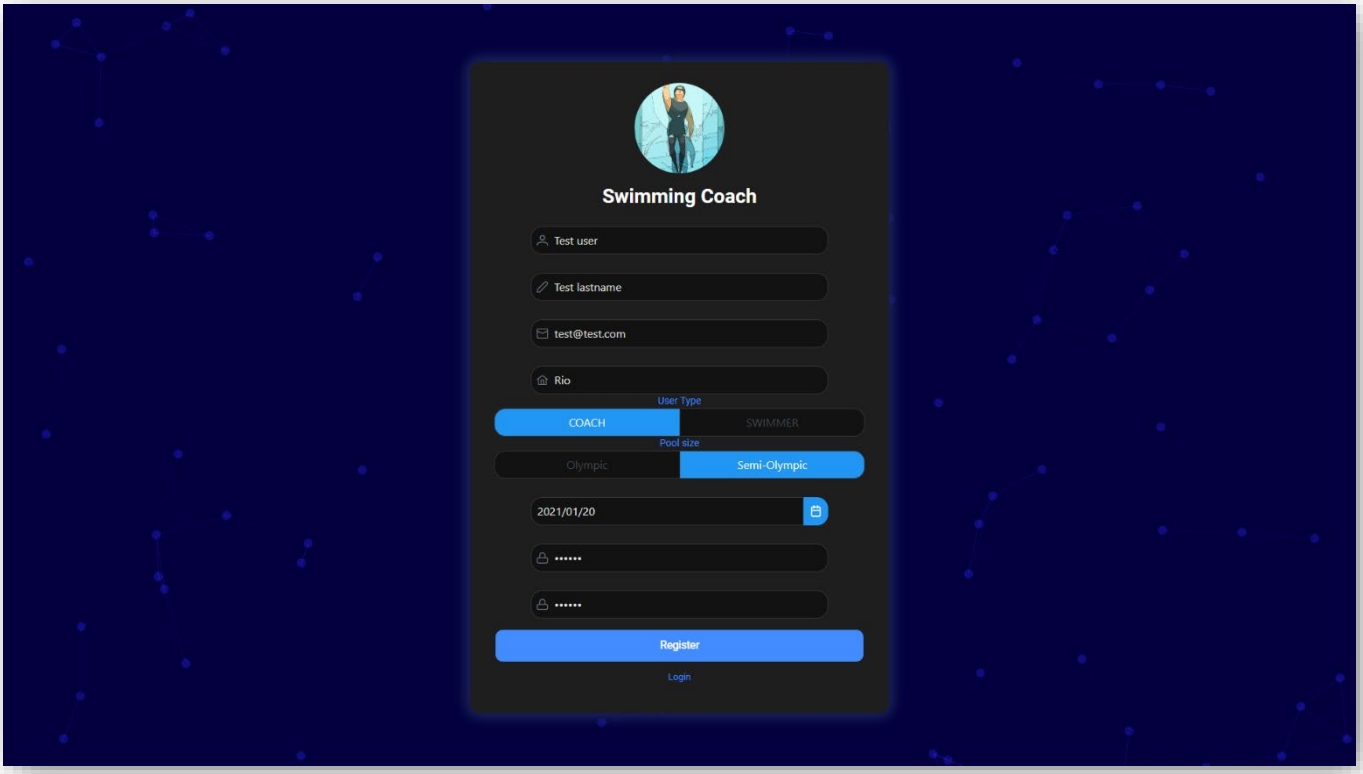


Figura 33: Pantalla inicio sesión formulario registro de usuario, escritorio.

5.1.3 Pantalla Grupos de entrenamiento:

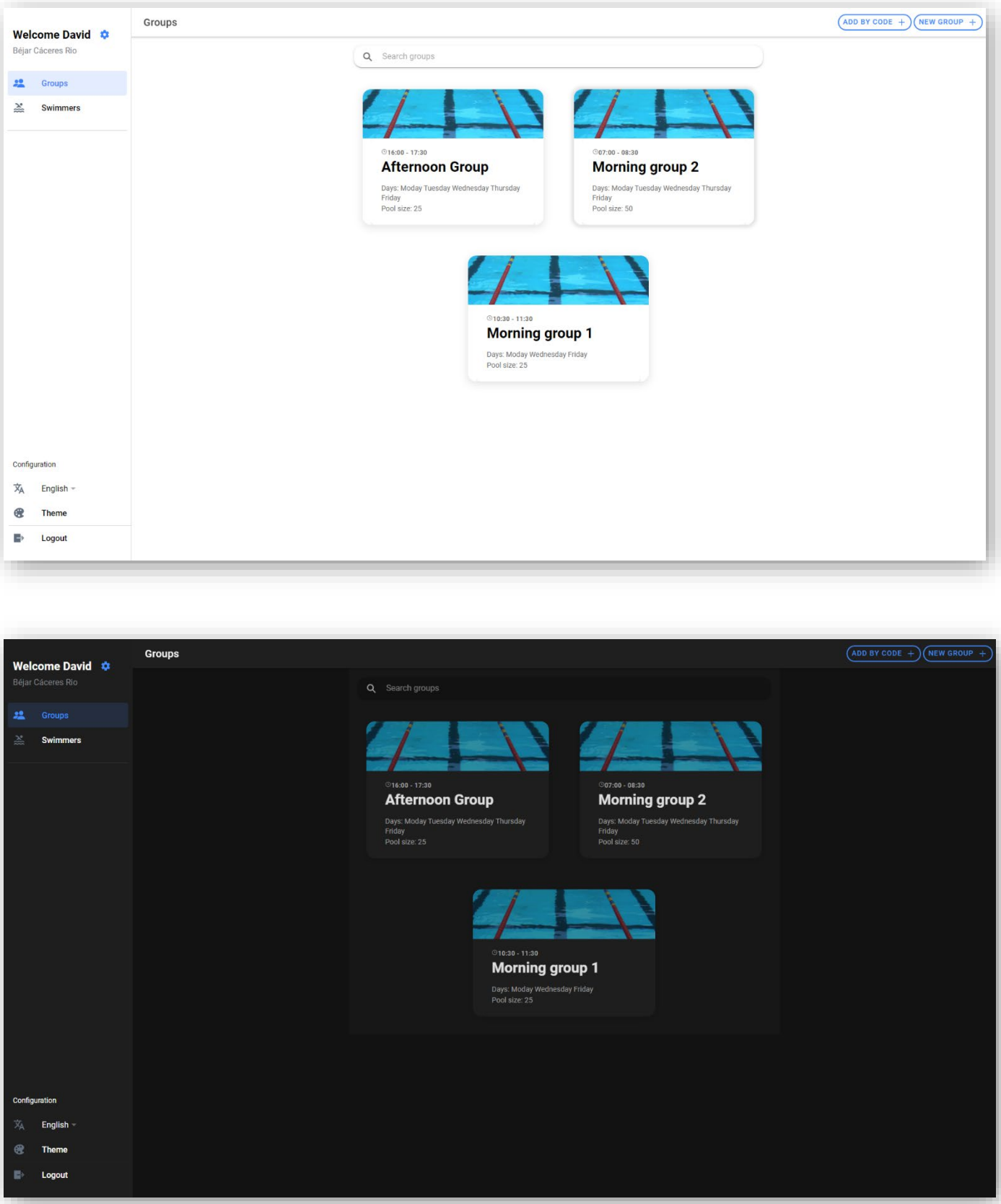


Figura 34: Pantalla Grupos de entrenamiento, escritorio.

5.1.4 Diálogo datos de usuario:

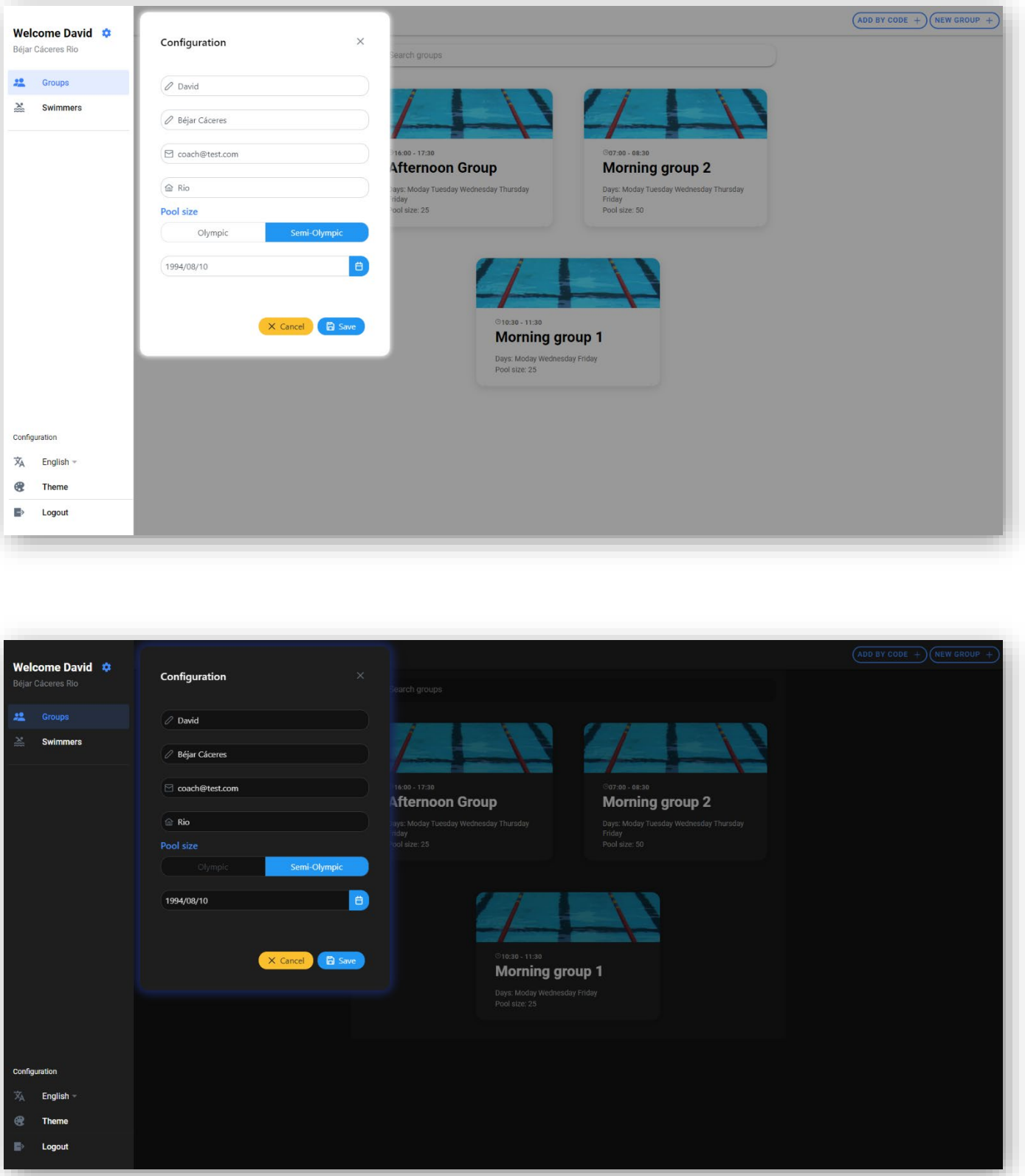


Figura 35: Diálogo datos de usuario, escritorio.

5.1.5 Diálogo crear/editar usuario:

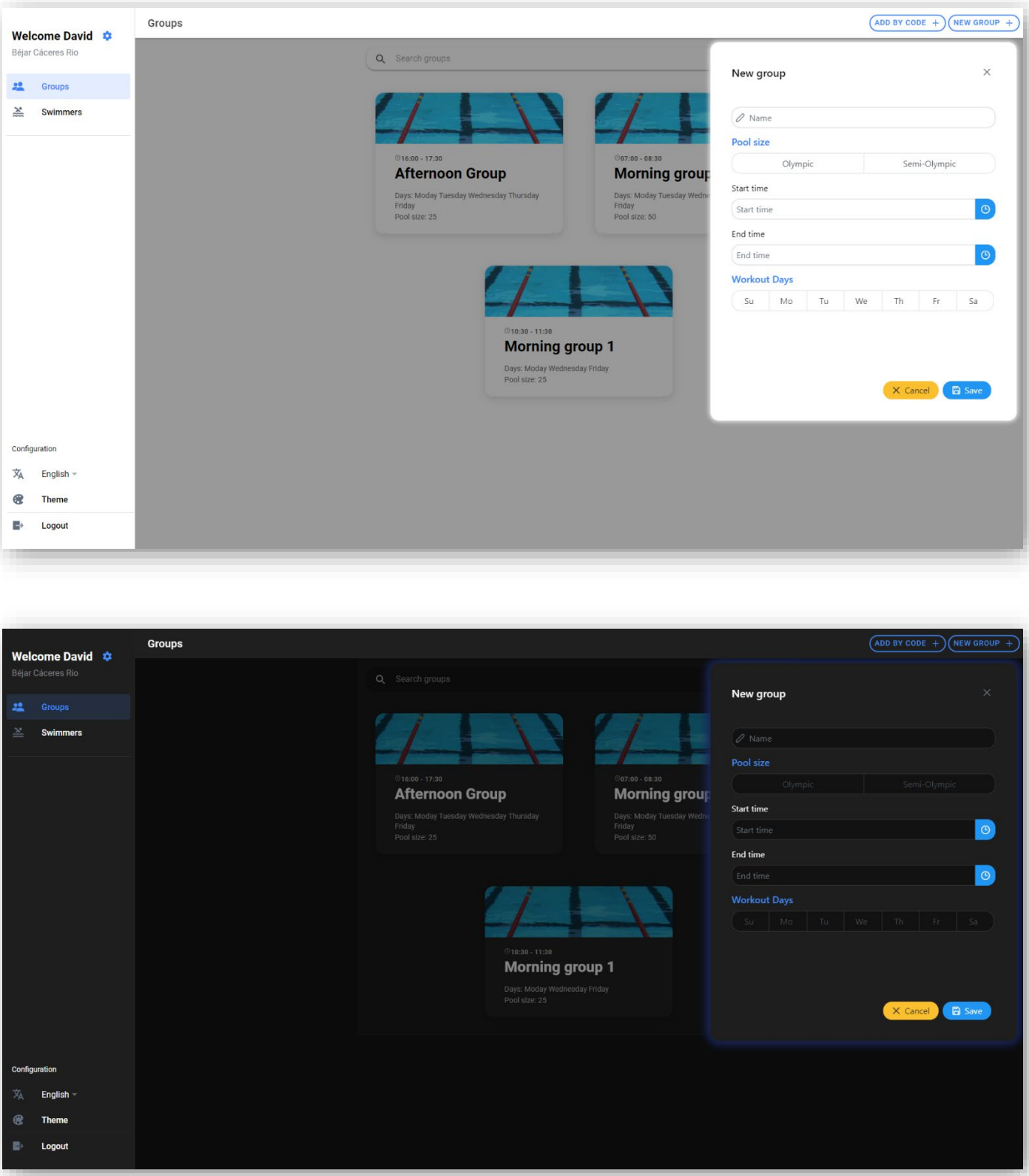


Figura 36: Diálogo crear/editar usuario, escritorio.

