



UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén

Trabajo Fin de Grado

GESTIÓN Y ACCESO A LOS SERVICIOS DE UNA ORGANIZACIÓN O EMPRESA MEDIANTE APLICACIÓN MÓVIL

Alumno: Ángel Moral Cano

Tutor: Prof. D. Pedro José Sánchez Sánchez
Dpto: Informática

Septiembre, 2018



Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Don Pedro J. Sánchez Sánchez , tutor del Proyecto Fin de Carrera titulado: Gestión y Acceso a los Servicios de una Organización o Empresa mediante Aplicación Móvil, que presenta Ángel Moral Cano, autoriza su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, Septiembre de 2018

El alumno:

Los tutores:

Ángel Moral Cano

Pedro J. Sánchez Sánchez

Índice

CAPÍTULO 1 INTRODUCCIÓN	5
1.1. Introducción al trabajo	6
1.2. Motivación	7
1.3. Objetivos	8
CAPÍTULO 2 SERVICIO WEB	11
2.1. ¿Qué es un Servicio Web?	12
2.2. ¿Qué podemos hacer con los Servicios Web?.....	13
2.3. Estándares Servicio Web.....	14
2.4. Ventajas de los Servicios Web	15
2.5. Inconvenientes de los Servicios Web.....	16
2.6. Por qué utilizar un Servicio Web	16
2.7. XAMPP	17
2.7.1. Apache	18
2.7.2. MySQL	20
2.7.3. MariaDB	23
2.7.4. PHP	25
2.7.5. phpMyAdmin.....	28
2.8. Diferentes tecnologías para realizar Servicios Web.....	28
2.8.1. SOAP	28
2.8.2. REST.....	29
CAPÍTULO 3 ANDROID	33
3.1. ¿Qué es Android?	34
3.2. Características de Android	35
3.3. Arquitectura de Android.....	38
3.4. Comparativa con los sistemas operativos móviles más usados.....	40
3.5. Por qué elegir Android	45
CAPÍTULO 4 ANÁLISIS DEL PROYECTO.....	47
4.1. Análisis de Requerimientos.....	48
4.2. Análisis del sistema.....	53
4.2.1. Actores	53
4.2.2. Casos de uso.....	54
4.2.3. Diagramas de secuencia.....	63
4.2.4. Escenarios	70

4.3. Diseño.....	75
4.3.1. Diseño de la base de datos	75
4.3.2. Diseño de la interfaz	80
CAPÍTULO 5 IMPLEMENTACIÓN.....	95
5.1. Arquitectura del Sistema	96
5.2. Lenguajes de programación	97
5.2.1. Servicio Web.....	97
5.2.2. Aplicación Android.....	101
5.3. Pruebas y validación.....	115
CAPÍTULO 6 CONCLUSIÓN Y TRABAJOS FUTUROS.....	123
6.1. Conclusión.....	124
6.2. Trabajos futuros.....	124
Bibliografía	127
ANEXO I MANUAL DE INSTALACIÓN DEL SERVIDOR WEB.....	131
ANEXO II MANUAL DE INSTALACIÓN DE LA APLICACIÓN ANDROID.....	139
ANEXO III MANUAL DE USUARIO.....	143

CAPÍTULO 1



INTRODUCCIÓN

1.1. Introducción al trabajo

Vivimos en un mundo en continuo cambio, especialmente en el ámbito tecnológico, podemos observar como día tras día aparecen nuevas herramientas tecnológicas, tecnologías y ramas de investigación, pero no solo podemos observar este cambio en el habito global sino también en lo cotidiano, con la salida al mercado de nuevos dispositivos, nuevas aplicaciones y en definitiva nuevas facilidades para el ser humano.

Esta evolución comenzó hace unas décadas pero no fue hasta la aparición del smartphone cuando alcanza su punto álgido, hasta llegar a un punto en el que hoy en día no concebimos el mundo sin el uso de este dispositivo y ya no solo eso, si no que se han ido incorporando nuevos dispositivos con la intención de sustituir otros dispositivos tradicionales como son el caso de los relojes inteligentes, coches inteligentes, televisiones inteligentes y una infinidad de dispositivos denominados “inteligentes” [42]. Aunque en esta ocasión nos vamos a centrar en los teléfonos inteligentes, los smathphones.

Son las grandes empresas las que ha visto una enorme oportunidad de negocio en el apartado de las aplicaciones móviles. Ya no solo existen aplicaciones de entretenimiento o juegos, si no también aplicaciones con las que el usuario puede tener una gran información útil y la capacidad de realizar grandes acciones desde la palma de su mano, como puede ser con la utilización de una aplicación de banca, donde el usuario desde cualquier parte del mundo puede gestionar todas sus finanzas. Con relación a esta reflexión aparece la siguiente cuestión, ¿Puede cualquier empresa aprovecharse del enorme potencial que tienen las aplicaciones móviles?.

La respuesta esta muy clara, definitivamente Sí, con el avance de las tecnologías también ha evolucionado la forma de desarrollarla, con lo cual podremos tener sistema que satisfagan incluso que supere las expectativas de pequeñas empresas o organizaciones sin necesidad que realizar una inversión muy grande para ello.

Y ¿por qué necesita una pequeña empresa tener una aplicación móvil? [43], los principales motivos son la comodidad y la eficiencia, como ya hemos visto podremos tener toda la información de nuestra empresa al alcance de nuestra mano y en

cualquier momento y lugar, pero a demás si los clientes de nuestra empresa también pueden utilizarla esta tendrá un gran valor añadido.

De la unión de estas dos ideas nace el proyecto que voy a desarrollar, una aplicación con la que el propietario de una empresa pueda tener en su smartphone la posibilidad de gestionar su negocio y que los usuarios puedan tener una aplicación útil para su día a día.

1.2. Motivación

En la actualidad todo el mundo dispone de un dispositivo móvil con el cual poder conectarnos a internet, ya sea para navegar, consumir contenido multimedia, aplicaciones de mensajería, etc.

Vamos a aprovechar este hecho para realizar aplicaciones que no sean simplemente para el entretenimiento del consumidor si no que realmente sean aplicaciones útiles con las cuales el usuario pueda realizar tareas de la vida cotidiana. De la misma manera a nivel empresarial para mejorar los servicios que ofrecen a estos consumidores.

En este proyecto, me centraré en crear una Aplicación Android para el acceso y gestión de los servicios que ofrece un gimnasio, tanto de la gestión de clientes y monitores como de las clases que el centro pone a disposición de sus clientes, además de la creación del Servicio Web que sustente la lógica de la aplicación. Ya que con la utilización de esta tecnología, el propietario del gimnasio tendrá en su smartphone toda la información relativa a su gimnasio. Tanto de los clientes como de los monitores podrán acceder a los servicios que ofrezca de manera rápida y sencilla, de la misma manera, podremos abandonar el tradicional sistema de inscripción a las distintas clases de manera presencial por parte de los clientes del centro así como solucionar el problema que tienen los centros con un gran numero de clientes, en los que los clientes puede que no puedan realizar la clase que deseen ya que al llegar al centro se encuentran que la clase esta completa, para pasar a un sistema donde la inscripciones se realicen mediante la aplicación proporcionando una mayor

comodidad y asegurar al cliente que va a tener su plaza reservada en la clase en la cual se inscriba. Por otro lado los monitores tendrá en su dispositivo móvil las clases que tienen que realizar y los usuarios que participarán en dichas clases.

Es importante destacar al publico al que va dirigido este proyecto, los usuarios de los gimnasios suelen ser personas jóvenes que están muy familiarizados con las nuevas tecnologías ya que el uso del smarthphone es indispensable en su día a día, por lo que es un incentivo más a la hora de desarrollar este proyecto.

1.3. Objetivos

La idea principal de mi trabajo es cubrir las necesidades de todas los usuarios del gimnasio. Por un lado el propietario podrá acceder y gestionar todo el personal que trabaja en el gimnasio, los clientes que utilizan sus instalaciones y las distintas clases que el centro pone a disposición de sus clientes. Los monitores podrán consultar las clases que tienen que realizar y los clientes podrán consultar las clases de las que dispone el gimnasio y las clases a las que ya se ha inscrito. Por tanto los objetivos principales del proyecto son:

- Gestión de clientes y monitores.
- Creación y asignación de clases.
- Permitir a los clientes inscribirse a las clases que quieran realizar.
- Consultar las clases que un monitor tiene que impartir.
- Consultar las clases a las que se ha inscrito un cliente.

1.4. Resumen de la memoria

A continuación presento un breve resumen de los diferentes capítulos que conforman esta memoria, donde encontraremos todo lo necesario para la realización de este proyecto.

En el siguiente capítulo, explicaremos que es un Servicio Web [1] y como podemos utilizarlo en la actualidad, así como los diferentes estándares que existen, ventajas, desventajas y por utilizar un Servicio Web. Definiremos también la plataforma que vamos a utilizar para la realización del Servicio Web, XAMPP [8] que es y los elementos que lo componen como son Apache [11] que es el servidor que utilizaremos, MariaDB [14] junto con MySQL [12] que serán los gestores de bases de datos y PHP [15] que es el lenguaje de programación que usaremos, todo ello acompañado con sus ventajas e inconvenientes a la hora de utilizarlo. Por último cerraremos este capítulo con las diferentes tecnologías con las que podemos realizar un Servicio Web y la razón por decantarnos por la tecnología REST [16].

A continuación, en el Capítulo 3, presentaremos Android [17] para ello explicaremos que es, cuales son sus características más relevantes, su arquitectura así como una comparativa con otros sistemas operativos móviles de la actualidad y el por que hemos elegido Android para realizar este proyecto.

En el Capítulo 4 entramos en profundidad en todo lo referente al proyecto, incluyendo una pequeña introducción de este. Posteriormente pasaremos a desglosar todo lo referente con la Ingeniería Software como son el Análisis de requerimientos, Análisis del Sistema así como el Diseño, incluyendo el Diseño de la Base de Datos y de la Interfaz de usuario.

Posteriormente en el Capítulo 5 y una vez conocido todo lo anteriormente mencionado, incluiremos todo lo referente a la implementación del proyecto como son las plataformas de desarrollo utilizadas, así como, la organización y función que tienen cada uno de los ficheros que lo componen.

Por último en el Capítulo 6 se expondrán las conclusiones obtenidas tras la realización del proyecto y los posibles trabajos que se podrá realizar en un futuro para que el proyecto evolucione.

CAPÍTULO 2



SERVICIO WEB

En el presente capítulo vamos a explicar que es un Servicio Web y cuales son sus aplicaciones, que plataforma vamos a utilizar para poner en marcha nuestro Servicio Web así como las diferentes tecnología que lo conforman.

2.1. ¿Qué es un Servicio Web?

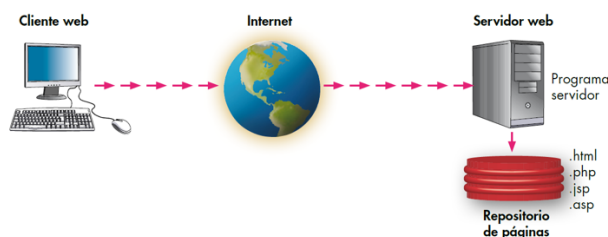


Ilustración 2.1 - Servicio Web

La World Wide Web Consortium [2] (W3C) define un servicio web [1] como “...un sistema de software diseñado para soportar interacción interoperable máquina a máquina sobre una red. Este tiene una interface descrita en un formato procesable por una máquina (específicamente WSDL (Web Services Description Language)). Otros sistemas interactúan con el servicios web en una manera prescrita por su descripción usando mensajes SOAP (Simple Object Access Protocol) [28], típicamente enviados usando HTTP (HiperText Transfer Protocol) [29] con una serialización XML (Extensible Markup Language) [30] en relación con otros estándares relacionados con la web”.

Un servicio web es una tecnología que emplea un conjunto de estándares y protocolos para poder intercambiar datos entre aplicaciones. Los servicios web pueden ser utilizados por diferentes aplicaciones software ejecutadas sobre cualquier plataforma y desarrolladas en lenguajes de programación dispares, con el fin de intercambiar datos en redes de ordenadores como internet. La interoperabilidad se consigue utilizando estándares abiertos. Las organizaciones OASIS [3] (Organization for the Advancement of Structured Information Standards) y W3C son los comités responsables de la arquitectura y reglamentación de los servicios Web.

Para mejorar la interoperabilidad entre distintas implementaciones de servicios Web se ha creado el organismo WS-I [4] (Web Services Interoperability Organization),

encargado de desarrollar diversos perfiles para definir de manera más exhaustiva estos estándares.

2.2. ¿Qué podemos hacer con los Servicios Web?

En un principio la idea de un servicio web era la de una función a la cual se le entregaban algunos parámetros, esta función devolvía un valor en base a una serie de cálculos.

A medida que fue pasando el tiempo y con ello la tecnología, esta primera idea evolucionó de manera que cada objeto tenía varias funciones para poder realizar, además de sus propias variables de datos privados. Ya no se basaba únicamente en variables externas del sistema, con lo que el desarrollo de aplicaciones era mas sencillo. Conforme las aplicaciones empezaron a comunicarse entre ellas, cobro especial importancia el concepto de interfaz universal, para poder permitir que objetos pertenecientes a distintas plataformas puedan comunicarse sin problema a pesar de haber sido escritos en otros lenguajes de programación y operen en otros sistemas operativos.

Más recientemente, los servicios web se han acercado al este concepto de interfaces y las comunicaciones se han definido en XML, para de esta manera poder unir aplicaciones de todo tipo y tener la libertad de cambiar a lo largo del tiempo.

El aspecto distintivo de los servidores web con las tecnología que componen la generación anterior es la versatilidad de XML. Al permitirnos separar el significado gramatical (semántica) de la estructura (sintaxis) posibilitando que cada servicio del entorno procese la parte que le corresponde. Por esta razón en la actualidad podemos definir objetos como servicios que se comunican con otros servicios en formato XML. Por lo tanto, una aplicación puede estar formada por distintas entidades con sus correspondientes construcciones y diseños pero deberá cumplir con las reglas de su arquitectura orientada a servicios.

Así, los servicios web permiten [5]:

- Interacción entre servicios escritos en cualquier lenguaje de programación y ejecutados sobre cualquier plataforma.
- Conceptualizar funciones de aplicaciones en tareas, lo que lleva al desarrollo y a flujos de trabajo orientados a tareas.
- Evitar que se rompan las interacciones entre aplicaciones cuando se produzca un cambio en la implementación o en el diseño.
- Adaptar las aplicaciones existentes a las diferentes necesidades de los clientes y a las cambiantes condiciones a nivel empresarial.
- Proporcionar aplicaciones de software ya existentes o legadas con interfaces de servicio sin cambiar las aplicaciones originales, lo que permite operar totalmente en el entorno de servicios.
- Incorporar diferentes funciones como la seguridad, confidencialidad, etc, sin alterar la función original de la aplicación.

2.3. Estándares Servicio Web

Como hemos visto anteriormente existen una serie de estándares para permitir esta interoperabilidad entre las aplicaciones. Estos estándares son:

- **Web Services Protocol Stack:** Lo conforman el conjunto de servicios y protocolos de los servicios web.
- **XML** (Extensible Markup Language): formato estándar para los datos que se vayan a intercambiar.
- **SOAP** (Simple Object Access Protocol) o **XML-RPC** (XML Remote Procedure Call): Son protocolos sobre los que se establece el intercambio de datos.
- **WSDL** (Web Services Description Language): es el lenguaje de la interfaz pública para los servicios web.
- **UDDI** (Universal Description, Discovery and Integration): protocolo que permite publicar y encontrar información a cerca de servicios Web. Permite comprobar qué servicios web están disponibles.

