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

Aplicación móvil de prescripción de ejercicio con acceso a servicio web Spring

Autor: Laura García Corredera
Tutor: María Teresa Ariza Gómez

Departamento de Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2018

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

Aplicación móvil de prescripción de ejercicio con acceso a servicio web Spring

Autor:
Laura García Corredera

Tutor:
María Teresa Ariza Gómez
Profesor titular

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

Proyecto Fin de Grado: Aplicación móvil de prescripción de ejercicio con acceso a servicio web Spring

Autor: Laura García Corredera

Tutor: María Teresa Ariza Gómez

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

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2018

El Secretario del Tribunal

A mi familia

A mis maestros

Agradecimientos

Durante todos estos años de carrera he vivido momentos buenos, malos y muy malos y son, sobre todo, por estos últimos, por lo que quiero dar las gracias a todas las personas que los han vivido conmigo y me han acompañado durante todo este largo camino.

En primer lugar, como no podría ser de otra forma, a mis padres. Necesitaría mil hojas para poder agradeceros todo lo que habéis hecho por mi durante estos veintitrés años. Y es que vosotros sois el motivo principal por el que hoy estoy pudiendo escribir esto, porque gracias a vuestro esfuerzo y vuestro apoyo he logrado ver la luz al final del túnel. Me habéis enseñado que la perseverancia y el creer en uno mismo es la mayor de las fuerzas. Por eso y por absolutamente todo lo que soy ahora, muchísimas gracias.

Luego está mi hermana, mi ejemplo a seguir, gracias por demostrarnos a todos que los sueños se cumplen, no puedo estar más orgullosa de ti. Y quiero que sepas que siempre has sido y serás mi calma en la tempestad.

Y no puedo olvidarme de ti, gracias por andar conmigo durante todos estos años y ayudarme a levantarme tras cada caída. Gracias por darle color a los días más grises. Gracias por ser tan opuesto como complementario y hacer de esta etapa algo maravilloso.

Por último y no por ello menos importante, agradecer a todos mis amigos el apoyo que me han brindado todo este tiempo y haber sido parte fundamental en conseguir que cada día fuese un buen día si estaban ellos.

Laura García Corredera

Sevilla, 2018

Resumen

Hoy en día vivimos en un mundo cambiante, revolucionado por la tecnología. Un mundo que nos ofrece una gran variedad de herramientas que ponen al alcance de nuestra mano todo un sinfín de información.

Las aplicaciones móviles son herramientas que nos ofrecen la movilidad y la conexión en tiempo real que otras no pueden y, es por ello por lo que, en poco tiempo, han logrado abrirse paso en el campo de la tecnología convirtiéndose en un elemento indispensable para muchos y para el desarrollo de sus actividades diarias.

Es por esto, que en el presente proyecto se ha desarrollado una aplicación móvil que permite al usuario obtener una serie de rutinas de entrenamientos personalizadas acorde con sus necesidades físicas y respetando sus características físicas. De tal manera que el usuario podrá acceder a ellas en cualquier momento dándole la posibilidad de mantener un estilo de vida saludable y adaptable a su día a día.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

"Es intentando lo imposible como se realiza lo posible"
Henry Barbusse

En este primer capítulo se va a proceder a dar una breve introducción con el fin de poner en situación al lector de la presente memoria y así abrirle camino en la comprensión de este proyecto además de hacerle conocedor del motivo que impulsó a llevarlo a cabo.

1.1 Contexto

Hoy en día, el mundo ha sido sacudido por la tecnología y el acceso a Internet hasta llegar a convertir estos elementos en herramientas indispensables en nuestro día a día, de tal manera que en muchas ocasiones llegamos a depender del acceso a datos para poder realizar nuestro trabajo y/o nuestras tareas diarias. A esto se le suma la ferviente necesidad de mantenernos conectados y comunicados siempre que sea posible.

Estas necesidades dan lugar a la aparición de los teléfonos móviles inteligentes, o mejor conocidos como Smartphones. Gracias a estos dispositivos cualquier persona puede estar conectado y transmitiendo/recibiendo datos continuamente desde cualquier lugar y en cualquier momento del día.

Esta oferta de servicios ha provocado una abrumadora demanda de aplicaciones que satisfagan las necesidades y deseos de los usuarios en todos los campos imaginables de tal manera que éstos puedan realizar tareas de forma rápida y fácil a través de aplicaciones especializadas.

Si nos paramos a estudiar la tendencia social actual, podríamos afirmar que la imagen de uno mismo juega un importante papel en el ámbito social, lo que lleva al ser humano a implantar un hábito de vida saludable a través de la alimentación y el deporte. Por otro lado, el ritmo de vida, en general, suele ser exigente de manera que no suele quedar mucho tiempo libre para invertir en los aspectos señalados, Así, surge la necesidad de buscar una alternativa para poder cumplir con dichos objetivos.

Es aquí donde entran las aplicaciones móviles como un papel importante en el cumplimiento y éxito de unos hábitos de vida saludable. En concreto, la aplicación desarrollada está enfocada a respetar el estado físico de cada individuo para que éste pueda realizar un entrenamiento personalizado, evitando futuras dolencias o lesiones, y enseñándole a seguir un hábito de entrenamiento responsable y saludable igual que si estuviese en un centro deportivo asistidos por un entrenador.

Es por esto, que muchas aplicaciones móviles permiten mejorar aspectos de la vida cotidiana que antes nos quitaban tiempo y podían llevar consigo un coste económico, ya que hoy por hoy nos permiten ahorrar tiempo y dinero y nos proporcionan la comodidad y el libre albedrío que antes no teníamos.

1.2 Presentación del problema

Centrándonos en el ámbito que nos ocupa, el de la actividad física y el deporte, podemos afirmar que existe una inmensa variedad de aplicaciones móviles dedicadas a facilitar este tipo de información al usuario que las use, pero un gran porcentaje de éstas no contemplan las condiciones físicas de cada individuo, tratándolos a todos como uno y ofreciendo los mismos ejercicios a una persona en perfecto estado de salud que a una con alguna dolencia física, siendo contraproducente para ésta última pudiendo llegar a dañar su salud.

Es por esto, que nace la necesidad de crear una aplicación que ofrezca asistencia personal por parte de un profesional cualificado que asigne a cada usuario los ejercicios que necesiten teniendo presente sus características físicas y estado de salud, tal y como sucede en un gimnasio, pero con la ventaja de poder realizar estas rutinas dónde y cuándo el usuario desee.

De esta manera, cualquier usuario con conexión a internet podrá obtener tanto sus rutinas personalizadas, previamente asignadas a él por el entrenador, como acceder a toda la información existente para realizar, por libre y bajo su responsabilidad, todos los ejercicios contenidos en la aplicación.

En definitiva, el objetivo de este proyecto es desarrollar una aplicación única y personalizable para cada individuo respetando sus condiciones y necesidades físicas y permitiéndole cumplir sus objetivos para que sea capaz de seguir un estilo de vida saludable compatible con su día a día.

1.3 Antecedentes

El presente proyecto es la mejora y continuación de uno llevado a cabo por José María Valverde Baena en la Escuela Técnica Superior de Ingeniería de Sevilla en el año 2015.

Dicho trabajo tenía como objetivo solventar los problemas planteados en el apartado anterior (1.2 Presentación del problema). La principales diferencias a resaltar entre ambos proyectos son una estructura e interfaz de usuario totalmente redefinida y mejorada y un nuevo tipo de servicio web que gestiona la comunicación e intercambio de datos entre la aplicación móvil y la base de datos. Todos estos cambios se explicarán y detallarán en próximos capítulos.

Otras referencias

Además del proyecto anteriormente mencionado, el cuál ha sido la línea de continuación para la realización del presente, se debe mencionar el proyecto de Juan García Piosa, exalumno de la Escuela Técnica Superior de Ingeniería, quién fue el primero en desarrollar tanto la aplicación de entrenamiento como el servidor con el que ésta se comunicaba para hacer uso de las funcionalidades ofrecidas. Gracias a este trabajo se ha podido ir añadiendo mejoras y actualizaciones a lo largo de los años para llegar al que se presenta en esta memoria.

Finalmente, cabe destacar el realizado por Francisco José Díaz Romero en la Escuela Técnica Superior de Ingeniería de Sevilla en el año 2017. Este trabajo consiste en una aplicación web que, en términos generales, permite crear y gestionar rutinas. De esta manera, podría clasificarse este proyecto como el lado del profesional encargado de crear las rutinas y asignarlas a cada usuario, quien, posteriormente, podrá acceder a ellas a través de la aplicación móvil que podría ser definida como el lado del cliente.

Es por esto, que en el presente proyecto se ha estudiado la funcionalidad de dicha aplicación y, aunque no comparten un servidor en común, se ha intentado orientar el desarrollo de la aplicación móvil de manera que cubriese y satisficiese las necesidades del cliente concordando con lo ofrecido por la aplicación web mencionada, para así en unas líneas futuras se pueda integrar ambos proyectos como uno solo que usen un mismo servidor web.

1.4 Descripción de la solución

En este apartado se detallarán los objetivos impuestos en la realización del proyecto, así como la funcionalidad y arquitectura que finalmente presenta la solución dada.

1.4.1 Objetivos

El objetivo principal que se ha perseguido durante la realización de la aplicación móvil ha sido el ofrecer al usuario la posibilidad de incluir la actividad física en su rutina diaria con la total libertad de decidir él mismo dónde y el cuándo llevarla a cabo.

1.4.2 Funcionalidades

Una vez que el usuario haya descargado la aplicación MyTraining en su dispositivo móvil, podrá comenzar a usarla. La navegación por dicha aplicación comprenderá las siguientes funcionalidades:

- Inicio de sesión rápido y cómodo a través de su número de teléfono.
  - Si es un nuevo usuario, se le solicitará un registro y tras él podrá acceder al menú principal donde podrá hacer uso de la aplicación en su totalidad.
  - En el caso de ser un usuario anteriormente registrado, accederá inmediatamente al menú principal.
- Menú principal donde se ofrecen diversas posibilidades tales como:
  - Acceder a todos los ejercicios de la aplicación y realizar un entrenamiento por libre.
  - Acceder a las rutinas de ejercicios que le han sido asignadas por un profesional.
  - Mapa donde ver los locales más cercanos y poder buscar un centro deportivo, gimnasio, parque u otro lugar donde realizar el entrenamiento.
  - Configuración de la cuenta y posible modificación de datos personales.
  - Cierre de cuenta.
  - Eliminación permanente de cuenta.
- Listados de rutinas personalizadas para el usuario que contendrán a su vez los ejercicios que la comprenden, pudiendo conocer los detalles de cada uno de ellos y visualizar un vídeo explicativo de cómo realizar correctamente cada ejercicio.
- Listados de ejercicios varios, clasificados según objetivo deseado, igualmente se podrá acceder al detalle y vídeo de cada uno de ellos.

1.4.3 Arquitectura

Nuestra solución implementa un servicio web REST que sigue la arquitectura software dada por el Modelo Vista Controlador (MVC), consistente en dividir el problema en tres componentes, interfaz de usuario, datos y lógica de control, los cuáles interaccionan entre sí para dar vida al servicio.

En la imagen adjunta a continuación podemos ver cuáles son los componentes de nuestro servicio y cómo se comunican entre sí.

Ilustración 1: Arquitectura del servicio

Aplicación Móvil

El usuario hace uso de la aplicación mediante su teléfono móvil con conexión a internet gracias a la cual obtiene los datos que ofrece el servicio web y los muestra en pantalla tal y como esté configurada la interfaz de usuario en la App.

Servicio web REST

El servicio web implantado tiene como objetivo realizar operaciones CRUD (Create, Read, Update, Delete) sobre la información residente en la base de datos con el fin de ofrecerla al usuario y

poder gestionarla según las peticiones que éste realice a través de la App. Todo este intercambio de datos será realizado mediante peticiones HTTP y la información será codificada en objetos JSON.

Base de datos

La información que el usuario va a recibir en su dispositivo se encuentra almacenada en una base de datos MySQL a la que el servidor web accederá mediante consultas SQL siguiendo el patrón DAO (Data Access Object) que se explicará posteriormente.

1.5 Estructura de la memoria

La presente memoria está dividida en una serie de capítulos numerados en los que se irá profundizando en el funcionamiento y estructura de la aplicación y servicio que comprenden este proyecto. A continuación, se da una breve descripción de cada uno de esos capítulos para poner en situación al lector.

Capítulo 1: Introducción

Sitúa al lector en el ámbito que concierne a este proyecto y da una breve descripción de la funcionalidad y arquitectura de éste con el fin de ir profundizando en dichos aspectos en próximos capítulos.

Capítulo 2: Tecnologías utilizadas

Enumera y explica en detalle las tecnologías que han sido implementadas y utilizadas durante el desarrollo del proyecto.

Capítulo 3: Herramientas usadas

Describe las herramientas y frameworks utilizados para llevar a cabo con éxito la aplicación y el servicio web creados y el porqué de la elección de éstos.

Capítulo 4: Arquitectura y funcionalidad

Se realiza un esquema explicativo de la arquitectura y funcionalidad de lo implementado en el proyecto mediante diagramas UML para facilitar la comprensión al lector y darle una visión más clara de en qué consiste la aplicación desarrollada.

Capítulo 5: Servicio web

Se detalla en mayor profundidad la estructura y funcionalidad del servicio web REST implementado mediante Spring y cómo éste realiza las conexiones requeridas para otorgar un correcto funcionamiento a la aplicación móvil.

Capítulo 6: Aplicación móvil

Se muestra en detalle cómo es la interfaz de la aplicación desarrollada junto con la estructura y funcionamiento que ésta posee.

Capítulo 7: Base de datos

Se muestra cada una de las tablas que componen la base de datos, parándose en detallar cada campo que contienen y explicando la relación entre ellas.

Capítulo 8: Líneas futuras

Se mencionan posibles futuras mejoras o cambios de tecnologías para mejorar la funcionalidad y eficiencia del servicio ofrecido.

Capítulo 9: Conclusiones

Se incluye una conclusión tras la finalización del proyecto analizando lo realizado y haciendo una valoración personal de ello.

Anexos

Anexo A: Manual de instalación y despliegue del servicio.

Anexo B: Manual de uso de las APIs y servicios externos implementados en la aplicación móvil.

2 TECNOLOGÍAS UTILIZADAS

En este capítulo se va a proceder a presentar las tecnologías utilizadas en el proyecto y que han permitido su funcionamiento. Además, se detallará cada una de ellas y se enfatizará en la finalidad con la que han sido usadas.

2.1 Tecnologías en el servicio web

2.1.1 Arquitectura REST

REST o Representational State Transfer es un estilo de arquitectura software usada en la comunicación entre un cliente y su servidor. La comunicación se realiza mediante el protocolo HTTP que se encarga de la transferencia de mensajes, usualmente en formato JSON o XML.

A diferencia de otro tipo de arquitecturas, REST no utiliza servicios con métodos diversos, sino que manipula recursos, que no son otra cosa que referencias a un concepto determinado de nuestro servicio. Estos recursos son manipulados a partir de sus respectivos identificadores, conocidos como URIs. Este nuevo elemento permite acceder a la información fácilmente para operar sobre ella a través de los siguientes verbos:

- GET: Obtener un recurso.
- POST: Crear un recurso en el servidor.
- PUT: Cambiar el estado de un recurso o actualizarlo.
- DETELE: Eliminar un recurso.

La ventaja de los servicios REST frente a otros es que aporta una solución más sencilla para la manipulación de datos y una mayor escalabilidad ya que el cliente y el servidor son sistemas independientes el uno del otro pudiéndose desarrollar de forma autónoma sin afectar la edición de uno al otro y el lenguaje y tecnologías usadas para cada uno. Por todo esto, queda aclarado el porqué de implementar un servicio REST para llevar a cabo la comunicación entre la aplicación móvil desarrollada y el servidor web implantado.

Ilustración 2: Funcionamiento servicio REST

2.1.2 Formato JSON

JSON o JavaScript Object Notation es un formato para el intercambio de datos, básicamente JSON describe los datos con una sintaxis dedicada que se usa para identificar y gestionar los datos. Nació como una alternativa a XML y, una de las mayores ventajas que tiene su uso es que puede ser leído por cualquier lenguaje de programación. Por lo tanto, puede ser usado para el intercambio de información entre distintas tecnologías. Además, debido a su naturaleza y al ser más compacto suele ser mucho más rápido trabajar con JSON antes que con XML.

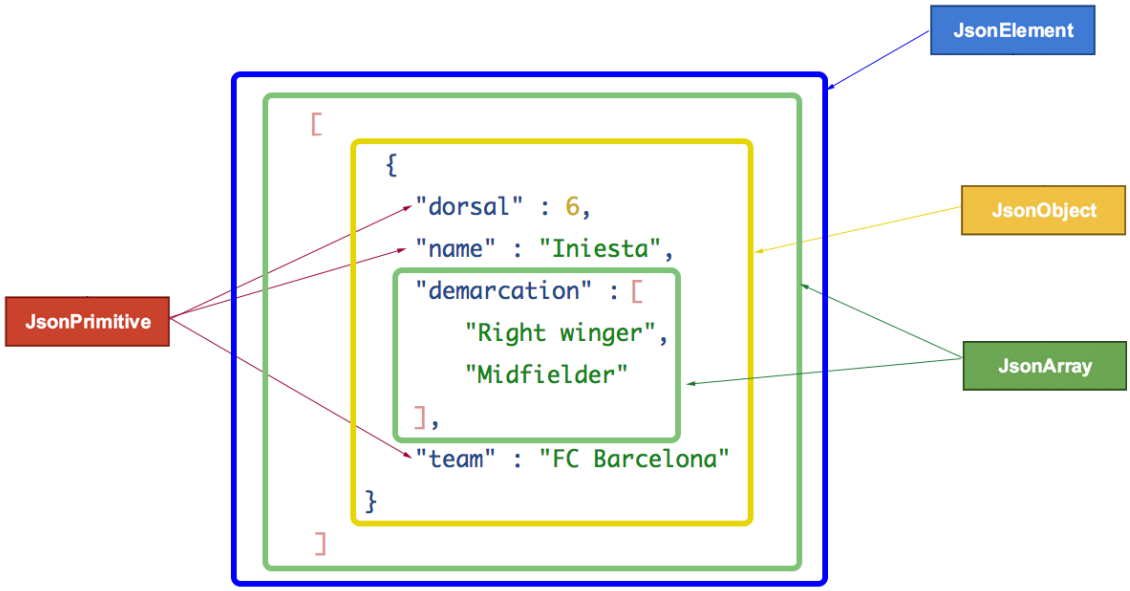


Ilustración 3: Arquitectura de un objeto JSON

2.2 Tecnologías en la aplicación móvil

2.2.1 Android

Android es un sistema operativo libre, gratuito y multiplataforma basado en Linux que posee una gran capacidad de adaptación a todo tipo de dispositivos lo que le confiere un gran potencial de desarrollo.

En el desarrollo de aplicaciones Android, el lenguaje de programación más utilizado actualmente es Java, uno de los lenguajes más conocidos y extendidos en el mundo. Sin embargo, el nuevo lenguaje diseñado por JetBrains, Kotlin, le está haciendo la competencia habiéndose incorporado como segundo lenguaje oficial para desarrollo Android y habiéndose incluido en las nuevas versiones de los framework de programación de este sistema operativo.

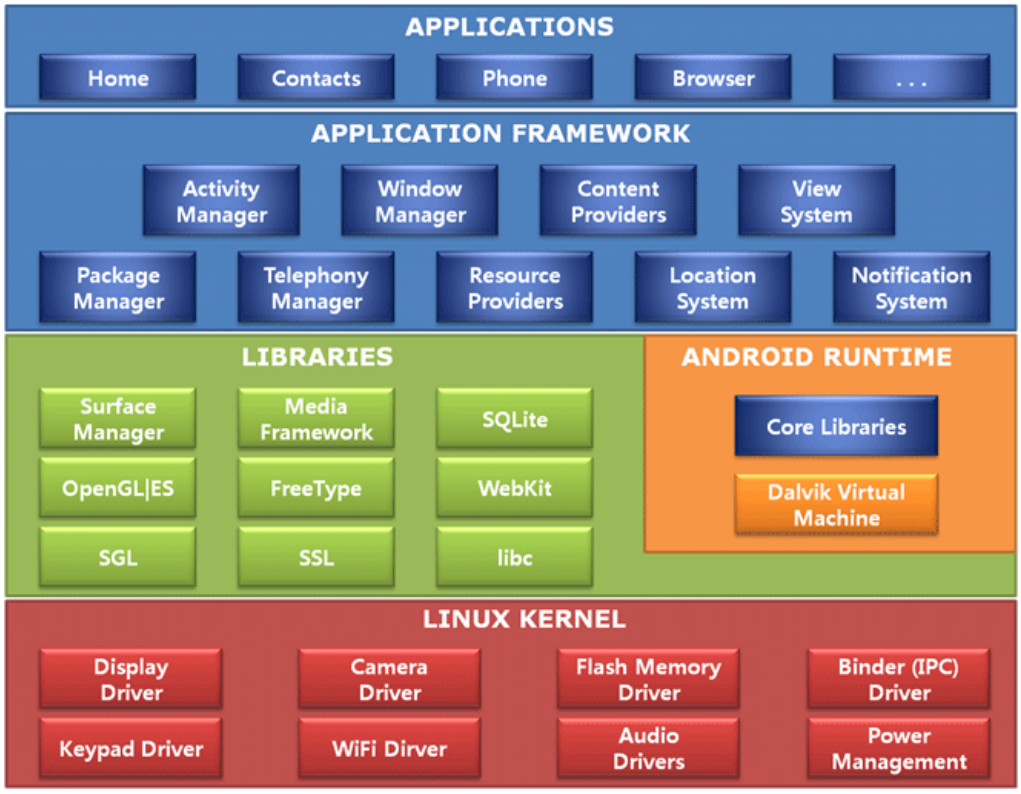


Ilustración 4: Estructura Android

Independientemente del lenguaje usado para desarrollar la aplicación hay que destacar una de las mayores ventajas que presenta este sistema operativo y es la libertad. Gracias a ella, cualquier persona puede crear una aplicación fácilmente y adaptarla a otros dispositivos además de teléfonos móviles, como tablets, gps, relojes, microondas, incluso internet.

Además, si hablamos de potencial de distribución de una aplicación, Android domina el mercado mundial. Para ser más específicos, actualmente Apple tiene el 13% del mercado global, y Android en torno al 80%.

Es por todas estas razones, que se ha decidido desarrollar la aplicación "MyTraining" en lenguaje Java para el sistema operativo android.

2.3 Tecnologías en el servidor web

2.3.1 Spring

Spring es un framework del lenguaje de programación java que permite desarrollar aplicaciones de manera más rápida, eficaz y corta, ahorrando al programador la realización de tareas repetitivas y al programa, líneas de código. Cabe destacar que este framework es verdaderamente extenso y crece día a día para ayudar al desarrollo de aplicaciones web, por lo que se va a proceder a aportar una breve explicación de una de sus funciones básicas, dejando en mano del lector la búsqueda de conceptos más detallados y complejos.

Inversión de control

Inversión de control es una expresión que se usa para referirse al cambio en el flujo de ejecución y vida de los objetos con respecto a la programación tradicional, básicamente de lo que se trata es de invertir la forma en que se controla la aplicación, lo que antes dependía del programador, como por ejemplo, el orden en que se llaman los métodos para darle un comportamiento particular a la aplicación, ahora depende completamente de otro ente, en este caso, el framework, todo esto con la finalidad de crear aplicaciones más complejas y con funcionamientos más encapsulados y automáticos.

Inyección de dependencias

La inyección de dependencias es un tipo de inversión de control, donde el manejo de las propiedades de un objeto es inyectado a través de un constructor, un setter, un servicio, etc. Creando de esta manera un control diferente para un comportamiento más conveniente en nuestra aplicación, como se mencionó anteriormente.