5.1.6 Diálogo añadir grupo por código para suscribirse:

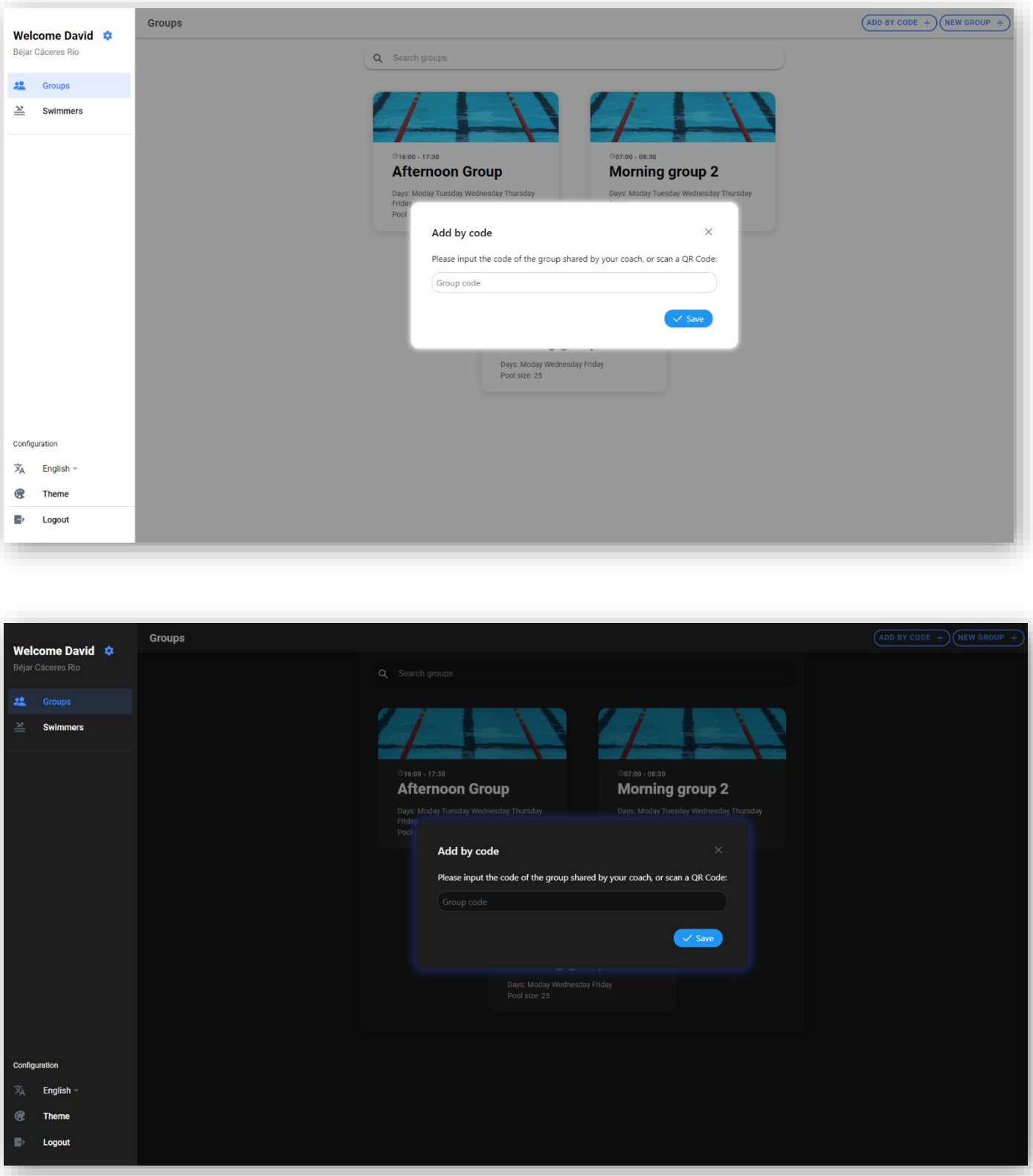


Figura 37: Diálogo añadir grupo por código para suscribirse, escritorio.

5.1.7 Lista de entrenamientos de grupo y estadísticas:

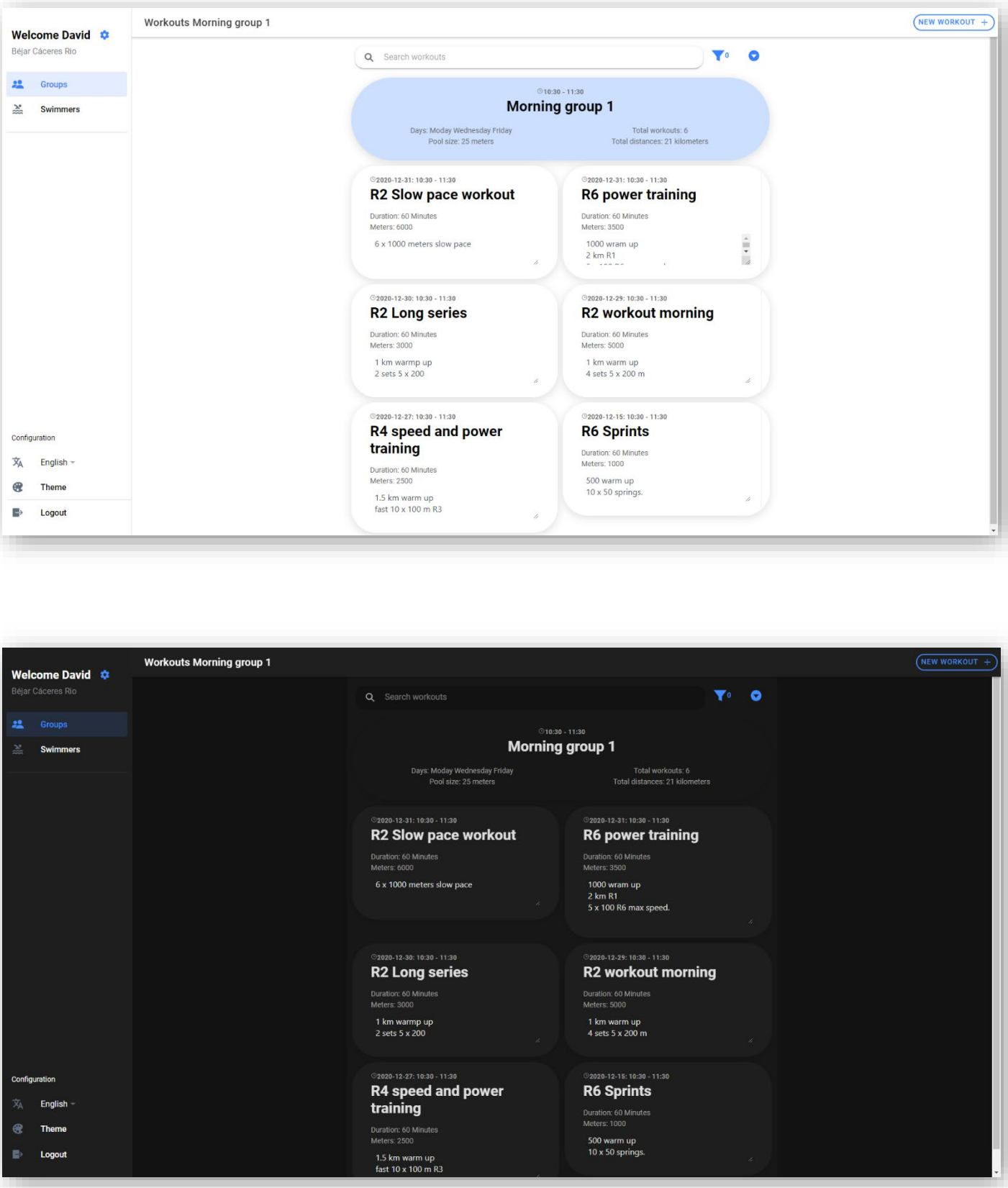


Figura 38: Lista de entrenamientos de grupo y estadísticas, escritorio.

5.1.8 Diálogo de filtros avanzados para entrenamientos de grupo:

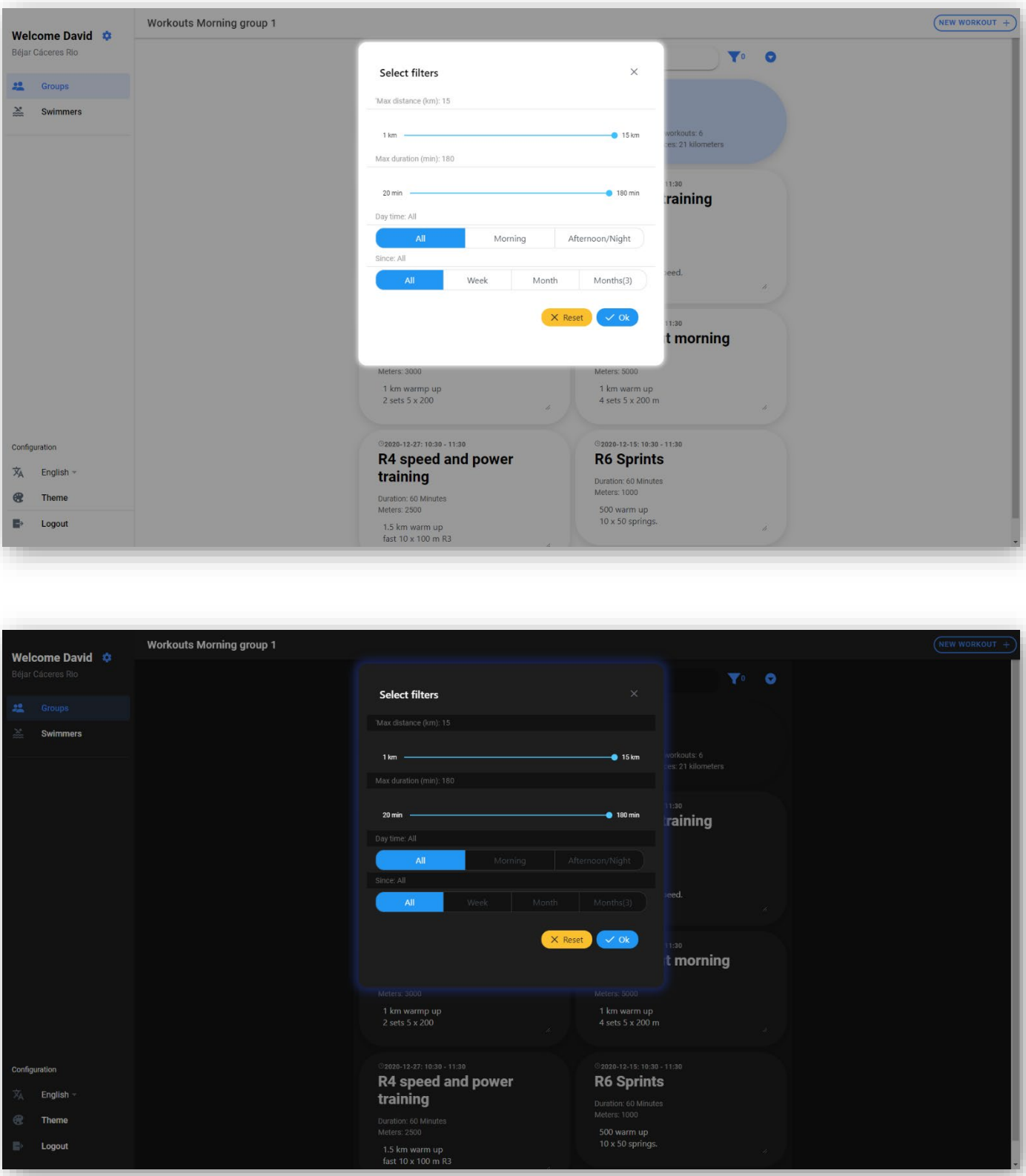


Figura 39: Diálogo de filtros avanzados para entrenamientos de grupo, escritorio.

5.1.9 Menú contextual sobre grupo de entrenamiento:

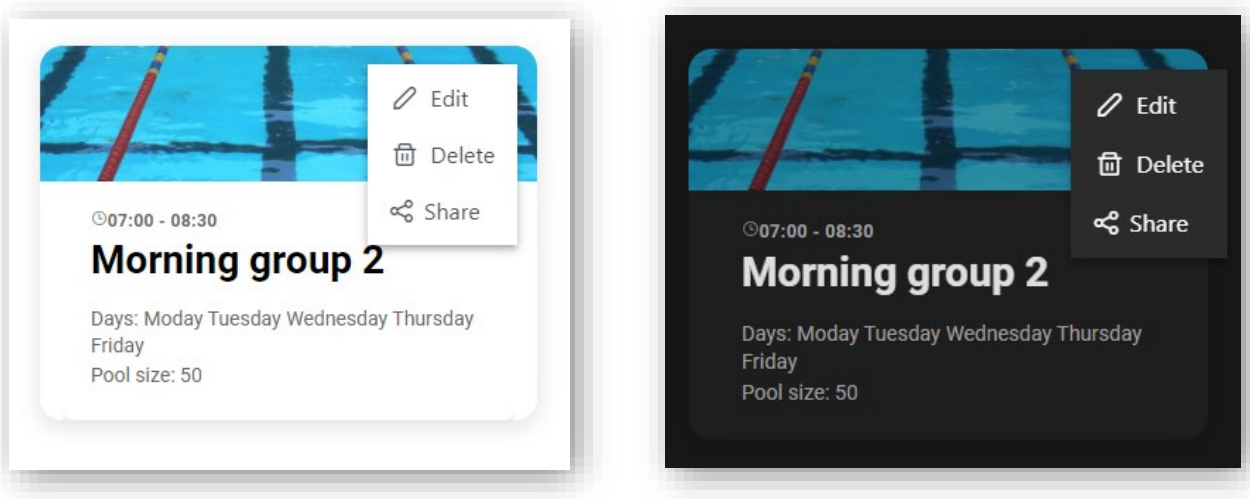


Figura 40: Menú contextual sobre grupo de entrenamiento, escritorio.

5.1.10 Diálogo selección de idioma:

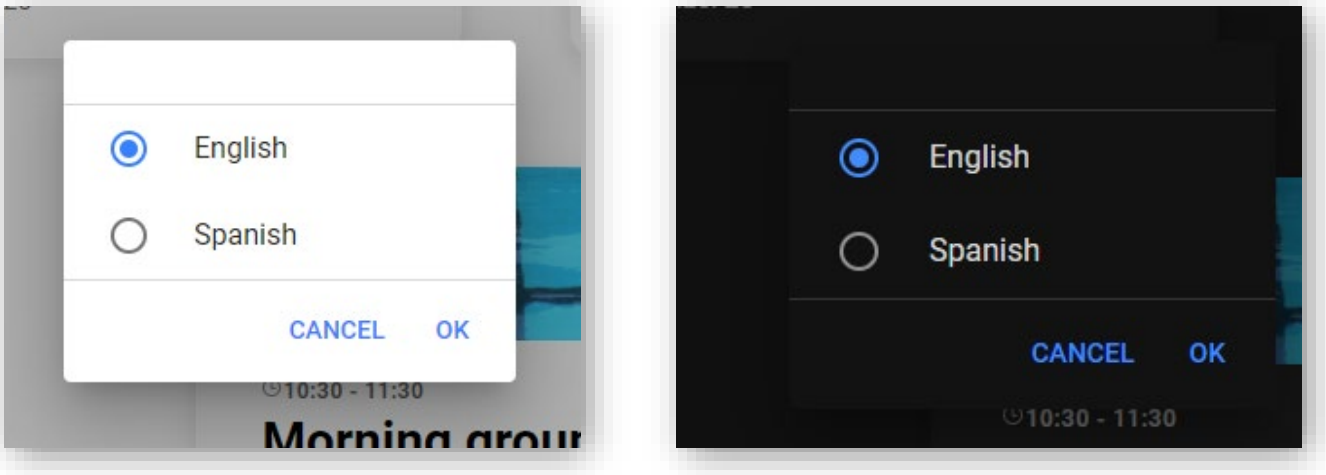


Figura 41: Diálogo selección de idioma, escritorio.