- **WS-Security** (Web Service Security): protocolo de seguridad aceptado como estándar por OASIS (Organization for the Advancement of Structured Information Standards). Garantiza la autenticación de los actores y la confidencialidad de los mensajes enviados.

2.4. Ventajas de los Servicios Web

- Permiten la interoperabilidad de aplicaciones con diferentes propiedades e instaladas en diferentes plataformas [6].
- Fomentar el uso de protocolos y estándares basados en texto para permitir un funcionamiento más sencillo y accesible.
- Proveer servicios integrados mediante la combinación de diferentes servicios y software ubicados en distintos lugares geográficos.
- Permiten la integración “justo-a-tiempo”: El proceso de descubrimiento se ejecuta dinámicamente, mientras que los solicitantes de servicio utilizan a los agentes para encontrar proveedores de servicio [7].
- Uso de encapsulamiento para disminuir la complejidad. Los proveedores de servicios no saben como el cliente utiliza el servicio, así mismo, el cliente no sabe como se implemento el servicio que proporciona el proveedor.
- Permiten compartir servicios con un alto grado de integración, facilitando las comunicaciones entre distintas organizaciones.
- Mayor rapidez en el desarrollo de aplicaciones gracias a la filosofía de orientación a objetos.

2.5. Inconvenientes de los Servicios Web

- No pueden compararse con los estándares abiertos de comunicación distribuida como COBRA (*Common Object Request Broker Architecture*), para la realización de transacciones, ya que estos están mucho mas desarrollados.
- Tiene un rendimiento menor con respecto a otros modelos de computación distribuida debido a adoptar un formato basado en texto.
- Al apoyarse en HTTP, pueden esquivar medidas de seguridad basadas en *firewall* cuyas reglas tratan de bloquear o auditar la comunicación.

2.6. Por qué utilizar un Servicio Web

Uno de los motivos mas importantes para utilizar un servicio Web es que se puede utilizar desde cualquier ordenador o plataforma, facilitando así su uso.

Otro motivo a tener en cuenta es que los servicios Web utilizan el protocolo HTTP sobre TCP utilizando para ello el puerto 80 por lo que no obtendremos bloqueos a la hora de comunicarnos en cualquier red ya que aunque haya organizaciones que protejan sus redes filtrando el trafico y bloqueando parte de este, el puerto 80 no lo suelen bloquear ya que es el usado por HTTP. Mencionar que los servicios web se pueden utilizar sobre cualquier protocolo, sin embargo, TCP es el más común.

La última razón es la independencia que tienen los servicios web con las aplicaciones que los utilizan, esto hace que los cambios que se realicen en un sistema no afectan al otro.

2.7. XAMPP

En este apartado de la memoria vamos a realizar un análisis de la plataforma elegida para realizar el Servicio Web. Esta plataforma nos facilitará todas las herramientas y funcionalidades necesarias para realizar la comunicación entre la aplicación y el resto de sistemas implicados. Esta plataforma es XAMPP.



Ilustración 2.2 XAMPP – Página Oficial de XAMPP
(<https://www.apachefriends.org/es/index.html>)

XAMPP [8] es un paquete de software libre, que formado por un sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script PHP y Perl. El nombre es en realidad un acrónimo:

- **X**, para cualquiera de los diferentes sistemas operativos.
- **A**pache, como servidor Web
- **M**ariaDB/MySQL, como gestor de base de datos
- **P**HP, lenguaje de programación.
- **P**erl, lenguaje de programación.

A partir de la versión 5.6.15, XAMPP cambió la base de datos MySQL por MariaDB, una adaptación de MySQL con licencia GPL [9].

La principal ventaja de XAMPP es que puede ejecutarse sobre cualquier sistema operativo, con lo cual podemos estar ejecutándolo en una máquina y trasladarlo a otra con un sistema operativo diferente sin que esto comprometa el correcto funcionamiento de nuestro sistema.

La filosofía de XAMPP, como lo indican su sitio web, es crear una distribución fácil de instalar, de tal manera que los desarrolladores web principiantes cuenten con todo lo necesario ya configurado. XAMPP solamente requiere descargar y ejecutar un

archivo zip, tar , exe o fkl, con unas pequeñas configuraciones en alguno de sus componentes para poner en funcionamiento el servidor Web.

XAMPP se actualiza regularmente para incorporar las últimas versiones de Apache/MySQL/PHP y Perl. También incluye otros módulos como OpenSSL y phpMyAdmin, este ultimo los analizaremos en profundidad más a delante.

2.7.1. Apache

El **servidor HTTP Apache** [10] es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1, se utiliza para la creación de páginas y servicios web. Se caracteriza por ser un servidor multiplataforma, gratuito, muy robusto y destaca por su seguridad y rendimiento.



Ilustración 2.3 – Apache – Página Oficial de Apache (<http://httpd.apache.org>)

El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation [11].

Apache es el servidor web por excelencia en todo el mundo, esto se debe a su estabilidad y seguridad. Fue una pieza clave en el desarrollo de la World Wide Web, alcanzando su máximo apogeo en el año 2005 al ser el servidor que utilizaban el 70% de los sitios web.

Ventajas

- **Instalación/Configuración.** Software de código abierto. Tanto la instalación como la configuración son sencillas para el usuario.
- **Funcional y Soporte.** Alta aceptación en la red y muy popular, al ser el servidor web mas usado hace que tenga una gran comunidad que lo

sustenten y contribuya a mejorarlo. Esto permite que Apache se actualice constantemente.

- **Coste.** Apache es un servidor web gratuito.
- **Multi-plataforma.** Se puede instalar en distintos sistemas operativos, como Windows, Linux y MacOS.
- **Soporte de seguridad SSL (Secure Socket Layer) y TLS (Transport Layer Security).** SSL y TLS son protocolos que proporcionan comunicaciones seguras en la red.
- **Rendimiento.** Apache tiene la capacidad de soportar más de un millón de visitas al día.
- **Modular:** El servidor se compone de una sección que compone su núcleo y distintos módulos que aportan distintas funcionalidad al servidor web.
- **Extensible**

Desventajas

Las desventajas de utilizar apache como servidor web son prácticamente nulas esto es lo que hace que sea el mas utilizado como hemos visto anteriormente, por mencionar algunas sería:

- Formatos de configuración no estándar.
- No cuenta con una buena administración.
- Falta de integración.

Usos de Apache

Apache se utiliza para enviar paginas web tanto dinámicas como estáticas a la World Wide Web. Estas páginas al estar diseñadas con Apache utilizan las características de esta servidor.

Este servidor web se redistribuye como parte de varios paquetes propietarios de software, incluyendo la base de datos Oracle y el IBM WebSphere application server. Mac OS X también integra apache como parte de su propio servidor web y como soporte de su servidor de aplicaciones WebObjects.

También utilizaremos Apache cuando queramos realizar tareas cuyo contenido requiere ser puesto a disposición de forma confiable y segura. Por ejemplo cuando un usuario quiera compartir archivos en internet, este puede colgar los archivos a la raíz de Apache y desde ahí compartirlos.

Por ultimo mencionar que otro de los usos de Apache es la de previsualizar y probar código en desarrollo utilizando una versión local.

Para la realización de este proyecto se ha utilizado la versión de Apache 2.4.33

2.7.2. MySQL

“MySQL es la base de datos de código abierto más popular del mercado. Debido a su fiabilidad, facilidad de uso y rendimiento probado, MySQL se ha convertido en la base de datos mas utilizada para las aplicaciones basadas en web y utilizada por importantes web como Facebook, Twitter, YouTube” [12] Así es como MySql se define así misma en su página web, aunque la definición va mas allá.



Ilustración 2.4 – MySQL – Página Oficial MySQL (<https://www.mysql.com>)

MySQL [13], es un sistema de gestión de base de datos relacional o SGBD. Es muy versátil permitiendo ser utilizado por varios usuarios a la vez y realizar varias consultas simultáneamente al ser multiusuario y multihilo.

En un principio era un proyecto de Software Libre aunque con el tiempo se ha convertido en Software Propietario que requiere una licencia para poder utilizarlo a nivel empresarial. Si es cierto que para usuarios particulares sigue siendo gratuito.

MySQL permite crear bases de datos simples o más complejas con mucha facilidad, esto se a la sintaxis que utiliza cuyo uso es muy simple y fácil de aprender.

Aplicaciones de MySql

Como comentábamos anteriormente este gestor de base de datos es muy utilizado en desarrollo web, MySQL permite trabajar en un sistema de gestión de datos centralizado permitiendo a los desarrolladores modificar sus sitios web de forma sencilla sin necesidad de realizar grandes cambios en la estructura y en el código del sitio.

MySQL tiene aún más valor añadido cuando se utiliza conjuntamente con PHP, es por ello por lo que actualmente cada vez se realizan más aplicaciones del tipo cliente/servidor que necesitan un base de datos muy potente a la par que rápida y segura.

Una de las aplicaciones más interesantes de MySQL su utilización en gestores de contenido como Wordpress o Drupal, estos gestores de contenido son herramientas accesible a todo tipo de usuarios y muy fáciles de utilizar, lo que permite que usuarios con conocimientos básicos de informática puedan utilizar MySQL aunque no sea de forma directa.

Por ultimo destacar la gran efectividad de MySQL en paginas web donde la concurrencia sea baja a la hora de modificar los datos pero sin embargo se produzcan un gran número de lecturas de estos datos. En el caso de que en la base de datos se produzcan muchas modificaciones de datos no sería tan interesante hacer uso de esta tecnología.

Características de MySQL

La integridad referencial y las transacciones son elementos de los cuales en un principio, carecía MySQL. A pesar de ello y gracias a su simplicidad muchos desarrolladores se decantaron por usar este gestor de base de datos. Con el avance y desarrollo de esta tecnología se han ido incorporando estos elementos. Son muchas las características que posee MySQL ya hemos mencionado algunas de ellas, aunque cabrían destacar algunas de las que incorporan sus versiones más recientes.

- Amplio subconjunto de su lenguaje propio (SQL [\[31\]](#)).
- Posibilidad de selección de mecanismos de almacenamiento que ofrecen diferentes velocidades de operación, capacidad variable, soporte físico, transacciones...
- Disponibilidad en gran cantidad de plataformas y sistemas. MySQL puede ejecutarse en los principales sistemas operativos e infinidad de aplicaciones.
- Búsqueda e indexación de campos de texto.
- Transacciones y claves foráneas. Las transacciones sirven para dar una mayor fiabilidad a las bases de datos ya que nos garantizan que se realicen todas las consultas de un conjunto sin que falle la ejecución. Las claves foráneas son un elemento para mantener la integridad de la base de datos.
- Conectividad segura.
- Replicación. MySQL permite tener distintas copias de la base de datos para en el caso de que se produzca un error poder subsanarlo utilizando alguna de estas copias.

La característica principal de MySQL es que al ser un gestor de base de datos relacional, los datos no se almacenan en un solo archivo si no que se organizan en tablas independientes. Estas tablas están conectadas mediante relaciones permitiendo cambiar datos de tablas distintas.

Las siguientes características son implementadas únicamente por MySQL:

- Permite escoger entre diferentes motores de almacenamiento para cada tabla. A partir de la versión 5.1 de MySQL estos motores se pueden añadir dinámicamente en tiempo de ejecución. Estos motores pueden ser nativos de MySQL como es el caso de MyISAM [\[32\]](#) InnoDB [\[33\]](#), desarrollados por empresa asociadas como solidDB [\[34\]](#) y desarrollados por la comunidad como memcache [\[35\]](#).
- Agrupar transacciones, reuniendo múltiples transacciones de varias conexiones para incrementar el número de transacciones por segundo.

Futuro de MySQL

A pesar de MySQL ser uno de los gestores de bases de datos más utilizados como hemos visto anteriormente, su futuro esta en entre dicho, se debe principalmente a la incertidumbre por saber si pasará a ser software completamente propietario, es por esto por lo que surgen nuevas vertientes de MySQL con un enorme potencial como es el caso de MariaDB.

MariaDB garantiza total compatibilidad con MySQL, a demás de incorporar distintas características adicionales que mejoran su rendimiento con respecto a MySQL aunque la característica diferenciadora con MySQL es que MariaDB usa licencias GPL, es decir es software libre y de código abierto pudiendo ser usada por cualquier usuario.

MariaDB es la base de datos que he utilizado para la realización de este proyecto, ya que es la base de datos que incorpora XAMPP.

2.7.3. MariaDB

MariaDB [14] es una derivación de MySQL, nace con la idea de ser una alternativa a MySQL tras ser esta adquirida por Oracle. Su objetivo es ser un gestor de base de datos libre con licencia GPL que se actualiza con cada nueva versión de MySQL incluyendo los nuevos avances que incorpore.

El principal objetivo de MariaDB es garantizar la compatibilidad con MySQL para que en el caso de que un sistema quiera cambiar de gestor de base de datos lo pueda hacer de la manera más limpia posible sin ningún tipo de error.



Ilustración 2.5 – MariaDB – Página
Oficial MariaDB (<https://mariadb.com>)

Ventajas y desventajas de MariaDB

- **Ventajas de MariaDB**
 - Incorpora nuevos motores de almacenamiento más eficientes, dejando a un lado los motores que utiliza MySQL como son MyISAM e InnoDB y dando paso a Aria [\[36\]](#) y XtraDB [\[37\]](#).
 - El motor de almacenamiento Aria tienen un mayor rendimiento al recibir consultas complejas utilizando para ello tablas temporales almacenadas en la memoria caché en lugar de almacenarlas en el disco.
 - Incorpora nuevas tablas para recoger información relativa a las estadísticas de los índices y las tablas, ayudando así a la optimización de la base de datos.
 - En general, muchas mejoras para aumentar el rendimiento y la eficiencia con respecto a MySQL.

- **Desventajas de MariaDB**
 - La principal desventaja de utilizar MariaDB aparece si estamos utilizando MySQL, ya que deberíamos de considerar si es necesario o beneficioso realizar este cambio, en el caso de que nuestro sistema no sea de gran tamaño la respuesta puede ser más clara hacia la migración, en cambio si nuestro sistema es muy grande tendría que valorar muchos más aspectos para valorar si es verdaderamente beneficioso cambiar a MariaDB.

2.7.4. PHP



Ilustración 2.6 – PHP – Página Oficial
PHP (<http://php.net/download-logos.php>)

PHP [15], Hypertext Preprocessor (preprocesador de hipertexto), es un lenguaje de programación de propósito general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación en el lado del servidor que en lugar de llamar a un archivo externo para procesar los datos, era capaz de incorporarse en un archivo HTML [38]. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera el HTML resultante.

PHP se ha ido desarrollando e incorporando nuevas funcionalidades, pudiendo ser utilizado en la mayor parte de los sistemas operativos, plataformas y servidores web actuales, incluso incorporando su interfaz de línea de comandos en aplicaciones gráficas, todo ello sin coste alguno.