Contenedor de Inversión de Control

Sabiendo lo anterior, se puede hablar ya del Contenedor de Inversión de Control, es la parte del framework encargada de realizar la inyección de dependencias a través de archivos de metadata, por ejemplo, un archivo XML. De esta manera, el programador se ahorra la creación, relación y configuración de objetos y manejo de sus ciclos de vida, simplificando así el código y haciéndolo más legible.

Hoy en día es la referencia mundial como framework de programación orientado al desarrollo web gracias a la constante labor de investigación e innovación que lleva a cabo su equipo de desarrollo.

La decisión de hacer uso de Spring para desarrollar el servicio Web REST del proyecto actual, ha sido debida a la agilidad y diversidad de herramientas que este framework provee, lo cual hace que su implementación y uso sea más sencillo de llevar a cabo. De todos los módulos que posee este framework, hemos hecho uso de los siguientes: Spring Boot, Spring Web Services y Spring Data, los cuales se describirán brevemente en apartados que siguen.

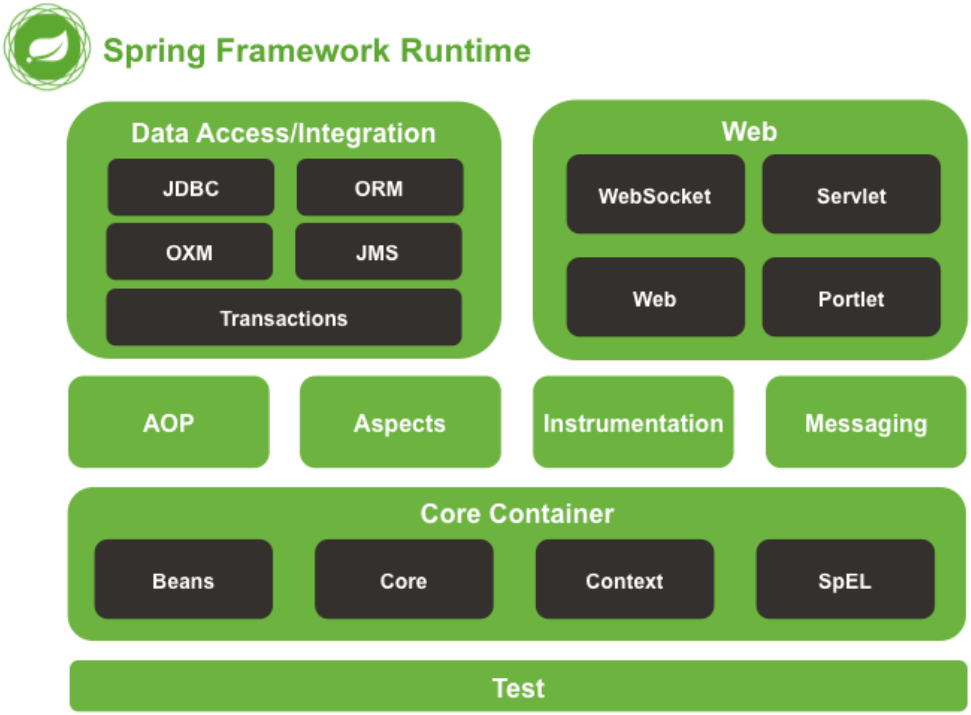
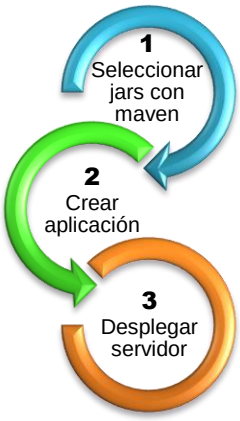


Ilustración 5: Spring Framework runtime

2.3.2 Spring Boot

En la imagen adjunta podemos ver los pasos fundamentales a seguir para la creación de proyectos en el framework de Spring.



Spring Boot es un módulo de Spring que busca facilitarnos el desarrollo de estos proyectos simplificando los pasos 1 y 2 de manera que se elimine la necesidad de crear largos archivos de configuración XML proveyendo configuraciones y gran cantidad de librerías y facilitando el arranque de la aplicación haciendo uso de una clase principal "Main", que se desplegará en un servidor web ya implementado por Spring Boot ahorrando al programador la configuración de uno nuevo.

Ilustración 6: Spring Boot

2.3.3 Spring y JDBC

Java Database Connectivity es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice. El clásico ciclo de acceso a datos de esta tecnología es el siguiente:

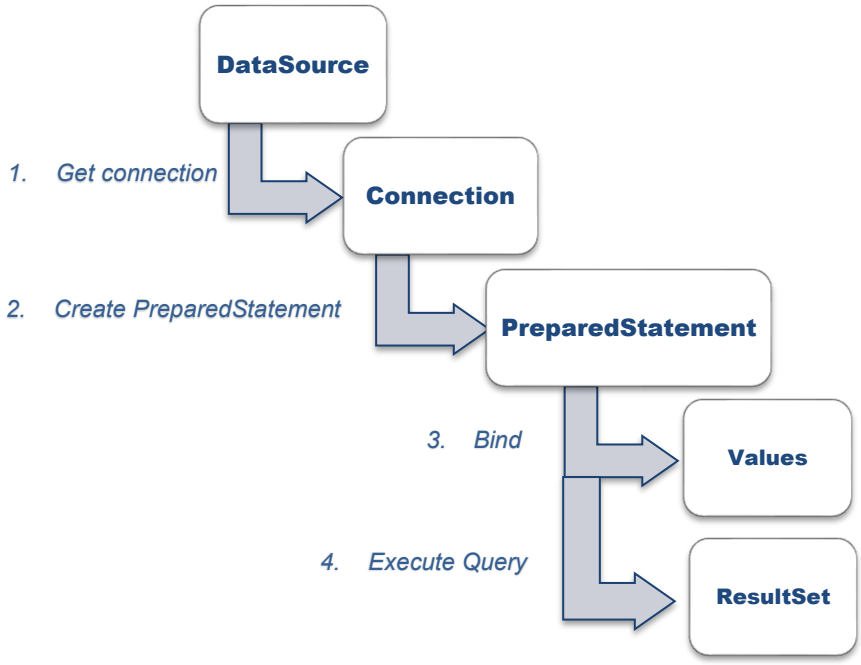


Ilustración 7: Ciclo de acceso a datos de JDBC

Para evitar al programador todo este proceso, Spring ha implementado la clase JdbcTemplate que es la encargada de realizar todo ese trabajo reduciendo el código de la aplicación y permitiendo al desarrollador centrarse en lo que realmente interesa.

Patrón DAO

El patrón de diseño software conocido como "Patrón DAO" ofrece abstraer y encapsular el acceso a datos envolviendo todo gracias al uso de objetos con el mismo nombre.

Un objeto DAO permite obtener y guardar datos, manejando la conexión con el sistema de persistencia implementado. Esto lo consigue mediante operaciones de creación, actualización, borrado y obtención de registros dentro de la base de datos.

El uso de estos objetos en la aplicación permite que ésta sea lo más transparente posible al sistema de almacenamiento usado, permitiendo la migración y adaptación a otros sistemas de almacenamiento simplemente ajustando la sintaxis de las consultas a realizar.

En este proyecto se ha hecho uso de este patrón para generar tantos objetos DAO como entidades creadas, de forma que dichos objetos serán los utilizados para realizar las consultas a la base de datos y obtener la información registrada en ella de cada entidad relacionada.

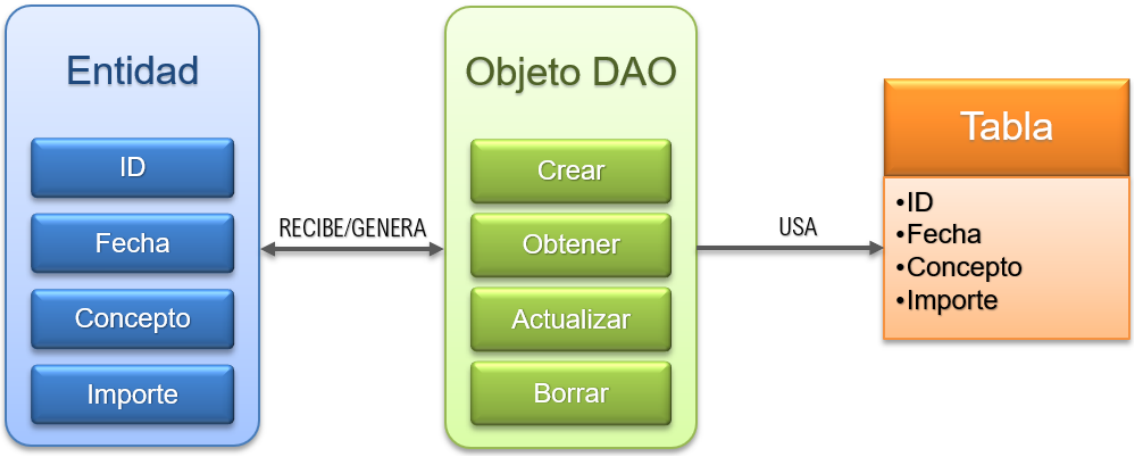


Ilustración 8: Funcionamiento de los objetos DAO

2.3.4 Spring Beans

Antes de entrar en la temática de Spring Beans, es importante pararse a entender qué es un bean. Este elemento es una clase destinada a almacenar una cantidad de datos necesarios para nuestra aplicación con el fin de encapsular información y reutilizar código fuente.

Un bean debe cumplir con los siguientes requisitos para poder cumplir con su función mencionada en el párrafo anterior:

- Implementar la interfaz "Serializable" para tener persistencia.
- Tener introspección, es decir, permitir analizar al IDE o herramienta de programación cómo trabaja dicho bean.
- Poseer un constructor público por defecto.
- Privatizar las variables.
- Implementar los métodos get y set de cada variable.

Una vez entendido el concepto y las características de un bean general, se puede profundizar en la temática que ocupa a este proyecto: Spring.

Un bean en Spring no es más que un objeto configurado e instanciado en el contenedor de este framework. Todos estos objetos permanecerán en el contenedor durante toda la vida de la aplicación o hasta que el desarrollador los destruya, lo cual permitirá inyectar un bean en otro, reutilizarlo o acceder a él desde cualquier lugar del programa cuando se desee.

Es por esto por lo que en el proyecto se ha hecho uso de los beans de Spring para así poder definir los objetos DAO y el dataSource y acceder a ellos desde cualquier parte de la aplicación.

2.4 Tecnologías en la base de datos

2.4.1 MySQL

MySQL, es un sistema de administración y gestión de bases de datos relacionales que ofrece las cuatro operaciones conocidas como C.R.U.D (Create, Read, Update, Delete) para poder alterar la información contenida.

Este gestor de base de datos destaca por su gran adaptación a diferentes entornos de desarrollo, permitiendo su interacción con los lenguajes de programación más utilizados como PHP, Perl y Java y su integración en distintos sistemas operativos. Además, es un sistema multihilo y multiusuario, lo que le permite ser utilizado por varias personas al mismo tiempo, e incluso, realizar varias consultas a la vez, lo que lo hace sumamente versátil.

También es muy destacable, la condición de open source de MySQL, que hace que su utilización sea gratuita e incluso se pueda modificar con total libertad, pudiendo descargar su código fuente. Esto ha favorecido muy positivamente en su desarrollo y continuas actualizaciones, para hacer de MySQL una de las herramientas más utilizadas por los programadores orientados a Internet.

Por lo mencionado en los párrafos anteriores es por lo que se ha escogido este gestor para administrar la base de datos del proyecto.

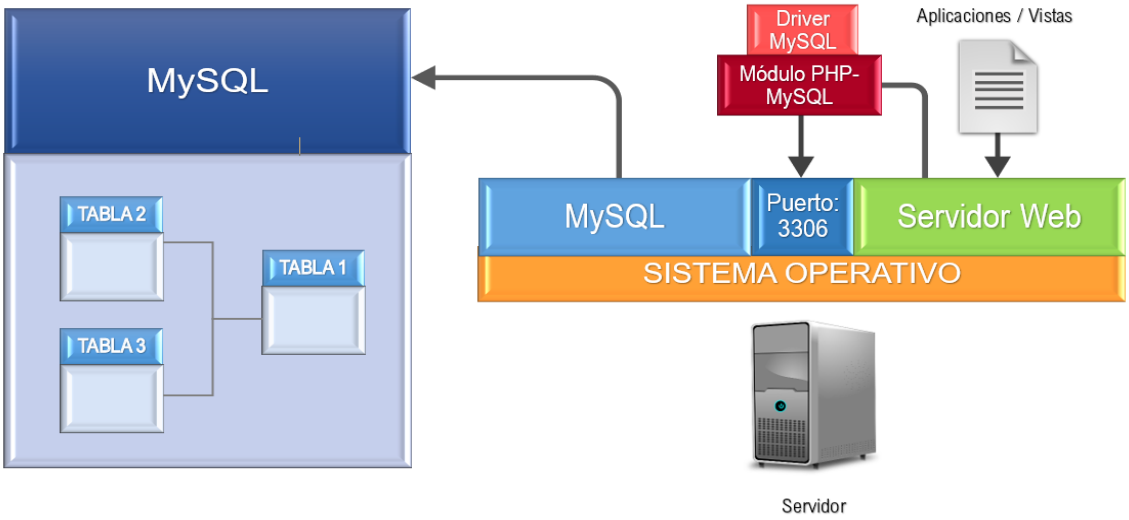


Ilustración 9: Comunicación con MySQL

3 HERRAMIENTAS USADAS

En el presente capítulo se va a proceder a presentar las herramientas utilizadas para crear el servicio que ocupa y que será descrito en próximos capítulos. Además, se detallará cada una de ellas y se enfatizará en la finalidad con la que han sido usadas.

3.1 Android Studio

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones con sistema operativo Android y se basa en IntelliJ IDEA. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece aún más funciones que aumentan la productividad durante la compilación de una aplicación, como las siguientes:

- Un sistema de compilación basado en Gradle flexible.
- Un emulador rápido con varias funciones.
- Un entorno unificado en el que puedes realizar desarrollos para todos los dispositivos Android.
- Instant Run para aplicar cambios mientras tu aplicación se ejecuta sin la necesidad de compilar un nuevo APK.
- Integración de plantillas de código y GitHub para ayudarte a compilar funciones comunes de las aplicaciones e importar ejemplos de código.
- Gran cantidad de herramientas y frameworks de prueba.
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versión, etc.
- Compatibilidad con C++ y NDK.
- Soporte incorporado para Google Cloud Platform, lo que facilita la integración de Google Cloud Messaging y App Engine.
- Compatible con Windows, Mac OS y Linux.

3.1.1 Gradle

Como se ha mencionado en el apartado anterior, Android Studio usa Gradle como la base del sistema de compilación, con más capacidades específicas de Android a través del complemento de Android para Gradle. Este sistema de compilación se ejecuta en una herramienta integrada desde el menú de Android Studio, y lo hace independientemente de la línea de comandos. Puedes usar las funciones del sistema de compilación para lo siguiente:

- Personalizar, configurar y extender el proceso de compilación.
- Crear múltiples APK para tu aplicación, con diferentes funciones utilizando el mismo proyecto y los mismos módulos.
- Volver a usar códigos y recursos entre conjuntos de archivos de origen.

Recurriendo a la flexibilidad de Gradle, se puede lograr todo esto sin modificar los archivos de origen de la aplicación. Los archivos de compilación de Android Studio denominados build.gradle, son archivos de texto sin formato que usan la sintaxis Groovy para configurar la compilación con elementos proporcionados por el complemento de Android para Gradle. Cada proyecto tiene un archivo de compilación de nivel superior para todo el proyecto y archivos de compilación de nivel de módulo independientes para cada módulo. Cuando se importa un proyecto existente, Android Studio genera automáticamente los archivos de compilación necesarios, liberando al desarrollador de realizar esta tediosa tarea.

Es por todas estas facilidades y ventajas que aporta Android Studio que ha sido el seleccionado para desarrollar la aplicación “MyTraining” frente a otras posibles herramientas como Eclipse o STS, el cual incorpora un módulo para desarrollo Android.

3.2 Spring Suite Tool

Spring Tool Suite (STS) proporciona un entorno ready-to-use para implementar, depurar, ejecutar y desplegar las aplicaciones Spring, incluyendo integraciones para Pivotal tc Server, Pivotal Cloud Foundry, Git, Maven, AspectJ, y viene encima de las últimas versiones de Eclipse.

Esta herramienta soporta el despliegue de aplicaciones tanto en servidores locales, virtuales y en la nube. Es de libre acceso para el desarrollo y uso en operaciones internas sin límite de tiempo, completamente de código abierto y licenciada bajo los términos de la Licencia Pública Eclipse.

Algunas de las ventajas que ofrece el uso de esta herramienta y que la hacen la mejor opción para desarrollar e implementar servicios Spring como es el caso de este proyecto son las siguientes:

- Analiza los archivos de configuración y muestra información detallada sobre los beans que están siendo definidos, sus dependencias entre sí, espacios de nombres utilizados, y extrae una visión general para ciertos estereotipos como controladores a petición, aspectos, los servicios y más.
- Ofrece un amplio conjunto de validaciones que se aplican de forma automática y se muestran en el mismo IDE sin tener que ejecutar la aplicación para ver los errores.
- Soporte Refactoring tanto para Java como para los elementos de Spring.
- Proporciona contenido de ayuda significativo en todos lados, junto con soluciones rápidas para los errores y problemas comunes.
- Se integra con las herramientas lenguaje AspectJ para Eclipse y proporciona el soporte más completo para AOP disponible hoy en día.
- Despliegue de las aplicaciones directamente en Cloud Foundry a elegir o en una instancia tc Server (incluyendo el soporte para la depuración, creación de la instancia, Spring Insight, servicios y más).

3.2.1 Maven

Se ha visto anteriormente que la herramienta de Spring, Spring Tool Suite, incorpora Maven para la inyección de dependencias. Maven es una herramienta capaz de gestionar un proyecto software completo, desde la etapa en la que se comprueba que el código es correcto, hasta que se despliega la aplicación, pasando por la ejecución de pruebas y generación de informes y documentación.

Para realizar todo esto, Maven presenta una serie de etapas para construir el software:

- Validación (validate): Validar que el proyecto es correcto.
- Compilación (compile).
- Test (test): Probar el código fuente usando un framework de pruebas unitarias.
- Empaquetar (package): Empaquetar el código compilado y transformarlo en algún formato tipo .jar o .war.
- Pruebas de integración (integration-test): Procesar y desplegar el código en algún entorno donde se puedan ejecutar las pruebas de integración.

- Verificar que el código empaquetado es válido y cumple los criterios de calidad (verify).
- Instalar el código empaquetado en el repositorio local de Maven, para usarlo como dependencia de otros proyectos (install).
- Desplegar el código a un entorno (deploy).

En el caso que ocupa a este documento, que es el desarrollo del servicio web Spring, es necesario aclarar que la funcionalidad de Maven usada ha sido la inyección de dependencias, de tal manera, que se han añadido las librerías/módulos requeridos al archivo POM.xml que es el fichero de configuración que aporta Maven para gestionar lo dicho.

3.3 XAMPP

XAMPP es un paquete de software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script PHP y Perl. El programa se distribuye con la licencia GNU y actúa como un servidor web libre, fácil de usar y capaz de interpretar páginas dinámicas. Actualmente, XAMPP está disponible para Windows, GNU/Linux, Solaris y Mac OS X.

3.3.1 phpMyAdmin

Además de lo ya dicho, XAMPP incorpora también gestores para facilitar al programador el manejo de los servicios aportados. Entre ellos cabe destacar la herramienta phpMyAdmin que ha sido utilizada en este proyecto para la creación y gestión de la base de datos de la aplicación facilitando la manipulación de tablas y las relaciones entre ellas.

3.4 Wireshark

Se trata de un potente sniffer de red de software libre heredero de Ethereal, que permite, a través de una amigable interfaz, capturar y monitorizar todos los paquetes de red que pasan por un equipo con el solo hecho de indicar la tarjeta de red a escuchar de manera que capture todo el tráfico que pase por ella.

4 FUNCIONALIDAD Y COMUNICACIÓN

En este capítulo se detallará la funcionalidad de la aplicación desarrollada y cómo ésta se comunica con el servicio web implementado. Para ello, se usarán dos tipos de diagramas UML: diagramas de casos de uso y diagramas de secuencia.

4.1 Diagramas de casos de uso

A continuación, se presentarán los diagramas de casos de uso con el fin de especificar la comunicación y el comportamiento del sistema desarrollado mediante la interacción de éste con los usuarios y/u otros sistemas. Además, se adjuntarán unas tablas explicativas para cada caso de uso.

4.1.1 Términos a tener en cuenta

En este apartado se mencionarán una serie de términos con el fin de facilitar la comprensión de los casos de uso que se listarán en la siguiente sección de este capítulo.

- Usuarios:
  - No autenticado: Será el usuario que inicie la aplicación sin tener una cuenta creada o teniéndola, pero no habiendo iniciado sesión, es decir, que habiéndose registrado anteriormente haya cerrado sesión y deba volver a introducir sus datos en el inicio de sesión.
  - Autenticado: Será el usuario que haya iniciado sesión en la aplicación directamente (si tenía una cuenta asociada) o tras haberse registrado en el sistema (si no había creado una cuenta hasta ahora).
- Abreviaciones:
  - MT: Abreviación para el nombre de la aplicación desarrollada, MyTraining.
  - UNA: Abreviación para identificar al Usuario No Autenticado.
  - UA: Abreviación para identificar al Usuario Autenticado.
  - BBDD: Abreviación para Base de Datos.