5.1.11 Página con lista de nadadores:

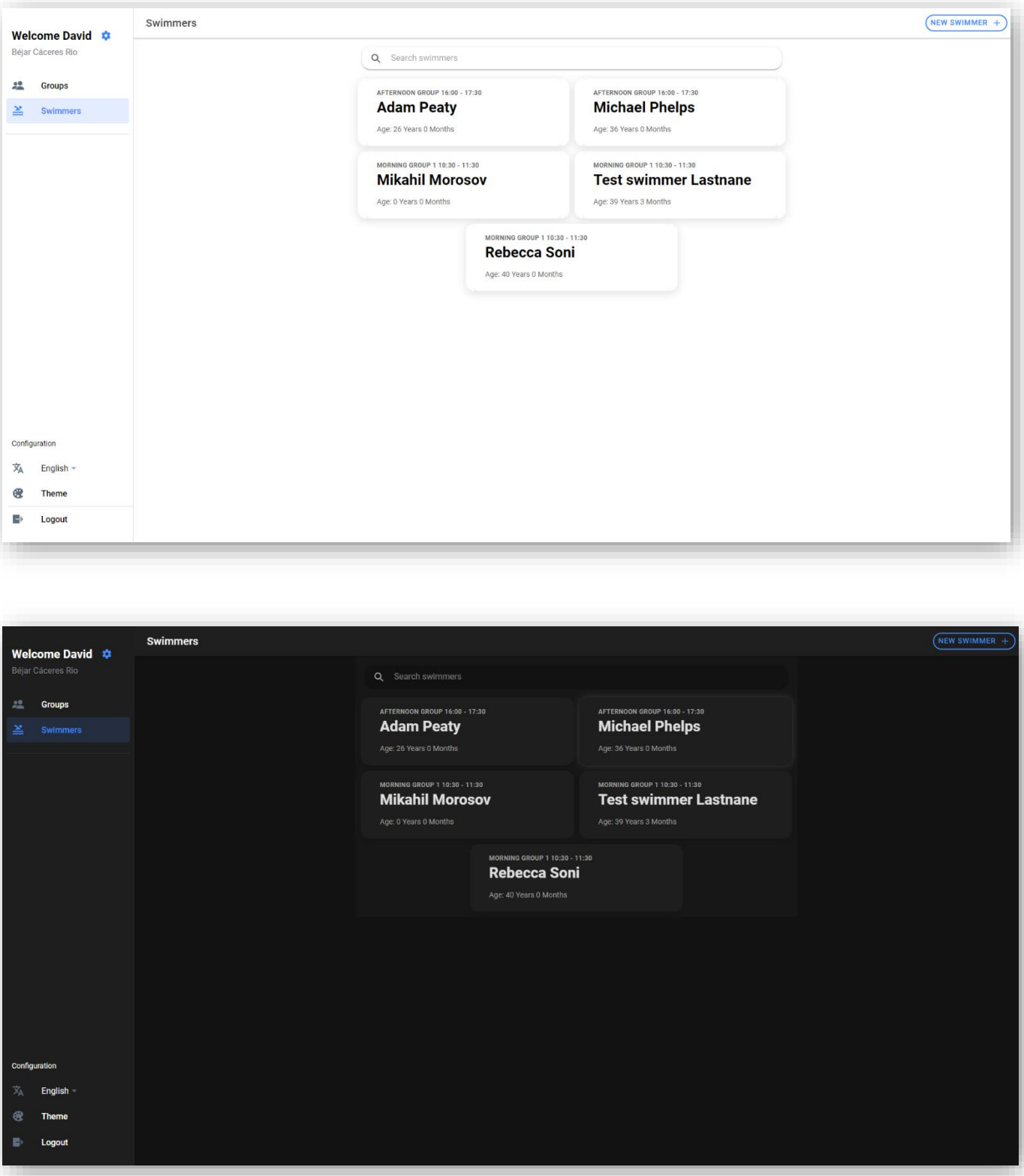


Figura 42: Página con lista de nadadores, escritorio.

5.1.12 Diálogo crear/editar nadador:

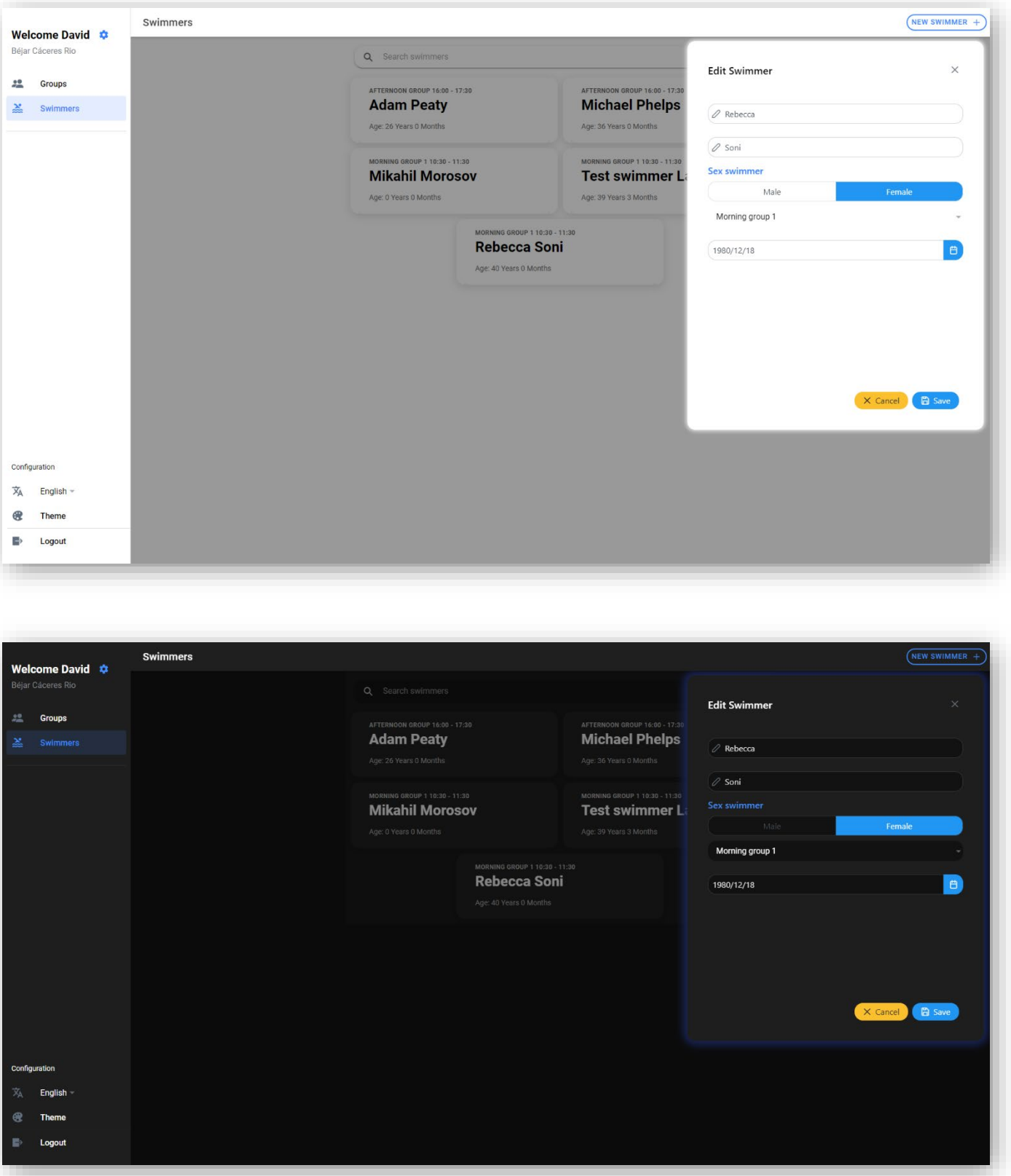


Figura 43: Diálogo crear/editar nadador, escritorio.

5.1.13 Diálogo compartir grupo:

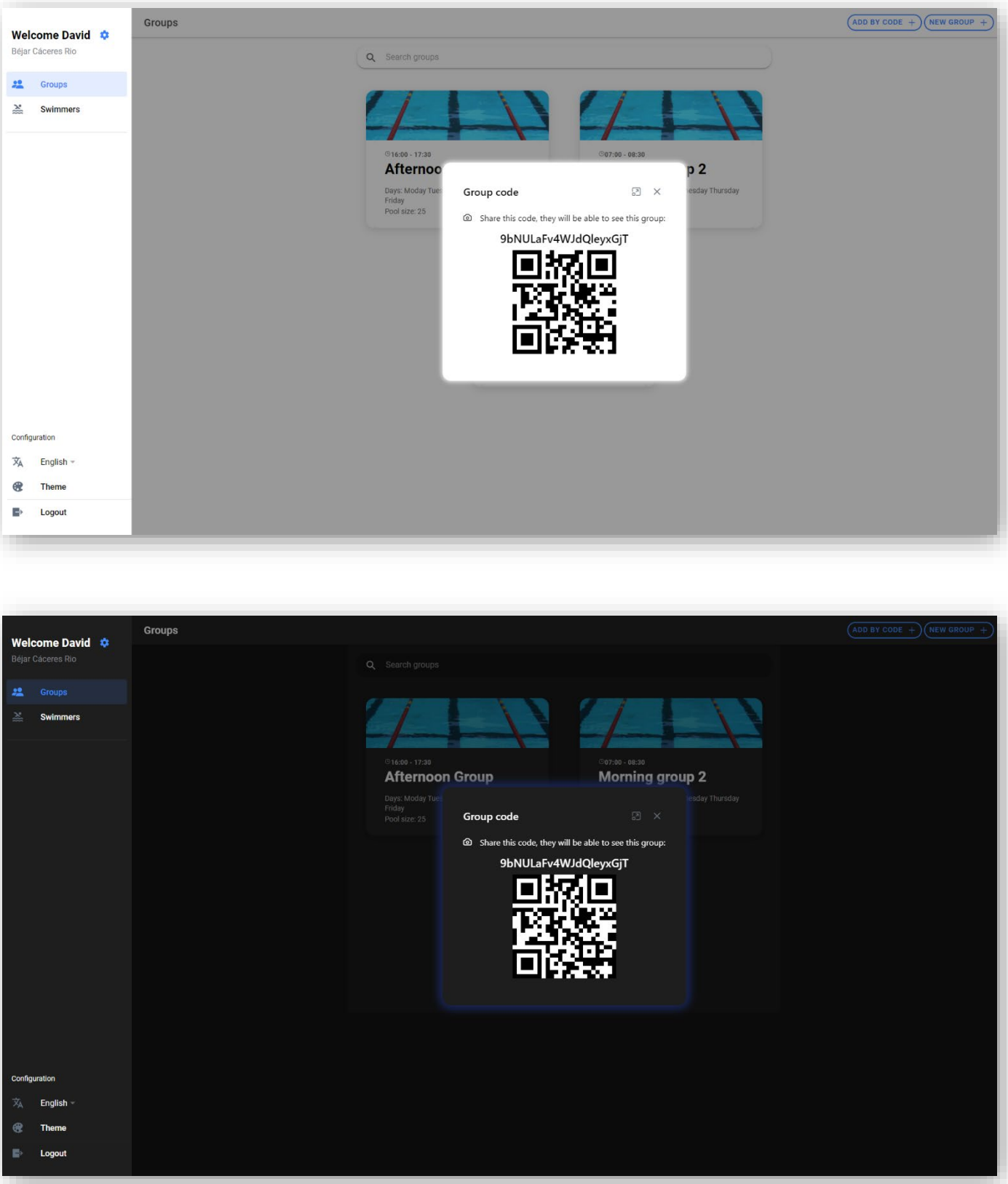


Figura 44: Diálogo compartir grupo, escritorio.

5.1.14 Diálogo confirmar eliminar elemento:

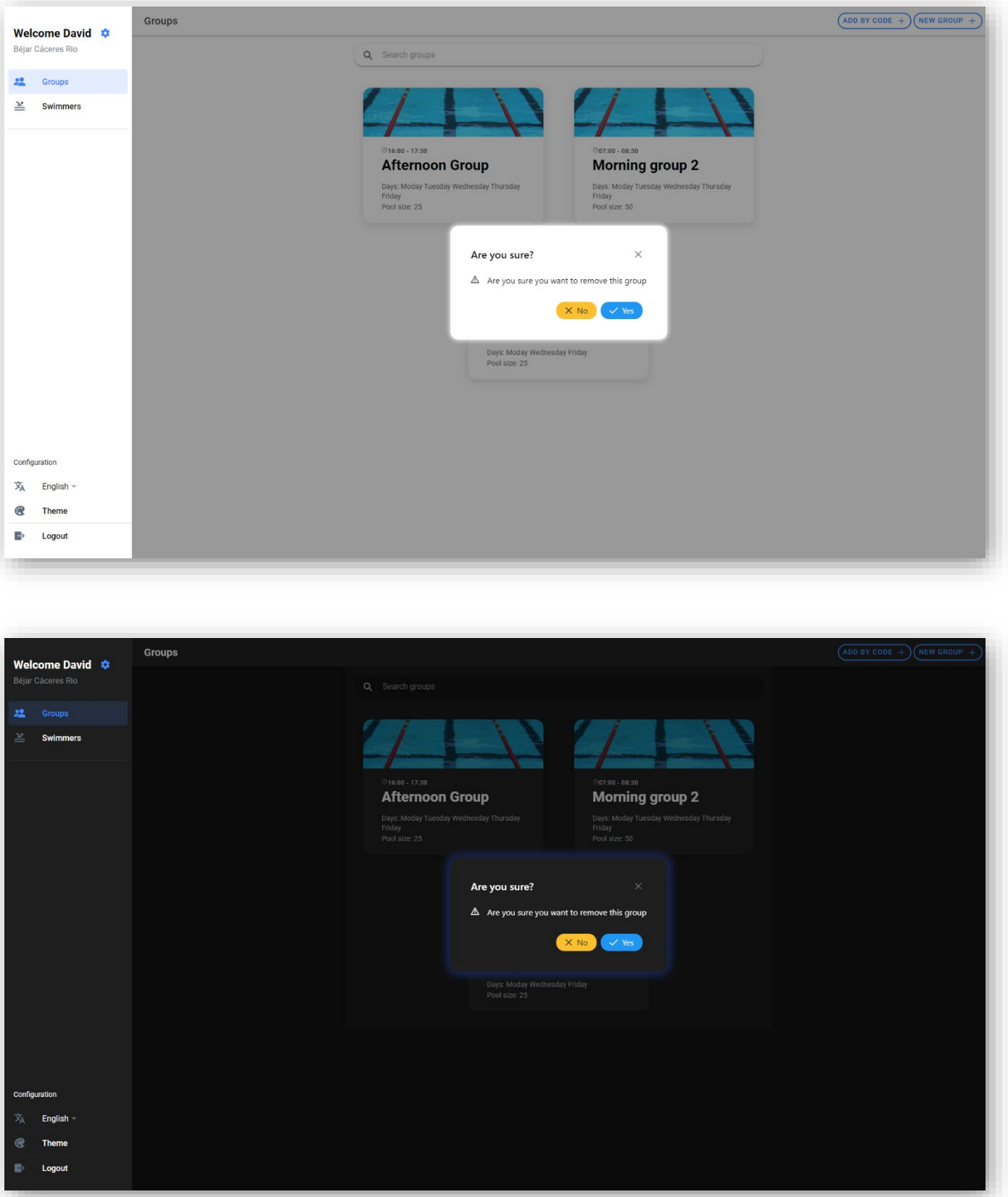


Figura 45: Diálogo confirmar eliminar elemento, escritorio.