Visión General

En la actualidad PHP es uno de los lenguajes de programación más populares y que más se utilizan para el desarrollo de sitios web con mas de 20 millones y de servidores con mas de un millón, si bien es cierto, que en la actualidad se está viendo desplazado por la aparición de nuevas tecnologías como Node.JS [39] o ASP.NET [40]. A pesar de ello al ser el modulo más popular de Apache y ser este, como hemos mencionado en pasados apartados, el servidor web por excelencia hace que PHP se mantenga en el tiempo.

Una de las causas por la que PHP se ha extendido muy rápidamente entre los desarrolladores es el ser muy similar a lenguajes de programación como C y Perl, lo que conlleva un periodo de aprendizaje muy corto para aquellos desarrolladores que ya dominaban estos lenguajes, pudiendo realizar aplicaciones con contenido dinámico sin necesidad de dominar todos las nuevas funciones de este lenguaje.

El funcionamiento de PHP dentro de un servidor web lo entenderemos mejor de forma esquemática:

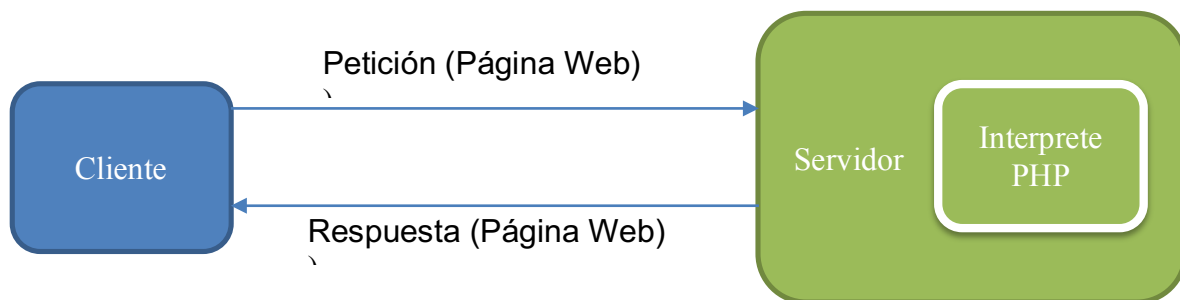


Ilustración 2.7 – Esquema PHP y Servicio Web

El cliente solicita al servidor web una página, esta petición se ejecuta en el intérprete PHP del servidor. Este genera una respuesta a la petición del cliente y la envía de nuevo al servidor para ser el servidor el que devuelva esta respuesta al cliente.

No es una característica de PHP pero es importante mencionar la aparición del modelo vista controlador (MVC), su funcionamiento consiste en separar el tratamiento de los datos y el acceso a los mismos, la lógica de control y la interfaz de usuario en tres elementos independientes.

Características de PHP

- Está orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos.
- Admite la conexión a distintos servidores de bases de datos tanto SQL con NoSQL.
- Puede generar archivos en distintos formatos (PDF, Flash) con el uso de extensiones.
- Puede ejecutarse en los sistemas operativos más usados (Linux, Mac OS X y Microsoft Windows)
- Fácil Aprendizaje.
- El servidor se encarga de enviar el resultado HTML al navegador web por lo que el código PHP es invisible tanto para el cliente como para el navegador web.

- Integración con los motores de bases de datos actuales como MySQL.
- Expandible mediante módulos denominados extensiones.
- Gran soporte de la comunidad y documentación muy completa, detallando todas sus funcionalidad mediante ejemplos.
- Al ser software libre puede ser utilizado por todo tipo de usuarios que lo desee.
- Permite aplicar técnicas de programación orientada a objetos.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- Manejo de excepciones (a partir de PHP5).
- Libertad en el código. PHP no obliga a seguir una metodología específica cuando se desarrolla una aplicación pudiendo cada programador aplicar sus propias técnicas para estructurar y ordenar su código.

Inconvenientes de PHP

- Código no oculto. Al ser un lenguaje que se interpreta durante la ejecución puede ser desfavorable el no poder ocultar el código.
- Más lento. Esto se debe a la interpretación del código cuyo tiempo es mayor que si utilizaremos un lenguaje de bajo nivel. Este inconveniente puede subsanarse con técnicas de caché.
- En versiones pasadas de PHP las variable no tienen tipo lo dificulta a los entornos de desarrollo integrado brindar asistencia para el tipificado de código, estos entornos suelen solventarlo incluyendo comentarios donde se incluye el tipo de las variables.

Para la realización de este proyecto se ha utilizado la versión PHP 7.2.5.

2.7.5. phpMyAdmin

phpMyAdmin es una herramienta desarrollada en PHP cuyo objetivo es manejar el gestor de base de datos MySQL a través de páginas webs. PhpMyAdmin puede realizar todas las tareas para la gestión de una bases de datos como es la creación, borrado, editado y actualización de tablas, también posibilita la realización de consultas en lengua SQL, administrar claves, privilegios, exportar datos, etc. Se encuentra disponible bajo la licencia GPL.

phpMyAdmin es la herramienta que incorpora XAMPP y que utilizaremos como nexo de unión entre el lenguaje de programación que vamos a utilizar y el sistema de gestión de base de datos MariaDB.

2.8. Diferentes tecnologías para realizar Servicios Web

En cuanto a tecnologías se refiere, hay dos esquemas relevantes, los cuales serán explicados a continuación. Estos son Soap y Rest.



Ilustración 2.8 - SOAP vs REST - <http://testingalert.com/difference-rest-soap/>

2.8.1. SOAP

Es el protocolo que utiliza la tecnología comúnmente denominada Servicio Web. En sus inicios todo el mundo utilizaba SOAP a pesar de no soportar casi ningún escenario avanzado y ser un protocolo complejo. Su idea principal era dar solución a cualquier problema en el ámbito de las comunicaciones, incluyendo mensajería asegurada, seguridad avanzada, etc. A medida que fueron evolucionando los Servicios Web estas soluciones se volvían mas engorrosas por lo que su utilización era cada vez más compleja.

El funcionamiento de SOAP se basaba en un paquete(SOAP envelope) compuesto por la información relativa al consumidor del servicio(receptor), el mensaje que se envía y por la respuesta.

Si este mensaje lo queremos enviar por internet tendremos que nuevamente “envolverlo” pero ahora utilizando el protocolo HTTP/HTTPS.

SOAP ha conseguido obtener múltiples capacidades las cuales la mayoría no se utilizan.

2.8.2. REST

REST [\[16\]](#) (Representational State Transfer- Transferencia de Estado Representacional) es un nuevo planteamiento para desarrollar servicios web y proyectos definido por Roy Fielding (impulsor de la especificación HTTP) un experto en la Arquitectura de Redes. En la actualidad las APIs REST son uno de los pilares fundamentales en el desarrollo de los servicios de las aplicaciones.

REST es cualquier interfaz entre sistemas que use HTTP para obtener datos o generar operaciones sobre esos datos en todos los formatos posibles, como XML y JSON.

Actualmente REST es el estándar mas utilizado en la creación de APIs entre otras virtudes por su eficiencia, la mayoría de proyectos utilizan estas APIs REST para la creación y utilización de sus servicios. Grandes compañías como son Facebook o Twitter sacan el máximo partido a sus datos con el uso de esta tecnología.

REST es la alternativa principal a protocolos como SOAP (mencionada anteriormente) se diferencia de esta en su complejidad, REST es mucho mas sencilla que SOAP, otro aspecto diferencial es la disminución de los tiempos de consumo y respuesta, SOAP realiza doble empaquetamiento con lo que los tiempos aumentan, sin embargo REST mediante la supresión de la “envoltura” de SOAP y de la capa XML consigue disminuir los tiempos de consumo y respuesta.

Características de REST

- **Protocolo cliente/servidor sin estado.** Ni el cliente ni el servidor requieren guardar ningún estado previo para poder ejecutar una petición HTTP, ya que esta no necesita información adicional para ser ejecutada. Existe la posibilidad de que algunas aplicaciones utilicen memoria caché para que el cliente pueda ejecutar las mismas respuestas para peticiones iguales, aunque esto no significa que esto sea lo más recomendable.
- **Existen cuatro operaciones relacionadas con los datos.** Estas operaciones son POST para crear nuevos datos, GET para leer y consultar datos existentes, PUT para editar datos y DELETE para eliminar datos.
- **REST maneja los objetos mediante la URI.** La URI es el identificador único de cada elemento en un sistema REST, nos permite acceder a la información de ese elemento para realizar alguna de las operaciones mencionadas en el punto anterior.
- **Interfaz uniforme.** Al realizar la transferencia de datos mediante operaciones concretas, sobre recursos y mediante una URI, permite tener una interfaz uniforme.
- **Organizado en capas.** REST se organiza de manera jerárquica mediante la utilización de capas, cada capa desempeña una función determinada dentro del sistema.
- **Uso de hipermedios.** Hipermedia es una extensión del término hipertexto, en el caso de REST este concepto se utiliza para explicar la capacidad de una interfaz de desarrollo de aplicaciones de facilitar al cliente y al usuario los enlaces correctos para ejecutar acciones determinadas sobre los datos del sistema.

Ventajas que ofrece REST para el desarrollo

- **Separación entre el cliente y el servidor.** El protocolo REST divide el sistema en tres partes, por un lado la interfaz de usuario, por otro el servidor y por otro el almacenamiento de datos, lo que proporciona notables ventajas en el desarrollo principalmente por que las distintas partes pueden desarrollarse de manera independiente, de la misma manera también mejora la portabilidad de la interfaz y la escalabilidad.
- **Visibilidad, fiabilidad y escalabilidad.** La mayor ventaja que tiene separar el cliente y el servidor tiene la ventaja de que el producto puede ser escalado por un equipo de desarrollo sin muchos inconvenientes. Se pueden realizar cambios en el cliente o en el servidor siempre y cuando los datos que se envíen y se reciban lo realicen de forma correcta.
- **La API REST siempre es independiente del tipo de plataformas o lenguajes.** REST permite cambiar a nuevos entornos de desarrollo ya que se adapta fácilmente a la sintaxis o plataformas que se utilicen. REST permite tener servidores PHP, Java o Node.js. Estos cambios se pueden realizar siempre que las respuestas se realicen en XML o JSON (lo explicaremos en profundidad en el Capítulo 5 - Implementación).

Una vez estudiadas ambas tecnologías podemos observar como la tecnología REST se impone en la actualidad a SOAP, al aportar mejoras en los tiempos de respuesta y de consumo, organizarse en capas independientes mejorando el rendimiento y poder realizar operaciones sobre los datos accediendo a ellos mediante URIs por todo esto he decidido usar esta tecnología para la realización de mi aplicación.

CAPÍTULO 3



ANDROID

En el presente capítulo se procederá a la explicación detallada del sistema operativo en el que se va a realizar el proyecto analizando sus características y arquitectura, también se realizará una comparativa con otros sistemas operativos y se argumentará el por que elegir este sistema operativo en lugar de cualquier otro.

3.1. ¿Qué es Android?

Android [17] es un sistema operativo que se basa en el núcleo Linux. Inicialmente fue diseñado para dispositivos móviles con pantalla táctil, como es el caso de los smartphone y las tabletas pero con el paso del tiempo y el avance de la tecnología este sistema operativo ha evolucionado y se ha incorporado a televisores inteligentes (Smart TV), relojes (Smart Watch) e incluso en el ámbito de la automatización con coches que incorporan este sistema. Android fue desarrollado por Android Inc, una empresa financiada por Google, esta ultima la acabó adquiriendo en 2005 por lo que en la actualidad Android es propiedad de Google.



Ilustración 3.1 – Android –
Página Oficial de
desarrolladores Android
(<https://developer.android.com/?hl=es-419>)

En 2007, a la par que Open Handset Alliance [41] se presentó Android, aunque no fue hasta 2008 cuando se empezó a comercializar el primer móvil que utilizaba este sistema operativo, fue un dispositivo de la compañía HTC, el HTC Dream.

Existen otros sistema operativos orientados a dispositivos móviles, como veremos en este capítulo, sin embargo Android es el más popular del mundo superando el 80% de cuota de mercado. Esto se debe, en parte, a ser software libre, ya que su núcleo es Linux con licencia GPL, y a ser de código abierto, es decir, que los desarrolladores pueden acceder al código fuente de Android. Por esto las gran mayoría de fabricantes de dispositivos móviles eligen Android para sus dispositivos.

Como consecuencia de ser el sistema operativo más usado posee una comunidad de usuarios y de desarrolladores muy amplia y que va *in crescendo*, superado a principios de este año 2018 las dos millones de aplicaciones en su tienda

oficial(Google Play). Estas aplicaciones también se pueden adquirir de forma externa a la tienda oficial.

3.2. Características de Android

En este apartado vamos a ver las características más relevantes que posee Android.

- **Diseño de dispositivos.** Android se puede utilizar en dispositivos con distintos tamaños y distintas resoluciones de pantalla.
- **Almacenamiento.** Para el almacenamiento de datos utiliza una versión reducida de MySQL, SQLite, es una base de datos de menor tamaño optimizada para dispositivos móviles.
- **Conectividad.** Android soporta diferentes tecnologías de conectividad:
 - GSM/EDGE, GSM. Sistema global para las comunicaciones móviles, mientras que EDGE es la tecnología que se utiliza para las redes 2G y 3G.
 - IDEN. Tecnología de comunicaciones que permite la interconexión telefónica, la transmisión de datos, el envío de mensajes, etc.
 - CDMA. Protocolo de comunicación alternativo a GSM.
 - EV-DO, es un estándar de telecomunicaciones destinado a la transmisión de datos inalámbrica a través de las redes de telefonía móvil.
 - UMTS. Tecnología que proporciona una alta velocidad de acceso a internet, una gran capacidad multimedia y una transmisión de voz de calidad.
 - Bluetooth. Permite la comunicación entre dispositivos mediante radiofrecuencia.
 - Wi-Fi. Permite la interconexión inalámbrica.
 - LTE. Estándar para la transmisión de datos de alta velocidad.
 - HSDPA/HSPA. Mejora de la tecnología UMTS.
 - NFC. Es una tecnología que permite la comunicación entre dispositivos que se encuentren próximos.

- **Mensajería.** Android incorpora tecnología que permiten la mensajería de texto con SMS y MMS. En la actualidad también incorpora un sistema de mensajería en la nube denominado Firebase Cloud Messaging.
- **Navegador Web.** Android incorpora un navegador web con el cual podremos consultar internet siempre que tengamos acceso a una red.
- **Soporte de Java.** Java es el lenguaje que predomina en las aplicaciones Android, estas aplicaciones se ejecutan en un entorno de ejecución de aplicaciones que incorpora Android denominado Android Runtime(ART).
- **Soporte multimedia.** Android soporta los siguientes formatos multimedia:
 - WebM, es un formato de video de código abierto.
 - H.263 y H.264. Estándares para la compresión de video.
 - MPEG-4 SP. Técnica de compresión de video.
 - AAC, HE-AAC, Ogg Vorbis, AMR y MP3. Formatos de compresión de audio.
 - MIDI. Estándar que permite a los instrumentos musicales electrónicos puedan comunicarse.
 - WAV. Formato de audio sin compresión.
 - JPEG
 - PNG y JPEG. Formatos de compresión de imagen.
 - GIF. Formato para imágenes y animaciones.
 - BMP. Formato para imágenes de mapa de bits.
- **Soporte para streaming.** Android soporta la reproducción de video en directo con el uso de tecnología como Adobe Flash Streaming (RTMP).
- **Soporte para hardware adicional.** Android soporta el acoplamiento de diferentes hardware que se han convertido indispensables en la actualidad como son las pantallas táctiles, las cámaras, acelerómetros, sensores de proximidad, termómetros, GPS, etc.
- **Entorno de desarrollo.** El entorno de desarrollo integrado (IDE) de Android comenzó siendo Eclipse, posteriormente aparición una herramienta específica para el desarrollo de aplicaciones Android, Android Studio.