4.1.2 Inicio de sesión

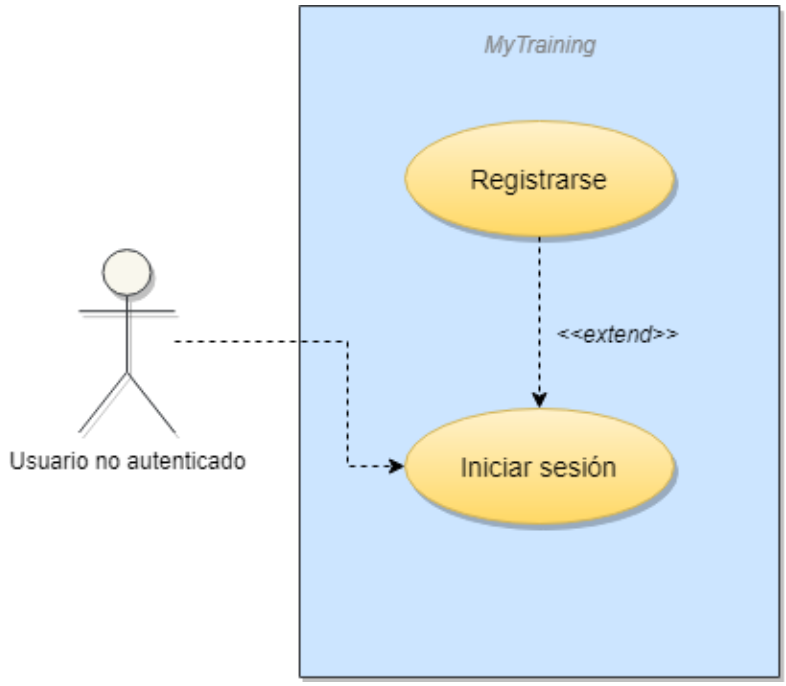


Ilustración 10: Diagrama de caso de uso CU-01

Nombre	*CU-01: Operaciones posibles para un UNA.*	
Objetivo	*Realizar las operaciones disponibles en la pantalla de inicio.*	
Disparador	*UNA inicia la aplicación en su dispositivo móvil.*	
Precondiciones	*El actor no debe tener cuenta creada o, en el caso de tenerla, haber cerrado sesión anteriormente.*	
Descripción	*El UNA iniciará la aplicación y solo podrá autenticarse/crear cuenta para poder hacer uso de las funcionalidades que ofrece MT.*	
Secuencia normal	1	*El UNA descarga MT en su dispositivo e inicia la aplicación.*
	2	*El UNA inicia sesión a través de su número de teléfono.*
	3	*Si el UNA no tenía una cuenta creada se registrará en la aplicación poniendo sus datos personales. En caso de tenerla, accederá directamente al menú de la aplicación.*

Tabla 1: Tabla de caso de uso CU-01

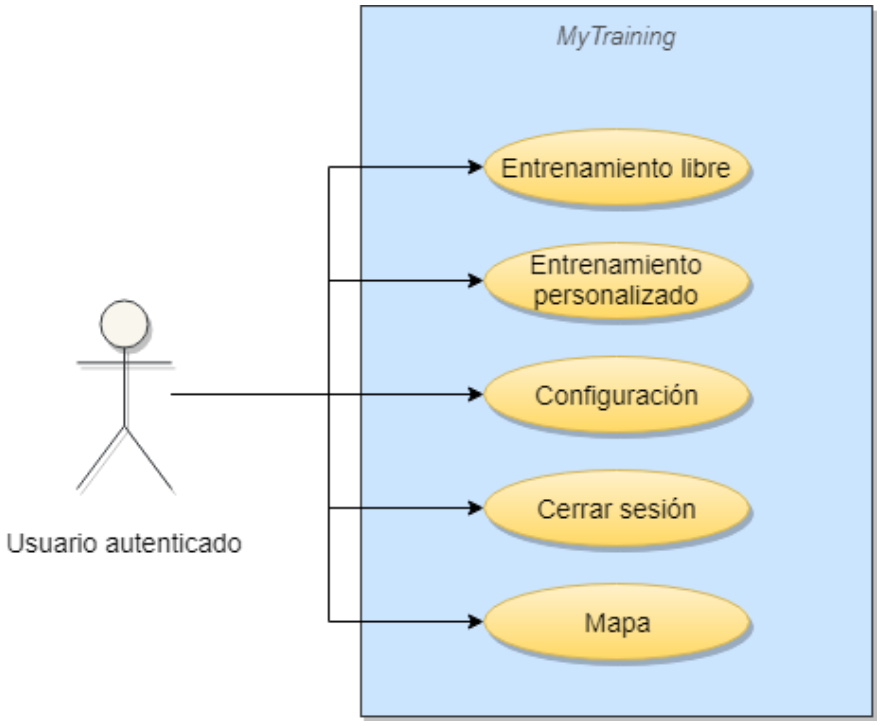


Ilustración 11: Diagrama de caso de uso CU-02

Nombre	*CU-02: Operaciones posibles en el menú de MT para un UA.*	
Objetivo	*Realizar las operaciones disponibles en la pantalla del menú.*	
Disparador	*UA inicia la aplicación y se autentica en el caso de tener la sesión cerrada.*	
Precondiciones	*El actor debe haberse autenticado con éxito.*	
Descripción	*El UA accederá al menú tras iniciar sesión y podrá hacer uso de toda la funcionalidad que ofrece MT.*	
Secuencia normal	1	*El actor accede al menú de MT.*
	2	*El UA accede a sus rutinas personalizadas y a todos los ejercicios existentes clasificados por objetivos.*
	3	*El UA accede al mapa.*
	4	*El UA accede a la configuración de la cuenta.*
	5	*El UA accede a la información acerca de MT.*

Tabla 2: Tabla de caso de uso CU-02

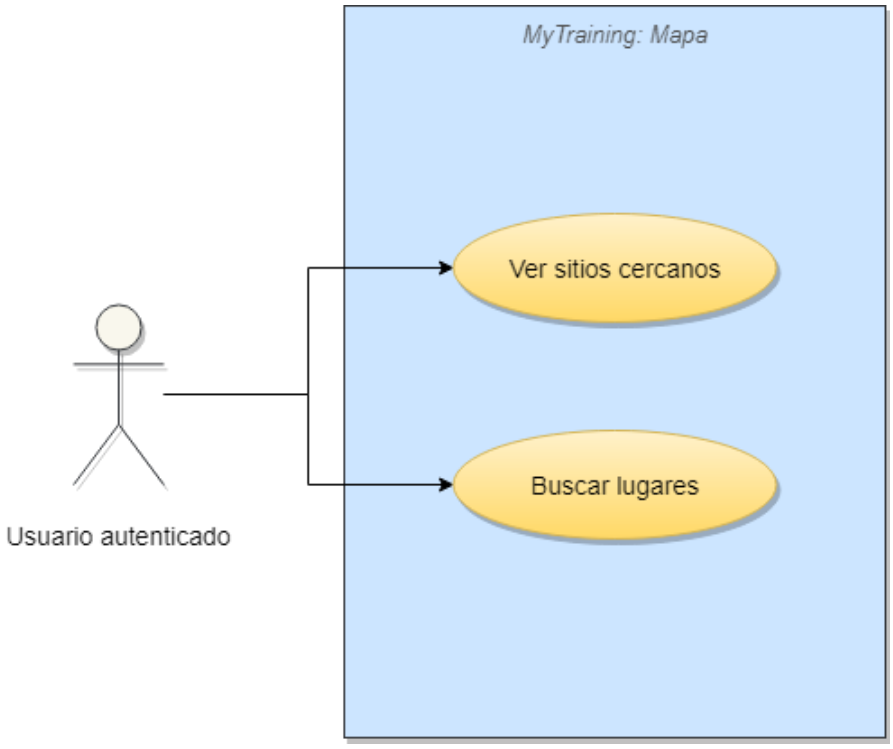


Ilustración 12: Diagrama de caso de uso CU-03

Nombre	*CU-03: Operaciones posibles en el mapa de MT para un UA.*	
Objetivo	*Realizar las operaciones disponibles en la pantalla del mapa.*	
Disparador	*UA selecciona la opción del mapa.*	
Precondiciones	*El actor debe haberse autenticado con éxito.*	
Descripción	*El UA accederá al mapa y podrá hacer uso de la funcionalidad que esta pantalla ofrece.*	
Secuencia normal	1	*El UA accede al mapa.*
	2	*La pantalla muestra la ubicación actual del actor.*
	3	*El UA ve los sitios cercanos a él.*
	4	*El UA hace uso del buscador del mapa para localizar cualquier sitio o establecimiento.*

Tabla 3: Tabla de caso de uso CU-03

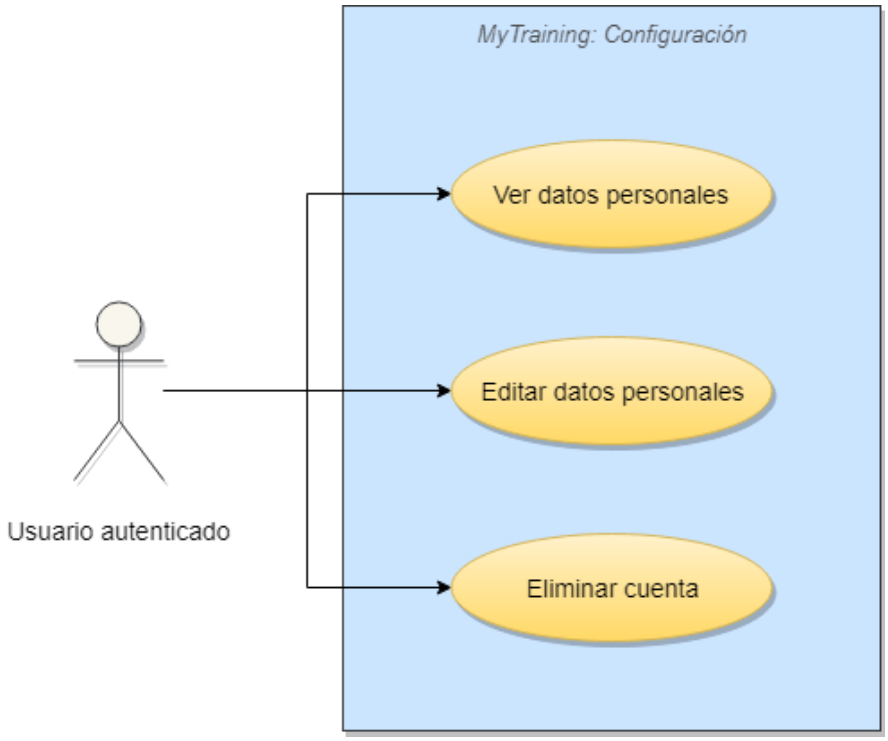


Ilustración 13: Diagrama de caso de uso CU-04

Nombre	*CU-04: Operaciones posibles en la configuración de un UA.*	
Objetivo	*Realizar las operaciones disponibles en la pantalla de configuración.*	
Disparador	*UA selecciona la opción de la configuración.*	
Precondiciones	*El actor debe haberse autenticado con éxito.*	
Descripción	*El UA accederá a la configuración y podrá hacer uso de la funcionalidad que esta pantalla ofrece.*	
Secuencia normal	1	*El UA accede a la configuración.*
	2	*El actor ve sus datos personales.*
	3	*El actor edita dichos datos y se guardarán solo sí son válidos.*
	4	*El UA elimina su cuenta y sus datos con ella.*

Tabla 4: Tabla de caso de uso CU-04

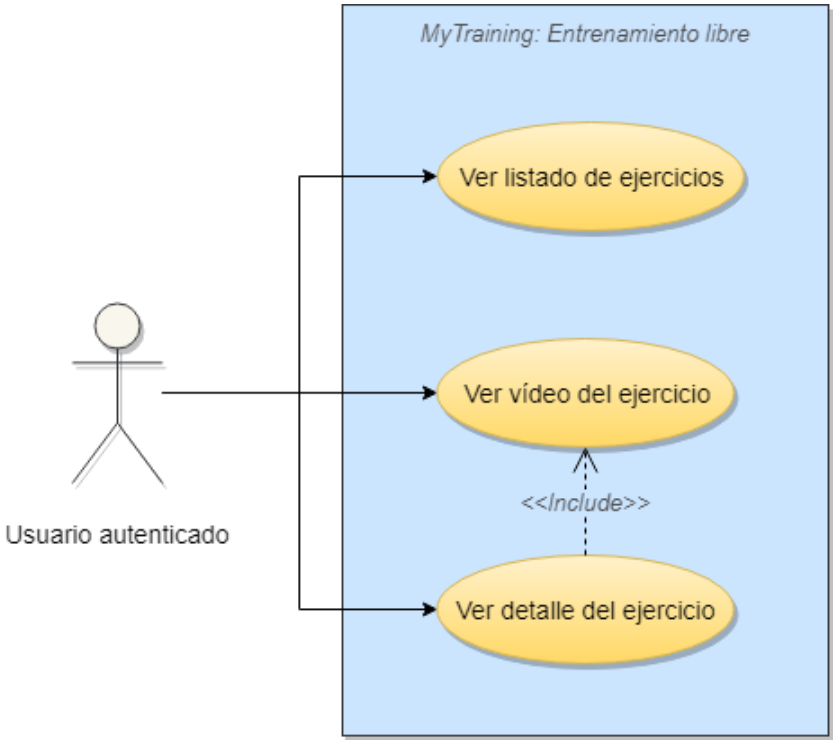


Ilustración 14: Diagrama de caso de uso CU-05

Nombre	*CU-05: Operaciones posibles en la opción de entrenamiento libre.*	
Objetivo	*Realizar las operaciones disponibles en la pantalla de entrenamiento libre.*	
Disparador	*UA selecciona la opción de entrenamiento libre del menú.*	
Precondiciones	*El actor debe haberse autenticado con éxito.*	
Descripción	*El UA accederá al entrenamiento libre y podrá hacer uso de la funcionalidad que esta pantalla ofrece.*	
Secuencia normal	1	*El UA accede al entrenamiento libre.*
	2	*El actor navega por las pestañas para ver los listados de ejercicios existentes en MT.*
	3	*El actor selecciona un ejercicio y accede a su descripción.*
	4	*El actor reproduce el vídeo del ejercicio.*

Tabla 5: Tabla de caso de uso CU-05

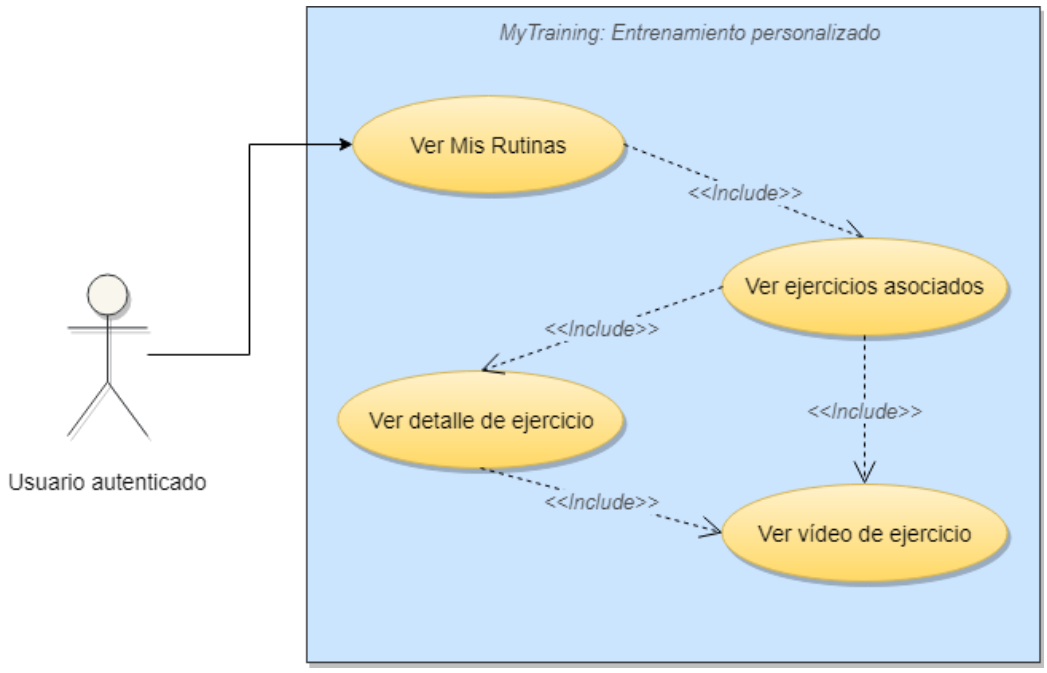


Ilustración 15: Diagrama de caso de uso CU-06

Nombre	*CU-05: Operaciones posibles en la opción de entrenamiento personalizado.*	
Objetivo	*Realizar las operaciones disponibles en la pantalla de entrenamiento personalizado.*	
Disparador	*UA selecciona la opción de entrenamiento personalizado del menú.*	
Precondiciones	*El actor debe haberse autenticado con éxito.*	
Descripción	*El UA accederá al entrenamiento personalizado y podrá hacer uso de la funcionalidad que esta pantalla ofrece.*	
Secuencia normal	1	*El UA accede al entrenamiento personalizado.*
	2	*El actor ve su lista de rutinas personalizadas y su descripción.*
	3	*El UA selecciona una rutina y accede al conjunto de ejercicios que la forman.*
	4	*El actor reproduce el vídeo de un ejercicio.*
	5	*El UA selecciona un ejercicio y accede a su detalle.*

Tabla 6: Tabla de caso de uso CU-06

4.2 Comunicación entre cliente y servidor

Esta sección del capítulo está centrada en explicar al lector cómo se ha establecido la comunicación entre la aplicación móvil o cliente y el servicio web Spring desarrollado que actúa como servidor y qué elemento ha interferido entre ellos para poder garantizar el éxito de dicha comunicación.

Para establecer el canal de comunicación por donde cliente y servidor se intercambian información e interactúan entre sí se ha hecho uso de la plantilla proporcionada por Spring conocida como "RestTemplate" que será detallada a continuación junto con una breve explicación de su implementación en la aplicación creada.

RestTemplate es la clase que ofrece Spring para el acceso desde la parte cliente a un servicio REST y es la encargada de realizar la conexión HTTP a través de la URI proporcionada por el desarrollador durante su implementación y de gestionar sus propios convertidores de mensajes con los que parseará los datos JSON/XML recibidos/enviados hacia/desde el servidor.

Profundizando en el ámbito que concierne al desarrollador cabe destacar dónde y cómo se ha implementado este elemento en la aplicación móvil para que ésta sea capaz de establecer una comunicación con el servidor creado e intercambiar información con él.

Dentro de los activities pertinentes se ha hecho uso de llamadas asíncronas al servidor, mediante la clase AsyncTask aportada por Android, de manera que la aplicación pueda seguir funcionando en primer plano y sin que el usuario note lentitud alguna mientras hace uso de ella, y, mientras tanto, en un segundo plano se realizan todas las llamadas e intercambio de datos con el servidor necesarios para satisfacer las peticiones que el usuario haga desde la interfaz. A continuación, se presenta un trozo de código de la aplicación para ejemplificar el funcionamiento de AsynTask y el uso de RestTemplate.

```
// ********************************************************************//
// ********************** GETTING THE USER ***********************//
// ********************************************************************//

private class getUserHttpRequestTask extends AsyncTask<Void, Void, User[]> {

    @Override
    protected User[] doInBackground(Void... params) {
        try {

            final String url = "http://192.168.1.92:8080/myTraining/Account/{phone}";

            // URI (URL) parameters
            HashMap<String, String> uriParams = new HashMap<>();
            uriParams.put("phone", phoneNumberString);

            // Create a new RestTemplate instance
            RestTemplate restTemplate = new RestTemplate();
            restTemplate.getMessageConverters().add(new MappingJackson2HttpMessageConverter());


            // Make the HTTP GET request, getting an User type and sending the phone to the URI
            User[] user = restTemplate.getForObject(url, User[].class, uriParams);

            return user;


        } catch (Exception e) {
            Log.e( tag: "ConfigurationActivity", e.getMessage(), e);
        }

        return null;
    }

    protected void onPostExecute(User[] user) {
        super.onPostExecute(user);
        ;

        fieldName.setText(user[0].getName());
        fieldPhone.setText(user[0].getPhone());
        fieldEmail.setText(user[0].getEmail());

    }

}
```

Como vemos en la captura anterior la clase RestTemplate hace uso de un método llamado "getForObject" para obtener el usuario y sus datos y almacenarlo en un objeto User y es que esta clase posee una gran variedad de métodos que permiten realizar las distintas peticiones HTTP conocidas como GET, PUT, POST y DELETE. En la siguiente tabla se presentan los principales métodos y las llamadas que realizan se listan a continuación para conocimiento del lector.

MÉTODOS HTTP	MÉTODOS REST TEMPLATE
DELETE	`delete(java.lang.String, java.lang.Object...)`
GET	`getForObject(java.lang.String, java.lang.Class<T>, java.lang.Object...)`
	`getForEntity(java.lang.String, java.lang.Class<T>, java.lang.Object...)`
POST	`postForLocation(java.lang.String, java.lang.Object, java.lang.Object...)`
	`postForObject(java.lang.String, java.lang.Object, java.lang.Class<T>, java.lang.Object...)`
PUT	`put(java.lang.String, java.lang.Object, java.lang.Object...)`

Tabla 7: Métodos de RestTemplate.

Como se puede apreciar, la comunicación entre cliente y servidor es fácilmente implementable gracias a esta plantilla y sus métodos, los cuales dan la posibilidad de realizar muchas peticiones aportando a cualquier aplicación que los implemente una funcionalidad diversa.

5 SERVICIO WEB

En este quinto capítulo se profundizará en el funcionamiento del servicio web Spring desarrollado mediante la exposición de diagramas UML que permitan al lector comprender su estructura y las operaciones que éste realiza contra la base de datos para la obtención de información.

Además de la exposición de dichos diagramas se hará especial hincapié en la API del servicio REST, detallando los recursos implicados y sus respectivas operaciones CRUD.

5.1 Diagrama de clases

Los diagramas de clases son un tipo de diagramas estáticos que describen la estructura de un sistema mostrando sus clases, atributos, métodos y las relaciones entre clases con independencia de su implementación.

La estructura de estos diagramas es muy similar para los distintos tipos de datos (Usuarios, Rutinas y Ejercicios) que maneja el sistema creado. Esta estructura se caracteriza por las siguientes clases:

Ilustración 16: Estructura del servicio web Spring.

De esta manera, se presentará a continuación un diagrama de clase para cada paquete encargado de gestionar su parte correspondiente de la aplicación y garantizar su correcto funcionamiento.

5.1.1 Diagrama de clases para la gestión de usuarios

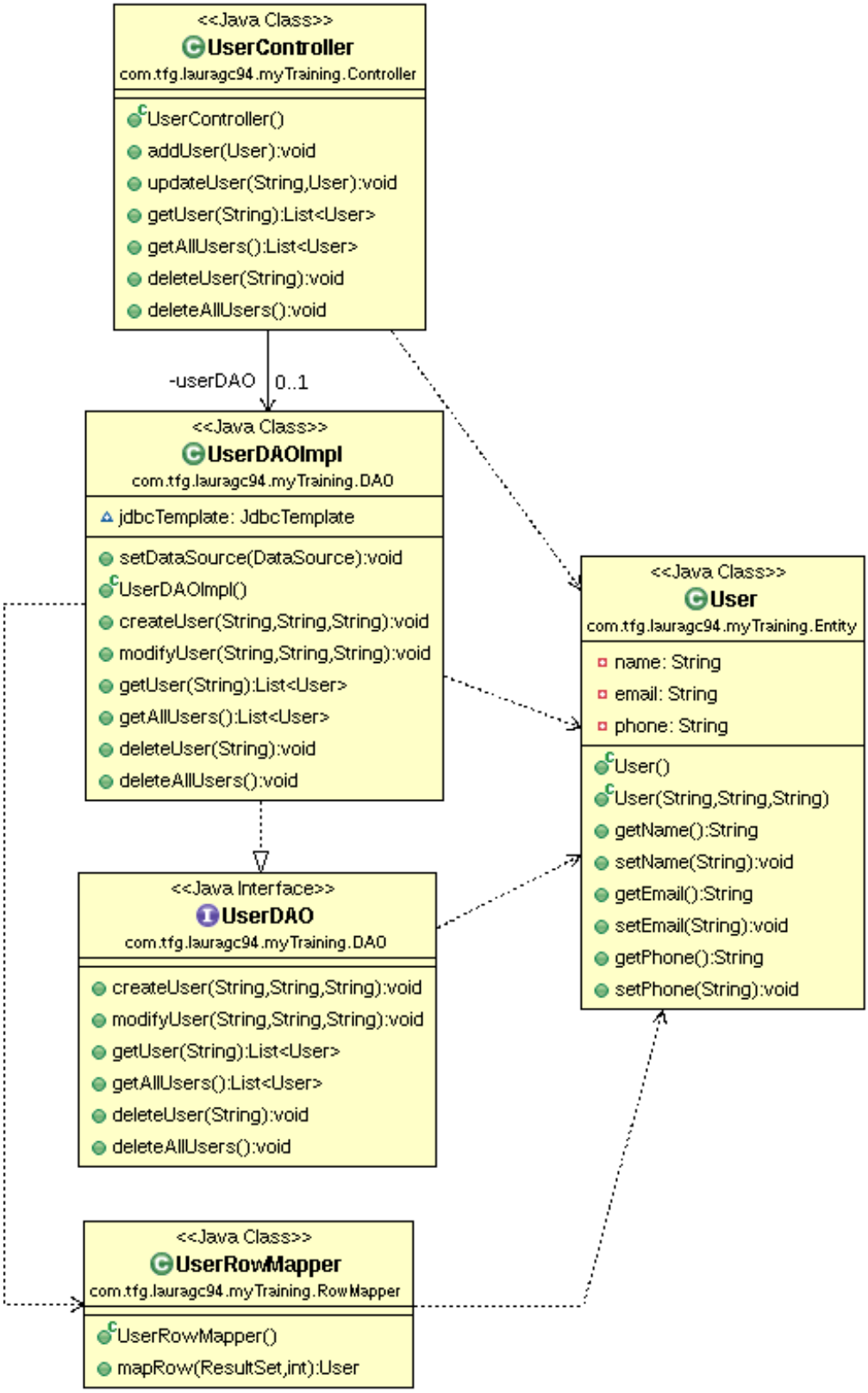


Ilustración 17: Diagrama de clases para la gestión de usuarios.