5.1.15 Diálogo crear/editar entrenamiento:

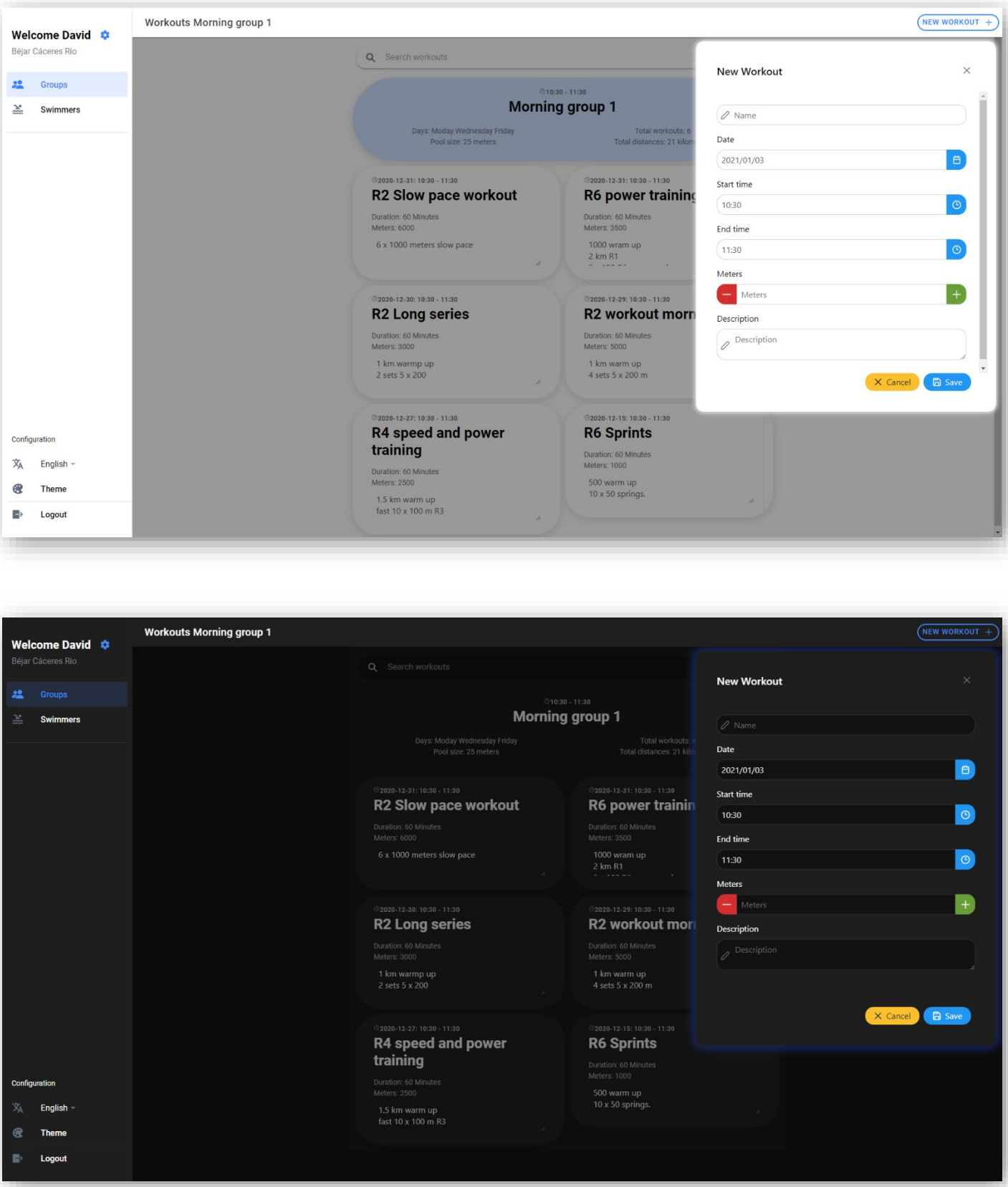


Figura 46: Diálogo crear/editar entrenamiento, escritorio.

5.1.16 Animaciones de carga de listas “Skelleton screen”:

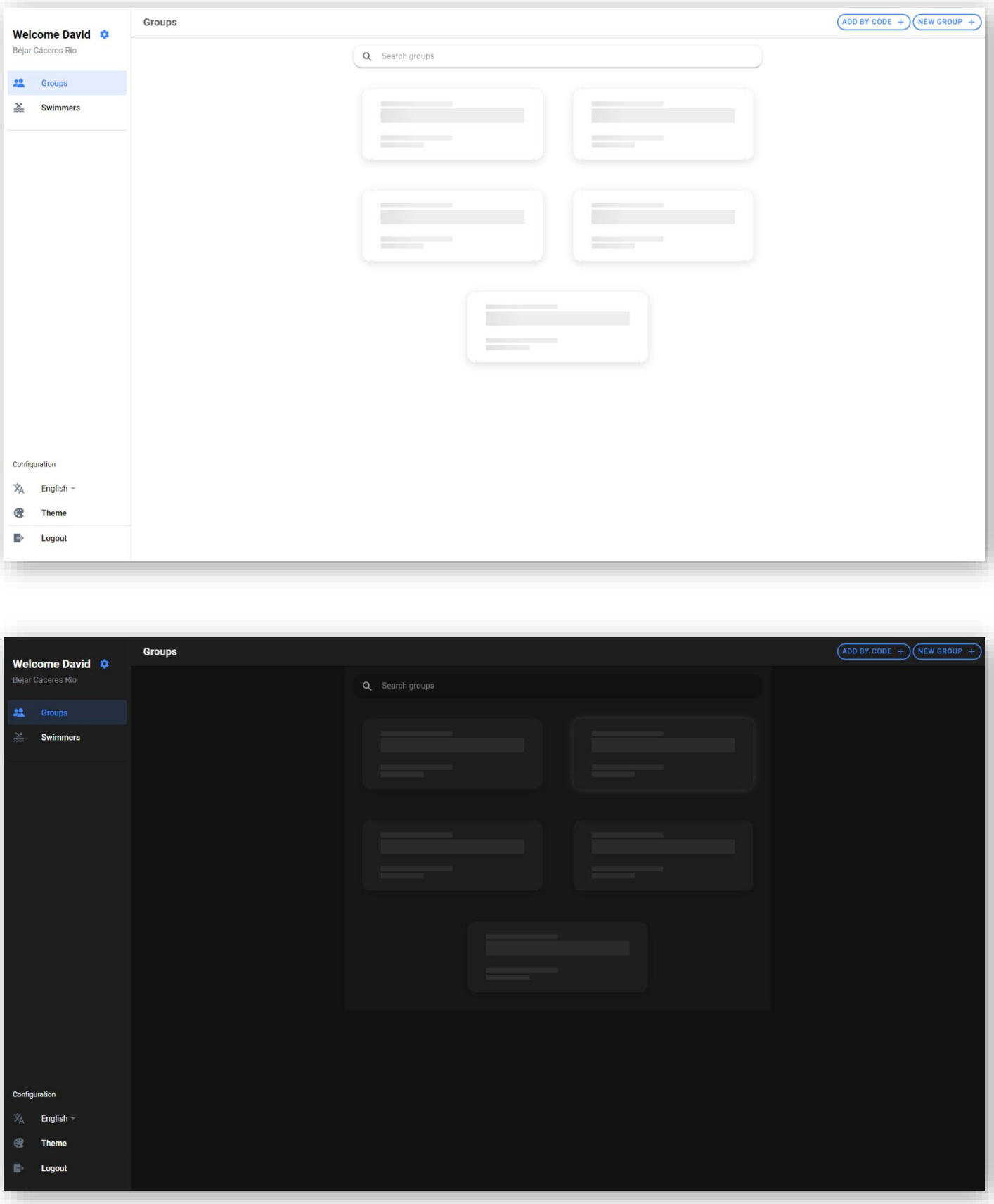


Figura 47: Animaciones de carga de listas "Skelleton Screen", escritorio.

5.2 Dispositivos móviles temas claro/oscuro (resolución 412 x 732 Nexus 6P)

5.2.1 Pantalla inicio de sesión:

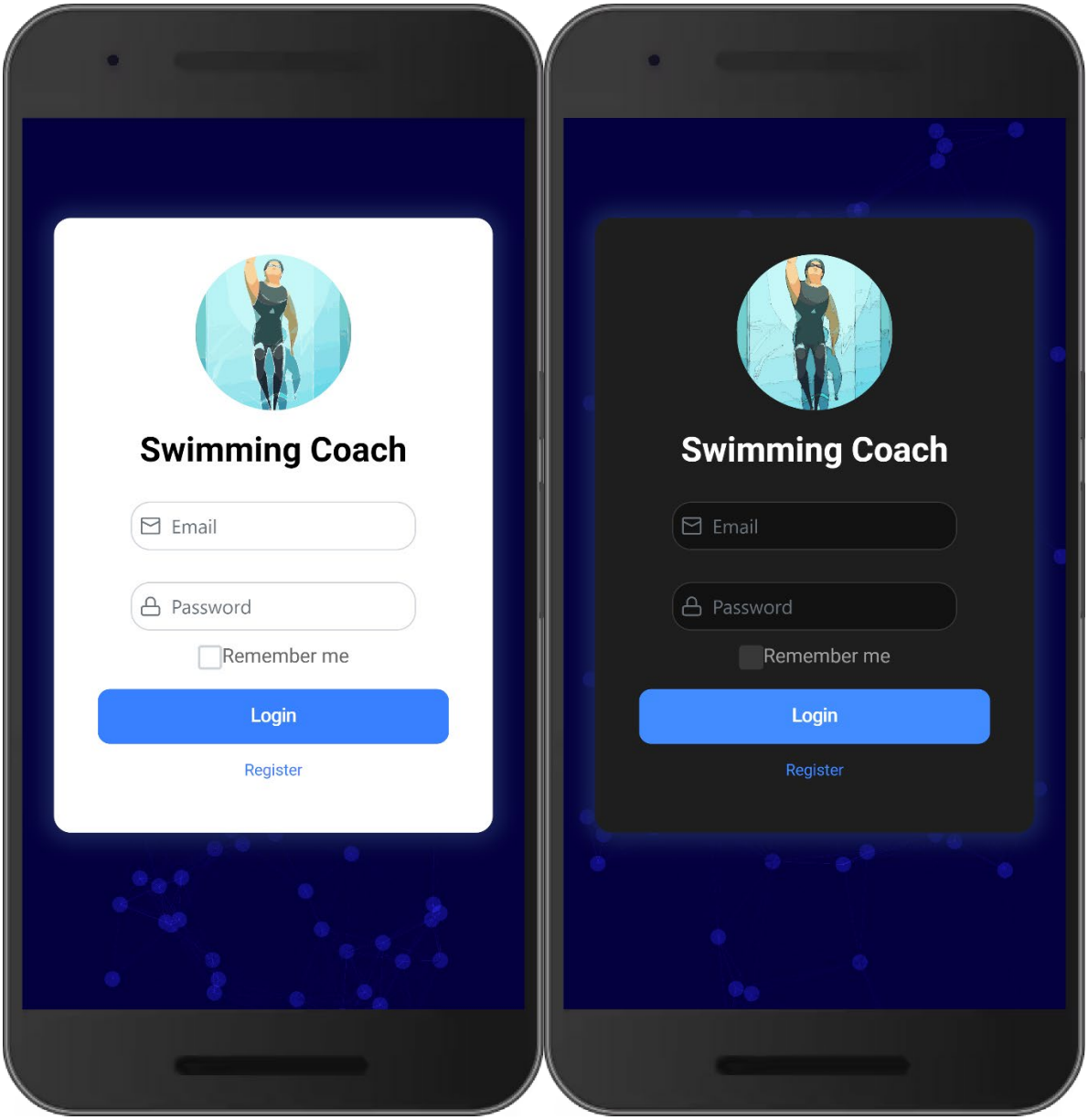


Figura 48: Pantalla inicio de sesión, móvil.

5.2.2 Pantalla inicio de sesión formulario registro de usuario:

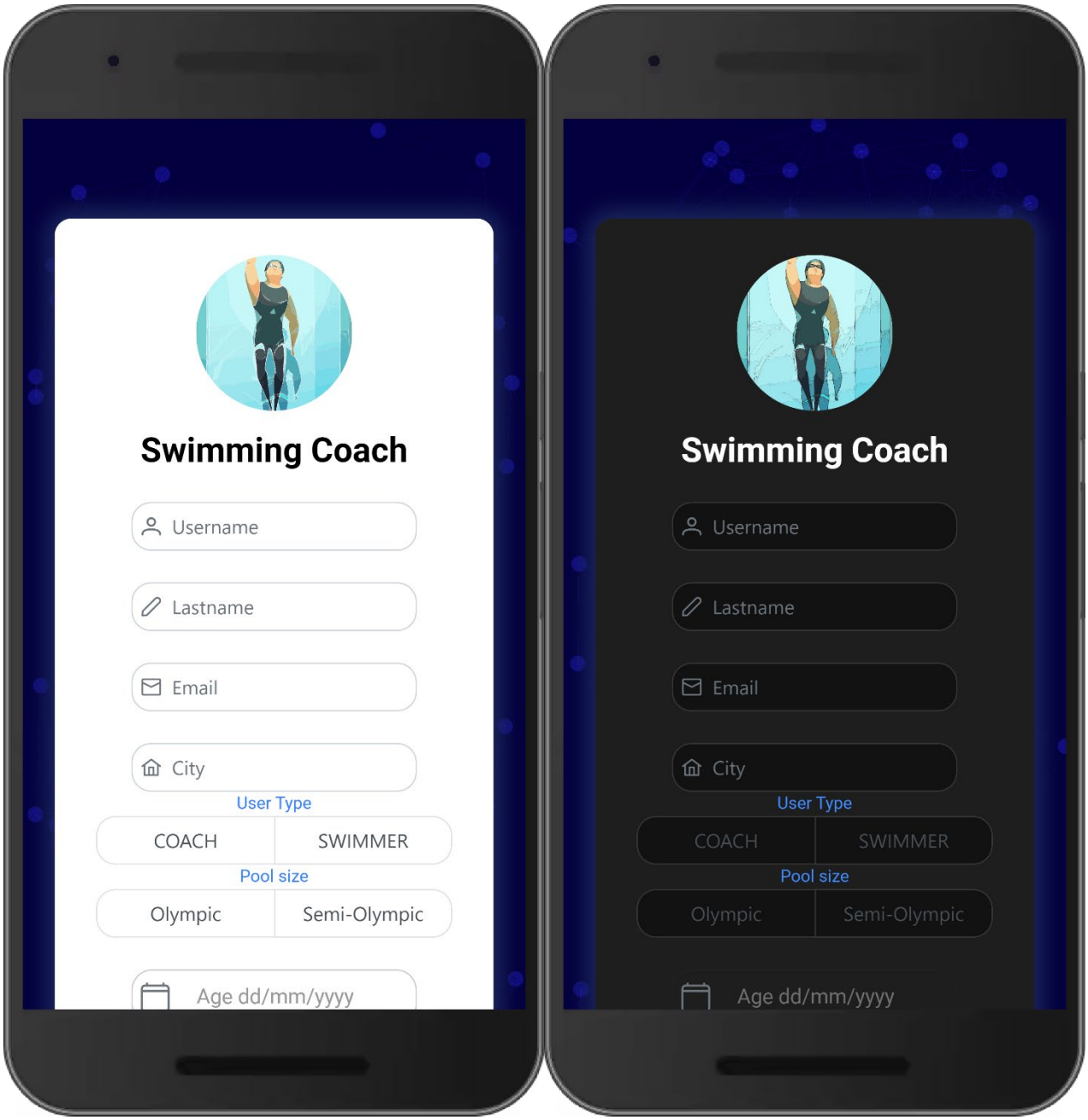


Figura 49: Pantalla inicio de sesión formulario registro de usuario, móvil.