- **Google Play.** Como hemos mencionado con anterioridad, Google Play es la tienda oficial de Android donde podemos encontrar todas las aplicaciones disponibles para este sistema operativo. Se accede a ella directamente desde el dispositivo móvil.
- **Multi-táctil.** Android tiene soporte nativo para la tecnología multi-táctil, esta tecnología consiste en que podemos tocar la pantalla táctil de nuestro dispositivo en varias posiciones a la vez y el dispositivo será capaz de interpretarlo y responder de manera adecuada.
- **Videollamada.** Para poder realizar video llamadas Android utiliza una aplicación denominada Hangouts, lo que nos permite realizar una conversación con otro dispositivo incorporando señal de video.
- **Multitarea.** La multitarea consiste en que varias aplicaciones se están ejecutando al mismo tiempo aunque estas no se encuentren en primer plano, esta es una funcionalidad que incorpora Android.
- **Características basadas en voz.** Android permite la utilización de voz para utilizar los dispositivos desde su comienzo utilizando Google Now, en 2016 esta tecnología se mejoró y dio paso a Google Assistant que es el que se utiliza en la actualidad.
- **Tethering.** Es una tecnología que nos permite usar un dispositivo Android como punto de acceso alámbrico o inalámbrico pudiendo conectarse a otros dispositivos móviles o computadores.

3.3. Arquitectura de Android

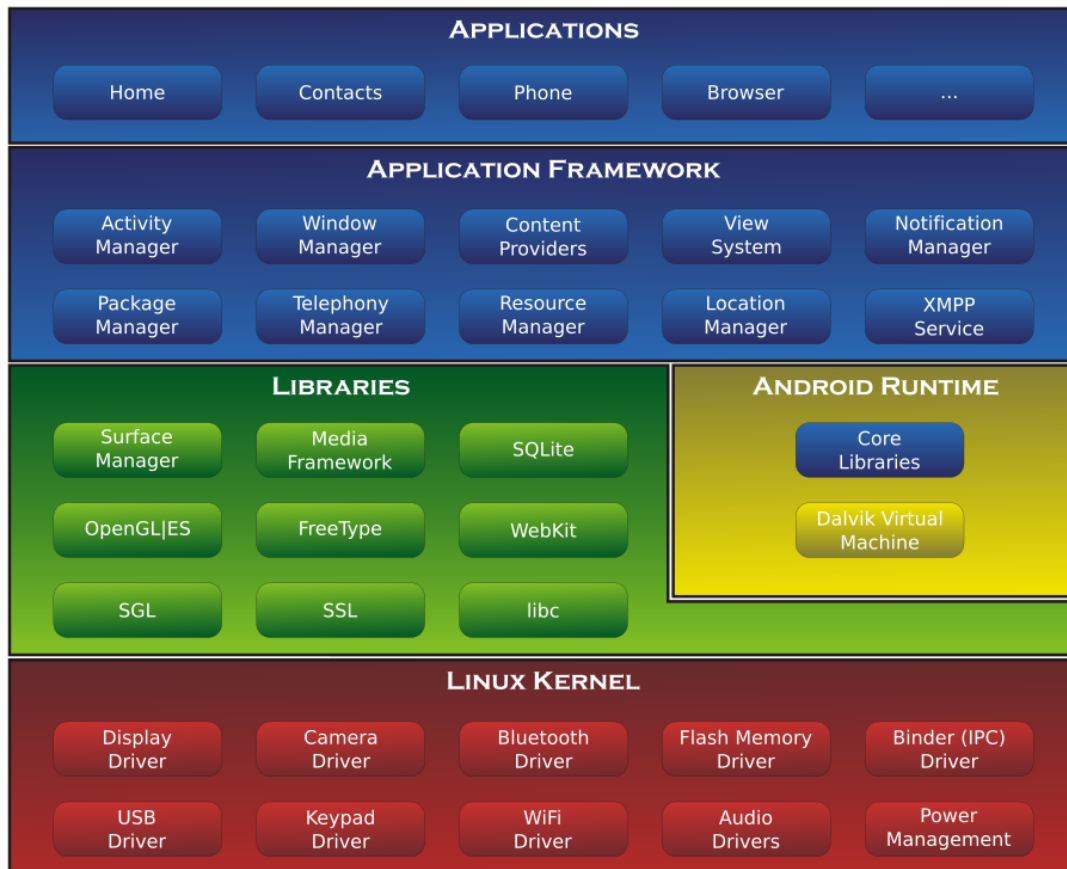


Ilustración 3.2 - Arquitectura Android - <https://commons.wikimedia.org/wiki/File:Android-System-Architecture.svg>

Los componentes principales del sistema operativo de Android [18]:

- **Aplicaciones:** En el nivel más alto de la arquitectura Android encontramos las aplicaciones, Android trae preinstaladas una serie de aplicaciones como pueden ser el correo o los contactos. A demás como ya hemos mencionado anteriormente se pueden instalar nuevas. Estas aplicaciones se desarrollaran en el lenguaje de programación Java.
- **Framework de aplicaciones:** Es el marco de aplicaciones Android, como hemos visto, Android es un sistema operativo de código abierto con lo que los desarrolladores tendrán acceso a las mismas APIs que utilizan las aplicaciones preinstaladas. Las APIs más relevantes de Android son
 - **Activity Manager:** APIs encargadas de gestionar el ciclo de vida de las aplicaciones.

- **Windows Manager** : Encargada de gestionar las distintas pantallas o ventanas de aplicaciones que se están ejecutando.
- **Telephone Manager** : APIs encargadas de las funciones propias de telefonía móvil como son la gestión de llamadas, de mensajes, etc.
- **Content Provider** : Es la API que permita que las aplicaciones puedan compartir datos entre sí como por ejemplo una aplicación que quiera acceder a la cámara, al micrófono, consultar los contactos, etc.
- **View System**: Es la API con la que se controle la interfaz gráfica de las aplicaciones Android, es decir, todo lo que queda a la vista del usuario como son los botones, los campos de texto, listas, teclados, etc.
- **Location Manager** : Con esta API podemos acceder a la geolocalización de nuestro dispositivo con el uso del GPS y facilitar esta información a las aplicaciones.
- **Notification Manager** : Cada vez más las aplicaciones se comunican con el usuario sin que este las esté utilizando. para ello emplean las denominadas notificaciones, que son eventos que se muestran al usuario informándole de distintas situaciones como que la batería es baja, que tiene un mensaje nuevo, etc.
- **XMPP Service**: Nos permiten usar el protocolo de intercambio de mensajes basado en XML.
- **Bibliotecas**: Hay elementos del sistema Android que necesitan una serie de bibliotecas para su correcto funcionamiento, estas bibliotecas están desarrolladas en los lenguajes de programación C y C++. Estas bibliotecas están a disposición de los desarrolladores a través del Framework de Android.
- **Runtime de Android(ART)**: Es el entorno de ejecución de aplicaciones que utiliza Android, este incluye una serie de bibliotecas con las funciones específicas del lenguaje de programación Java. Android Runtime desplaza a Dalvik que era la máquina virtual que utilizó Android en un principio para ejecutar las aplicaciones. La diferencia principal entre

Dalvik y ART es que Dalvik compilaba las aplicaciones cuando se utilizaban mientras que ART realiza una compilación previa. ART tienen un mejor rendimiento sin embargo el tiempo de instalación con respecto a Dalvik es mayor y el espacio que ocupa la aplicación es mayor.

- **Núcleo Linux:** Los sistemas básicos del sistema están desarrollados bajo un sistema Linux, este se encarga de la seguridad, administración de memoria y procesos y resto de funciones básicas del sistema operativo. Además el núcleo actúa como capa de abstracción entre hardware y el software.

3.4. Comparativa con los sistemas operativos móviles más usados

A parte de Android existen otros sistemas operativos para dispositivos móviles [\[19\]](#).

Algunos de los sistemas operativos móviles más populares fueron Symbian, BlackBerry, Debian, etc. Aunque en la actualidad los tres sistemas por excelencia son Android, iOS y Windows Phone, este último en menor medida.

- **iOS.** Es propiedad de Apple Inc. Es el sistema operativo que utilizan todos los dispositivos móviles (iPhone y iPad) de esta compañía. Es el segundo sistema operativo más utilizado en el mundo después de Android, esto se debe fundamentalmente a que el precio de los dispositivos Apple tiene un coste mayor y menor variedad que los dispositivos Android, además iOS es un sistema operativo de código cerrado.
- **Windows Mobile o Windows Phone.** Es propiedad de Microsoft, diseñado para smartphones y tablets. Una de las características principales es su integración con el resto de servicios que ofrece Microsoft como son OneDrive, Microsoft Office, etc. a demás de servicios que no son de su propiedad como es el caso de Google. Sin embargo el verdadero objetivo de Windows Phone era integrarse a la perfección con los computadores y tablets que utilizan Windows 10, a pesar de ser estos dispositivos los que monopolizan el

mercado de los computadores, Windows Phone no ha tenido el éxito esperado y en la actualidad cada vez son menos los usuarios de este sistema operativo. Por ultimo mencionar que al igual que el sistema operativo móvil de Apple, es de código cerrado.

A continuación mostraré una tabla comparativa de estos tres sistemas operativos.

Android	iOS	Windows 10 Mobile
Compañía		
Open Handset Alliance	Apple Inc.	Microsoft
Licencia		
Libre y de código abierto, pero por lo general se incluye con aplicaciones y drivers propietarios.	Propietaria excepto para componentes de código abierto.	Propietaria
Programado en		
C, C ++, Java.	C , C ++ , Objective-C , Swift.	.NET C#, VB.NET, Silverlight, native C/C++, WinRTP (XMLA), DirectX
Tienda oficial de aplicaciones		
Google Play	App Store	Windows Store

Android	iOS	Windows 10 Mobile
Coste de desarrollo para el OS móvil		
Gratis	Gratis con Xcode 7	Gratis
Coste para publicar aplicación en la tienda oficial		
US\$25 una vez por individuo	US\$99 al año	US\$19, una vez por un individuo; y \$99 para una cuenta de la compañía
Motor de navegador web por defecto		
Blink	WebKit	Trident (EdgeHTML después de la versión 10)
Navegadores web disponibles		
Chrome para Android, Opera, Firefox	Safari, Chrome para iOS, Opera Mini, Firefox	Internet Explorer, Opera Mini, UC Browser, Microsoft Edge
Motor de búsqueda de los navegadores		
Muchos (entre ellos Google)	Bing, Google, Yahoo! Search, DuckDuckGo	Muchos (entre ellos Bing)

Android	iOS	Windows 10 Mobile
Voz sobre IP		
Protocolo SIP o software de terceros	FaceTime y software de terceros	8+ Skype
Software de pago con tecnología NFC		
Disponible en cualquier dispositivo compatible con el hardware. Android Pay para pagos NFC disponible en Play Store.	8+: vía Apple Pay	8+
USB On-The-Go		
3.1 +	No	10+
Reproducción de audio		
AAC LC/LTP 3GPP, HE-AACv1 (AAC+), HE-AACv2 (AAC+ mejorado) AMR-NB, AMR-WB, MP3, MIDI (tipo 0 y 1, versiones DLS 1 y 2), Ogg Vorbis, PCM/WAVE, FLAC, WAVE, Opus	AAC, AAC protegido (del iTunes Store), HE-AAC, MP3, MP3 VBR, Audible (formatos 2, 3, 4, Audible Enhanced Audio, AAX y AAX+), Apple Lossless, AIFF, WAV	MP3, WMA Std 9.2, WMA Pro, FLAC, AMR-NB, AAC-LC, AAC+, eAAC+

Android	iOS	Windows 10 Mobile
Reproducción de vídeo		
H.263, H.264 (hasta Baseline Profile), H.265 HEVC, MPEG-4 SP, DivX, XviD, VP8, VP9	H.264 (hasta High Profile), MPEG-4, M-JPEG	H.263, H.264, WMV, MPEG4, MPEG4 @ HD 720p 30fps, DivX, XviD
Teclado Bluetooth		
2.3+, en versiones anteriores a través de software de terceros	Sí	8.1u2+
Teclado USB		
3.1+	Con el kit de conexión de cámara	10+ algunos dispositivos
Cliente SSH		
Sí	Sí	Sí
VPN		
Sí	Sí	8.1+
Reconocimiento de voz		
Sí	5+ (Siri)	8.1+ Microsoft Cortana

Tabla 3.1 - Tabla comparativa SO móviles

3.5. Por qué elegir Android

No hay un motivo único por el cual he elegido Android como el sistema operativo en el cual desarrollar la aplicación. Son varios los aspectos que han hecho que me decante por este sistema.

En primer lugar y uno de los motivos más importantes la cuota de mercado de Android, al haber muchos mas smartphones que utilizan este sistema operativo hay muchos mas usuarios que lo usan, con lo cual la aplicación estará destinada un mayor numero de usuarios.

Otro aspecto importante es Android Studio, este nos proporciona disponibilidad, es decir, podemos realizar una aplicación Android en cualquier sistema operativo (Mac OS, Windows, Linux) mientras que si elegimos IOS necesitaríamos obligatoriamente Xcode que es el Software de desarrollo específico para las aplicaciones IOS y que solo lo podemos utilizar si tenemos una maquina con sistema operativo Mac OS.

Uno de los motivos de más peso para elegir Android es la comunidad de desarrollo que lo respalda, al ser un sistema operativo como hemos visto anteriormente que utilizan tantos usuarios y que se puede desarrollar en cualquier sistema operativo existe una gran cantidad de desarrolladores que realizan aplicaciones Android con lo que hay mucha información y soporte en internet a cerca de este sistema.

Por ultimo, por la motivación personal que suponía realizar una aplicación en un sistema completamente nuevo. Durante el grado he realizado la asignatura “Desarrollo de aplicaciones móviles” pero en dicha asignatura las practicas y el proyecto final de las mismas se realiza sobre el sistema operativo IOS utilizando su lenguaje propio(Swift) con lo que me parecía una oportunidad para conocer un sistema nuevo, un software de desarrollo nuevo como es Android Studio y conocer otro apartado más de Java.

CAPÍTULO 4

ANÁLISIS DEL PROYECTO

Una vez introducidos aspectos en la memoria como son la motivación y los objetivos de este proyecto, así como la descripción teórica de aquellas tecnología que vamos a utilizar para la realización del mismo, pasará a definir todos aquellos aspectos relacionados con la Ingeniería del Software del proyecto. Para la realización de este proyecto se han utilizado una serie de técnicas muy conocidas dentro de la Ingeniería del Software como son el Análisis de Requerimientos, el Análisis del sistema mediante la utilización de casos de uso, diagramas de secuencia y escenarios, por ultimo se procederá a la explicación del diseño de la base de datos mediante el modelo Entidad-Relación y el diseño de la interfaz con el uso de Storyboards.

4.1. Análisis de Requerimientos

Lo primero que tenemos que hacer a la hora de desarrollar un proyecto software es un análisis de Requerimientos, es un proceso en el cual el cliente indica cuales son los servicios que quiere realice el sistema así como las posibles restricciones para el desarrollo del mismo.