5.1.2 Diagrama de clases para la gestión de rutinas

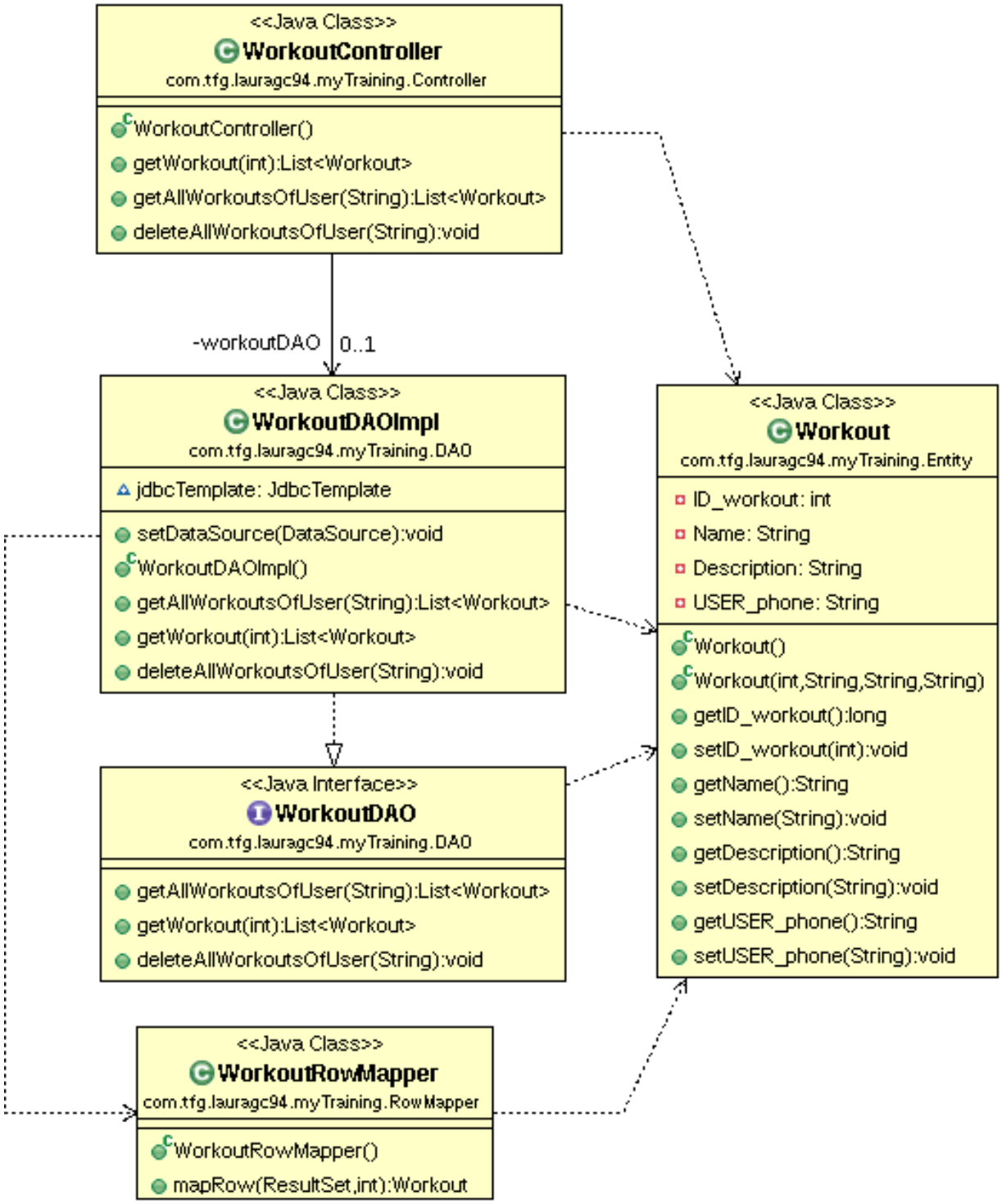


Ilustración 18: Diagrama de clases para la gestión de rutinas.

5.1.3 Diagrama de clases para la gestión de ejercicios

Ilustración 19: Diagrama de clases para la gestión de ejercicios.

5.2 API del servicio REST

En esta sección se va a mostrar la API del servicio REST implementado, concepto el cual ya fue explicado en la sección Arquitectura REST del presente documento, por lo que no se parará en este aspecto, pero sí en detallar los recursos y sus respectivas operaciones CRUD usadas para dar funcionalidad a la aplicación.

5.2.1 Métodos y URIs de un usuario

Recurso	Función	Método	URI	Consulta SQL	Resultado
USUARIO	addUser	POST	/myTraining/Register//	UPDATE	*Crea un usuario con los datos introducidos desde la aplicación.*
	updateUser	POST	/myTraining/Account/Modify/{phone:.+}	UPDATE	*Modifica el usuario con ID=phone cambiando los datos introducidos desde la aplicación.*
	getUser	GET	/myTraining/Account/{phone:.+}	QUERY	*Obtiene todos los datos del usuario con ID=phone.*
	deleteUser	DELETE	/myTraining/Account/Delete/{phone:.+}	UPDATE	*Elimina al usuario con ID=phone.*

Tabla 8: API REST de usuario.

5.2.2 Métodos y URIs de una rutina

Recurso	Función	Método	URI	Consulta SQL	Resultado
RUTINA	getWorkout	GET	/myTraining/Workout/{ID_workout:.+}	QUERY	*Obtiene todos los datos de una rutina con ID=ID_workout.*
	getAllWorkoutsOfUser	GET	/myTraining/Workouts/User/{USER_phone:.+}	QUERY	*Obtiene todas las rutinas asignadas al usuario con ID=phone.*
	deleteAllWorkoutsOfUser	DELETE	/myTraining/Workouts/User/Delete/{USER_phone:.+}	UPDATE	*Elimina todas las rutinas del usuario con ID=phone.*

Tabla 9: API REST de rutina.

5.2.3 Métodos y URIs un ejercicio

Recurso	Función	Método	URI	Consulta SQL	Resultado
EJERCICIO	getExercise	GET	/myTraining/Exercises/ {ID_exercise:.+}	QUERY	*Obtiene todos los datos de un ejercicio con ID=ID_exercise.*
	getAllExercises	GET	/myTraining/Exercises/Objective/ {Objective:.+}	QUERY	*Obtiene todos los ejercicios con objetivo=Objective.*
	getAllExercises OfWorkout	GET	myTraining/Workout/Exercises/ {ID_workout:.+}	QUERY	*Obtiene todos los ejercicios de la rutina con ID=ID_workout.*
	deleteExercises OfWorkout	DELETE	myTraining/Workout/Exercises /Delete/{ID_workout:.+}	UPDATE	*Elimina todos los ejercicios de la rutina con ID=ID_workout.*

Tabla 10: API REST de ejercicio.

6 APLICACIÓN MÓVIL

El presente capítulo va a dividirse en distintas secciones que cubran conjuntamente la funcionalidad, estructura y diseño de la aplicación móvil objeto de este proyecto.

En primer lugar, se expondrá la estructura y composición de la aplicación desarrollada para posteriormente adentrar al lector en el funcionamiento de ésta mostrando a su vez la interfaz diseñada que se presentará al usuario ocultando el backend, pero dándole la funcionalidad que éste ofrece.

Todo esto va a ser implementado en este documento mediante esquemas representativos, diagramas de actividad y capturas de pantalla de la propia aplicación.

6.1 Estructura

A continuación, se va a presentar un esquema de los paquetes que componen la aplicación dando una breve descripción de lo que contienen sin llegar a profundizar ya que no es relevante para la comprensión del sistema diseñado.

BEGINNING	*Paquete que contiene las clases de inicio.*
LOGIN	*Paquete que contiene las clases de logueo y registro.*
MENU	*Paquete que contiene las clases del menú.*
FREE TRAINING	*Paquete que contiene las clases del entrenamiento libre.*
CUSTOM TRAINING	*Paquete que contiene las clases del entrenamiento personalizado.*
VIDEOS	*Paquete que contiene las clases de los vídeos.*
ENTITIES	*Paquete que contiene las clases de las entidades.*

Ilustración 20: Estructura de la aplicación móvil

6.2 Diagrama de clases

En este apartado se presentarán los diagramas de clases que muestran las clases que componen la aplicación y las relaciones entre ellas. Para evitar demasiada información condensada en una misma imagen, se van a mostrar cuatro diagramas de clases distintos clasificados según los criterios explicados a continuación.

- **Diagrama de clases referentes a la entidad Usuario**. Este diagrama comprenderá las clases contenidas en los paquetes anteriormente definidos y conocidos como "Beginning","Login" y "Menu" y sus relaciones y dependencias entre ellas. Además se mostrarán las peticiones HTTP realizadas al servidor contenidas en cada clase y su asociación con la entidad referente.

- **Diagrama de clases referentes a la entidad Rutina**. Este diagrama comprenderá las clases contenidas en los paquetes anteriormente definidos y conocidos como "Menu" y "CustomTraining" y sus relaciones y dependencias entre ellas. Además se mostrarán las peticiones HTTP realizadas al servidor contenidas en cada clase y su asociación con la entidad referente.

- **Diagrama de clases referentes a la entidad Ejercicio**. Este diagrama comprenderá las clases contenidas en los paquetes anteriormente definidos y conocidos como "Menu" y "FreeTraining" y sus relaciones y dependencias entre ellas. Además se mostrarán las peticiones HTTP realizadas al servidor contenidas en cada clase y su asociación con la entidad referente.

- D**iagrama de clases referentes a la carga de Videos**. Este diagrama comprenderá las clases contenidas en el paquete anteriormente definido y conocido como "Videos" y sus relaciones y dependencias entre ellas.

6.2.1 Diagrama de clases referentes a la entidad Usuario

En el diagrama adjunto se podrán ver las clases que manipulan información del usuario durante su registro, inicio de sesión, acceso al menú y a la configuración de su cuenta haciendo llamadas al servidor mediante peticiones HTTP para obtener los datos deseados para posteriormente mostrarlos al usuario en la interfaz de la aplicación.

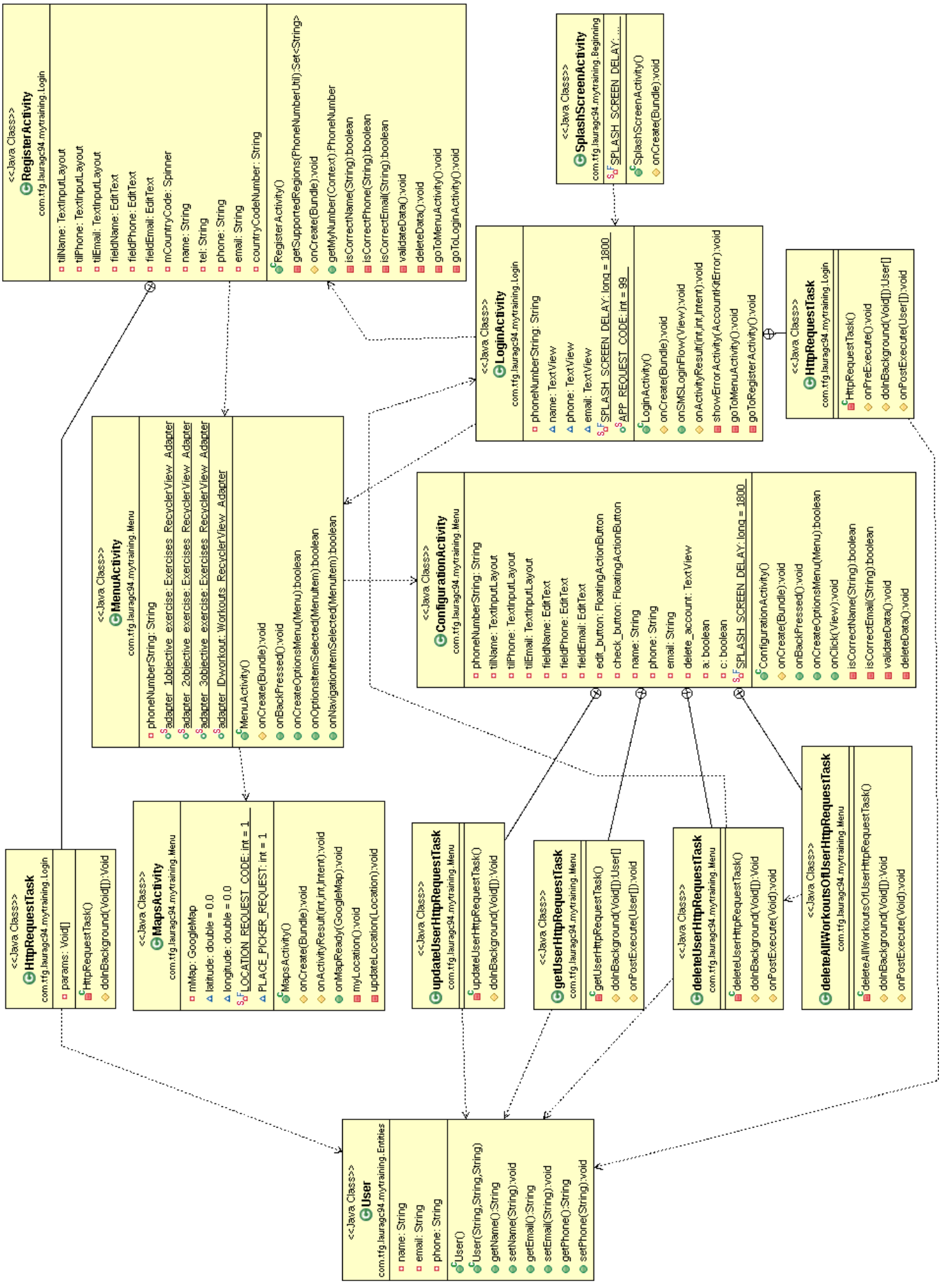


Ilustración 21: Diagrama de clases referente a la entidad Usuario.

6.2.2 Diagrama de clases referentes a la entidad Rutina

En el diagrama adjunto se podrán ver las clases que manipulan las rutinas de un usuario, accediendo a ellas y mostrando su información adquirida a través de las peticiones HTTP que la aplicación realiza al servidor para obtener dichos datos almacenados en la tabla de rutinas de la base de datos.

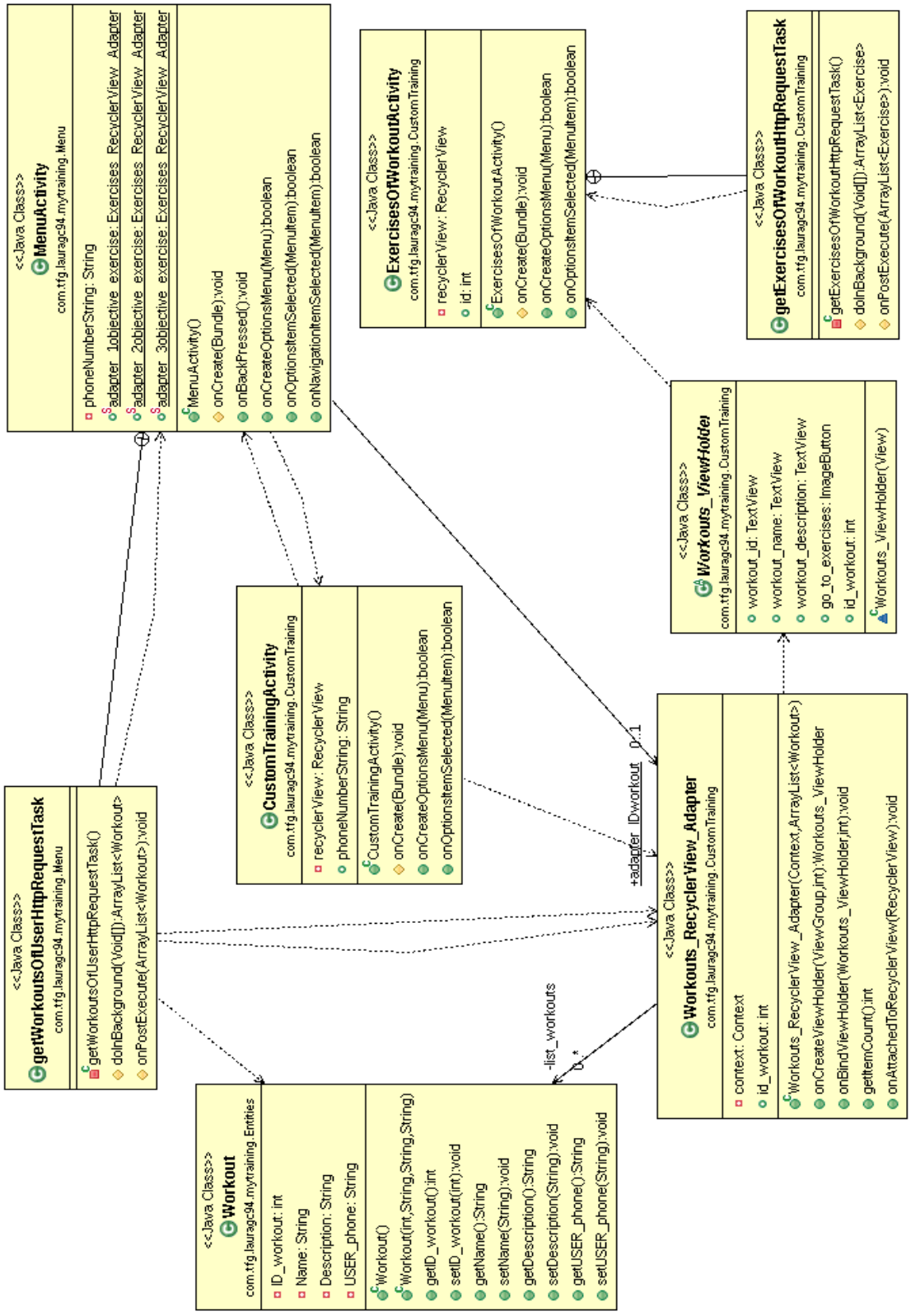


Ilustración 22: Diagrama de clases referente a la entidad Rutina.

6.2.3 Diagrama de clases referentes a la entidad Ejercicio

En el diagrama adjunto se podrán ver las clases que manipulan los ejercicios de manera que habrá unas clases encargadas de mostrar al usuario los ejercicios según el objetivo que tengan éstos y, otras, que filtrarán los ejercicios según la rutina a la que pertenezcan.

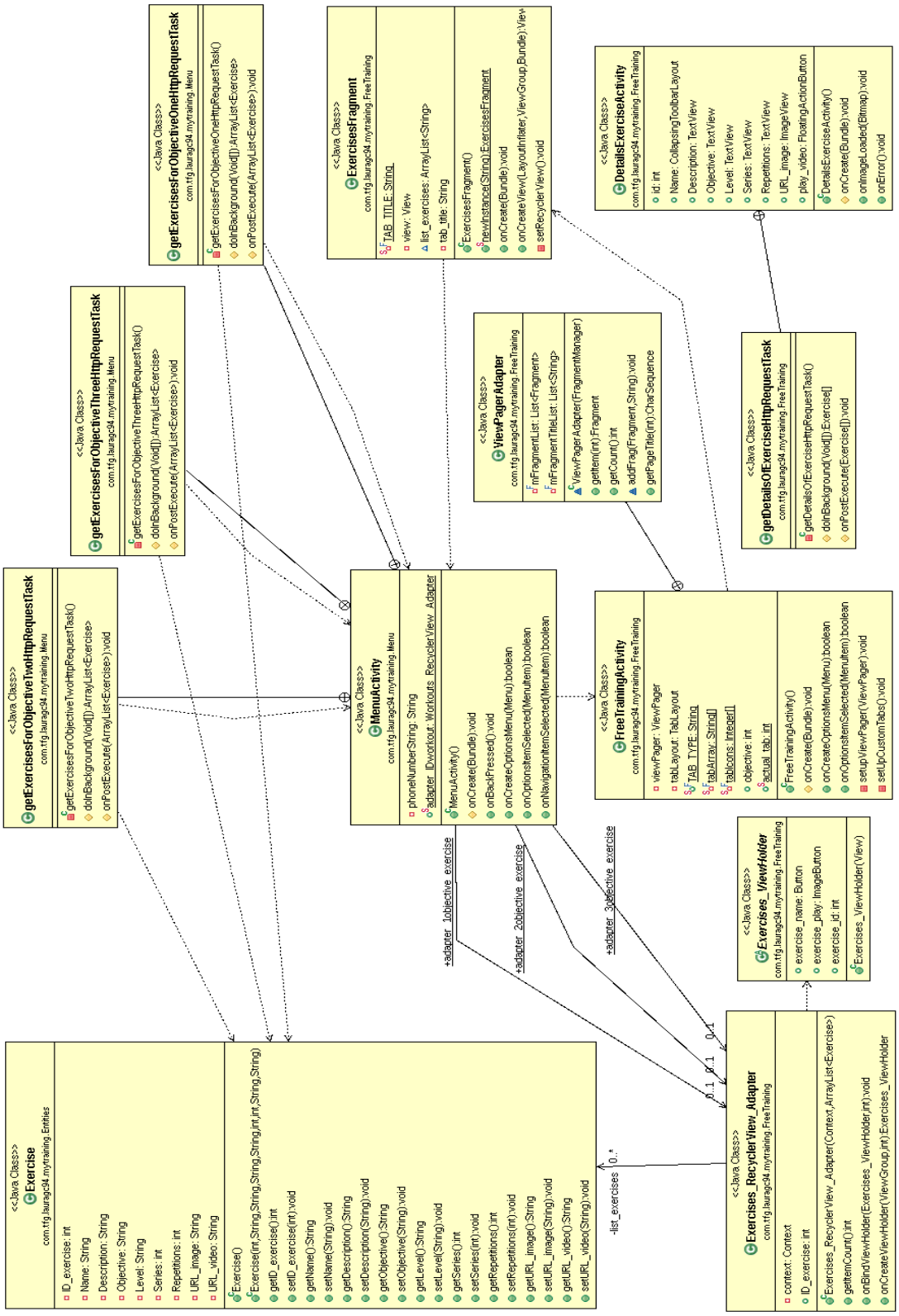


Ilustración 23: Diagrama de clases referente a la entidad Ejercicio.

6.2.4 Diagrama de clases referentes a la carga de Videos

En el diagrama adjunto se podrán ver las clases que manipulan los vídeos de los ejercicios asociados obtenidos gracias a la implementación de la API de Youtube para Android que será explicada en la sección Servicios externos y APIs implementadas de este capítulo.