5.2.3 Pantalla inicio de sesión feedback al usuario por errores:

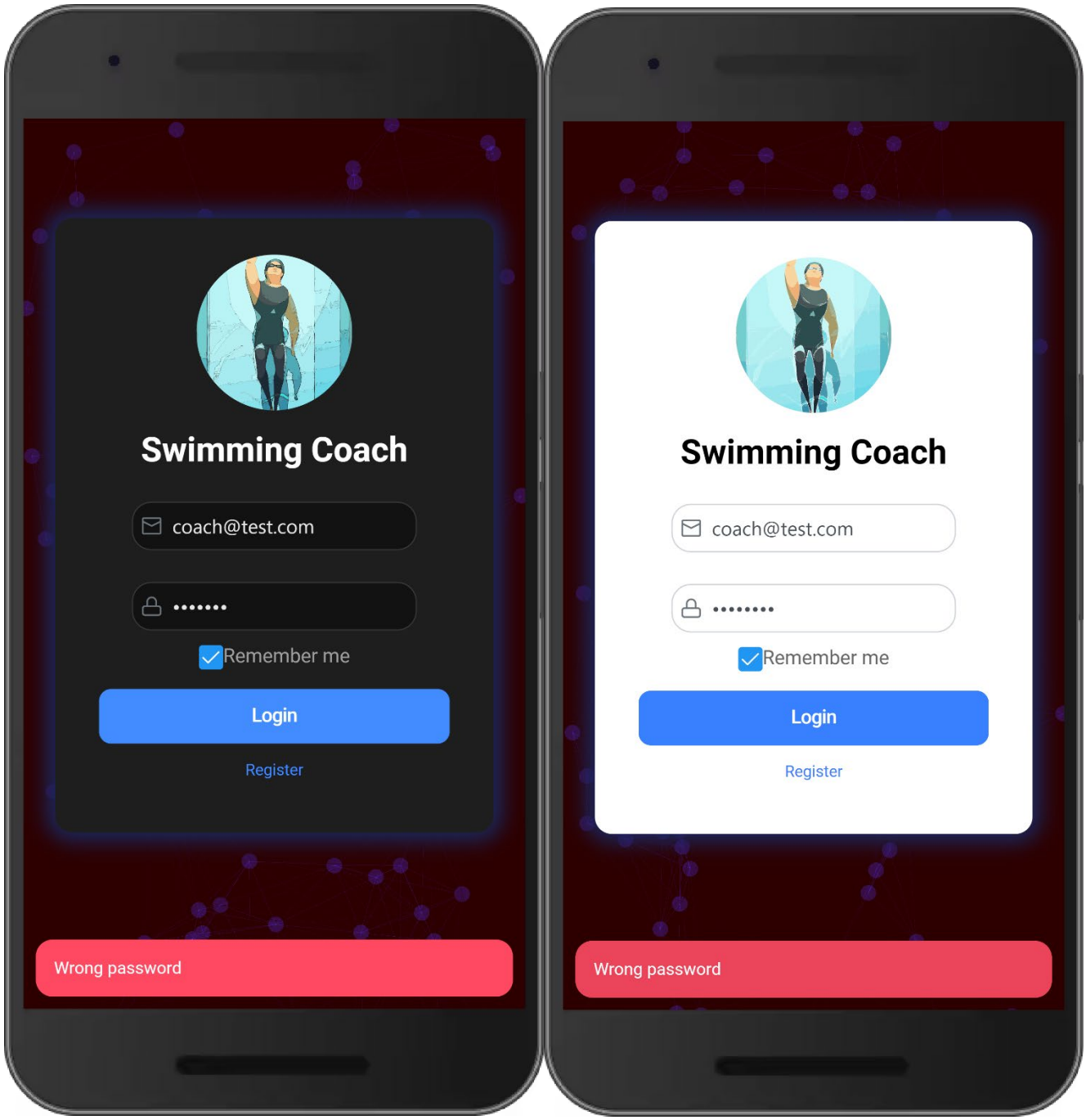


Figura 50: Pantalla inicio de sesión feedback al usuario por errores, móvil.

5.2.4 Menú desplegable lateral:

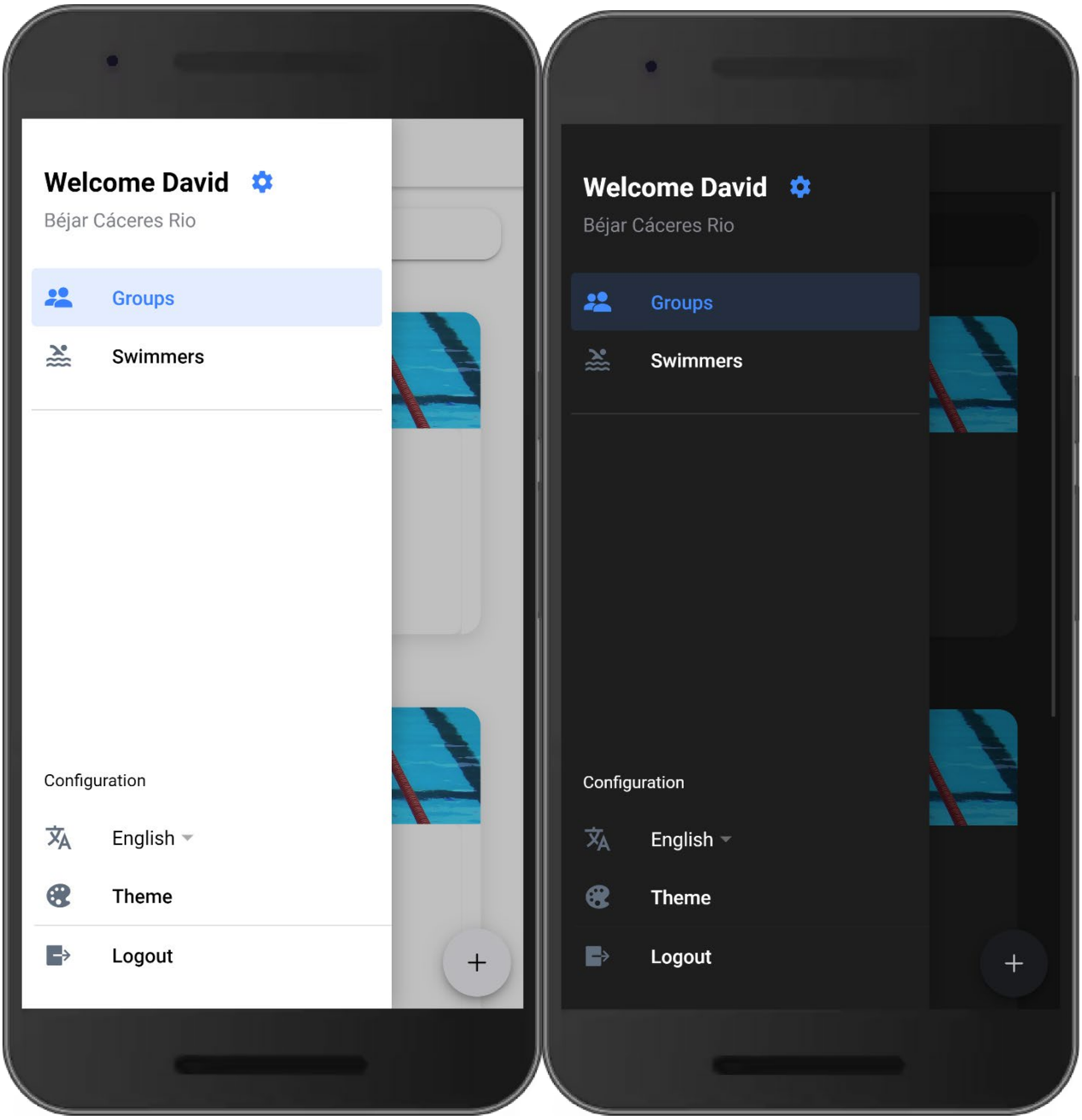


Figura 51: Menú desplegable lateral, móvil.

5.2.5 Pantalla Grupos de entrenamiento:

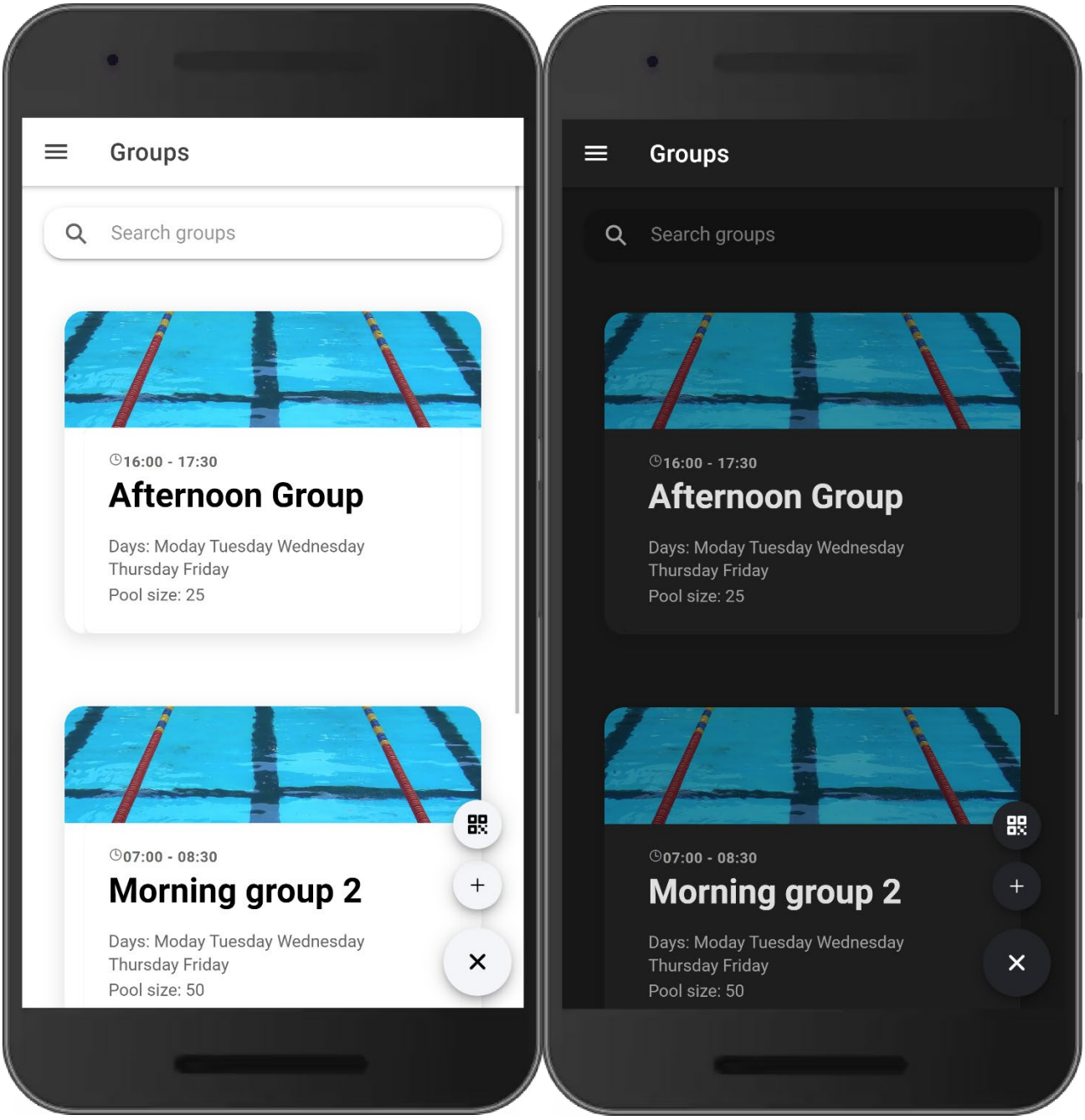


Figura 52: Pantalla Grupos de entrenamiento, móvil.

5.2.6 Página datos de usuario:

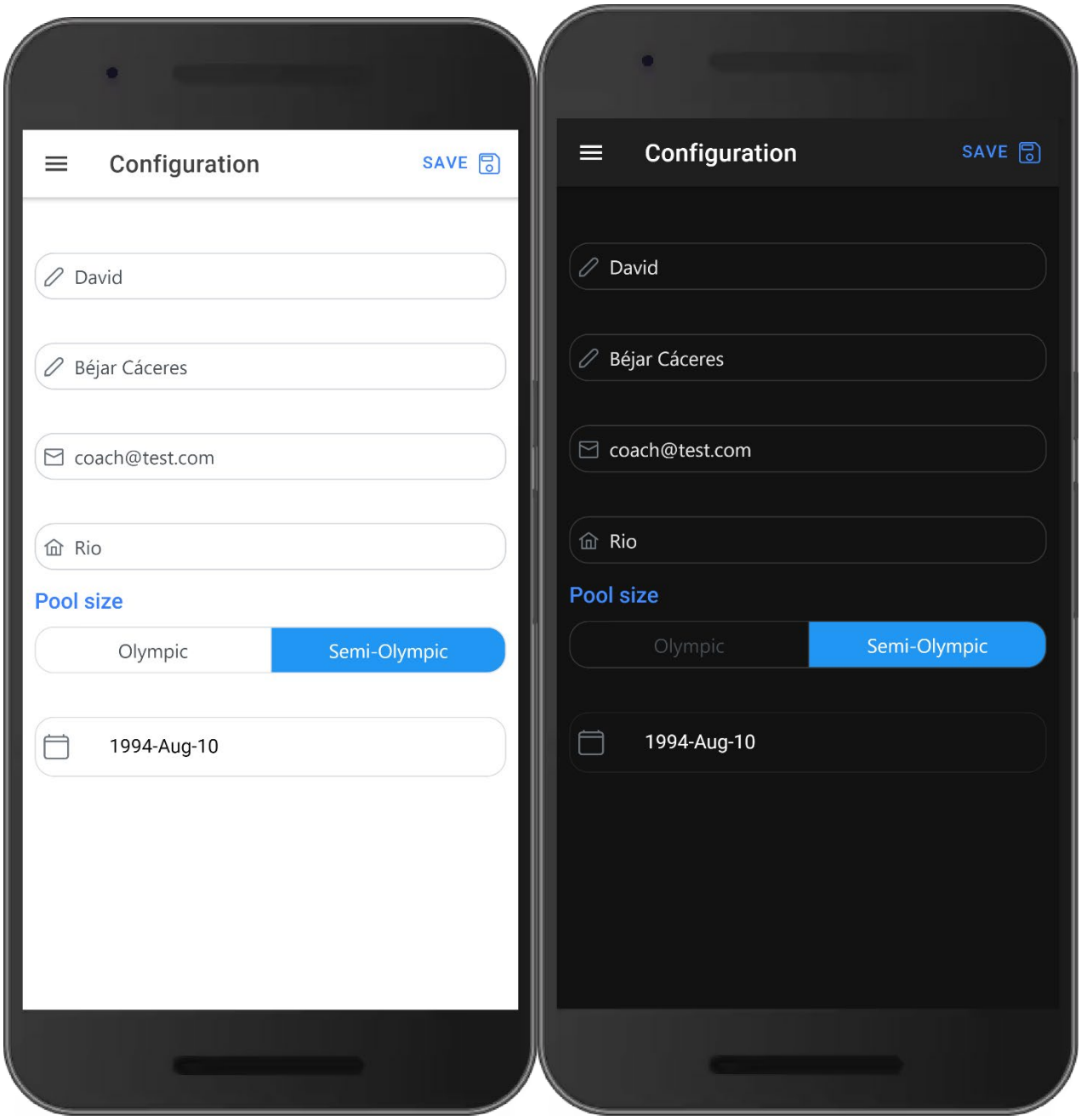


Figura 53: Página datos de usuario, móvil.