Dentro de un proyecto software podemos distinguir dos tipos de requerimientos:

- **Requerimientos funcionales:** Describen los servicios que prestará el sistema.
- **Requerimientos no funcionales:** Requisitos a tener en cuenta para que la aplicación pueda llevarse a cabo en todos sus factores y cubriendo sus objetivos. No se refieren explícitamente a las funciones específicas del sistema sino a sus propiedades (rendimientos, seguridad, etc).

Requerimientos funcionales

El marco de trabajo en el que nos encontramos es el de un gimnasio que ofrece diferentes clases a sus clientes y el propietario de este gimnasio quiere que estos clientes se puedan apuntar a dichas clases a través de la aplicación móvil para de esta manera llevar un control más exhaustivo de las clases a la par que poder

gestionar tanto los clientes como el personal (Monitores) que forman parte del gimnasio.

Después de realizar un estudio exhaustivo del gimnasio analizando tanto la forma de trabajo del personal como la comportamiento de los clientes hacia las clases, se han definido las funcionalidades que se consideran necesarias para un correcto funcionamiento de la aplicación. Mencionar que habrá distintos roles dentro de la aplicación que se diferenciarán mas adelante.

- Iniciar sesión
- Listar los clientes
- Listar los monitores
- Listar las clases
- Crear cliente
- Crear monitores
- Ver detalles de una clase
- Ver un cliente
- Ver un monitor
- Crear clase
- Eliminar un cliente
- Eliminar un monitor
- Inscribirse a diferentes clases
- Ver clases asignadas

Esta sería la definición de las funcionalidades que debe realizar el sistema, a continuación detallaremos cada una de ellas diferenciando cada una en función del tipo de usuario que acceda al sistema. Encontramos tres tipos de usuarios, Propietario del gimnasio, personal del gimnasio, de aquí en adelante nos referimos a este tipo de usuario como Monitores, y Clientes.

- **Para todos los usuarios.**
 - Iniciar sesión. Nos encontramos ante un sistema que ofrece distintas funcionalidades en función del tipo de usuario, por lo tanto es necesario

inicio de sesión para gestionar los servicios que se le ofrecen a cada tipo de usuario.

- **Para el propietario del gimnasio.** Funcionalidades que el propietario del gimnasio puede realizar.
 - Listar los clientes. Poder ver todos los usuarios del gimnasio que están dados de alta en el sistema.
 - Crear cliente. Dar de alta a un nuevo cliente del gimnasio en el sistema asignándole un identificador de usuario(DNI) y una contraseña.
 - Listar los monitores: Poder ver los monitores que tiene el gimnasio en el sistema.
 - Crear monitor. Dar de alta un nuevo monitor asignándole un identificador de usuario(DNI) y una contraseña.
 - Listar las clases. Ver todas las clases que hay en el gimnasio.
 - Ver detalles de una clase. Poder ver todo lo referente a una clase; monitor asignado, tipo de clase, fecha, hora, capacidad y usuarios que están apuntados a la misma.
 - Crear un clase. Crear una nueva clase que se vaya a realizar en el gimnasio, asignándole el monitor y definiendo el tipo, la fecha, la hora y la capacidad.
 - Eliminar un cliente. Dar de baja a un cliente del sistema.
 - Eliminar un monitor. Dar de baja a un monitor del sistema.

- **Para los monitores.** Funcionalidades que los monitores del gimnasio tiene disponibles dentro de la aplicación.
 - Lista de clases asignadas. Listado con las clases que tienen que realizar.
 - Ver detalles de una clase: Poder ver todo lo referente a una clase; tipo de clase, fecha, hora, capacidad y usuarios que están apuntados a la misma.

- **Para los clientes del gimnasio.** Funcionalidades que los clientes que estén registrados en la aplicación pueden realizar.
 - Lista de clases. Listado de las clases que hay disponibles en el gimnasio.
 - Ver detalles de una clase. Poder ver los detalles de una clase; monitor asignado, tipo de clase, fecha, hora y capacidad.
 - Apuntarse a una clase. Apuntarse a una clase que quiera realizar.
 - Lista de clases a las que se ha apuntado. Listado con las próximas clases a las que se ha apuntado ese cliente.

Requerimientos no funcionales

- Todos los usuarios de la aplicación necesitarán un móvil con sistema operativo Android en su versión Nougat 7.1.1 o superior API Level 25 [20]. Además de conexión a internet para poder utilizarla.
- El propietario puede acceder a todas las funcionalidades de la aplicación.
- Solo el propietario podrá gestionar los monitores y los clientes del sistema.
- Los clientes y los monitores no podrán acceder a la parte administrativa de la aplicación.
- Los clientes solo podrán apuntarse a aquellas clases en las que queden plazas libres.
- Los clientes no podrán apuntarse dos veces a la misma clase.
- Los monitores no podrán modificar la fecha y la hora de las clases que se le han asignado.

Requerimientos de la interfaz para la aplicación móvil

Uno de los aspectos más importante a la hora de realizar una aplicación móvil es la interfaz, ya que es la manera que tiene el usuario de interactuar con la aplicación y de esta manera hacer uso de todos los servicios que la aplicación le ofrece.

El objetivo de una buena interfaz de usuario será siempre el correcto funcionamiento de la misma, su capacidad para cumplir con lo que promete y para ayudar al usuario a encontrar lo que está buscando.

Para que un sistema interactivo cumpla con sus objetivos tiene que ser usable y accesible para la mayor parte de la población humana.

La usabilidad es la medida en la que un producto puede ser usado por determinados usuarios para conseguir unos objetivos específicos con efectividad, eficiencia y satisfacción en un contexto de uso dado. Los principios de usabilidad son los siguientes:

- Facilidad de aprendizaje. Es el tiempo mínimo necesario para que el usuario pueda utilizar un sistema de manera productiva. El objetivo de todo producto software tiene que ser minimizar este tiempo.
- Flexibilidad. Diversas maneras de intercambio de información entre el usuario y el sistema.
- Consistencia. Un sistema es consistente si los elementos que se utilizan son siempre empleados de la misma manera.
- Robustez. Implica que el sistema pueda realizar diversos procesos sin generar errores o bloqueos.
- Recuperabilidad. Grado en el que el sistema permite mantener y preservar los datos cuando se produce un fallo.
- Tiempo de respuesta. Tiempo que emplea el sistema en mostrar un cambio de estado al usuario.
- Adecuación de las tareas. Distintas formas en que el sistema permite al usuario realizar una acción y la forma que quiere realizarla.
- Disminución de la carga cognitiva. El usuario no tiene que recordar como utilizar la aplicación sino que realice las acciones de manera sistemática e intuitiva.

El diseño de sistemas interactivos implica realizar un diseño pensando en el usuario ya que el proyecto girará entorno a él.

4.2. Análisis del sistema

Una vez realizado un estudio de los requerimientos que son necesarios para realizar nuestro proyecto pasaremos a realizar un Análisis del Sistema, en este definiremos un modelo del proyecto lo mas integro posible, detallando todos los aspectos necesarios para entender de forma sencilla el funcionamiento de la aplicación.

Para realizar un correcto análisis del sistema tendremos que ir definiendo diferentes etapas como son la definición de los actores, los casos de uso, diagramas de secuencia, escenarios.

4.2.1. Actores

En este apartado explicaremos los diferentes tipos de usuario que podrán utilizar la aplicación en función del rol que desempeñen dentro del gimnasio.

Dentro de nuestro sistema diferenciaríamos los siguientes actores:

Propietario

Es el aquel que tiene conocimiento de todas las clases que se van a realizar en el gimnasio y el encargado de la gestión de los usuarios, de los monitores y de crear nuevas clases.

Monitor

Es aquel que podrá consultar las clases que se le han asignado así como los detalles de las mismas.

Cliente

Es el usuario propiamente dicho del gimnasio, podrá consultar todas las clases que hay disponibles en él, apuntarse a tantas como desee y consultar las clases a las que esta apuntado.

4.2.2. Casos de uso

Un caso de uso es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo algún proceso.

En el contexto de ingeniería del software, un caso de uso describe bajo la forma de acciones y reacciones el comportamiento de un sistema desde la perspectiva del usuario. Nos permite establecer los límites del sistema y las relaciones existentes entre este y el entorno.

Los elementos que conforman un caso de uso son:

- **Actores.** Cualquier elemento externo que interactúe con el sistema(maquina, persona...).
- **Casos de uso.** Acciones que realiza el sistema.
- **Asociaciones.** Marca una interacción entre dos elementos. Existen diferentes tipos:
 - Asociación(normal)
 - Include. Indica que un caso de uso engloba la acción del otro.
 - Extend. Indica que un caso de uso puede llevar a usar el caso extendido.
 - Generalización. Indica que un caso de uso es un caso particular de uno más general.

Una vez definido lo que es un caso de uso e identificado los actores que intervienen en nuestro proyecto podemos definir un diagrama que nos mostrará el funcionamiento general del sistema.

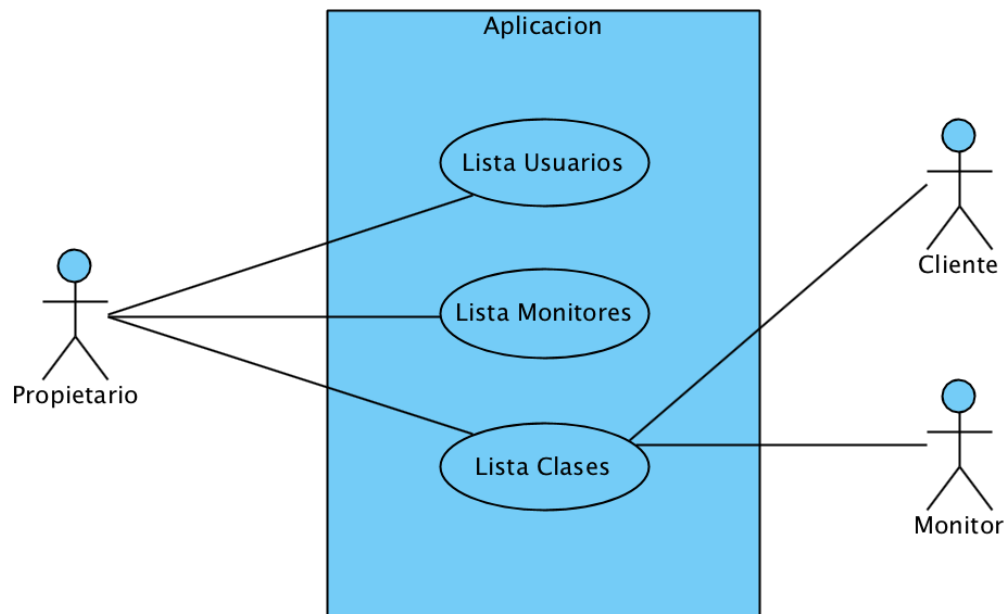


Ilustración 4.1 - Diagrama General del Sistema

Una vez definido este diagrama general procederemos a definir las distintas funcionalidades presentes en el.

Una vez definido esto, vamos a pasar a ver los diferentes Casos de Uso, incluiremos también el diagrama de caso de uso asociado a cada uno.

Caso de Uso 1

El primer caso de uso muestra las acciones que tendrá que realizar cualquier usuario para iniciar sesión en el sistema.

Caso de uso 1: Iniciar sesión	
Actores	Propietario, Monitor, Cliente
Prerrequisitos	Estar ejecutando la aplicación en primer plano y no haber iniciado sesión.
Propósito	
El usuario quiere acceder a los servicios que ofrece la aplicación. Para poder acceder la aplicación solicitará el usuario y la contraseña.	
Resumen	
El usuario introduce su identificador de usuario y la contraseña. El usuario envía los datos. El sistema comprueba que los datos sean correctos para poder acceder a los servicios de la aplicación.	
Curso normal	
1. El usuario introduce su usuario. 2. El usuario introduce su contraseña. 3. El sistema comprueba que los datos son correctos. 4. El sistema nos muestra los diferentes servicios que ofrece la aplicación dependiendo de nuestro rol.	
Curso Alternativo	
5a. Si los datos no son correctos el sistema mostrará un mensaje de error y dará la posibilidad de volver a intentarlo.	

Tabla 4.1 - Caso de Uso 1

Caso de Uso 2

El siguiente caso de uso muestra las distintas funcionalidades referentes a los Clientes que puede realizar el Propietario, así como su diagrama asociado.

Caso de uso 2: Clientes como Propietario	
Actores	Propietario.
Prerrequisitos	Estar ejecutando la aplicación en primer plano y haber iniciado sesión como propietario.
Propósito	
El propietario quiere acceder a un listado con todos los clientes dados de alta en la aplicación o dar de alta a un nuevo cliente.	
Resumen	
La aplicación muestra una lista donde aparezcan todos los clientes dados de alta en la aplicación, en la parte inferior derecha encontrará un opción para acceder a la creación de un nuevo cliente introduciendo su identificador de cliente, su nombre y su contraseña.	
Curso normal	
1. El sistema mostrará una lista con todos los clientes. 2. Seleccionar un cliente para ver sus datos. 2.1. Eliminar el cliente 3. Crear un nuevo cliente. 3.1. Introducir nombre del cliente 3.2. Introducir identificador del cliente. 3.3. Introducir contraseña del cliente. 3.4. Enviar datos.	
Curso Alternativo	
3.4a. El cliente ya existen dentro del sistema.	

Tabla 4.2 - Caso de Uso 2

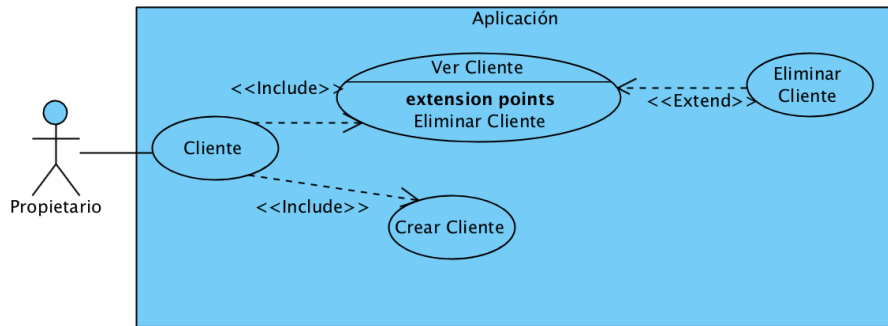


Ilustración 4.2-Diagrama Caso de Uso 2

Caso de Uso 3

El siguiente caso de uso muestra las distintas funcionalidades referentes a los Monitores que puede realizar el Propietario, así como su diagrama asociado.

Caso de uso 3: Monitores como Propietario	
Actores	Propietario.
Prerrequisitos	Estar ejecutando la aplicación en primer plano y haber iniciado sesión como propietario.
Propósito	
El propietario quiere acceder a un listado con todos los monitores dados de alta en la aplicación o dar de alta a un nuevo monitor.	
Resumen	
La aplicación muestra al propietario una lista con todos los monitores dados de alta en la aplicación, en la parte inferior derecha encontrará una opción para acceder a la creación de un nuevo monitor introduciendo su identificador de cliente, su nombre y su contraseña.	
Curso normal	
1. El sistema mostrará una lista con todos los monitores. 2. Seleccionar un monitor para ver sus datos. 2.1. Eliminar el monitor 3. Crear un nuevo monitor. 3.1. Introducir nombre del monitor. 3.2. Introducir identificador del monitor.	