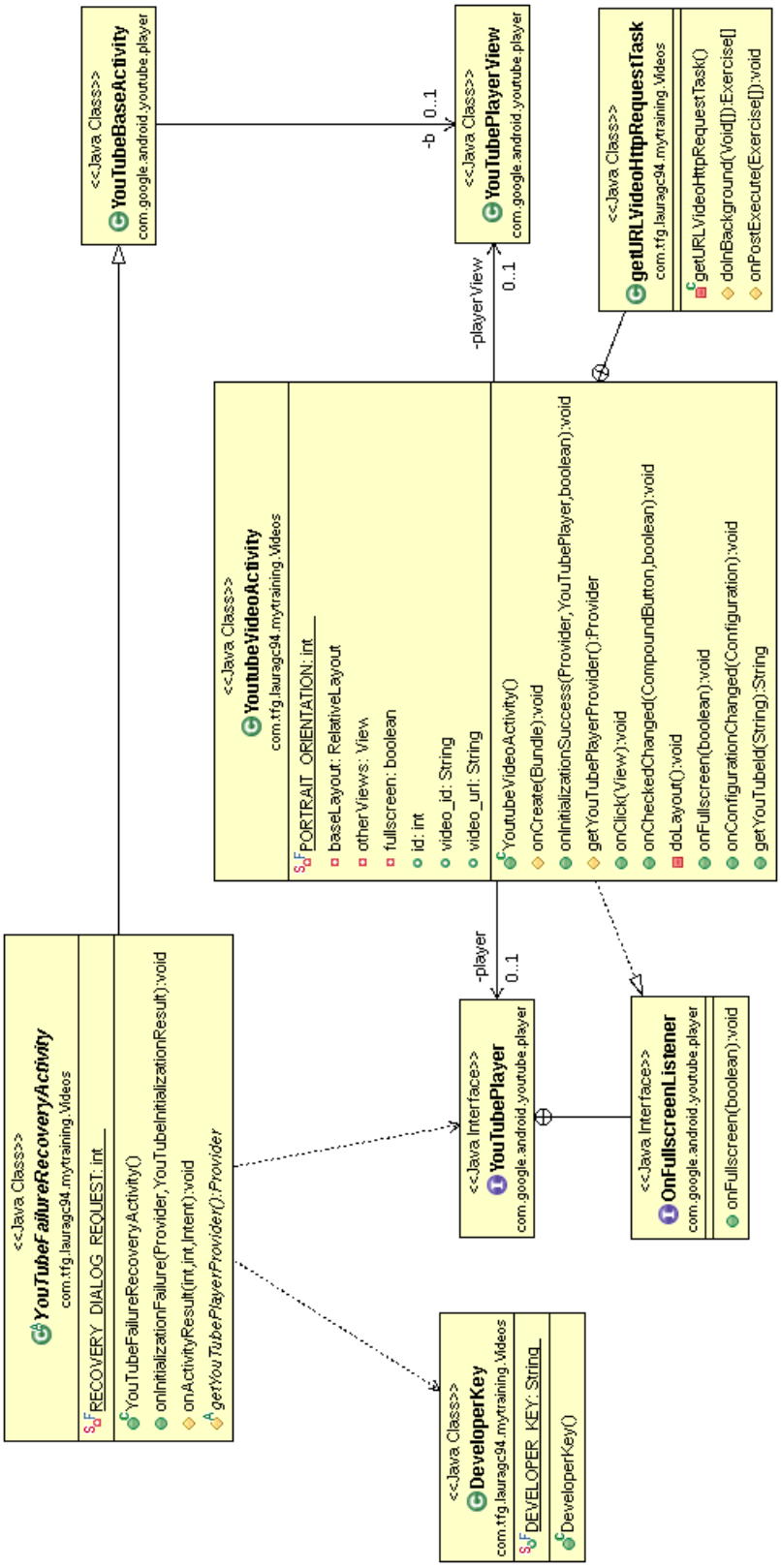


Ilustración 24: Diagrama de clases referente a la carga de Vídeos.

6.3 Diagramas de actividad

Para explicar al lector el funcionamiento de MyTraining se ha dedicado esta sección del capítulo a mostrar los diagramas de actividad más relevantes con el fin de explicar el flujo de trabajo de la aplicación. Hay que aclarar que un diagrama de actividad es otro tipo de diagrama UML que se caracteriza por presentar el flujo de un software a través de ciertas acciones realizadas sobre este, es decir, para nuestro caso, permite ver el resultado de las elecciones del usuario sobre la aplicación desarrollada.

6.3.1 Diagrama de actividad para el inicio de sesión

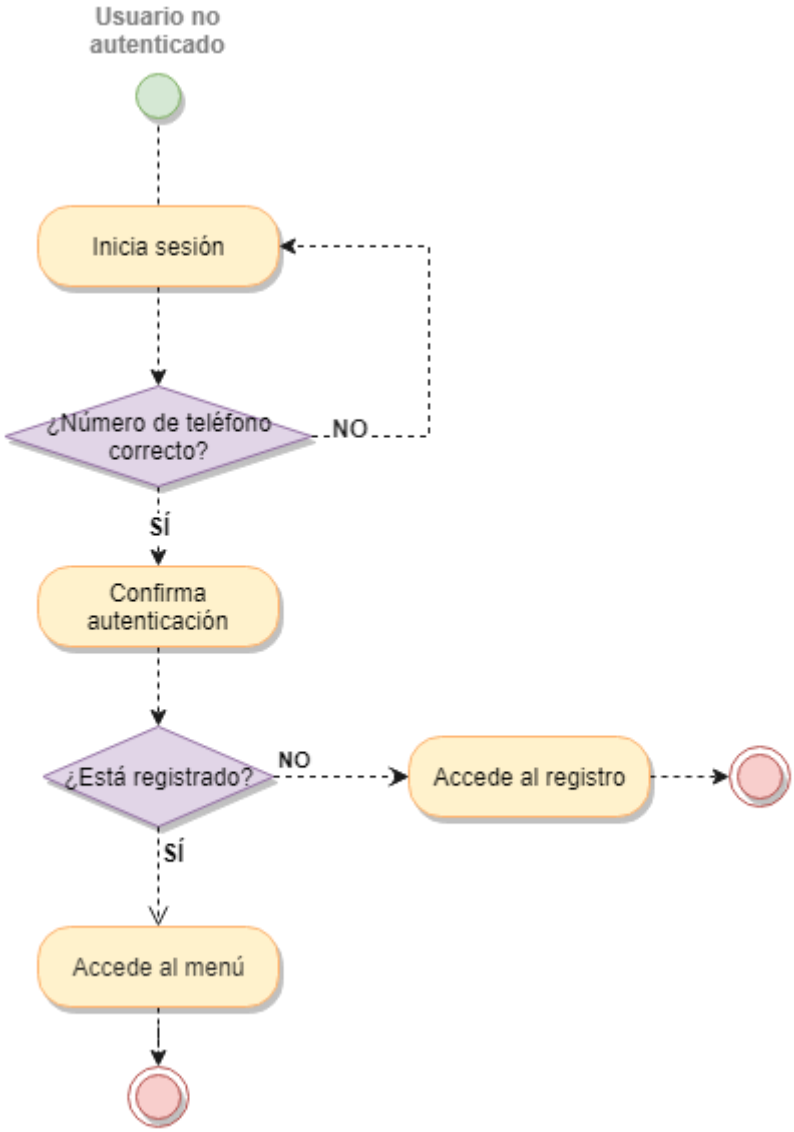


Ilustración 25: Diagrama de actividad para el inicio de sesión de un usuario.

6.3.2 Diagrama de actividad para el registro de un usuario

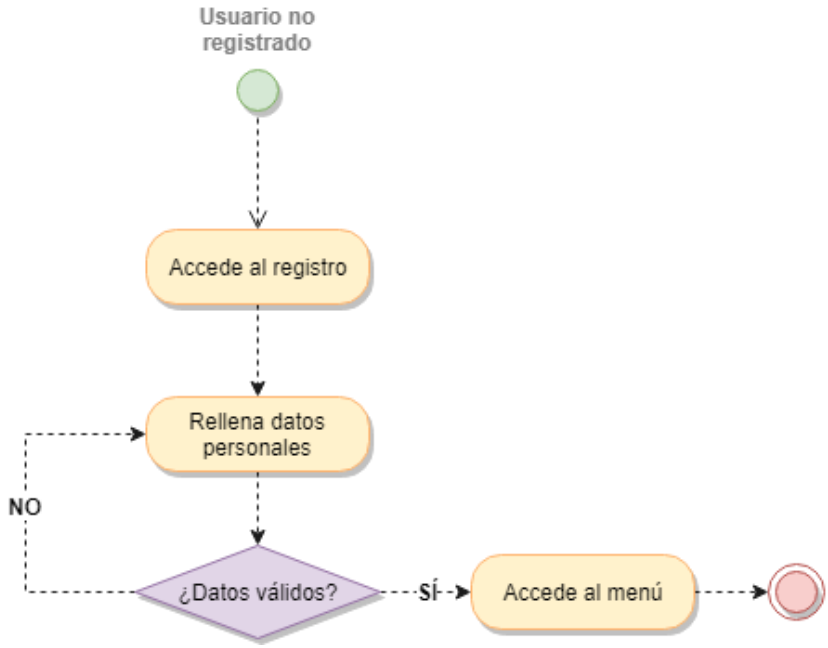


Ilustración 26: Diagrama de actividad para el registro de un usuario.

6.3.3 Diagrama de actividad para la configuración de un usuario

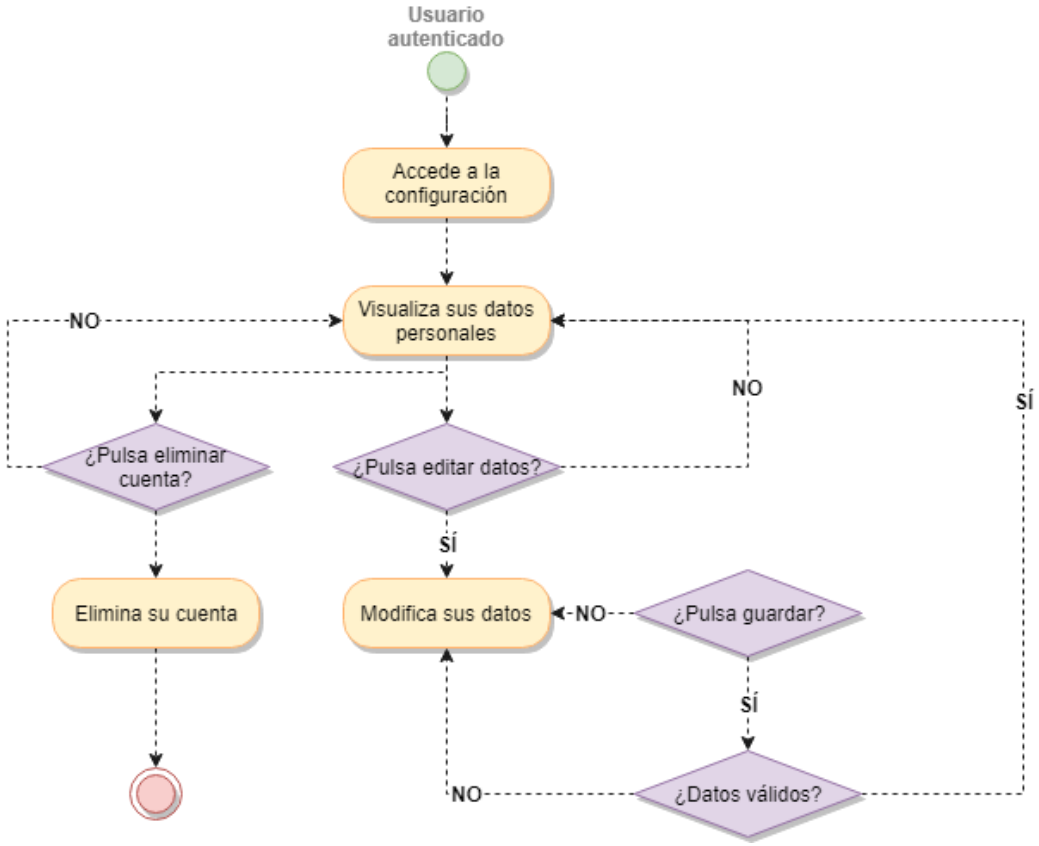


Ilustración 27: Diagrama de actividad para la configuración de un usuario.

6.3.4 Diagrama de actividad para la obtención de las rutinas de un usuario

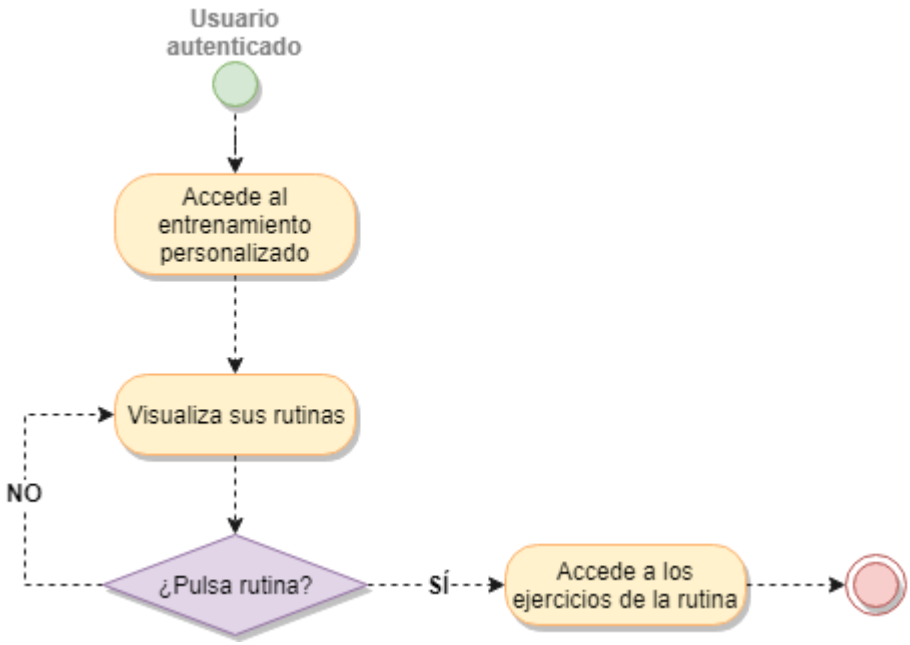


Ilustración 28: Diagrama de actividad para la obtención de rutinas de un usuario.

6.3.5 Diagrama de actividad para la obtención de los ejercicios de una rutina

Ilustración 29: Diagrama para la obtención de los ejercicios de una rutina.

6.3.6 Diagrama de actividad para la obtención de ejercicios según objetivo

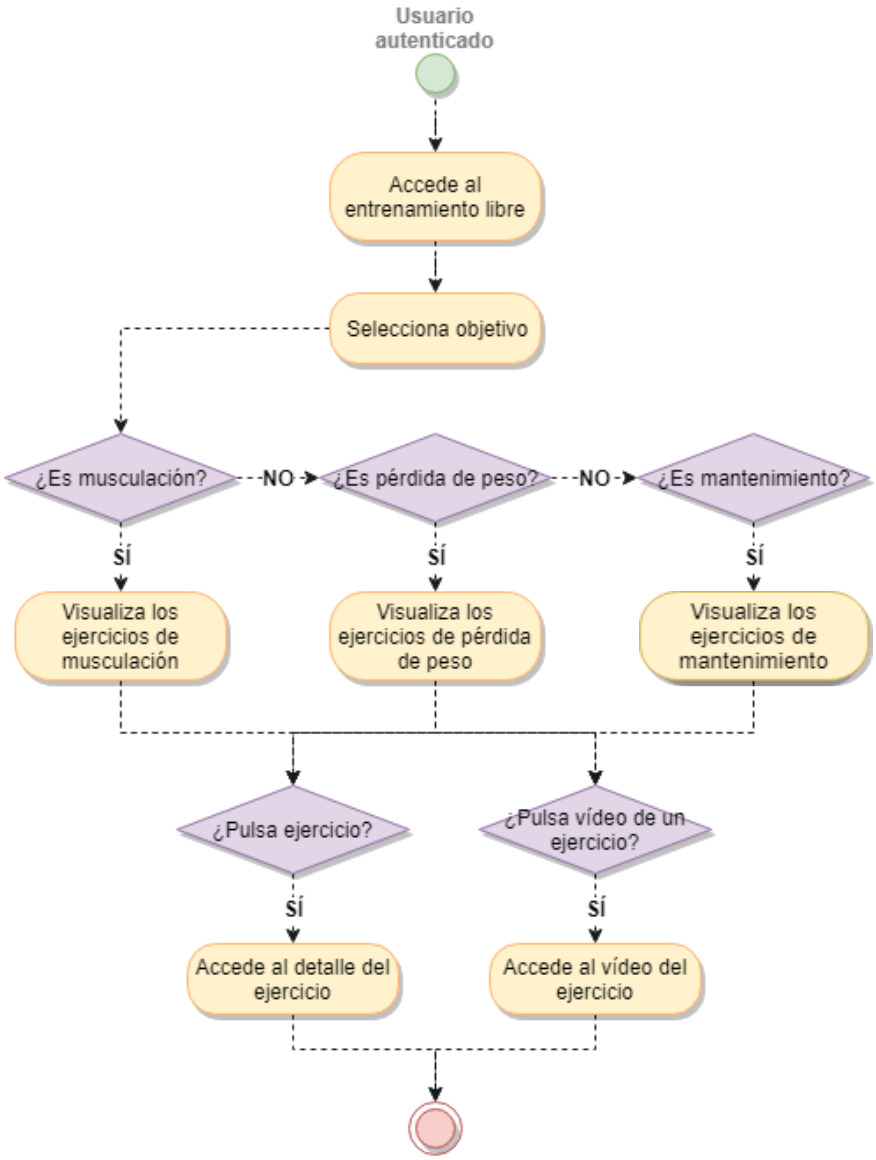


Ilustración 30: Diagrama de actividad para la obtención de ejercicios según objetivo.

6.3.7 Diagrama de actividad para la obtención del detalle de un ejercicio

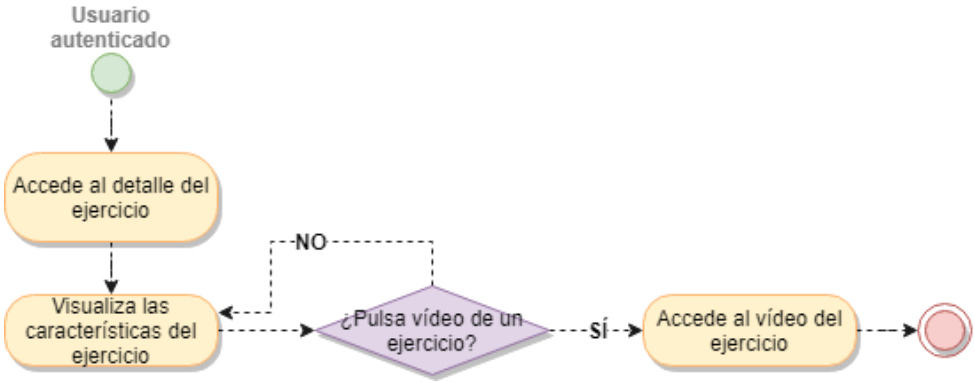


Ilustración 31: Diagrama de actividad para la obtención del detalle de un ejercicio.

6.3.8 Diagrama de actividad para la obtención del vídeo de un ejercicio

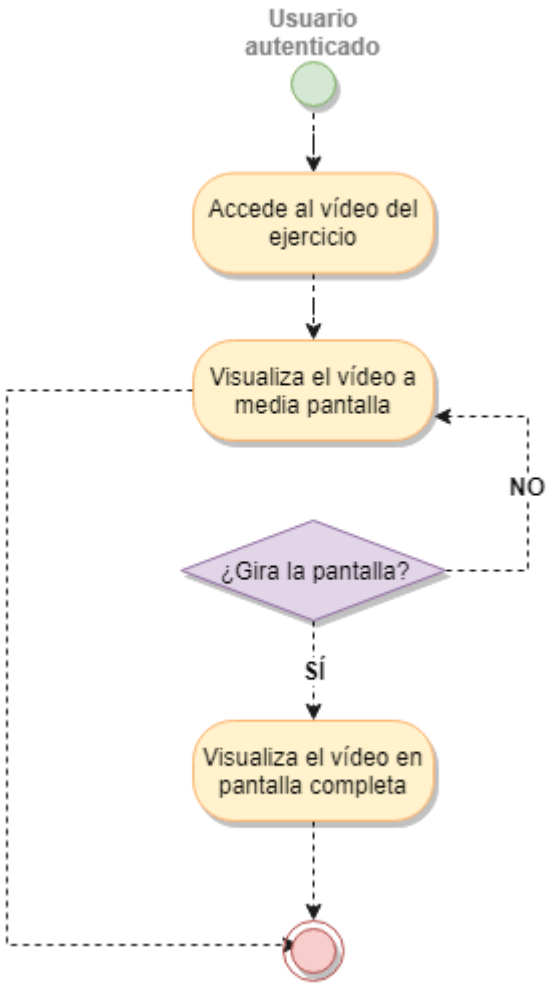


Ilustración 32: Diagrama para la obtención del vídeo del ejercicio.

6.4 Servicios externos y APIs implementadas

Tras comprender el funcionamiento general y el flujo de trabajo de la aplicación gracias a los diagramas UML expuestos en los apartados anteriores, se va a proceder a mencionar los servicios externos y las APIs que han sido usadas para completar la aplicación de MyTraining y otorgarle una serie de características que la hacen destacar frente al resto de aplicaciones móviles.

6.4.1 Inicio de sesión a través de Account Kit

Resulta necesario explicar el servicio implementado que gestiona el sistema de autenticación del usuario ya que es una de las novedades implantadas en este proyecto con respecto a sus antecesores.

Este servicio conocido como "Account Kit" es el ofrecido por Facebook a desarrolladores de aplicaciones Android cuyo objetivo es el de facilitar y simplificar el inicio de sesión/registro a sus clientes. Para poder hacer uso de este servicio, basta con que el programador posea una cuenta de Desarrollador Facebook y registre su aplicación para obtener el identificador que le dará acceso a la funcionalidad ofrecida.

Como se puede ver en la imagen adjunta Facebook permite a los desarrolladores hacer uso de sus servicios proporcionándoles el token para gestionar el inicio de sesión de los usuarios de la aplicación de tal manera que los desarrolladores pueden utilizarlo en sus servidores para gestionar como deseen el acceso a su aplicación.

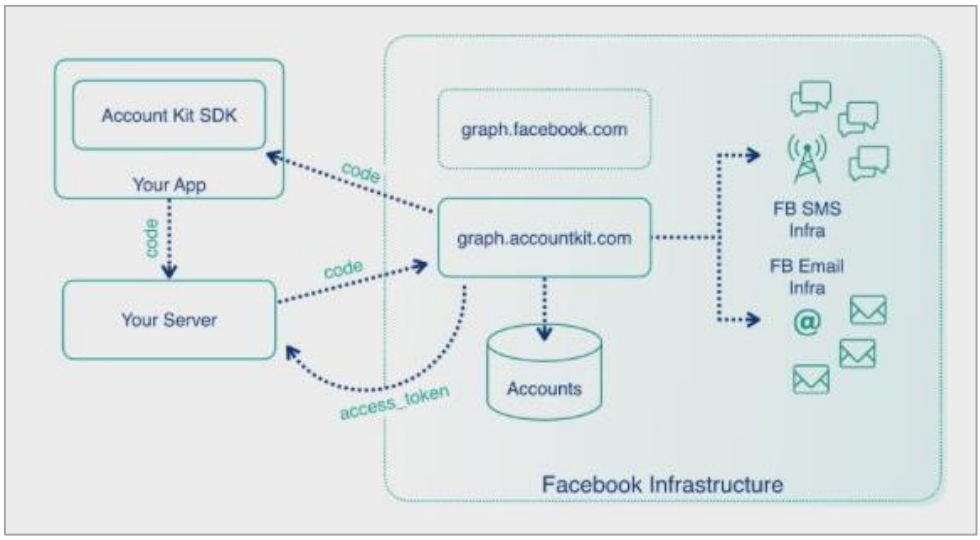


Ilustración 33: Funcionamiento Account Kit.

No se entrará en más detalle en lo referente a la utilización de este servicio por parte del desarrollador ya que se ha dedicado un anexo completo en este documento para cubrir ese aspecto. Por lo que, a continuación, se van a presentar las ventajas que dicha implementación aportará a una aplicación como la desarrollada aquí.

- Ofrece a los usuarios un registro e inicio de sesión rápido y sencillo utilizando únicamente su número de teléfono o dirección de correo electrónico, sin necesidad de contraseñas.
  - Si se elige inicio de sesión mediante SMS, se presentan ventajas como la verificación instantánea, siendo totalmente automático la introducción del código de verificación, evitando así al usuario tener que salir de la pantalla para ver el código enviado. Además, en el caso de producirse un error en la entrega del SMS, se podrá elegir entre recibir el código mediante una llamada telefónica o una notificación de Facebook (esta opción solo está disponible si han asociado su número de teléfono con una cuenta de Facebook).
  - En caso de seleccionar la opción del correo electrónico, el sistema permite al usuario elegir en cuál de todos los correos que tiene abierto en el dispositivo quiere recibir el código de verificación, redireccionándole a él automáticamente y tras confirmar la identidad del usuario, le devuelve la siguiente pantalla de la aplicación.
- Mantiene asegurado el acceso a la aplicación, ya que el usuario no reutilizará contraseñas o recurrirá a contraseñas fáciles de adivinar.
- Evita agotar a los usuarios con la necesidad de crear y recordar contraseñas únicas.
- Es fiable y fácil de usar tanto para los usuarios de la aplicación como para los desarrolladores que lo implementan, evitando a estos últimos llevar a cabo tediosas tareas de autenticación.
- Permite al desarrollador personalizar la interfaz de usuario para los dos tipos de procedimientos.