5.2.7 Página crear/editar grupo:

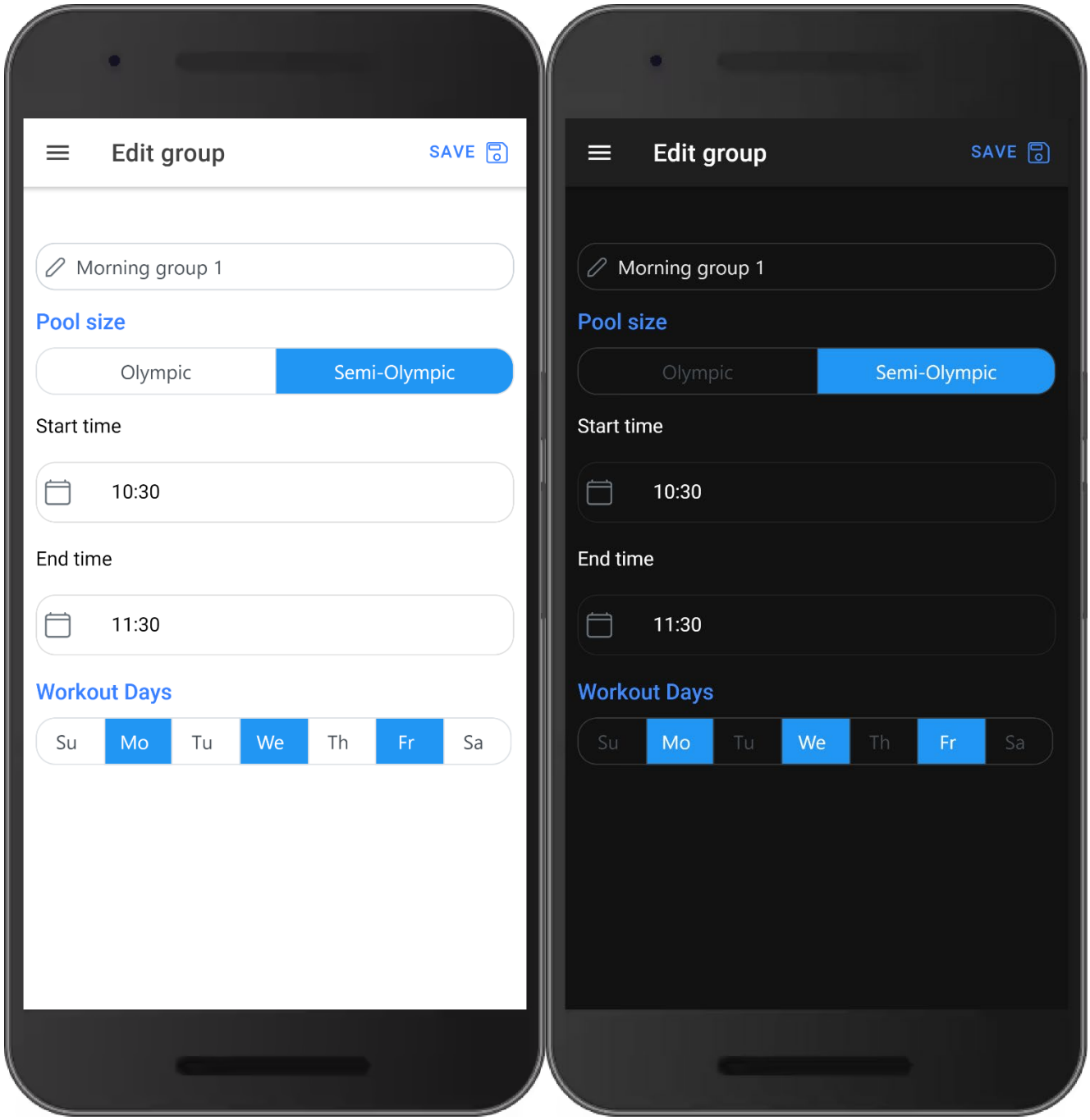


Figura 54: Página crear/editar grupo, móvil.

5.2.8 Lista de entrenamientos de grupo y estadísticas:

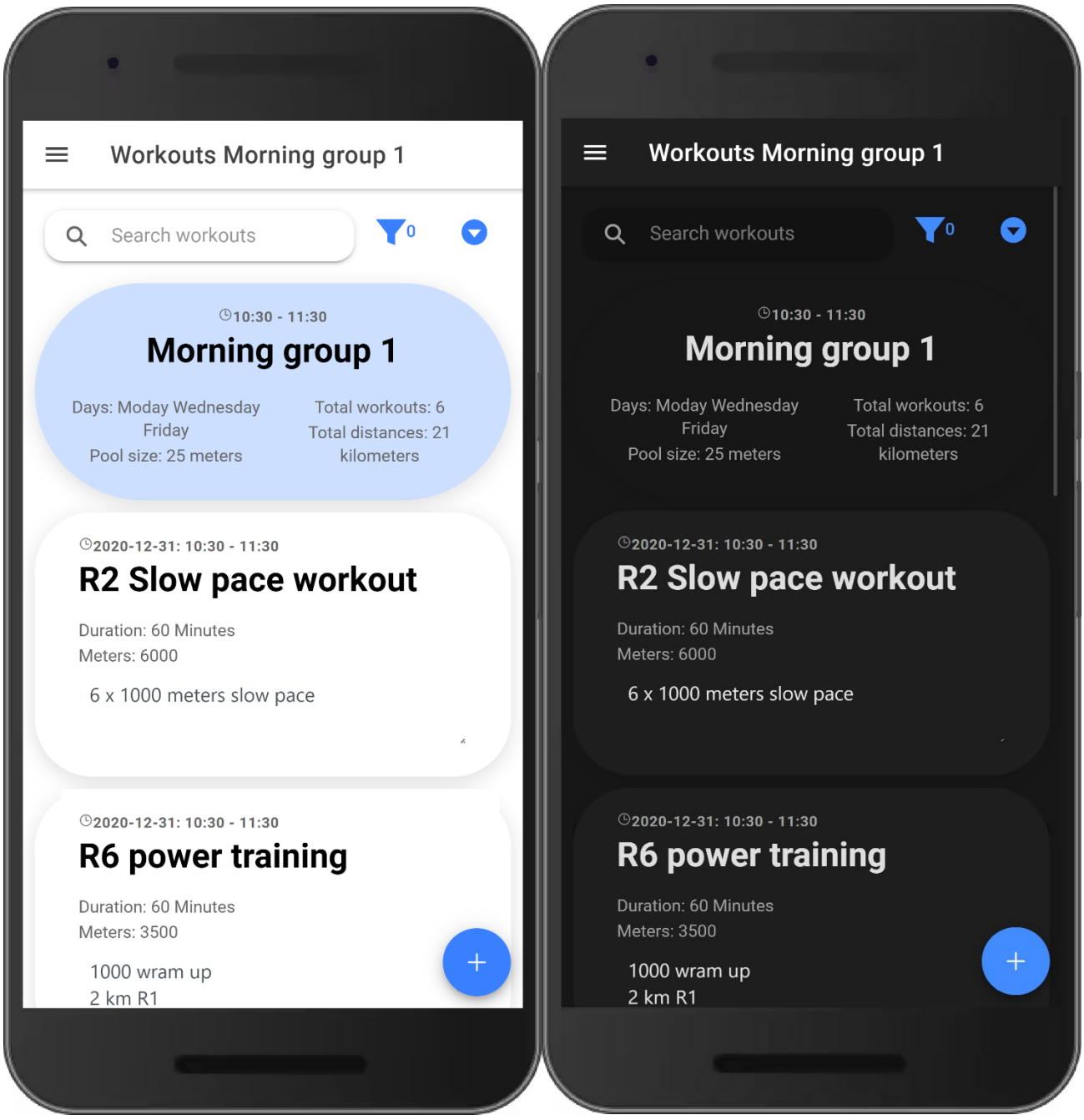


Figura 55: Lista de entrenamientos de grupo y estadísticas, móvil.

5.2.10 Página crear/editar entrenamiento:

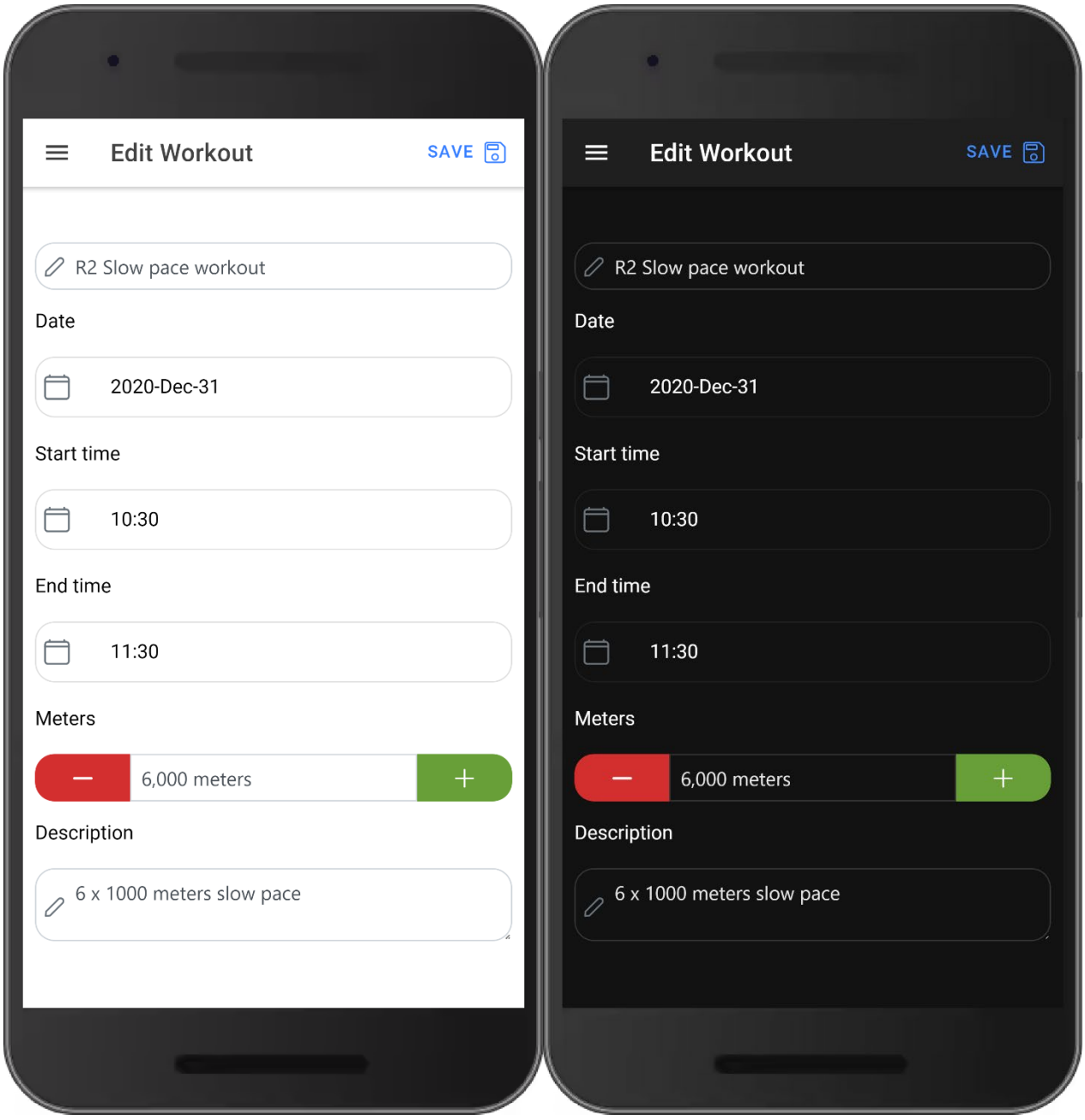


Figura 56: Página crear/editar entrenamiento, móvil.

5.2.11 Diálogo de filtros avanzados para entrenamientos de grupo:

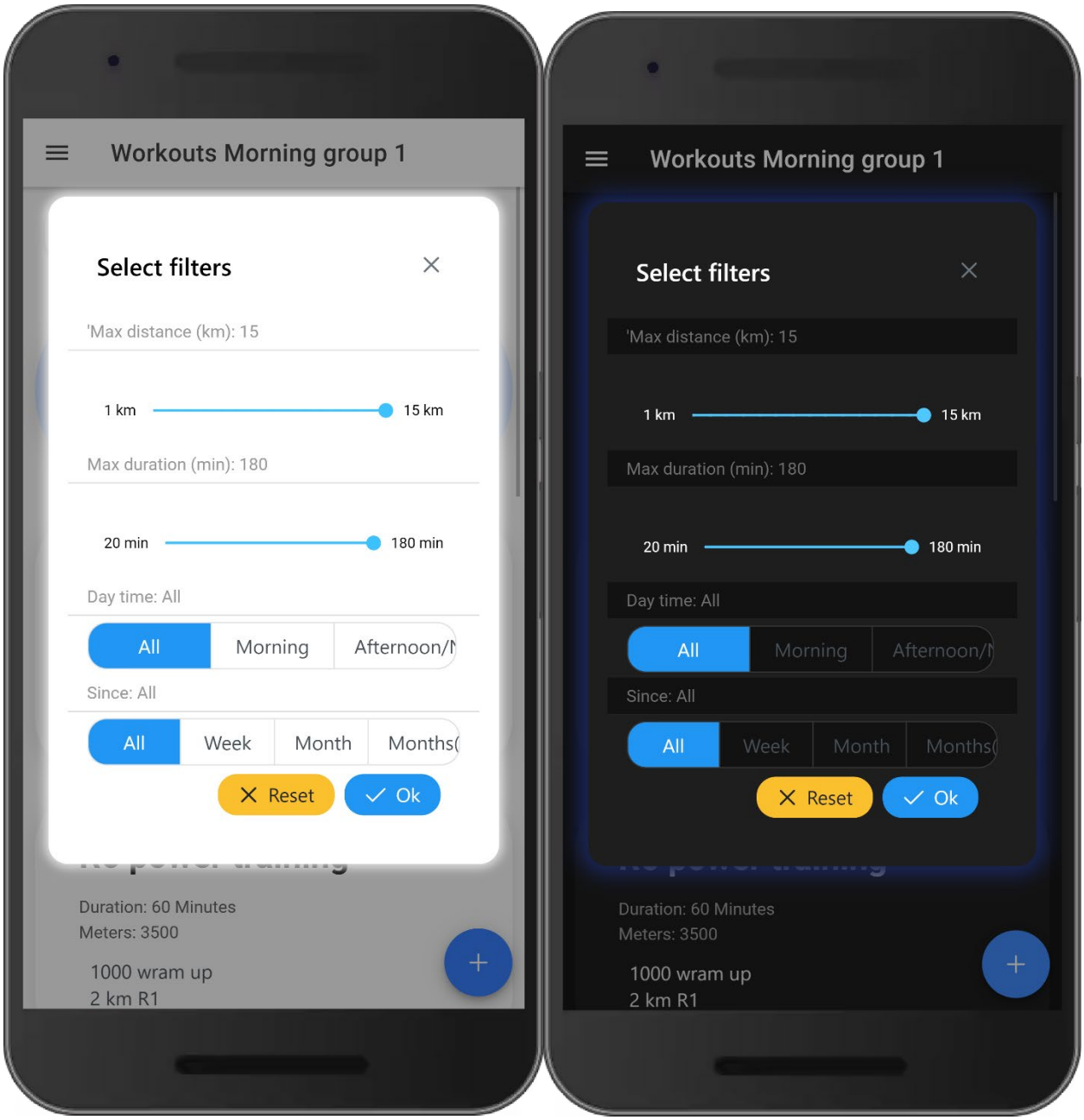


Figura 57: Diálogo de filtros avanzados para entrenamientos de grupo, móvil.