3.3. Introducir contraseña del monitor.
3.4. Enviar datos.
Curso Alternativo
3.4a. El monitor ya existen dentro del sistema.

Tabla 4.3 - Caso de Uso 3

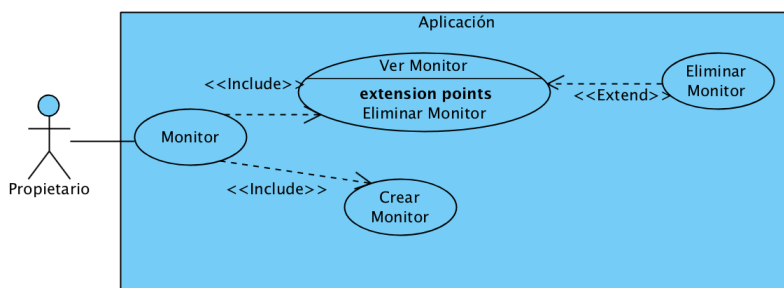


Ilustración 4.3-Diagrama Caso de Uso 3

Caso de Uso 4

El siguiente caso de usa muestras las distintas funcionalidades referentes a la gestión de Clases que puede realizar el Propietario, así como su diagrama asociado.

Caso de uso 4: Clases como Propietario	
Actores	Propietario.
Prerrequisitos	Estar ejecutando la aplicación en primer plano y haber iniciado sesión como propietario.
Propósito	
El propietario quiere acceder a un listado con todas las clases o crear una clase nueva.	
Resumen	
La aplicación muestra al propietario una pantalla donde aparezca una lista con las clases que se han o se van a realizar en el gimnasio, en la parte inferior derecha encontrará una opción para acceder a la creación de una nueva clase introduciendo el tipo de clase, asignándole un monitor, introducir la fecha, la hora y la capacidad.	

Pulsando sobre cada clase podrá ver todos los detalles de una clase(tipo, monitor, fecha, hora, capacidad y clientes apuntados).
Curso normal
1. El sistema mostrará una lista donde aparecerán todas las clases. 2. Acceder a los detalles de una clase. 3. Crear una nueva clase. 3.1. Introducir nombre del monitor. 3.2. Introducir identificador del monitor. 3.3. Introducir contraseña del monitor. 3.4. Enviar datos.
Curso Alternativo
1a. No se ha realizado ninguna clase hasta el momento.

Tabla 4.4 - Caso de Uso 4

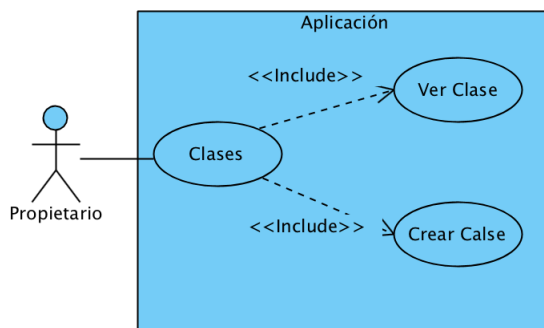


Ilustración 4.4-Diagrama Caso Uso 4

Caso de Uso 5

El siguiente caso de uso muestra las distintas funcionalidades referentes a las Clases que puede realizar un Monitor, así como su diagrama asociado.

Caso de uso 5: Clases como Monitor	
Actores	Monitor.
Prerrequisitos	Estar ejecutando la aplicación en primer plano y haber iniciado sesión como monitor.
Propósito	
Un monitor quiere acceder a un listado con las clases que tiene que realizar.	
Resumen	
La aplicación muestra al monitor una lista donde aparezcan todas las clases que tiene que realizar. Seleccionando cada clase podrá ver todos los detalles de una clase(tipo, monitor, fecha, hora, capacidad y clientes apuntados).	
Curso normal	
1. El sistema mostrará una lista donde aparecerán todas las clases que tiene ese monitor asignadas. 2. Acceder a los detalles de una clase.	
Curso Alternativo	
1a. No tiene ninguna clase asignada.	

Tabla 4.5 - Caso de Uso 5

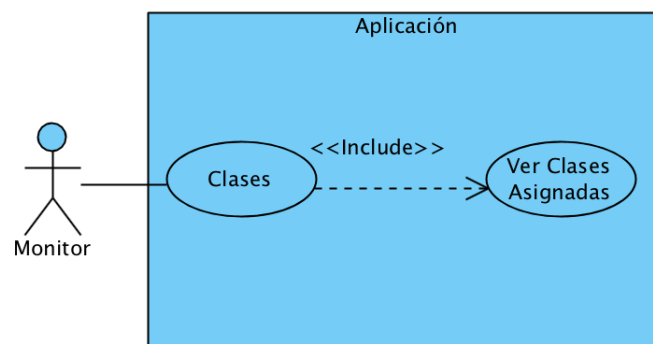


Ilustración 4.5-Diagrama Caso Uso 5

Caso de Uso 6

El siguiente caso de uso muestra las distintas funcionalidades referentes a la gestión de Clases que puede realizar un Cliente, así como su diagrama asociado.

Caso de uso 6: Clases como Cliente	
Actores	Cliente.
Prerrequisitos	Estar ejecutando la aplicación en primer plano y haber iniciado sesión como cliente.
Propósito	
Un cliente quiere acceder al listado de clases que tiene disponibles el gimnasio.	
Resumen	
La aplicación muestra al cliente una lista donde aparezcan todas las clases a las cuales puede apuntarse. Seleccionando una clase podrá ver todos los detalles de una clase(tipo, monitor, fecha, hora, capacidad y clientes apuntados) y apuntarse si lo desea.	
Curso normal	
1. El sistema mostrará dos opciones, “Clases” y “Mis Clases”. 2. Acceder a “Clases”. 2.1. Mostrar un listado con todas las clases. 2.1.1. Seleccionar una clase. 2.1.1.1. Apuntarse a una clase. 3. Acceder a “Mis clases”. 3.1. Mostrar un listado con las clases a las que ese usuario esta apuntado.	
Curso Alternativo	
2.1a. No existe ninguna clase en ese momento. 3.1a. Ese cliente aún no se ha apuntado a ninguna clase.	

Tabla 4.6 - Caso de Uso 6

Diagrama Caso de uso 6

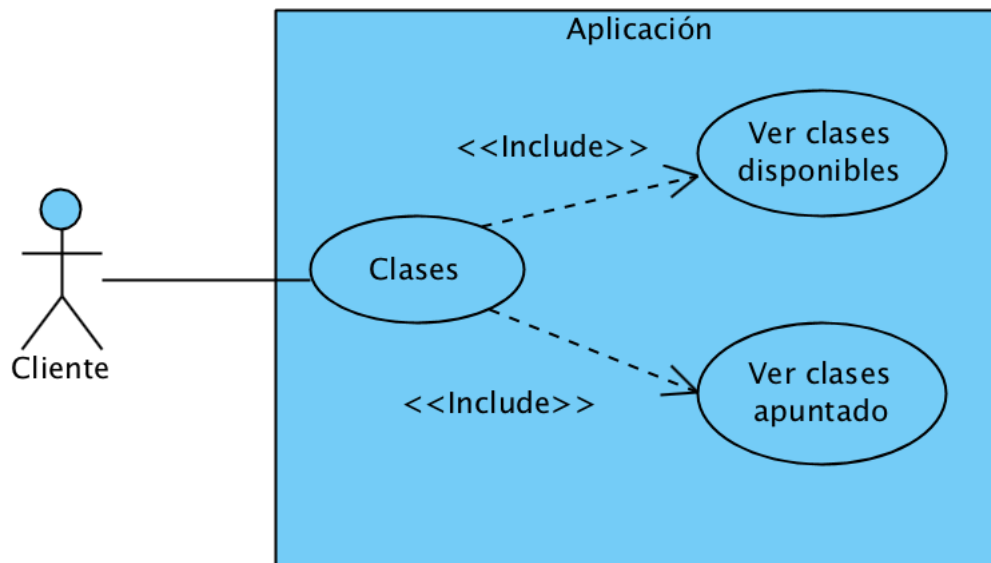


Ilustración 4.6-Diagrama Caso Uso 6

4.2.3. Diagramas de secuencia

El diagrama de secuencia muestra una interacción entre objetos organizados en una secuencia de tiempo. Los diagramas de secuencia pueden representarse con distintos niveles de detalle y para satisfacer distintos objetivos en las diversas etapas del ciclo de vida del desarrollo del sistema. Se suelen utilizar para representar la interacción entre objetos que se produce en un caso de uso.

Un diagrama de secuencia contiene:

- **Actor**

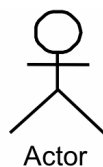


Ilustración 4.7 - Actor

- **Objetos**



Ilustración 4.7 - Objetos

- **Mensajes.** Un mensaje va de un objeto a otro, pasando de la línea de vida de un objeto a la del otro. Un objeto puede enviarse un mensaje a si mismo, es decir, de su línea de vida a su propia línea de vida. Los mensajes más utilizados son:
 - Simple. Es la transferencia del control de un objeto a otro.
 - Síncrono. Es aquel mensaje en el que el objeto espera la respuesta a ese mensaje antes de continuar su trabajo.
 - Asíncrono. Es aquel mensaje en el que el objeto no espera la respuesta a ese mensaje antes de continuar.

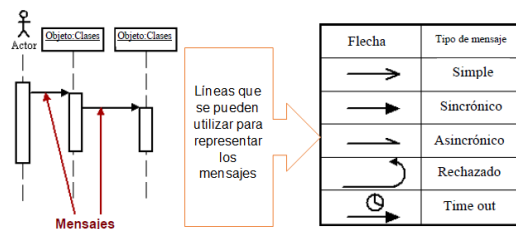


Ilustración 4.7 - Mensajes

- **Tiempo.** El tiempo se indica en la parte superior y avanza hacia la parte inferior. Un mensaje que esté más cerca de la parte superior ocurrirá antes que uno que esté cerca de la parte inferior.

• Representación:

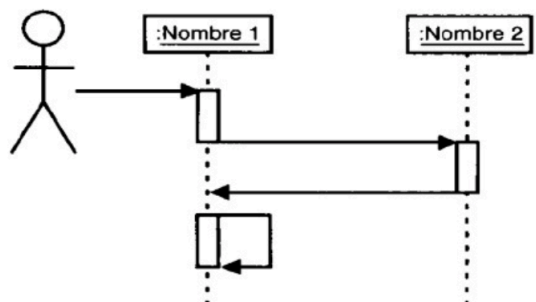


Ilustración 4.8 - Tiempo

Con ello el diagrama de secuencia tiene dos dimensiones. La dimensión horizontal es la disposición de los objetos y los mensajes, y la dimensión vertical muestra el paso del tiempo.

A continuación mostrare los diagramas de secuencia asociados a los casos de uso más relevantes dentro de la aplicación de este proyecto.

- **Iniciar sesión**

En la siguiente ilustración vemos como, una vez introducidos los datos de acceso, estos se comprueban y si son incorrectos nos muestra un error y nos permite volverlos a introducir, mientras que si son correctos, accederemos al sistema.

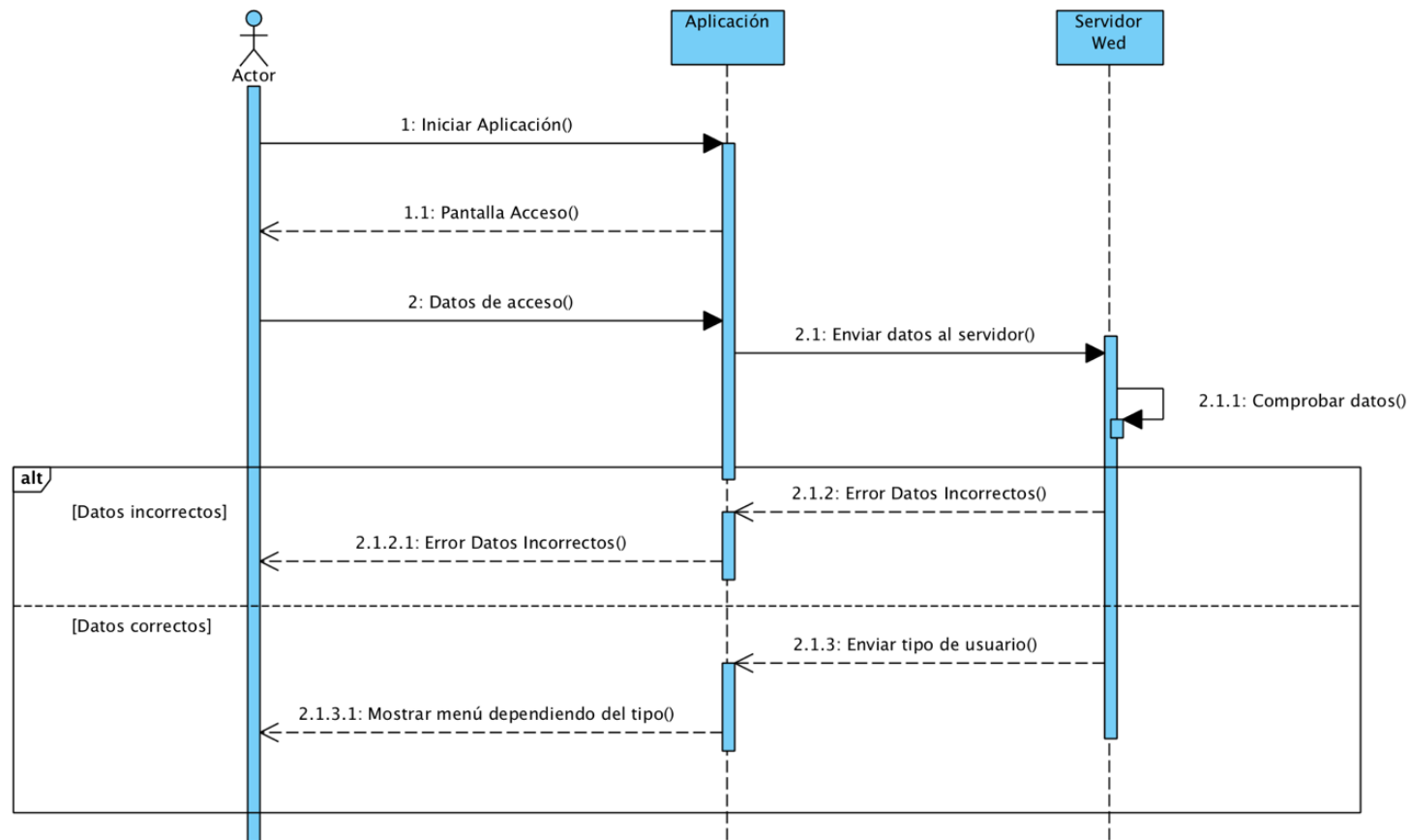


Ilustración 4.9 - Diagrama secuencia Iniciar Sesión

- **Dar de alta a un Cliente o a un Monitor**

En la siguiente ilustración vemos como dar de alta a un nuevo cliente o un nuevo monitor, esta acción dependiendo de si es un cliente o monitor se realizarían en sitios distintos dentro la aplicación pero el diagrama de secuencia para ello seria el mismo. Previamente habría que haber iniciado sesión (como hemos visto en la ilustración anterior) como propietario.