6.4.2 YouTube API

Para la reproducción de los vídeos de los ejercicios se ha decidido usar la API de YouTube para Android, la cual te permite incorporar la funcionalidad de reproducción de vídeo en las aplicaciones de Android. La API define métodos para cargar y reproducir videos y listas de reproducción de YouTube y, además, permite personalizar y controlar dicha reproducción.

Al implementar la API en una aplicación móvil se podrá cargar, insertar y controlar la reproducción de vídeos programáticamente únicamente usando el identificador del vídeo de YouTube. También se podrá registrar objetos de escucha de eventos para obtener devoluciones de llamada para ciertos eventos. Por último, la API tiene la función auxiliar de admitir los cambios de orientación, así como las transiciones y la reproducción en pantalla completa.

Como en el caso del apartado anterior sobre el servicio Account Kit usado, no se va a entrar en detalle en lo referente a la utilización de esta API por parte del desarrollador ya que se ha dedicado un anexo completo en este documento para cubrir ese aspecto.

6.4.3 Google Places API

Una de las opciones que se ofrecen al usuario dentro del menú de la aplicación es la de acceder al mapa para ver su ubicación y los establecimientos y zonas cercanos. Esto se ha podido hacer gracias al uso de la API de Google conocida como Google Places, que permite al desarrollador construir un mapa dinámico de la zona y localizar los locales próximos a través del servicio de localización de Android, además presenta un buscador que permite al usuario buscar sitios registrados de Google. Entre los elementos que se pueden implementar están:

- *Place Picker*. Permite a los usuarios seleccionar un lugar en un mapa interactivo.
- *Current Place*. Devuelve una lista de lugares donde se sabe por última vez que el dispositivo del usuario está ubicado junto con una indicación de la probabilidad relativa de cada lugar.
- *Autocomplete*. Completa automáticamente el nombre o la dirección de un lugar a medida que escriben los usuarios.
- *Place Photos.* devuelve imágenes de alta calidad de un lugar.
- *Place ID & Details*. Muestran información más detallada sobre un lugar.

El objetivo de utilizar este servicio en la aplicación ha sido ofrecer al usuario la posibilidad de buscar gimnasios o zonas al aire libre donde llevar a cabo su entrenamiento y así darle variedad de lugares dónde desarrollarlo rompiendo con la monotonía de hacerlo siempre en el mismo sitio.

6.5 Interfaz de usuario

En esta sección se va a profundizar en detalle en cada una de las pantallas que conforman la aplicación y en los elementos implementados en cada una de ellas. Todo esto estará acompañado de diagramas de actividad que expliquen el flujo de funcionamiento y de capturas de pantalla que muestren la interfaz de usuario diseñada.

6.5.1 Pantallas de inicio

El inicio de la aplicación comprende dos pantallas a su vez. La primera es un Splash Screen mostrado nada más lanzar la aplicación y consistente en una pantalla completa que muestra, durante unos segundos, el logo y el nombre de la aplicación, para luego dar paso a la segunda pantalla que comprende nuevamente el nombre de la aplicación y un botón que permitirá al usuario iniciar su autenticación a través de su número de teléfono, funcionamiento que será explicado detalladamente en el próximo subapartado.

Ilustración 35: Pantalla de inicio del Splash Screen.

Ilustración 34: Pantalla de inicio del Inicio de sesión.

6.5.2 Pantallas de Autenticación

Aquí se mostrarán y explicarán las cuatro pantallas que conforman la autenticación del usuario al iniciar sesión desde la pantalla de inicio visualizada en el apartado anterior. El esqueleto de estas interfaces es aportado por el servicio Account Kit ya explicado en el apartado Inicio de sesión a través de Account Kit de este capítulo, pero han sido personalizadas para concordar con la estética del resto de la aplicación.

A continuación, se adjuntan las capturas que muestran la interfaz de usuario personalizada para la aplicación desarrollada y en el orden temporal de funcionamiento.

Ilustración 37: Pantalla AccountKit del envío del SMS con éxito.

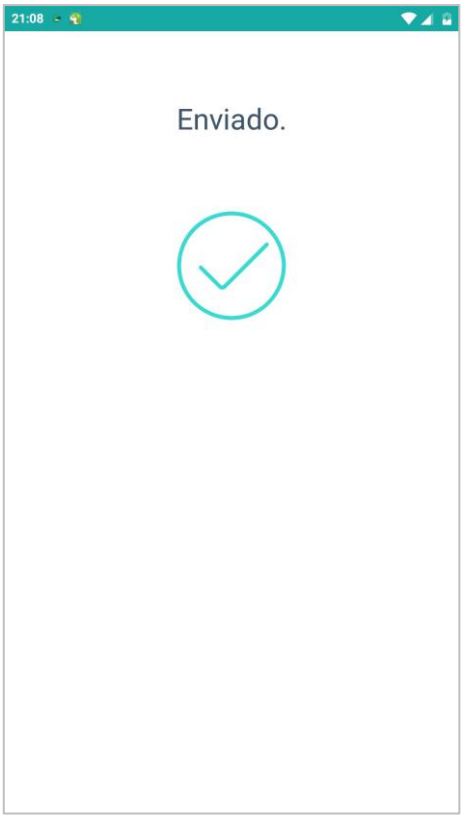


Ilustración 36: Pantalla Account Kit del registro del número de teléfono.

Ilustración 39: Pantalla AccountKit de la recepción del código de verificación.

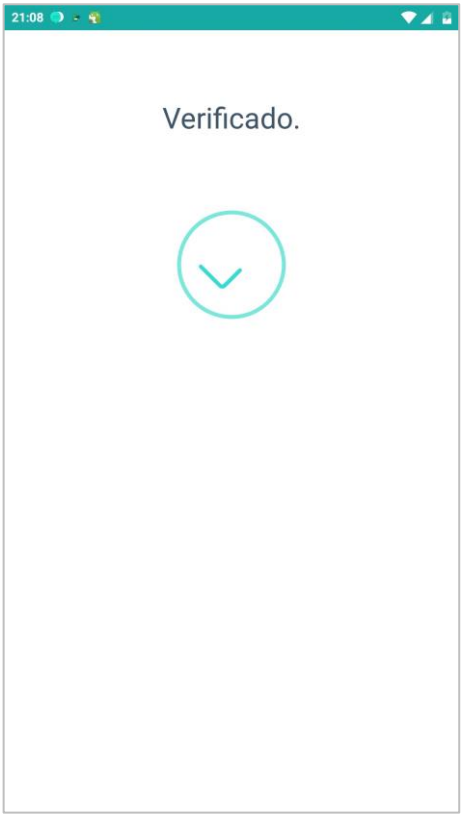


Ilustración 38: Pantalla de AccountKit de la comprobación del código.

6.5.3 Pantallas de Registro

La pantalla de registro únicamente será mostrada al usuario si tras iniciar sesión, el sistema detecta que el número de teléfono usado para dicha autenticación no se encuentra registrado en la base de datos, por tanto, será considerado como un nuevo usuario y deberá registrarse para almacenar sus datos y poder así acceder a sus rutinas personalizadas en próximas pantallas.

Como se puede apreciar en la imagen adjunta, el usuario deberá rellenar los campos "Nombre y Apellidos", "Teléfono" y "Correo electrónico" para poder registrarse con éxito, siempre y cuando estos valores sean válidos, aspecto que la propia aplicación detectará y avisará al usuario sobre ello al mismo tiempo que introduce texto en dichos campos. Además, el botón central que tiene el dibujo de una goma de borrar, como se puede intuir, permite al usuario vaciar todos los campos y comenzar de cero con el registro de sus datos.

Finalmente, quedan por explicar dos funcionalidades más correspondientes a los botones

"Cancelar" y "Aceptar" que, tal y como sus nombres indican, permitirán al usuario cancelar el registro volviendo, por tanto, a la pantalla de inicio o aceptarlo y guardar sus datos en la base de datos de MyTraining y acceder inmediatamente al menú principal de la aplicación. Además, en el caso de registrarse exitosamente la aplicación mostrará una pantalla dando la bienvenida al nuevo usuario.

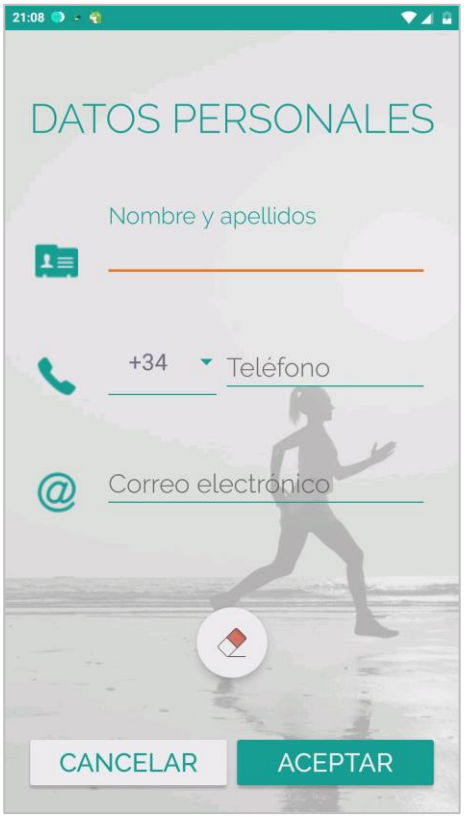


Ilustración 41: Pantalla de registro de un usuario.

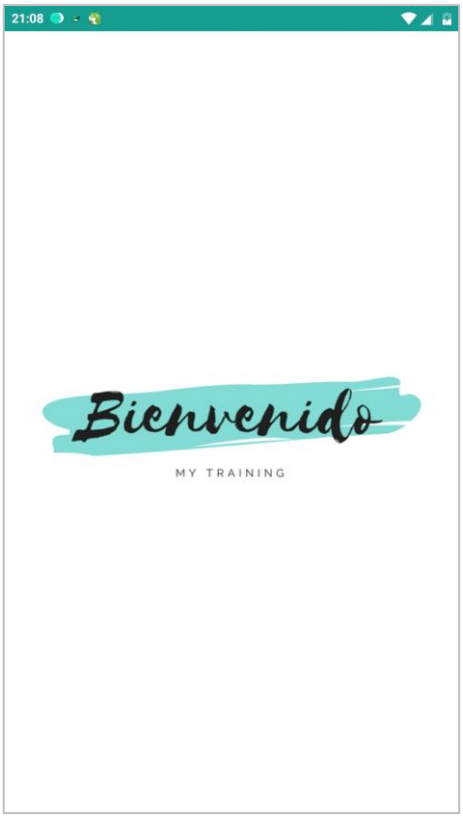


Ilustración 40: Pantalla de bienvenida para un nuevo usuario.

6.5.4 Pantallas de Menú

Una vez el usuario registrado haya iniciado sesión podrá acceder al menú principal y a todas sus funcionalidades.

Lo primero que verá será los dos tipos de entrenamiento pudiendo elegir cualquiera de ellos y acceder a sus características. Además, en la *toolbar* se verá a la derecha un icono de información que si se pulsa mostrará una ventana emergente con una explicación de en qué consiste MyTraining, y a la izquierda un icono para desplegar el menú lateral que, a su vez, permitirá al usuario elegir entre las opciones de "Mapa", "Configuración" y "Cerrar Sesión". Todo esto se puede apreciar en las capturas mostradas a continuación.

Ilustración 44: Pantalla del menú principal.

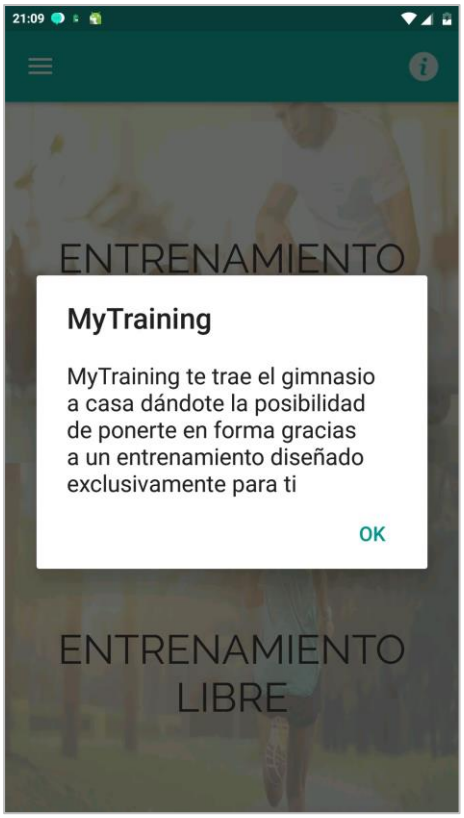


Ilustración 43: Pantalla de la ventana emergente informativa.

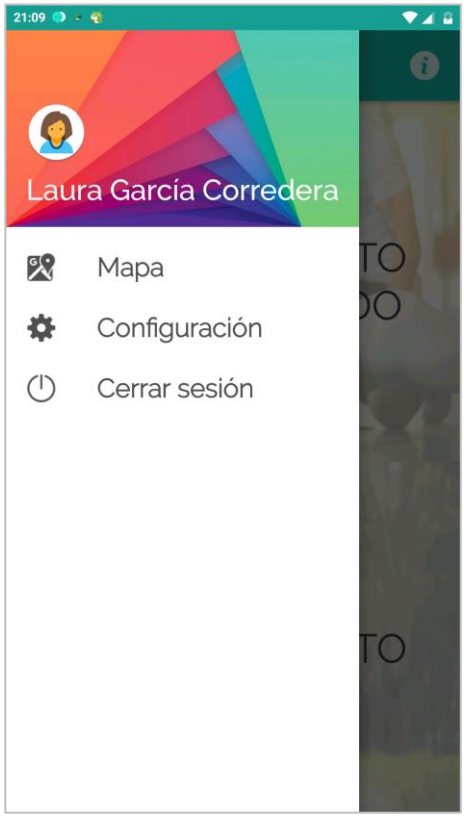


Ilustración 42: Pantalla del menú lateral desplegable.

6.5.5 Pantalla del Mapa

Si el usuario elige la opción del mapa dada por el menú lateral desplegable mostrado en la *Ilustración 42* podrá acceder a toda la funcionalidad ofrecida por la API de Google Places, anteriormente explicada en el punto Google Places API del presente capítulo, y que presenta la interfaz mostrada a continuación.

Ilustración 45: Pantalla del mapa.

6.5.6 Pantallas Configuración

Si el usuario elige la opción de la configuración dada por el menú lateral desplegable mostrado en la *Ilustración 42* podrá acceder a sus datos personales y modificarlos, siempre y cuando éstos sean válidos, en el caso de no serlos la aplicación lo detectará y mostrará un mensaje advirtiéndolo e impidiendo guardar (mismo funcionamiento que en el registro del usuario). Por último, el usuario podrá eliminar su cuenta definitivamente mediante el enlace inferior que mostrará una pantalla emergente para que el usuario confirme que realmente desea borrar sus datos de la base de datos MyTraining perdiendo consecuentemente toda la información (rutina, ejercicios y configuración) asociada a su cuenta. Todas estas funcionalidades se pueden apreciar en las capturas siguientes.

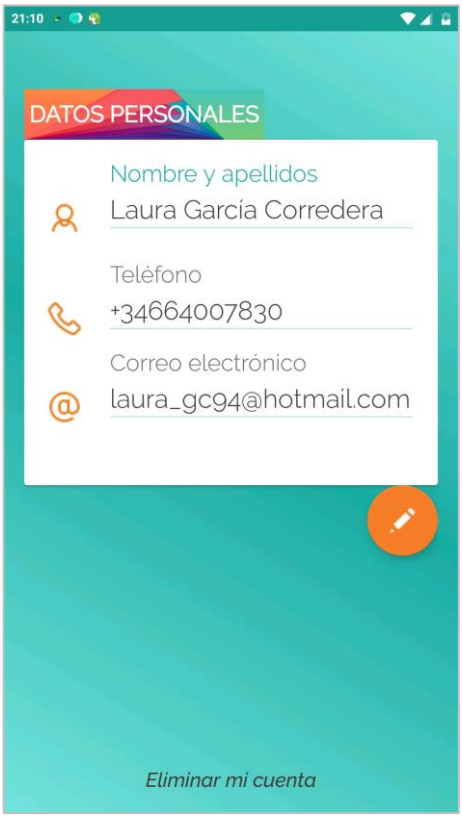


Ilustración 46: Pantalla de configuración: visualización de datos.

Ilustración 47: Pantalla de configuración: modificación de datos.

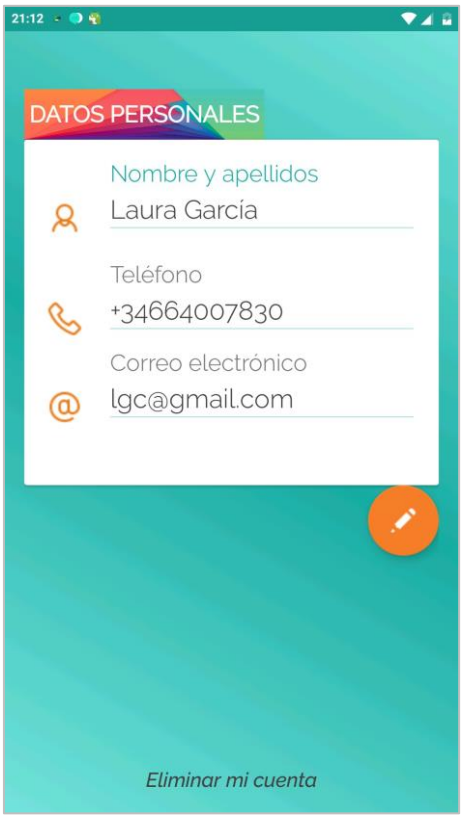


Ilustración 49: Pantalla de configuración: nuevos datos guardados.

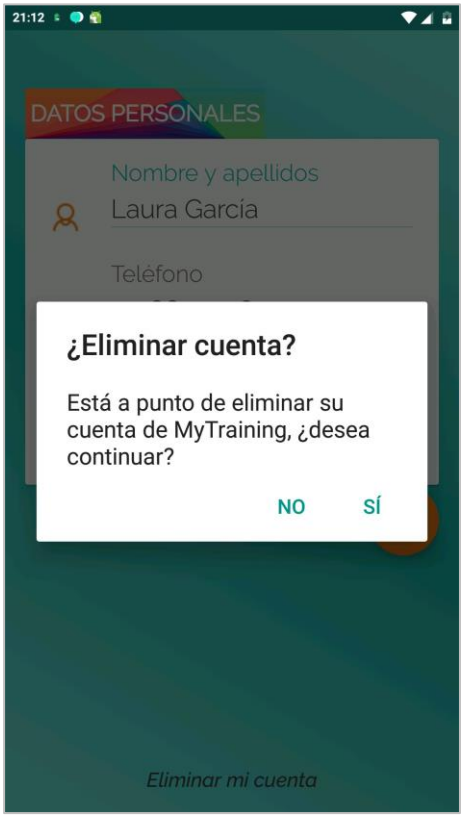


Ilustración 48: Pantalla de configuración: eliminación de cuenta.

6.5.7 Pantallas de Entrenamiento libre

Si el usuario elige la opción de entrenamiento libre dada por el menú principal mostrado en la *Ilustración 44* podrá acceder a todos los ejercicios registrados en la base de datos de la aplicación estando éstos catalogados por objetivo a cumplir.

Además, en la vista de estos ejercicios se podrá a su vez acceder al detalle del ejercicio y al vídeo pulsando sobre el nombre de éste o sobre el icono de *play* respectivamente. Estas pantallas serán mostradas más adelante.

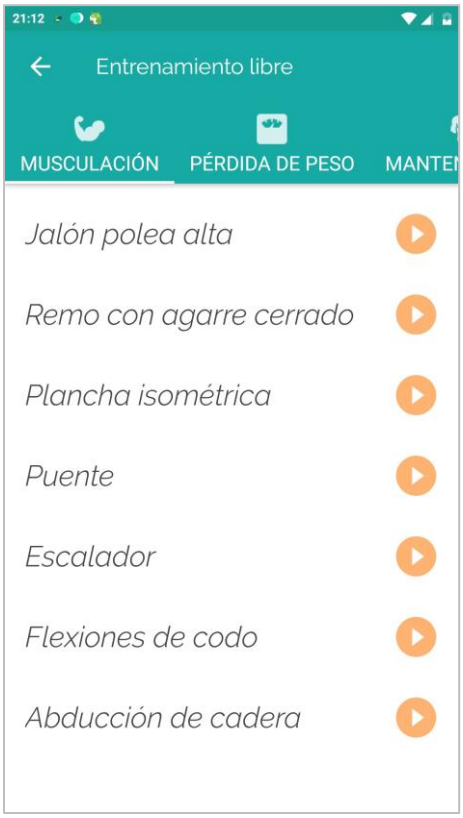


Ilustración 50: Pantalla entrenamiento libre con objetivo "Musculación".

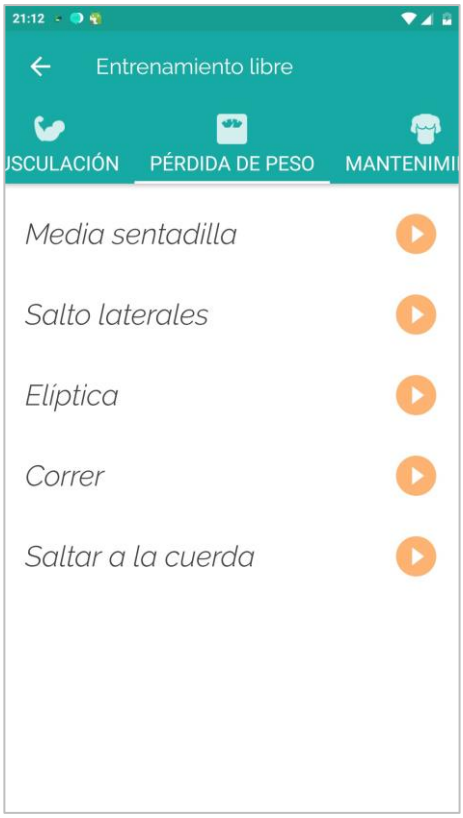


Ilustración 51: Pantalla entrenamiento libre con objetivo "Pérdida de peso".

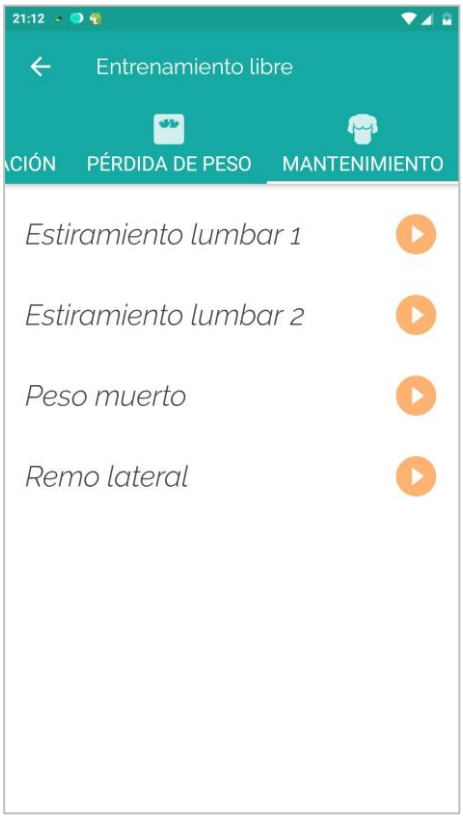


Ilustración 52: Pantalla entrenamiento libre con objetivo "Mantenimiento".