5.2.12 Página lista de nadadores:

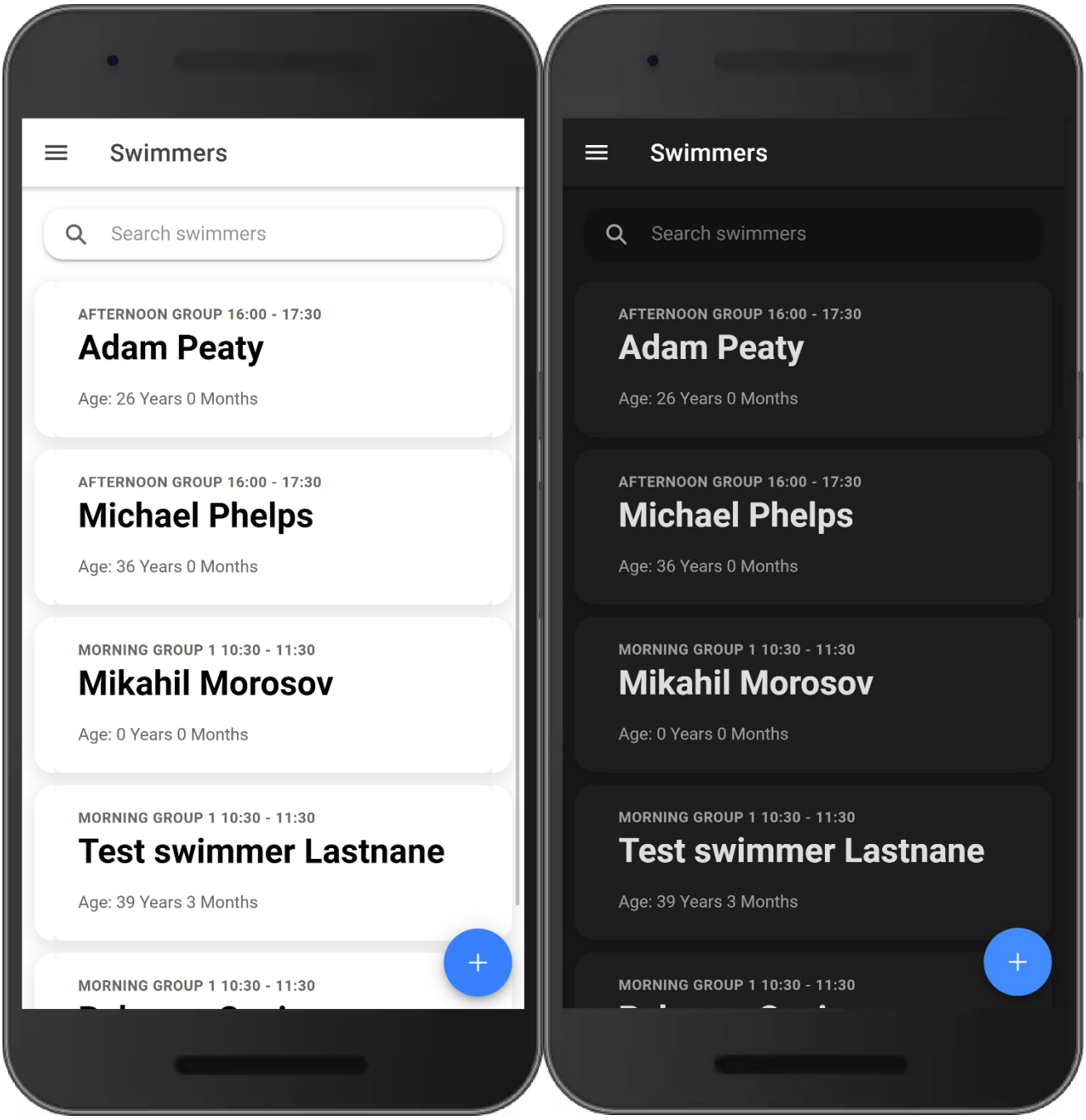


Figura 58: Página lista de nadadores móvil.

5.2.13 Página crear/editar nadador:

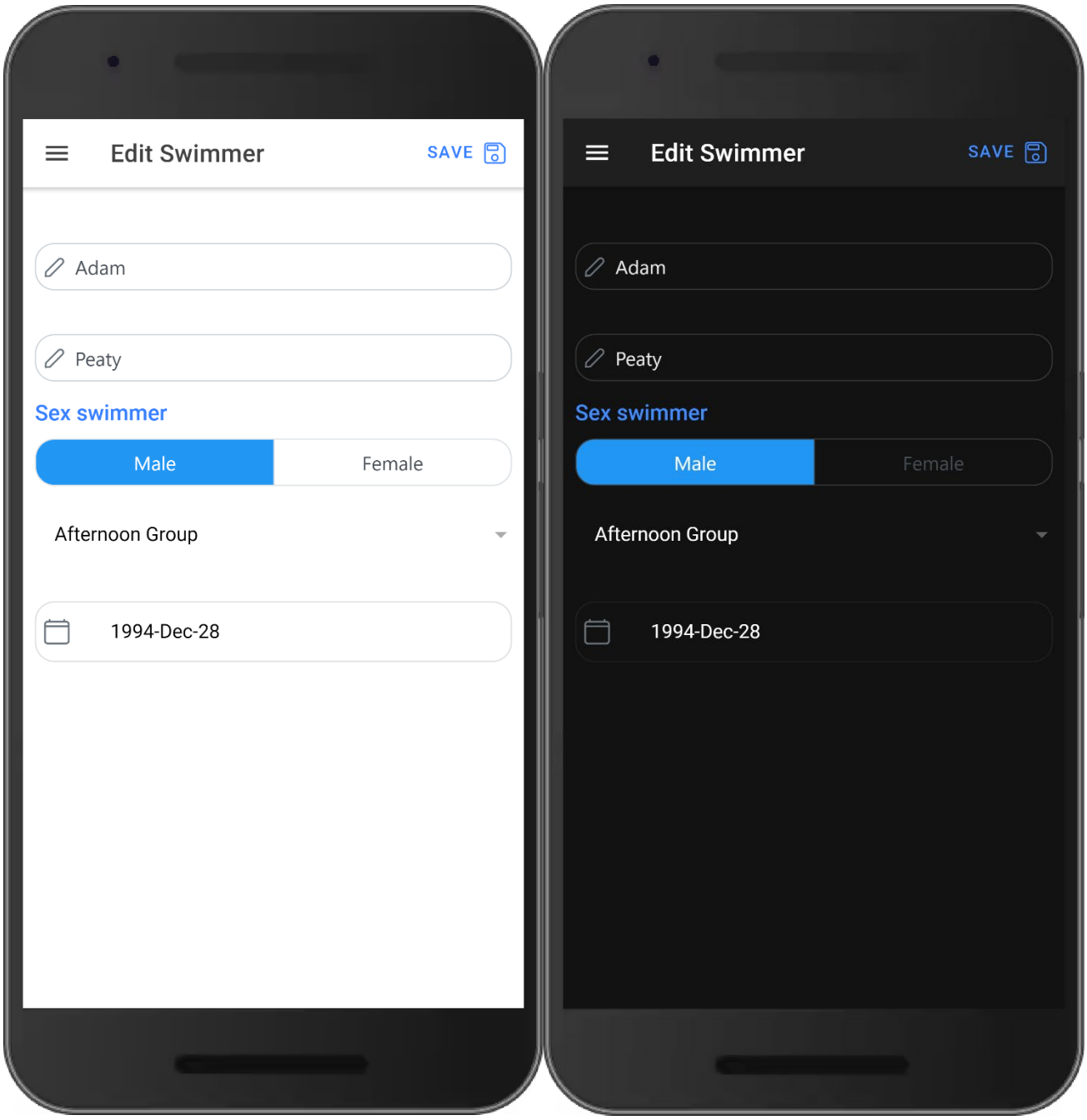


Figura 59: Página crear/editar nadador móvil.

5.2.14 Interacciones ítems en listas, deslizar derecha o izquierda para editar o eliminar:

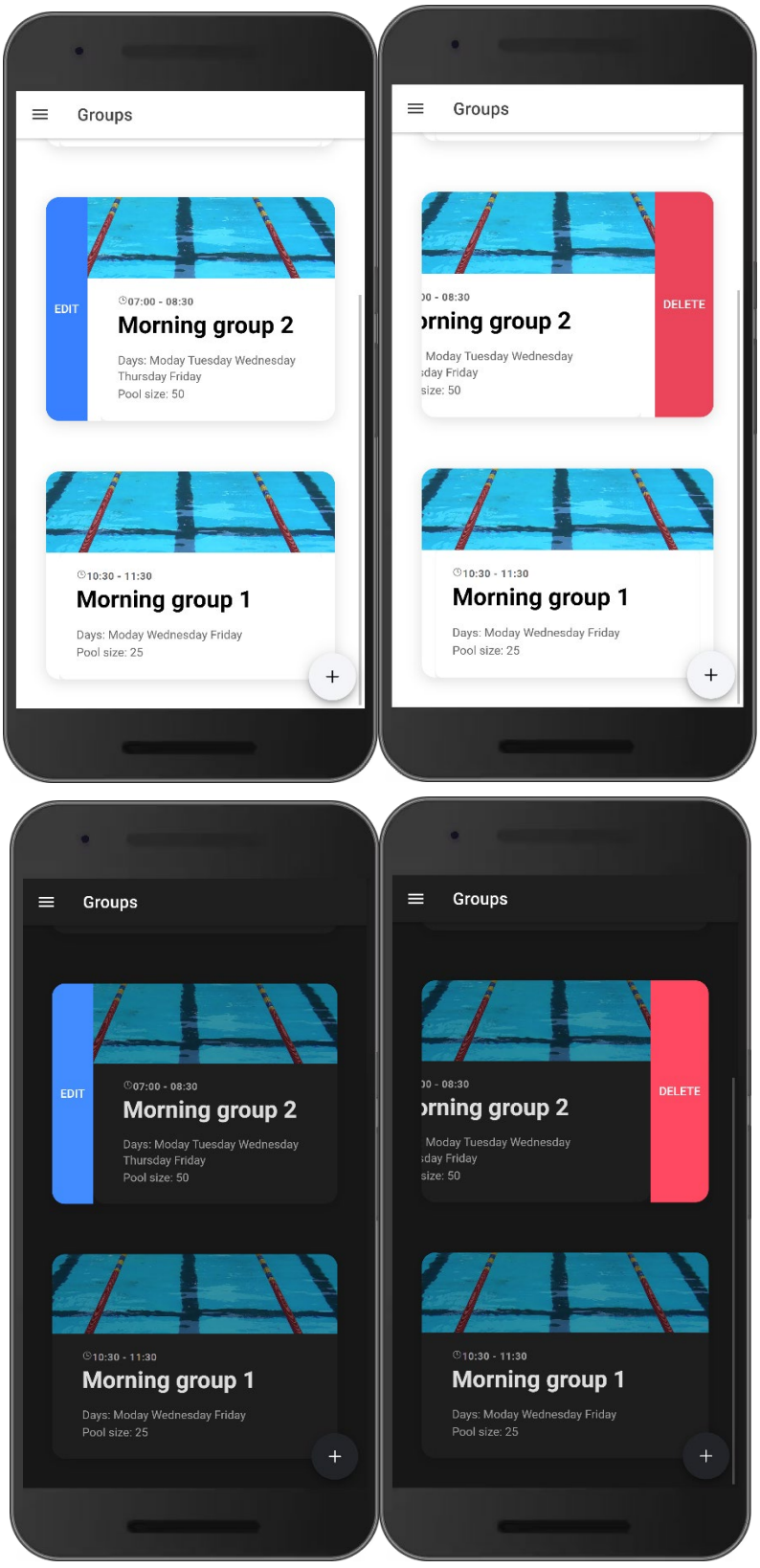


Figura 60: Interacciones ítems en listas, deslizar derecha o izquierda para editar o eliminar, móvil.

5.2.15 Diálogo añadir grupo para suscribirse mediante código de grupo o código QR:

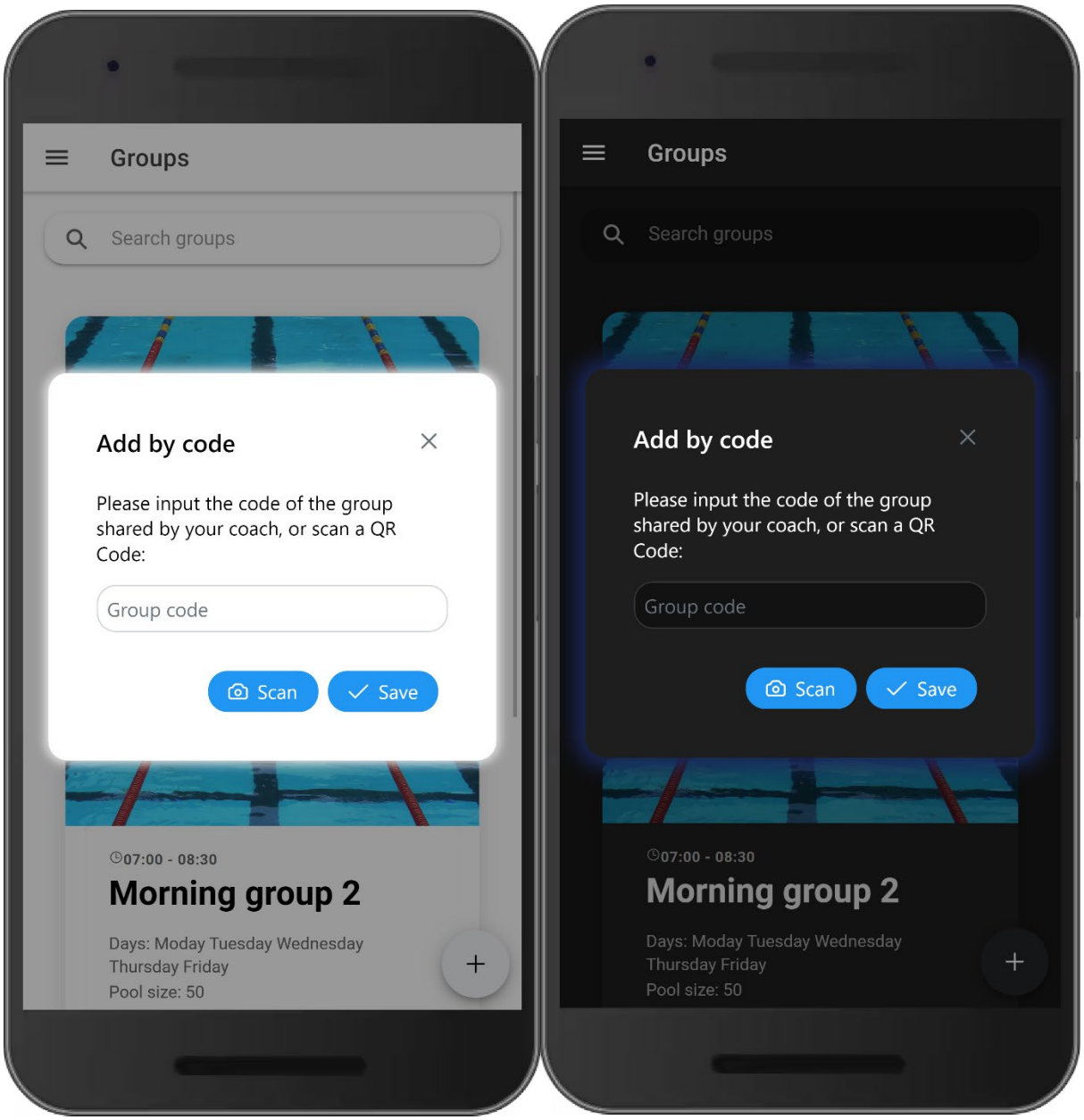


Figura 61: Diálogo añadir grupo para suscribirse mediante código de grupo o código QR, móvil.