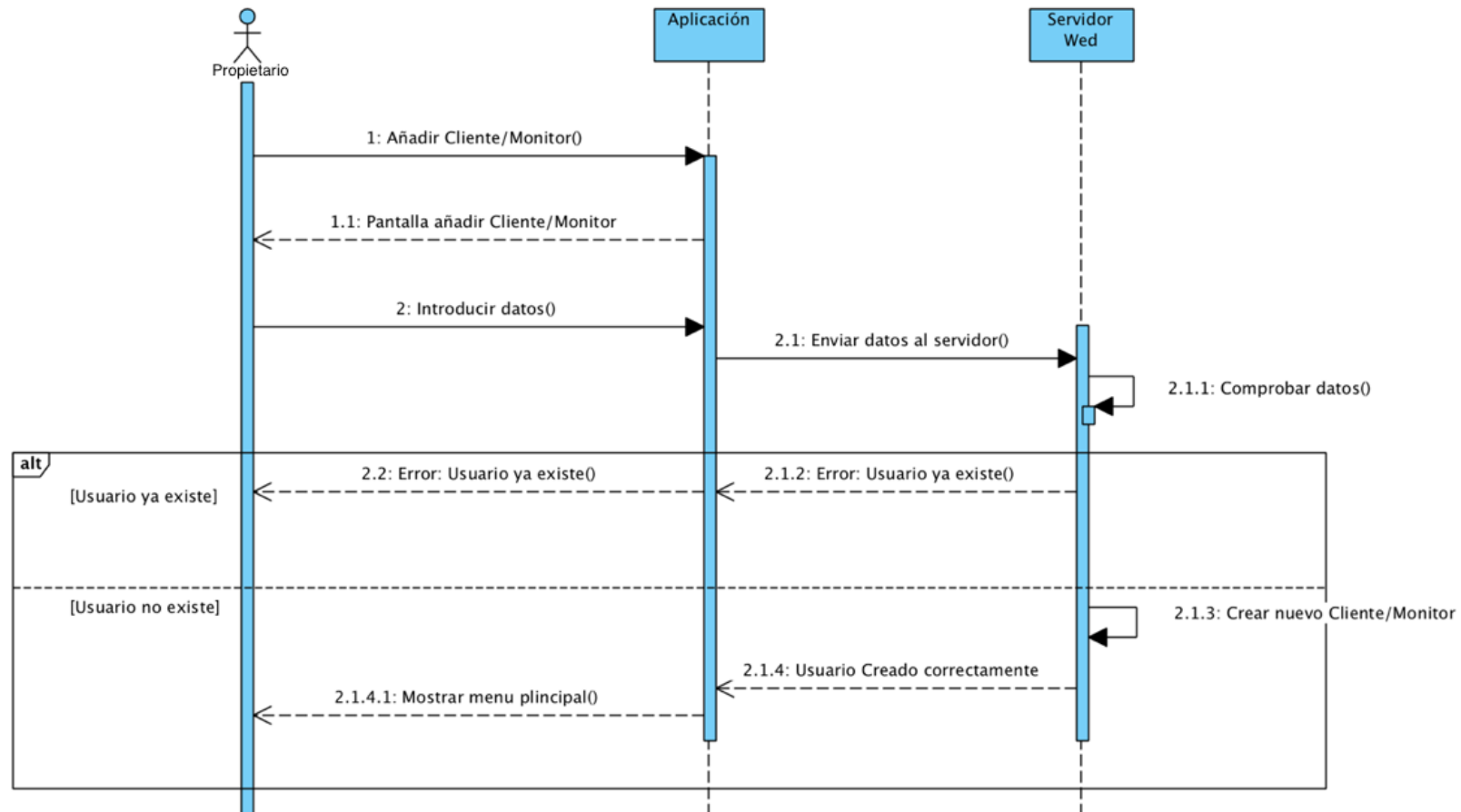


Ilustración 4.10 - Diagrama secuencia Dar de alta aun Cliente o Monitor

- **Apuntarse a una clase**

En la siguiente ilustración vemos como se apuntaría un cliente en alguna de las clases que el gimnasio tiene disponibles. Previamente tendría que haber iniciado sesión como hemos visto en el primer diagrama de secuencia.

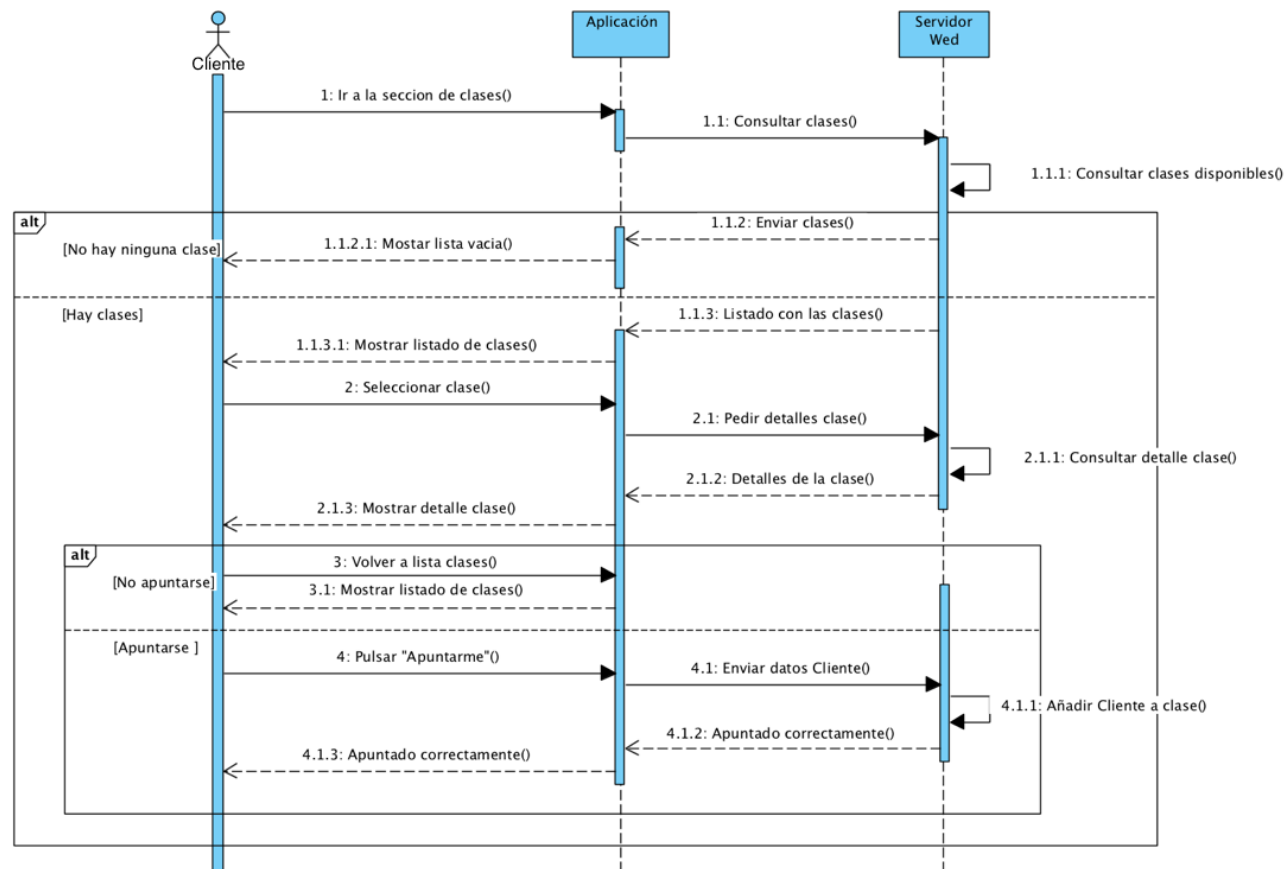


Ilustración 4.11 - Diagrama secuencia Apuntarse a una clase

- **Clases asignadas a un monitor**

En la siguiente ilustración mostraré como un monitor utilizaría la aplicación para ver las clases que tiene que impartir. Previamente tiene que haber iniciado sesión como hemos visto en la primera ilustración.

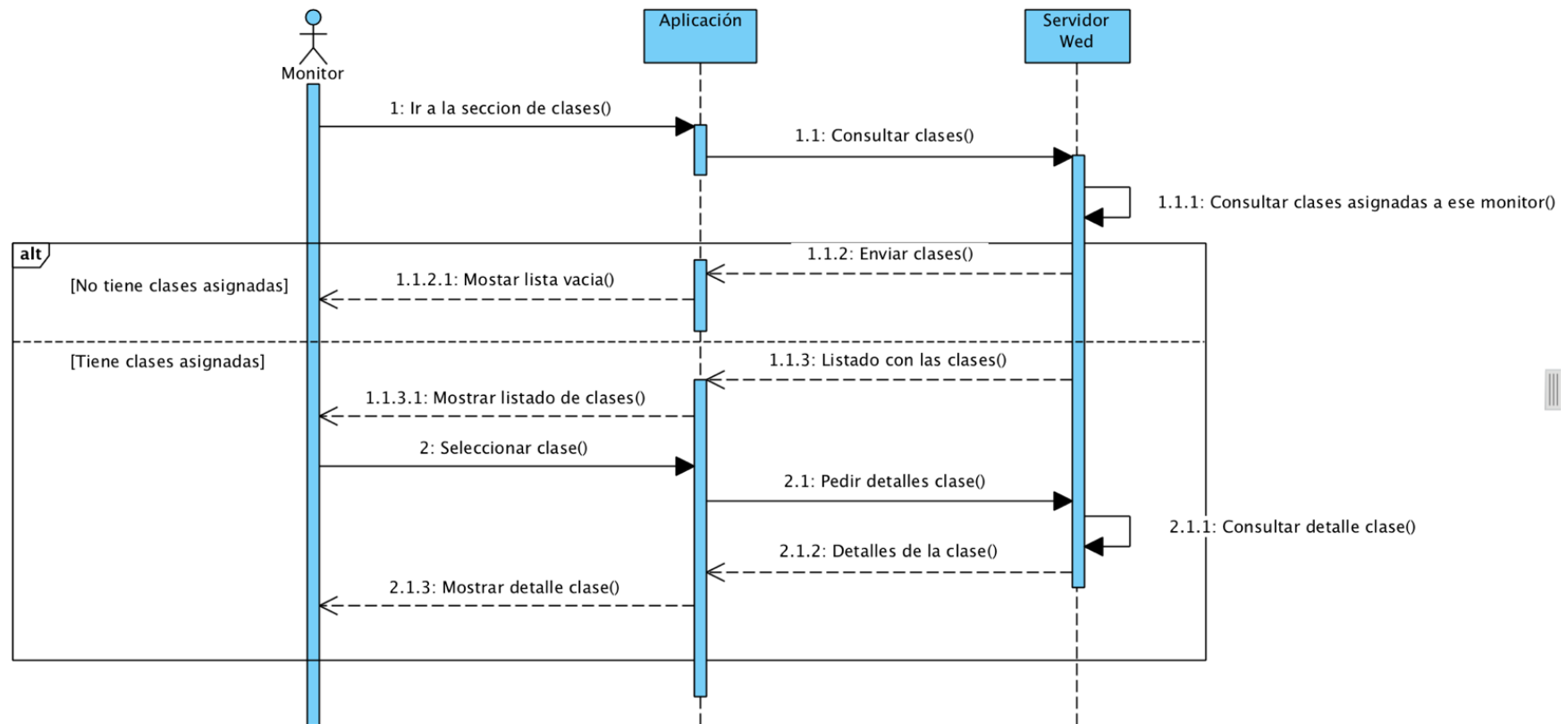


Ilustración 4.12 - Diagrama secuencia Clases asignadas a un monitor

- **Crear nueva clase**

En la siguiente ilustración veremos como tendría que crear un propietario una nueva clase. Previamente tiene que haber iniciado sesión como hemos visto en la primera ilustración.

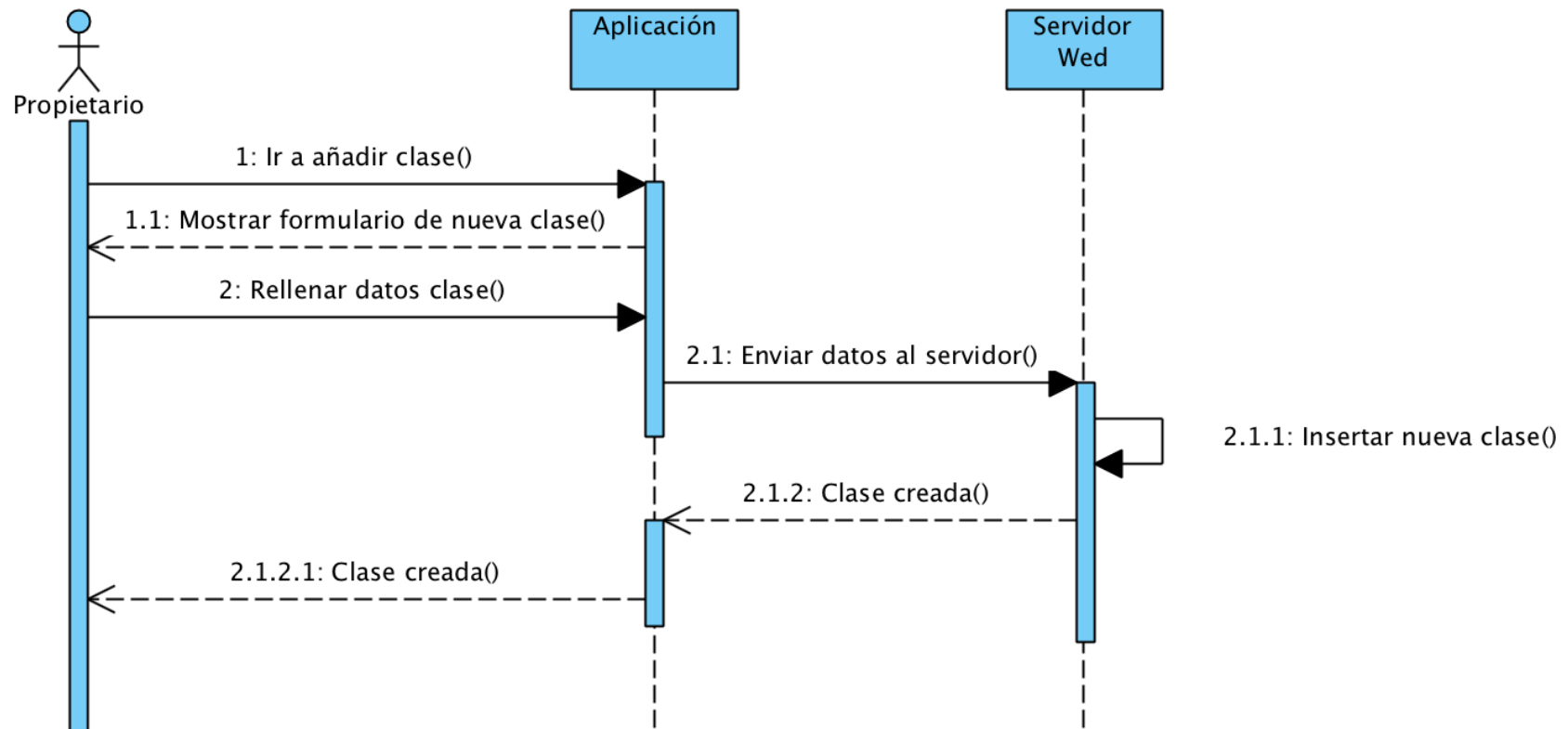


Ilustración 4.13 - Diagrama secuencia Crear Clase

4.2.4. Escenarios

Los escenarios [\[21\]](#) son casos particulares de uso del sistema en situaciones determinadas. Aunque los escenarios son útiles para adquirir y validar requisitos, no son requisitos en si mismos. Los escenarios pueden estar destinados a diversos objetivos en función de la fase de desarrollo software en la cual los realicemos, aunque lo mas interesante es realizarlos al principio del diseño para tener una visión general de las acciones que el usuario puede realizar en el sistema .