6.5.8 Pantallas de Entrenamiento personalizado

Si el usuario elige la opción de entrenamiento personalizado dada por el menú principal mostrado en la *Ilustración 44* podrá acceder a todas las rutinas personalizadas que el entrenador le ha asignado según sus características físicas y objetivos deseados.

Como se puede ver en las imágenes adjuntas, el usuario visualizará unas tarjetas que contienen el nombre y descripción de la rutina, pudiendo acceder a los ejercicios que la componen simplemente pulsando la flecha de la izquierda. Esto llevará a otra pantalla que mostrará un listado de ejercicios con el mismo diseño que la pantalla de entrenamiento libre y con la misma funcionalidad.

Ilustración 53: Pantalla de rutinas personalizadas.

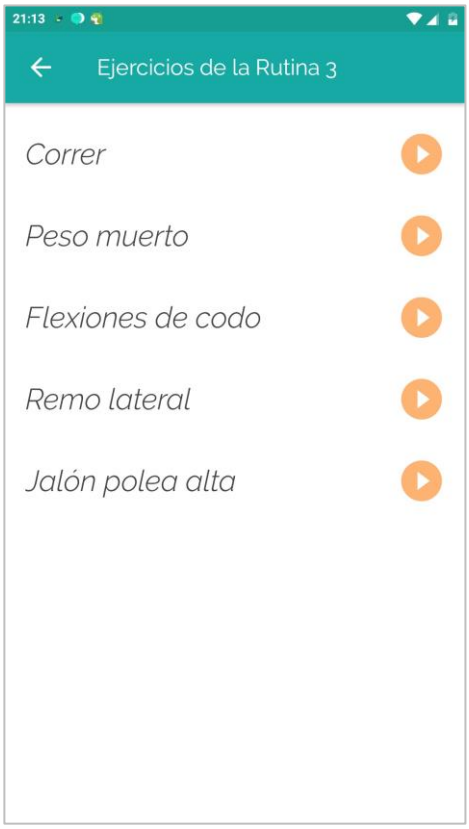


Ilustración 54: Pantalla de ejercicios de la rutina 3 seleccionada.

6.5.9 Pantallas de detalle de un ejercicio

Si el usuario elige un ejercicio cualquiera, ya sea desde la pantalla de entrenamiento libre o personalizado, podrá acceder al detalle del ejercicio seleccionado donde se mostrará al usuario, el nombre, una descripción de cómo realizar el ejercicio, el objetivo y nivel de dificultad que tiene y el número de series y repeticiones a realizar. Además, el entrenador tendrá la posibilidad de añadir en la base de datos la URL de la imagen que desee y que será mostrada en la división superior de la pantalla junto con el nombre de la aplicación.

Por último, se permitirá al usuario acceder al vídeo del ejercicio simplemente pulsando el icono flotante de la esquina inferior izquierda al igual que en el caso de pulsar el icono del *play* en el listado de ejercicios.

Ilustración 56: Pantalla del detalle del ejercicio 1.

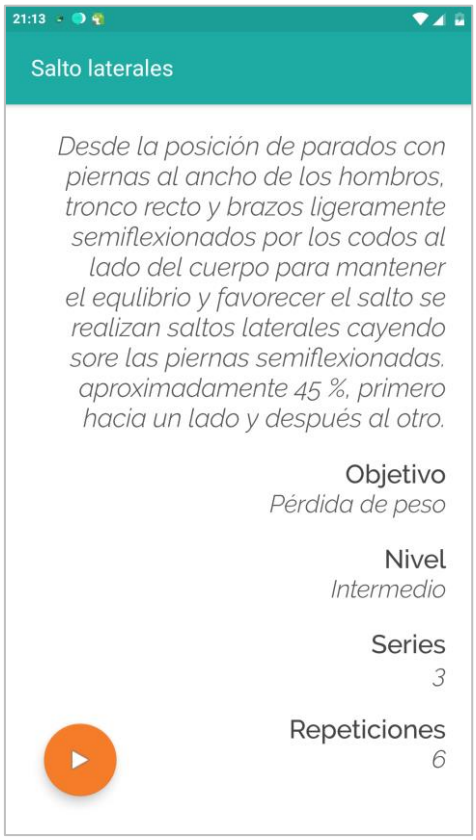


Ilustración 55: Pantalla del detalle del ejercicio 2.

6.5.10 Pantallas de detalle de un ejercicio

Si el usuario pulsa, en cualquier pantalla donde se encuentre, el icono del *play*, podrá acceder al vídeo del ejercicio seleccionado donde el usuario podrá visualizar el vídeo de YouTube desde la propia aplicación pudiendo hacer uso de las funcionalidades explicadas en la sección YouTube API.

Por último, se permitirá al usuario acceder al vídeo del ejercicio simplemente pulsando el icono flotante de la esquina inferior izquierda al igual que en el caso de pulsar el icono del *play* en el listado de ejercicios.

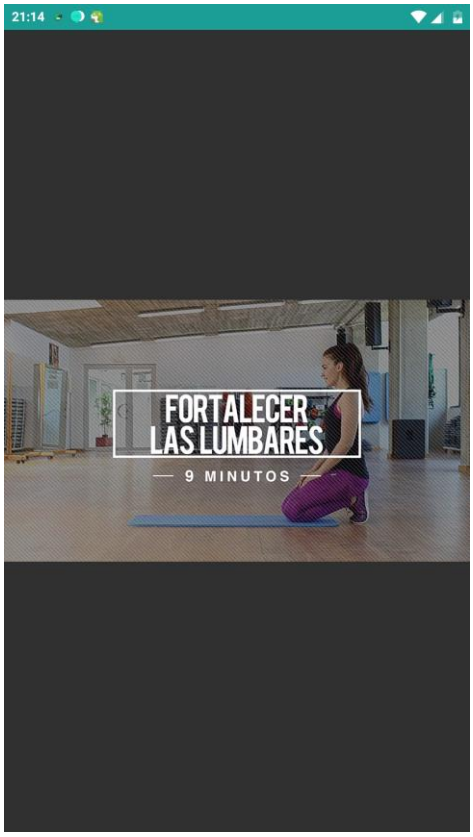


Ilustración 57: Pantalla del vídeo vertical.

Ilustración 58: Pantalla del vídeo horizontal.

7 BASE DE DATOS

En este capítulo se mostrará un esquema de la estructura de la base de datos creada para almacenar la información necesaria para el correcto funcionamiento del servicio ofrecido. Además, se explicará cada una de las tablas que conforman dicha base de datos junto con las relaciones que existen entre ellas parándose en cada campo para describir su finalidad y sus características.

7.1 Esquema de la base de datos

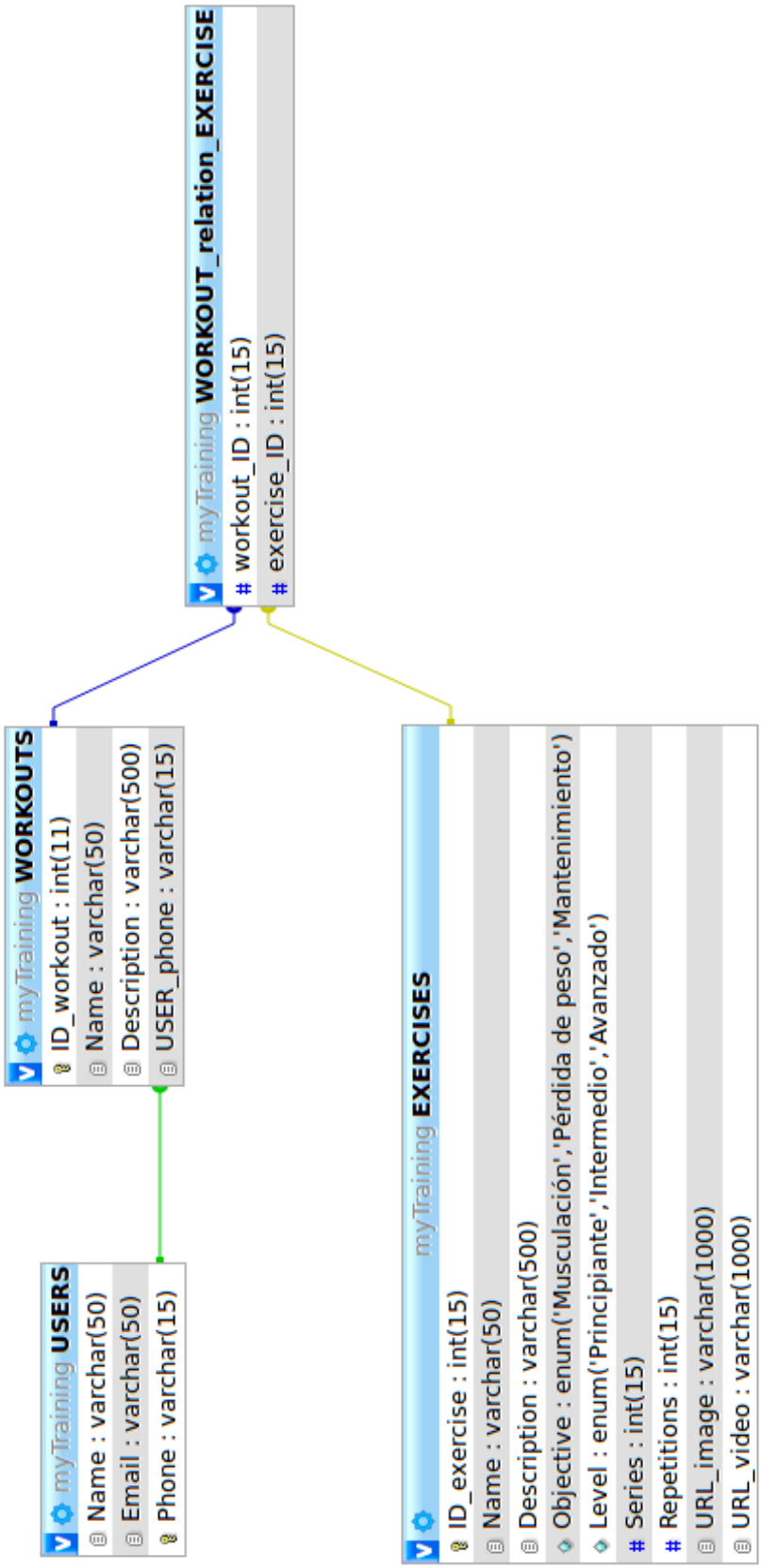


Ilustración 59: Esquema de la base de datos "MyTraining".

7.2 Tablas de la base de datos

En este apartado se van a mostrar las tablas expuestas en el esquema de la sección anterior, las cuáles conforman la base de datos creada para almacenar la información que el servicio manipula durante su funcionamiento. Además, en cada tabla se detallarán cada uno de los campos que la componen y destacando cuál es la clave primaria.

7.2.1 Tabla USERS

USERS	
Esta tabla almacena los datos personales de los usuarios registrados en la aplicación MyTraining.	
Name	*Nombre del usuario registrado en la aplicación, no puede tener valor nulo y puede ser editado por el usuario en la pantalla de configuración.*
Email	*Correo electrónico del usuario registrado en la aplicación, no puede tener valor nulo y puede ser modificado por el usuario en la pantalla de configuración.*
🔑 **Phone**	*Número de teléfono del usuario registrado. Éste no puede tener valor nulo y actúa como identificador del usuario, es decir es la clave primaria de esta tabla.*

Tabla 11: Tabla USERS de la base de datos.

7.2.2 Tabla WORKOUTS

WORKOUTS	
Esta tabla almacena las rutinas que el entrenador asigna a cada usuario.	
ID_workout	*Identificador de la rutina. Es un número auto incremental no nulo y único que actúa como clave primaria de la tabla.*
Name	*Nombre de la rutina creada por el entrenador que no puede tener valor nulo.*
Description	*Descripción de la rutina correspondiente con el fin de que el usuario pueda tener una idea de en qué consiste y con qué fin va a realizar los ejercicios que conformen dicha rutina.*
USER_phone	*Número de teléfono del usuario registrado. Éste no puede tener valor nulo y actúa como identificador del usuario, es decir una clave foránea que hace referencia a la clave primara "Phone" de la tabla de USERS.*

Tabla 12: Tabla WORKOUTS de la base de datos.

7.2.3 Tabla WORKOUT_relation_EXERCISE

WORKOUT_relation_EXERCISE	
Esta tabla almacena los IDs que relacionan una rutina con un ejercicio.	
Workout_ID	*Identificador de la rutina. Es un número no nulo y único siendo clave foránea, es decir, hace referencia al ID_workout de la tabla WORKOUTS.*
Exercise_ID	*Identificador del ejercicio. Es un número no nulo y único siendo clave foránea, es decir, hace referencia al ID_exercise de la tabla EXERCISES.*

Tabla 13: Tabla WORKOUT_relation_EXERCISE de la base de datos.

7.2.4 Tabla EXERCISES

EXERCISES	
Esta tabla almacena los ejercicios creados por el entrenador y todas sus características.	
ID_exercise	*Identificador del ejercicio. Es un número auto incremental no nulo y único que actúa como clave primaria de la tabla.*
Name	*Nombre del ejercicio creado por el entrenador que no puede tener valor nulo.*
Description	*Descripción del ejercicio correspondiente con el fin de que el usuario pueda tener una idea de en qué consiste.*
Objective	*Objetivo a conseguir con la realización de dicho ejercicio.*
Level	*Nivel de dificultad que tiene la realización del ejercicio.*
Series	*Número de series a realizar por el usuario.*
Repetitions	*Número de repeticiones que el usuario deberá de hacer en cada serie que realice.*
URL_image	*URL de la imagen del músculo o zona afectada durante el desarrollo del ejercicio.*
URL_video	*URL del vídeo que muestra cómo realizar correctamente el ejercicio para que el usuario evite posibles lesiones por mala práctica.*

Tabla 14: Tabla EXERCISES de la base de datos.

8 LÍNEAS FUTURAS

En este penúltimo capítulo se describirán algunas líneas de mejora que podrían seguirse en un futuro a la hora de mejorar el proyecto desarrollado y añadirle más funcionalidades con el fin de enriquecer la aplicación creada y aumentar sus posibilidades de éxito en el mercado actual.

En primer lugar, se hablará de algunas mejoras respecto a la funcionalidad ofrecida por la aplicación de manera que ésta sea ampliada para satisfacer aún más las necesidades del consumidor del servicio. A continuación, se destacan las siguientes:

- En la opción de entrenamiento libre se podría permitir al usuario seleccionar los ejercicios que desee de las listas mostradas para que pudiese crear así su propia rutina personalizada a su gusto y necesidad. Además, ésta sería añadida dentro del listado de las rutinas personalizadas del usuario para que éste pudiese consultarla y acceder a los ejercicios que la comprenden fácilmente.

- Se podría añadir más funcionalidades al menú lateral desplegable como, por ejemplo, ver las estadísticas del usuario para que así pueda seguir su progreso o un calendario donde pueda registrar y consultar la actividad realizada cada día.

- Como última mejora, sería una buena opción implementar un chat con el entrenador personal para que el usuario pudiese hacerle consultas y recibir ayuda personalizada de manera rápida y eficaz.

Dejando a un lado las posibles mejoras referentes al cliente consumidor de la aplicación móvil, se propone como una línea de desarrollo paralela a este proyecto, crear una aplicación específica para uso por parte del entrenador y que así, éste pueda registrar las rutinas y ejercicios a través de su teléfono, así como chatear (en el caso de implementarse el tercer punto anterior) con un cliente desde cualquier sitio y en cualquier momento.

Todas estas posibles mejoras harían a la actual aplicación una gran competidora frente a otras en el mercado de las aplicaciones móviles de deporte y entrenamiento.

9 CONCLUSIONES

Volviendo la vista hacia el principio del presente documento, resulta primordial para concluirlo, recordar en qué ha consistido a grandes rasgos.

MyTraining es una aplicación móvil desarrollada en Android que hace uso de un servicio web Spring creado para ofrecer a los clientes de esta aplicación ciertas funcionalidades como son un inicio de sesión rápido y cómodo usando únicamente su número de teléfono, un listado de ejercicios catalogados según objetivo a cumplir, pudiendo acceder a vídeos explicativos de los mismos, y una lista de rutinas personalizadas para cada usuario, recetadas por un profesional que ha tenido en cuenta las necesidades y características físicas de cada uno a la hora de prescribir los ejercicios pertinentes.

Teniendo presente lo anterior, se pueden sacar algunas conclusiones finales sobre la trayectoria seguida, las dificultades encontradas a lo largo del desarrollo del servicio y, sobre todo, el grado de satisfacción en cuanto al resultado final de este proyecto.

Se ha desarrollado una aplicación fácil de usar y comprender, destinada a cualquier usuario que quiera hacer uso de ella, y con cierta proyección hacia un futuro comercial de la misma. Su atractivo diseño y fácil manejo hacen de esta aplicación una plataforma llamativa y realmente útil para cualquier persona con la necesidad y/o el deseo de realizar ejercicio fuera del ambiente de un gimnasio o centro deportivo, pero manteniendo la profesionalidad que estos sitios ofrecen gracias a un entrenador personal cualificado que estudiará a cada usuario registrado en la aplicación y le dará el trato personal conveniente teniendo en cuenta sus patologías.

Un punto fuerte que destacar en estas conclusiones es el esfuerzo y la dedicación invertida en la realización de este proyecto que han conseguido, junto con la constancia, convertir la idea inicial propuesta para este trabajo en una aplicación y un servicio web totalmente funcional y listo para usar a pesar de todas las dificultades encontradas durante su realización debido al, hasta entonces, desconocimientos de las tecnologías implementadas y utilizadas.

Es por esto último, que puedo garantizar como autora del proyecto que gracias a su desarrollo he podido aprender multitud de conceptos, tecnologías y herramientas que hasta ahora desconocía y que, sin duda, enriquecen mi experiencia académica, habiéndome dado la oportunidad de ampliar mis conocimientos en el campo del desarrollo software, y personal al tener que hacer frente a todos los desafíos que se me han ido presentando durante el camino y habiendo descubierto una rama de la Ingeniería de Telecomunicaciones que me apasiona.

Finalmente, no puedo concluir este proyecto sin dar las gracias a mi Tutora María Teresa Ariza Gómez por proponerme este desafío y por ayudarme y asesorarme en todo momento.

10 REFERENCIAS

REFERENCIAS A IMÁGENES

- Ilustración 4) Estructura Android: https://es.slideshare.net/javacasm/21-android-cep-jaen-2014-estructura-de-aplicacin
- Ilustración 5) Spring Framework Runtime: https://docs.spring.io/spring/docs/4.2.x/spring-framework-reference/html/overview.html
- Ilustración 33) Funcionamiento Account Kit: https://ospreysecurity.com/blog/choosing-two-factor-authentication-to-secure-our-login-page/

REFERENCIAS A DOCUMENTACIÓN

- Libro *"Spring for Android Starter"* de Anthony Dahanne.
- Servicio web Spring: https://spring.io/
- Spring y Android. Comunicación y relación: https://docs.spring.io/autorepo/docs/spring-android/1.0.x/reference/htmlsingle/
- Documentación Android: https://developer.android.com/
- Clase RestTemplate para Spring Android: https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/web/client/RestTemplate.html
- Servicio Account Kit de Facebook para el inicio de sesión a través del número de teléfono: https://developers.facebook.com/docs/accountkit/android
- API Youtube para Android: https://developers.google.com/youtube/android/player/?hl=es-419
- API Google Places para Android: https://developers.google.com/places/android-api/?hl=es-419
- Tutoriales de Backend y Frontend para desarrollar la aplicación Android: http://www.hermosaprogramacion.com/
- Llamadas asíncronas desde android a servicio web: https://jarroba.com/asynctask-en-android/
- MySQL con phpMyAdmin: http://desarrolloweb.dlsi.ua.es/idesweb-2a-ed/modulo-09

11 ANEXO A: DESPLIEGUE DEL SERVICIO

En el presente anexo se documentará paso a paso cómo realizar el despliegue tanto de la aplicación Android como del servicio web creados para la reproducción de estos en un equipo.

Antes de entrar en materia, resulta necesario aclarar que el equipo en el que se vaya a reproducir el proyecto desarrollado debe cumplir con los siguientes requisitos para garantizar el éxito de la instalación y posterior funcionamiento del servicio.

- Sistema operativo Ubuntu 16.04 o superior.
- Java JDK 8 instalado y configurado correctamente.
- Servidor Apache instalado y corriendo.
- MySQL instalado y corriendo. Es recomendable, pero no necesario, tener instalado un gestor de base de datos MySQL, por ejemplo phpMyAdmin.
- Un IDE donde importar la aplicación móvil como Android Studio o Eclipse.
- El IDE de Spring "Spring Tool Suite" o STS para cargar el servicio web en el equipo.

La instalación y configuración de los aspectos anteriores no es materia de este anexo por lo que se deja en manos del lector.

11.1 Despliegue de la aplicación móvil MyTraining

En este apartado se va a detallar los pasos a seguir para desplegar la aplicación móvil desarrollada en un equipo con las características anteriores e instalarla posteriormente en un dispositivo móvil Android.

1. Descargar en el equipo el archivo comprimido llamado "MyTrainingApp.zip" y descomprimirlo en cualquier directorio.
2. Iniciar Android Studio.

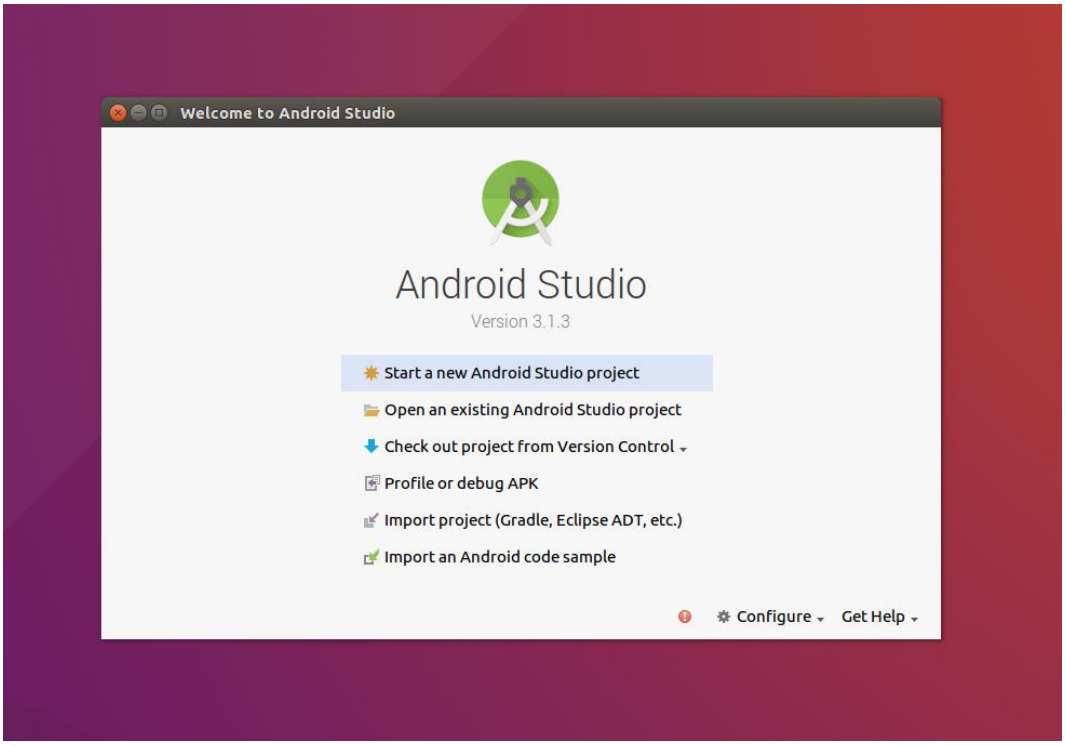


Ilustración 60: Selección del proyecto en AndroidStudio.

3. Tras hacerlo saldrá una ventana emergente como la mostrada. Seleccionar la penúltima opción "Import Project…" y seleccionar la carpeta descomprimida en el directorio elegido en el primer paso.
4. El proyecto se abrirá tras cargar todas las dependencias y ficheros de configuración y ya podrá navegar por él dentro del IDE.
5. El último paso será instalar la aplicación en el móvil. Para ello se conecta por USB el dispositivo al equipo y desde el IDE se pulsa el botón de play de la barra de herramientas.