5.2.16 Animaciones de carga de listas “Skelleton screen”:

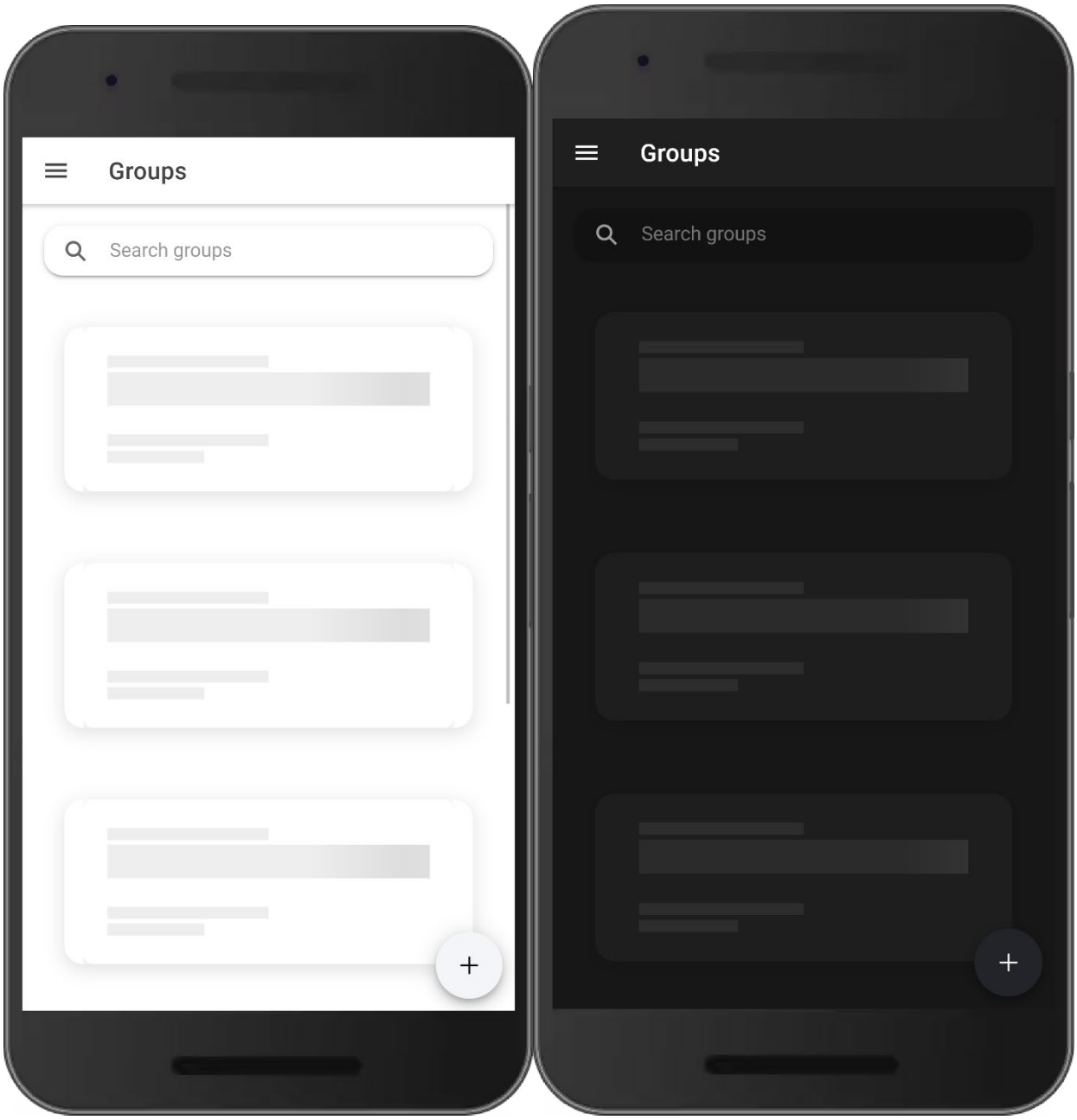


Figura 62: Animaciones de carga de listas "Skelleton Screen", móvil.

Anexo 6. Libro de estilo

6.1 Logotipo:

Figura 63: Logotipo app.

6.2 Paleta de colores tema claro y oscuro:

swimmingCoach - variables.css

```css
/** Ionic CSS Variables **/
:root {
  /** primary **/
  --ion-color-primary: #3880ff;
  --ion-color-primary-rgb: 56, 128, 255;
  --ion-color-primary-contrast: #ffffff;
  --ion-color-primary-contrast-rgb: 255, 255, 255;
  --ion-color-primary-shade: #3171e0;
  --ion-color-primary-tint: #4c8dff;

  /** secondary **/
  --ion-color-secondary: #3dc2ff;
  --ion-color-secondary-rgb: 61, 194, 255;
  --ion-color-secondary-contrast: #ffffff;
  --ion-color-secondary-contrast-rgb: 255, 255, 255;
  --ion-color-secondary-shade: #36abe0;
  --ion-color-secondary-tint: #50c8ff;

  /** tertiary **/
  --ion-color-tertiary: #5260ff;
  --ion-color-tertiary-rgb: 82, 96, 255;
  --ion-color-tertiary-contrast: #ffffff;
  --ion-color-tertiary-contrast-rgb: 255, 255, 255;
  --ion-color-tertiary-shade: #4854e0;
  --ion-color-tertiary-tint: #6370ff;

  /** success **/
  --ion-color-success: #2dd36f;
  --ion-color-success-rgb: 45, 211, 111;
  --ion-color-success-contrast: #ffffff;
  --ion-color-success-contrast-rgb: 255, 255, 255;
  --ion-color-success-shade: #28ba62;
  --ion-color-success-tint: #42d77d;

  /** warning **/
  --ion-color-warning: #ffc409;
  --ion-color-warning-rgb: 255, 196, 9;
  --ion-color-warning-contrast: #000000;
  --ion-color-warning-contrast-rgb: 0, 0, 0;
  --ion-color-warning-shade: #e0ac08;
  --ion-color-warning-tint: #ffca22;

  /** danger **/
  --ion-color-danger: #eb445a;
  --ion-color-danger-rgb: 235, 68, 90;
  --ion-color-danger-contrast: #ffffff;
  --ion-color-danger-contrast-rgb: 255, 255, 255;
  --ion-color-danger-shade: #cf3c4f;
  --ion-color-danger-tint: #ed576b;

  /** dark **/
  --ion-color-dark: #222428;
  --ion-color-dark-rgb: 34, 36, 40;
  --ion-color-dark-contrast: #ffffff;
  --ion-color-dark-contrast-rgb: 255, 255, 255;
  --ion-color-dark-shade: #1e2023;
  --ion-color-dark-tint: #383a3e;

  /** medium **/
  --ion-color-medium: #92949c;
  --ion-color-medium-rgb: 146, 148, 156;
  --ion-color-medium-contrast: #ffffff;
  --ion-color-medium-contrast-rgb: 255, 255, 255;
  --ion-color-medium-shade: #808289;
  --ion-color-medium-tint: #9d9fa6;

  /** light **/
  --ion-color-light: #f4f5f8;
  --ion-color-light-rgb: 244, 245, 248;
  --ion-color-light-contrast: #000000;
  --ion-color-light-contrast-rgb: 0, 0, 0;
  --ion-color-light-shade: #d7d8da;
  --ion-color-light-tint: #f5f6f9;
  --form-input-border: #DEE2E6;

  /* // Reduces width of the content for tablest */
  --ion-grid-width-xl: 50%; /* Tablet horizontal, Desktop (800px and above) */
  --ion-grid-width-lg: 50%; /* Tablet horizontal, Desktop (992px and above) */
  --ion-grid-columns: 12;

  --context-menu-container: #ffffff;
  --context-menu-active: #DEE2E6;

  --ion-card-background: #ffffff;
  --ion-card-float-shadow: 0px 0px 0.5rem 0.2rem #e2e2e2;

  --ion-toolbar-background: #ffffff;
  --form-input-border-hover: #2196F3;

  --ion-list-background: #ffffff;
  --ion-detail-card: #3880ff42;

}
```

swimmingCoach - variables.css

```css
body.dark {
  --ion-color-primary: #428cff;
  --ion-color-primary-rgb: 66,140,255;
  --ion-color-primary-contrast: #ffffff;
  --ion-color-primary-contrast-rgb: 255,255,255;
  --ion-color-primary-shade: #3a7be0;
  --ion-color-primary-tint: #5598ff;

  --ion-color-secondary: #50c8ff;
  --ion-color-secondary-rgb: 80,200,255;
  --ion-color-secondary-contrast: #ffffff;
  --ion-color-secondary-contrast-rgb: 255,255,255;
  --ion-color-secondary-shade: #46b0e0;
  --ion-color-secondary-tint: #62ceff;

  --ion-color-tertiary: #6a64ff;
  --ion-color-tertiary-rgb: 106,100,255;
  --ion-color-tertiary-contrast: #ffffff;
  --ion-color-tertiary-contrast-rgb: 255,255,255;
  --ion-color-tertiary-shade: #5d58e0;
  --ion-color-tertiary-tint: #7974ff;

  --ion-color-success: #2fdf75;
  --ion-color-success-rgb: 47,223,117;
  --ion-color-success-contrast: #000000;
  --ion-color-success-contrast-rgb: 0,0,0;
  --ion-color-success-shade: #29c467;
  --ion-color-success-tint: #44e283;

  --ion-color-warning: #ffd534;
  --ion-color-warning-rgb: 255,213,52;
  --ion-color-warning-contrast: #000000;
  --ion-color-warning-contrast-rgb: 0,0,0;
  --ion-color-warning-shade: #e0bb2e;
  --ion-color-warning-tint: #ffd948;

  --ion-color-danger: #ff4961;
  --ion-color-danger-rgb: 255,73,97;
  --ion-color-danger-contrast: #ffffff;
  --ion-color-danger-contrast-rgb: 255,255,255;
  --ion-color-danger-shade: #e04055;
  --ion-color-danger-tint: #ff5b71;

  --ion-color-dark: #f4f5f8;
  --ion-color-dark-rgb: 244,245,248;
  --ion-color-dark-contrast: #000000;
  --ion-color-dark-contrast-rgb: 0,0,0;
  --ion-color-dark-shade: #d7d8da;
  --ion-color-dark-tint: #f5f6f9;

  --ion-color-medium: #989aa2;
  --ion-color-medium-rgb: 152,154,162;
  --ion-color-medium-contrast: #000000;
  --ion-color-medium-contrast-rgb: 0,0,0;
  --ion-color-medium-shade: #86888f;
  --ion-color-medium-tint: #a2a4ab;

  --ion-color-light: #222428;
  --ion-color-light-rgb: 34,36,40;
  --ion-color-light-contrast: #ffffff;
  --ion-color-light-contrast-rgb: 255,255,255;
  --ion-color-light-shade: #1e2023;
  --ion-color-light-tint: #383a3e;

    /** forms inputs colors **/
  --form-input-background: #121212;
  --form-input-border-dark: #383838;
  --form-input-border: #383838;
  --form-input-error: #a04a22;
  --form-input-error-shadow: 0 0 0 0.1rem #a04a22;
  --form-input-shadow: 0 0 0 0.1rem #a6d5fa;
  --form-input-color: #e6e6e6;
  --form-container-background: #1E1E1E;

  --context-menu-container: #2B2B2B;
  --context-menu-active: #333333;
  --ion-card-float-shadow: 0px 0px 1rem 0.5rem #1a265b;

  --ion-list-background: #171717;
  --ion-detail-card: #171717;
}
```

Figura 64: Paleta de colores en variables CSS.

6.3 Fondos, iconos y otros elementos gráficos:

6.3.1 Fondo cabecera grupo de entrenamiento:

Figura 65: Fondo de pantalla cabecera grupo de entrenamiento.

6.3.2 Fondo animado pantalla inicio de sesión[71]:

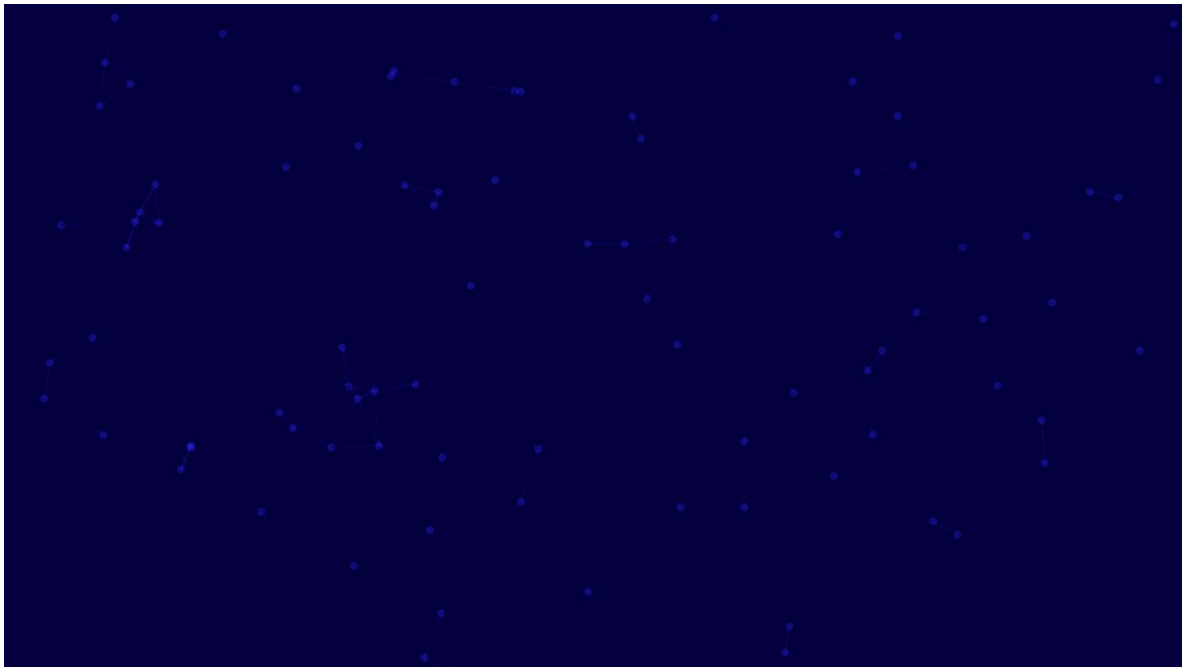


Figura 66: Fondo animado pantalla inicio de sesión y registro.

6.3.3 Otras reglas de uso:

- border-radius: 1em