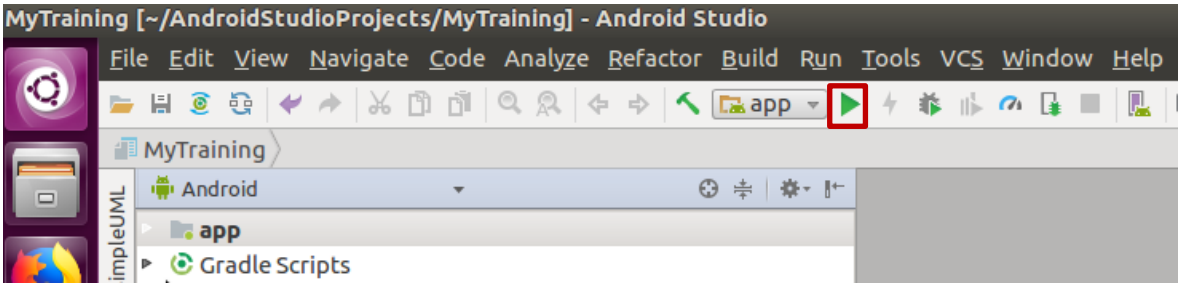


Ilustración 61: Reproducción de la aplicación en AndroidStudio.

Finalmente, saldrá la ventana emergente adjunta mostrando el dispositivo conectado al equipo y se seleccionará para instalar en él la aplicación y reproducirla en él finalmente.

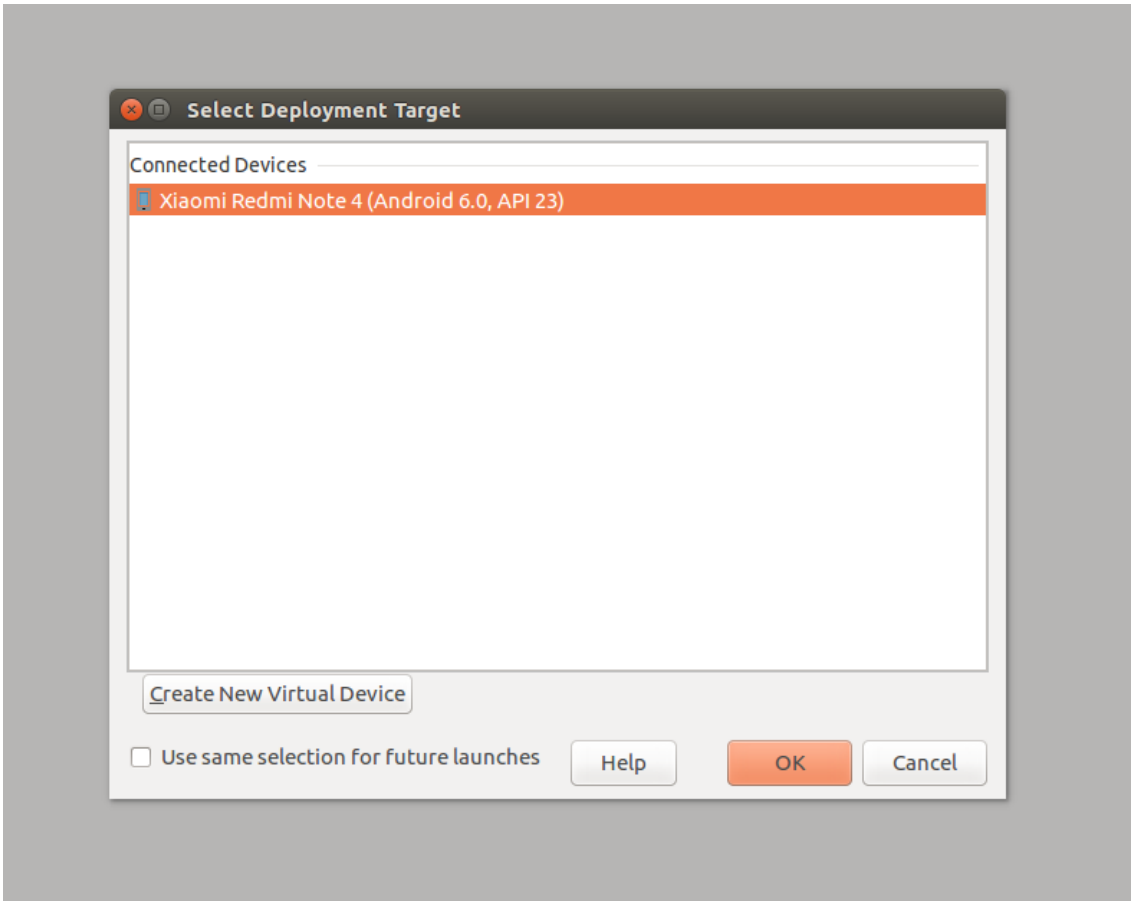


Ilustración 62: Instalación de la aplicación en el dispositivo seleccionado.

11.2 Despliegue del servidor web en STS

En este apartado se van a explicar los pasos a seguir para desplegar el servidor web desarrollado en un equipo con las características descritas al inicio de este capítulo y ponerlo a correr para que satisfaga las peticiones realizadas desde la aplicación móvil.

1. Descargar en el equipo el archivo comprimido llamado "MyTrainingServer.zip" y descomprimirlo en el *workspace* definido durante la instalación de Spring Tool Suite.
2. Iniciar Spring Tool Suite.
3. Cargar el proyecto accediendo a File → Import… → Maven → Existing Maven Projects y en la siguiente pantalla buscar el proyecto descomprimido en la ruta mencionada en el primer punto y cargarlo en el IDE. Todo este proceso se puede ver en las siguientes imágenes.

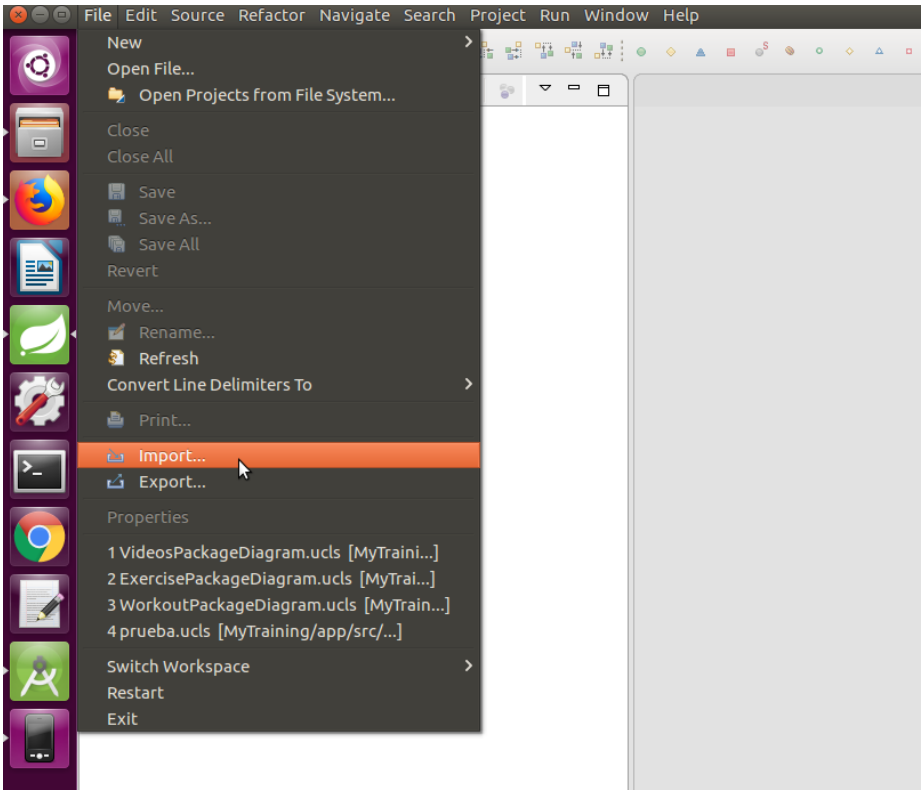


Ilustración 63: Importar proyecto en STS – parte I.

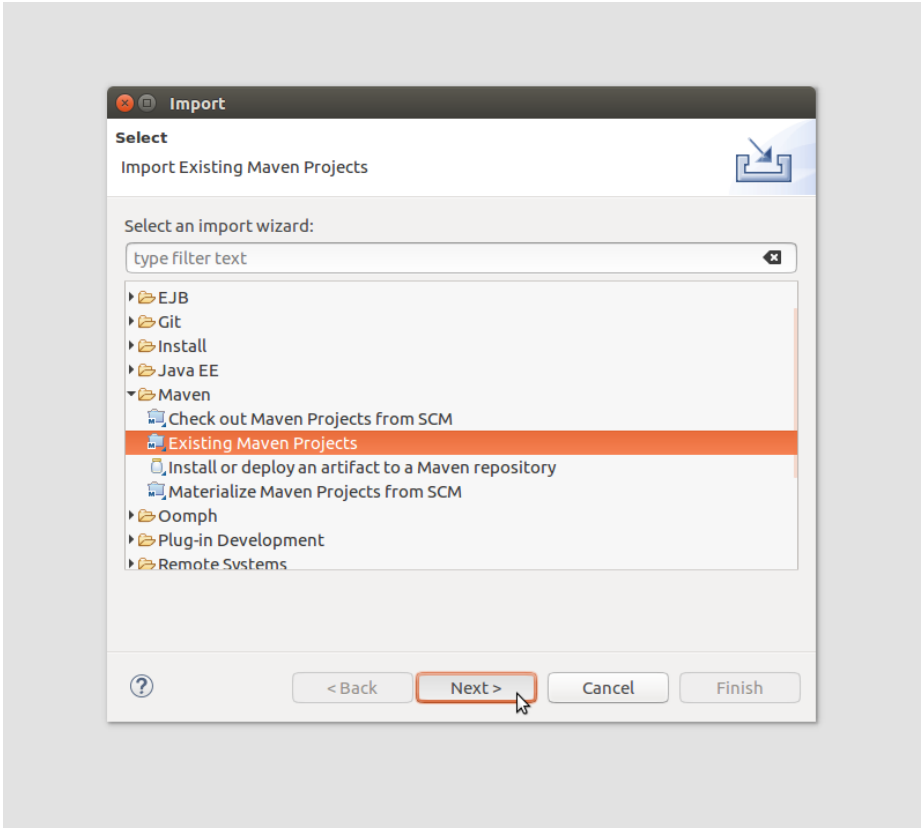


Ilustración 64: Importar proyecto en STS – parte II.

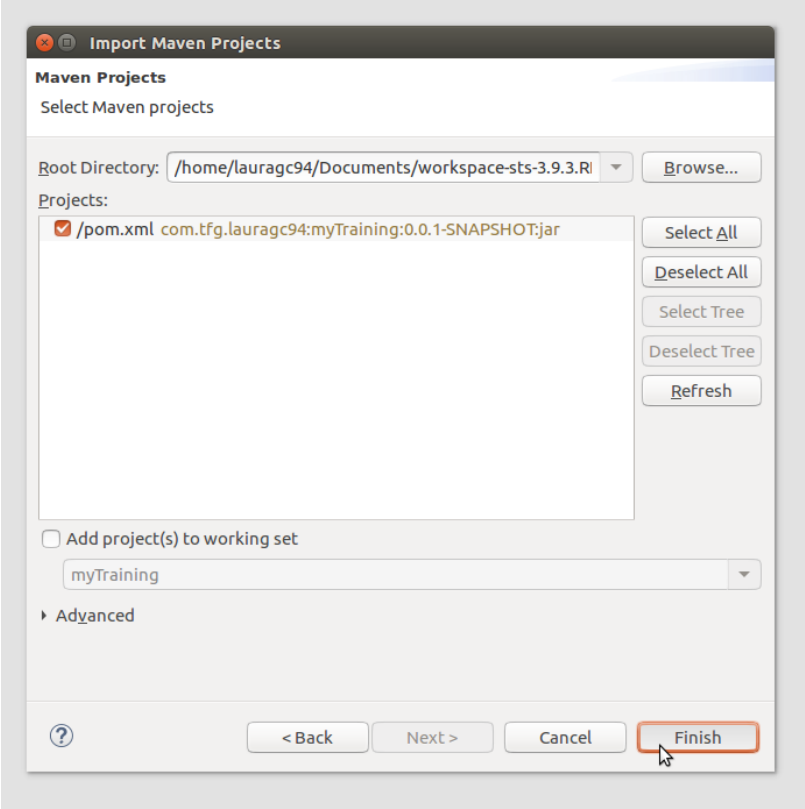


Ilustración 65: Importar el proyecto en STS - parte III.

4. Finalmente, para poner a correr el servidor proporcionado por Spring Boot con el fin de inicializar el servicio web creado se debe hacer lo mostrado en la captura siguiente.

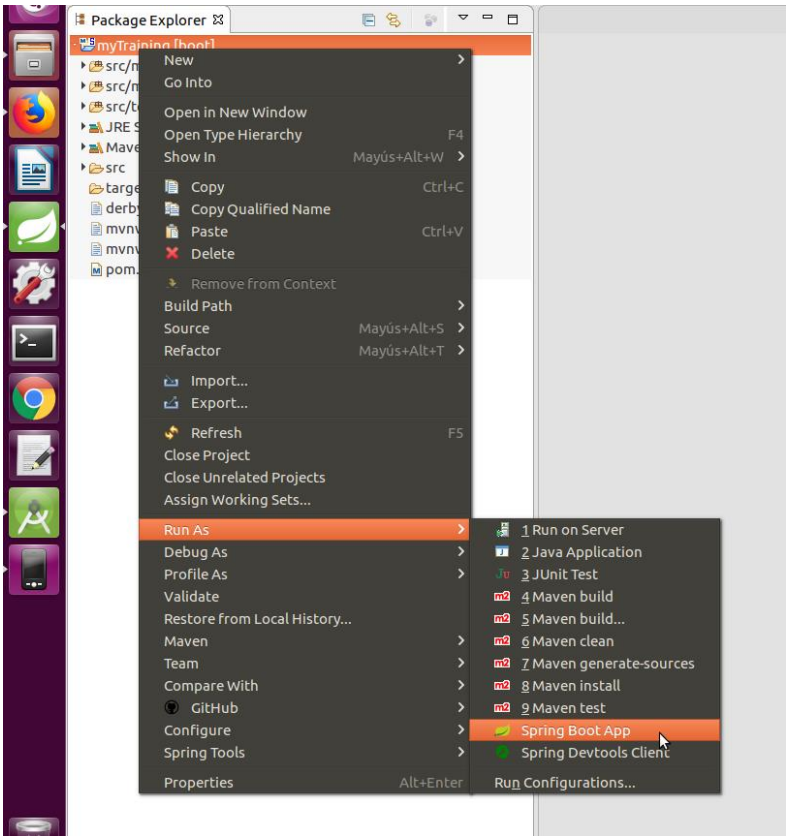


Ilustración 66: Poner a correr el servicio con Spring Boot.

De esta manera, como se puede observar en la imagen adjunta, ya se tendrá el servicio corriendo y funcionando para responder a las peticiones realizadas desde la aplicación móvil.

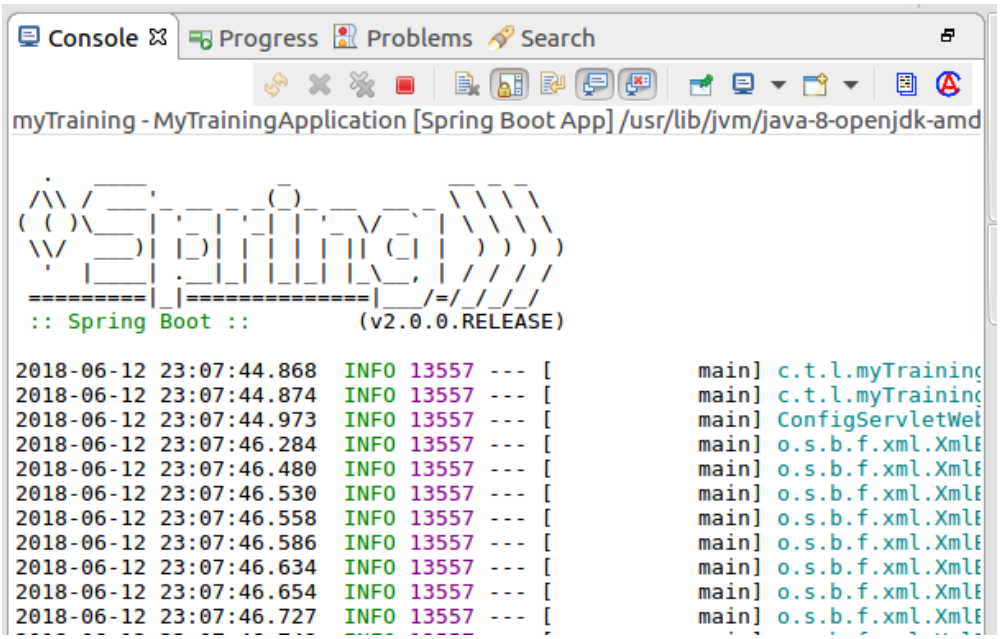


Ilustración 67: servicio web corriendo con Spring Boot.

12 ANEXO B: UTILIZACIÓN DE APIS

Este anexo consistirá en un manual de instalación y uso de los servicios externos y APIs utilizadas en la aplicación móvil ya mencionadas en la sección Servicios externos y APIs implementadas del sexto capítulo de la presente memoria.

12.1 Uso de Account Kit en Android

A continuación se van a mencionar los pasos que se deberán seguir si se quiere usar la funcionalidad ofrecida por Account Kit e implementarla en el inicio de sesión de una aplicación Android.

1. Registrar la aplicación en Facebook para obtener un identificador que haga referencia a ella.
2. Tras registrarla, es necesario definir su configuración para posteriormente implementarla en código.
3. Importar el SDK de Account Kit en el build.gradle del proyecto.

```
repositories {
  jcenter()
}

dependencies {
  implementation 'com.facebook.android:account-kit-sdk:4.+'
}
```

4. Añadir en el archivo Manifest el ID de Facebook anteriormente obtenido al registrar la aplicación y el TOKEN (en el caso de que el desarrollador haya configurado su aplicación para poder usarlo) y permitir el acceso a internet. Además, la clase AccountKitActivity también debe definirse aquí para que pueda iniciarse en la aplicación.

```
<uses-permission android:name="android.permission.INTERNET"/>
<meta-data android:name="com.facebook.accountkit.ApplicationName"
           android:value="@string/app_name" />
<meta-data android:name="com.facebook.sdk.ApplicationId"
           android:value="@string/FACEBOOK_APP_ID" />
<meta-data android:name="com.facebook.accountkit.ClientToken"
```

```
            android:value="@string/ACCOUNT_KIT_CLIENT_TOKEN" />

<activity
  android:name="com.facebook.accountkit.ui.AccountKitActivity"
  android:theme="@style/AppLoginTheme"
  tools:replace="android:theme" />
```

5. Añadir los hashes de clave de desarrollo y activación (este último en caso de publicar la aplicación). Para obtenerlos se puede usar los respectivos comandos:

```
keytool -exportcert -alias androiddebugkey -keystore
~/.android/debug.keystore | openssl sha1 -binary | openssl base64

keytool -exportcert -alias YOUR_RELEASE_KEY_ALIAS -keystore
YOUR_RELEASE_KEY_PATH | openssl sha1 -binary | openssl base64
```

6. Una vez configurado el sistema para poder usar Account Kit se implementa el código para gestionar el inicio de sesión usando el teléfono móvil. En primer lugar se deberá consultar si existe una sesión abierta para redireccionar al usuario a la pantalla pertinente o, en el caso contrario, comenzar con el proceso de inicio de sesión.

```
AccessToken accessToken = AccountKit.getCurrentAccessToken();

if (accessToken != null) {
  //Handle Returning User
} else {
  //Handle new or logged out user
```

7. En el caso de no haber sesión abierta se manda al usuario a iniciar sesión con su número de teléfono de la siguiente manera:

```
public static int APP_REQUEST_CODE = 99;

public void phoneLogin(final View view) {
  final Intent intent = new Intent(getActivity(),
AccountKitActivity.class);
  AccountKitConfiguration.AccountKitConfigurationBuilder
configurationBuilder =
    new AccountKitConfiguration.AccountKitConfigurationBuilder(
      LoginType.PHONE,
```

```
      AccountKitActivity.ResponseType.CODE); // or .ResponseType.TOKEN
  // ... perform additional configuration ...
  intent.putExtra(
    AccountKitActivity.ACCOUNT_KIT_ACTIVITY_CONFIGURATION,
    configurationBuilder.build());
  startActivityForResult(intent, APP_REQUEST_CODE);
}
```

8. Una vez iniciado sesión para gestionar el resultado de esta actividad es necesario implementar lo siguiente:

```
@Override
    protected void onActivityResult(
            final int requestCode,
            final int resultCode,
            final Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (requestCode == APP_REQUEST_CODE) { // confirm that this
response matches your request
            AccountKitLoginResult loginResult =
data.getParcelableExtra(AccountKitLoginResult.RESULT_KEY);
            String toastMessage;
            if (loginResult.getError() != null) {
                toastMessage =
loginResult.getError().getErrorType().getMessage();
                showErrorActivity(loginResult.getError());
            } else if (loginResult.wasCancelled()) {
                toastMessage = "Login Cancelled";
            } else {
                if (loginResult.getAccessToken() != null) {
                    toastMessage = "Success:" +
loginResult.getAccessToken().getAccountId();
                } else {
                    toastMessage = String.format(
                            "Success:%s...",

loginResult.getAuthorizationCode().substring(0,10));
                }

                // If you have an authorization code, retrieve it from
                // loginResult.getAuthorizationCode()
                // and pass it to your server and exchange it for an
access token.
```

```
                // Success! Start your next activity...
                goToMyLoggedInActivity();
            }

            // Surface the result to your user in an appropriate way.
            Toast.makeText(
                    this,
                    toastMessage,
                    Toast.LENGTH_LONG)
                    .show();
        }
}
```

9. Si se ha configurado el inicio de sesión para usar el TOKEN estará disponible la opción de cerrar sesión eliminando la clase AccessToken que se almacena en el dispositivo cuando se inicia sesión exitosamente.

```
AccountKit.logOut();
```

10. Finalmente si se desea acceder a la información del usuario que ha accedido a la aplicación y ha iniciado sesión mediante este método se podrá hacer implementando el siguiente trozo de código.

```
AccountKit.getCurrentAccount(new AccountKitCallback<Account>() {
  @Override
  public void onSuccess(final Account account) {
    // Get Account Kit ID
    String accountKitId = account.getId();

    // Get phone number
    PhoneNumber phoneNumber = account.getPhoneNumber();
    if (phoneNumber != null) {
      String phoneNumberString = phoneNumber.toString();
    }

    // Get email
    String email = account.getEmail();
  }

  @Override
  public void onError(final AccountKitError error) {
    // Handle Error
  }
});
```

12.2 Uso de YouTube API en Android

YouTube para desarrolladores ofrece una gran variedad de ejemplos de uso de su API para Android pudiendo el desarrollador instalar cualquier de ellos en su dispositivo y estudiar su código sin ningún tipo de restricción simplemente registrando la aplicación en Google APIs y obteniendo así la clave que se deberá introducir en la configuración de la aplicación.

Los siguientes documentos sirven de gran ayuda a la hora de configurar el entorno de desarrollo y el uso de la API del reproductor de YouTube para Android:

- Enlace para descargar la biblioteca de cliente API y JavaDocs.
- Instrucciones para registrar la aplicación en la Consola de API de Google.
- Instrucciones de instalación sobre cómo configurar un proyecto de API mediante Eclipse o IntelliJ.
- Aplicaciones de ejemplo que contienen todas las funciones ofrecidas por la API.
- Referencia JavaDoc que define de forma detallada las interfaces, categorías, métodos y enumeraciones de la API.