La utilización de escenarios tiene distintas finalidades como captar los requisitos esenciales del proyecto, dar unas pautas básicas del desarrollo del sistema, servir de enlace entre el cliente y el desarrollador para entender con exactitud que es lo que el cliente espera que realice su sistema.

Un escenario esta compuesto de varias partes, en nuestro caso centraremos la atención en el título, los objetivos, los actores que intervienen, los episodios o flujo de eventos y en caso de que existieran, las excepciones.

A continuación mostraré los escenarios que se corresponde con las acciones más relevantes que se pueden hacer en la aplicación.

Título: Iniciar sesión en la aplicación
Objetivo: Cualquier usuario(Propietario, cliente o monitor) quiere acceder a las funcionalidades de la aplicación.
Actores: Usuario
Episodios: 1. El usuario inicia la aplicación Android en su Smartphone. 2. La aplicación muestra la pantalla principal donde introducir los datos de acceso. 3. El usuario introduce su Identificador de usuario. 4. El usuario introduce su contraseña. 5. El usuario envía los datos. 6. Se muestran las funcionalidades que ese usuario tiene disponible.
Excepciones: Si el usuario introduce mal su usuario o su contraseña no accederá a las funcionalidades del sistema pero podrá intentarlo de nuevo.

Tabla 4.7 - Escenario 1

Título: Añadir un nuevo cliente(*)
Objetivo: El propietario quiere dar de alta en la aplicación a un nuevo cliente.
Actores: Propietario
Episodios: 1. El propietario entra en la aplicación y se identifica correctamente. 2. La aplicación muestra la pantalla con todas las funcionalidades disponible para él. 3. El propietario selecciona la opción "Clientes". 4. La aplicación muestra una pantalla con la lista de todos los usuarios. 5. El propietario selecciona el botón "+" de la parte inferior derecha. 6. La aplicación muestra una pantalla para rellenar con los datos del nuevo usuario. 7. El propietario introduce los datos del nuevo usuario. 8. El propietario envía los datos. 9. La aplicación devuelve un mensaje: "Usuario creado correctamente"

Excepciones: Si el propietario quiere dar de alta a un usuario que ya esta en la plataforma, el sistema le devolverá un mensaje de error.

Tabla 4.8 - Escenario 2

*En el caso de los monitores seria igual excepto en el episodio 3, el propietario tendría que seleccionar la opción “Monitores”.

Título: Añadir una nueva clase

Objetivo: El propietario quiere crear una nueva clase y asignársela a un monitor.

Actores: Propietario

Episodios:

1. El propietario entra en la aplicación y se identifica correctamente.
2. La aplicación muestra la pantalla con todas las funcionalidades disponible para él.
3. El propietario selecciona la opción “Clases”.
4. La aplicación muestra una pantalla con la lista de todas las clases.
5. El propietario selecciona el botón “+” de la parte inferior derecha.
6. La aplicación muestra una pantalla para rellenar con los datos de la nueva clase.
7. El propietario introduce los datos de la clase y selecciona al monitor que la imparte.
8. El propietario envía los datos.
9. La aplicación devuelve un mensaje: “Clase creado correctamente”

Excepciones:

Tabla 4.9 - Escenario 3

Título: Monitor ver sus clases
Objetivo: Un monitor quiere ver las clases que tiene asignadas para impartir y los detalles de alguna de ella.
Actores: Monitor
Episodios: 1. El monitor entra en la aplicación y se identifica correctamente. 2. La aplicación muestra la pantalla con todas las clases que tiene asignadas ese monitor. 3. El monitor selecciona alguna de sus clases para ver los detalles(hora, capacidad, tipo de clase) 4. La aplicación muestra la pantalla con el detalle de la clase así como los usuarios que están inscritos a la misma.
Excepciones: Si el monitor no tiene ninguna clase asignada PARA IMPARTIR DESDE ESE INSTANTE HACIA DELANTE la lista de clases aparecerá vacía.

Tabla 4.10 - Escenario 4

Título: Cliente apuntarse a clase
Objetivo: Un cliente quiere apuntarse a alguna de las clases que hay disponibles en el gimnasio.
Actores: Cliente.
Episodios: 1. El cliente entra en la aplicación y se identifica correctamente. 2. La aplicación muestra la pantalla con todas las funcionalidades disponible para él. 3. El Cliente selecciona la opción "Clases". 4. La aplicación muestra una pantalla con la lista de todas las clases. 5. El Cliente selecciona aquella clase a la que se quiera apuntar. 6. La aplicación muestra una pantalla con los detalles de la clase. 7. El Cliente selecciona la opción "Apuntarme". 8. La aplicación muestra un mensaje: "Se ha apuntado correctamente".
Excepciones: Si el gimnasio no tiene en ese momento ninguna clase disponible la lista de clases aparecerá vacía.

Tabla 4.11 - Escenario 5

Título: Cliente ver sus clase
Objetivo: Un cliente ver las próximas clases a las que se ha apuntado.
Actores: Cliente.
Episodios: 1. El cliente entra en la aplicación y se identifica correctamente. 2. La aplicación muestra la pantalla con todas las funcionalidades disponible para él. 3. El Cliente selecciona la opción “Mis Clases”. 4. La aplicación muestra una pantalla con la lista de las clases a las que esta apuntado ese cliente.
Excepciones: Si el cliente no se ha apuntado a ninguna clase la lista estará vacía.

Tabla 4.12 - Escenario 6

Estos escenarios nos muestran el comportamiento del sistema dependiendo de la acción que queramos realizar y de nuestro rol dentro de la aplicación. Podemos ver como cumple con los requisitos previamente establecidos y como seria su correcto funcionamiento.

4.3. Diseño

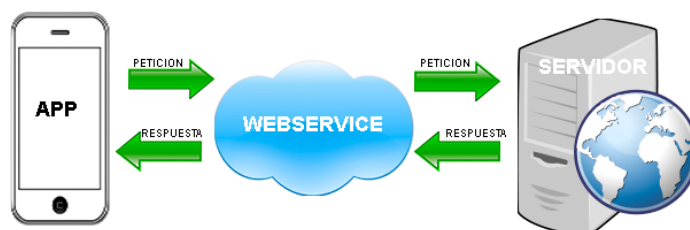


Ilustración 4.14 – Diseño Aplicación – Servicio Web <http://picarcodigo.blogspot.com/2014/05/webservice-conexiones-base-de-datos.html>

La siguiente etapa que compone el ciclo de vida de un producto software es el diseño, se define como el proceso de aplicar distintas técnicas y principios con el propósito de definir un dispositivo, proceso o sistema con suficientes detalles como para permitir su realización física [22].

Después de haber analizado los requisitos necesarios para poder desarrollar el proyecto y haber realizado un análisis del mismo, es entonces cuando entra en juego el diseño de software. El objetivo del diseño es realizar el esqueleto de nuestro sistema que dará las pautas para realizar una correcta implementación.

El diseño software es esencial para prevenir la aparición de errores en el correcto funcionamiento del sistema y para que visualmente sea tal y como desee el cliente.

Este apartado lo dividiré en dos partes, en una primera parte explicaré todo lo relacionado con el diseño de la base de datos y en una segunda parte el diseño de la aplicación.

4.3.1. Diseño de la base de datos

Una de las tareas mas importantes para la realización de cualquier producto software es el diseño de la base de datos ya que es esta la encargada de almacenar y gestionar todos los datos que use nuestra aplicación. Para un correcto diseño de la base de datos hay que saber que los distintos datos que manejará la aplicación así como las relaciones que hay entre ellos para estructurarlos de tal manera que proporcionen a la aplicación no solo eficacia sino también eficiencia para así tener el mejor rendimiento posible.

Para la realización de este proyecto se ha utilizado el modelo más utilizado dentro del diseño de base de datos que es el modelo entidad-relación.

Modelo Entidad-Relación

Es el más extendido de los modelos de datos para describir esquemas Conceptuales.

Destaca por la reproducción gráfica de esquemas de sencilla interpretación, en los cuales se representan distintas entidades que establecen relaciones entre ellas. En la representación se incluye información referente a las entidades y las relaciones.

Los conceptos básicos del modelo son:

- Entidad: Es un objeto real o abstracto con características diferenciadoras capaces de hacerse distinguir de otros objetos y del cual se almacena información. Una entidad toma como significado conceptos u objetos que juegan un papel importante en el sistema u organización.

Las entidades se representa con su nombre encerrado en un rectángulo.

Las entidades pueden ser de dos tipos:

- Débiles: No tienen atributos clave propios y no pueden existir sin participar en la relación.
 - Fuertes: Son aquellas que puede ser identificadas como únicas
- Atributos. Es una unidad básica e indivisible de información que sirve para identificarla o describirla. Pueden ser de diferentes tipos:
 - Simple o compuestos. Los simples no son divisibles mientras que los compuestos pueden dividirse en otros con significado propio.
 - Almacenados o derivados: Los atributos derivados es un valor calculado a partir de otra información ya existente(atributos, entidades relacionadas)
 - Monovalores o multivalores.
 - Opcionales(Nulos). Se desconoce el valor de cierta entidad.

Gráficamente los atributos se representan con su nombre encerrado en una elipse o rectángulo con los bordes redondeados.

Los atributos se unen a sus entidades mediante líneas.

Dentro de los atributos de una entidad encontramos lo que se denomina clave de una entidad o clave primaria. La clave primaria es un atributo de la entidad que sea lo mas restrictivo posible y que identifique de forma única a cada fila de una tabla. Si un atributo o conjunto de atributos de una entidad es clave primaria de otra entidad, a dicho conjunto de atributos se le denomina clave foránea.

- Relaciones. Las entidades por sí solas no describen la relación de un sistema de información. No basta con identificar los objetos, hay además que establecer las asociaciones existentes entre los mismos a través de Relaciones. Relación significa existencia de algo en común.

Las relaciones se representan mediante rombos y el nombre de la relación en su interior.

Las relaciones pueden tener distinto grado en función de la participación en la relación de cada una de las entidades afectadas. Este grado puede ser 1:1(una a una), 1:n(una a muchas), n:m(muchas a muchas).

Las relaciones pueden ser:

- Cíclicas: Las entidades que se relacionan son las mismas.
- Relaciones ternarias(n-arias): Una relación incluye a más de dos entidades.

Una vez definido lo que es el Modelo entidad relación y explicados sus elementos más importantes, pasaré a mostrar el modelo de la base de datos de la aplicación de este proyecto.

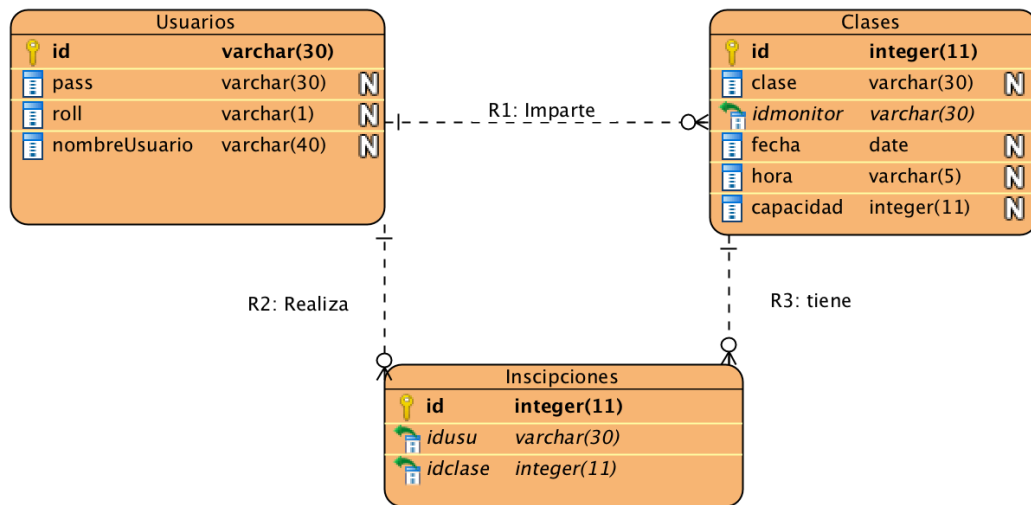


Ilustración 4.15 – Esquema Base de Datos

Encontramos tres Entidades que son las tres tablas que forman la base de datos.

• Usuarios

En esta tabla almacenaremos todos los usuarios de la aplicación independientemente de su función(roll) dentro del gimnasio, a continuación explicaremos como diferenciaremos entre cada tipo de usuario.

La tabla usuarios está compuesta de los siguientes atributos:

- Id: Será la clave primaria, ya que es un identificador único para cada usuario, en el caso de nuestra aplicación será el DNI de ese usuario.
- Pass: Será la contraseña que se le asigne a un usuario para acceder a la aplicación.
- Roll: Aquí diferenciaremos el tipo de usuario. Tenemos 3 tipos: 0- Propietario, 1-Monitor, 2-Cliente.

- nombreUsuario: Este atributo es para almacenar el nombre del usuario. Ej.: Juan Pérez Martínez.

La tabla usuarios define una relación (R1) con la tabla Clases, esta relación la tendría la tendría un usuario de tipo Monitor con una clase, ya que cada clase es impartida por un monitor. Un mismo monitor puede tener asignadas varias clases.

Por otro lado existe otra relación (R2) con la tabla Inscripciones(explicada mas a delante), esta relación la tendría un usuario de tipo Cliente, ya que un usuario realizará varía inscripciones.

- **Clases**

En esta tabla almacenaremos todas las clases que se realizarán en el gimnasio.

La tabla Clases está compuesta de los siguientes atributos:

- Id: Será la clave primaria, ya que es un valor único que identificará a cada clase, este atributo no se tiene que introducir al crear la clase, se asigna de manera automática e incremental a cada clase que se inserte.
- IdMonitor: Aquí almacenaremos el id del monitor que va a impartir la clase.
- Fecha: Este atributo almacenará la fecha en la que se realizará la clase.
- Hora Este atributo almacenará la hora en la que se realizará la clase.
- Capacidad: Este atributo almacenará la capacidad que tiene la clase, a medida que se vayan apuntando clientes esta disminuirá.

La tabla Clases establece una relación(R3) con la tabla Inscripciones ya que una clase tiene asociadas distintas inscripciones.

- **Inscripciones.**

En esta tabla almacenaremos todas las inscripciones a clases que se realicen en el gimnasio.

La tabla Inscripciones está compuesta de los siguientes atributos:

- **Id:** Será la clave primaria, ya que es un valor único que identificará a cada inscripción, este atributo no se tiene que introducir al realizar la inscripción, se asigna de manera automática e incremental a cada inscripción que se realice.
- **idUsuario:** Aquí almacenaremos el id(identificador) del usuario que realiza la inscripción.
- **idClase:** Aquí almacenaremos el id(identificador) de la clase en la cual se realiza la inscripción.

Tanto en la tabla Usuarios como en la tabla Clases guardamos únicamente los datos necesarios para un correcto funcionamiento de la aplicación, recordar que esto es solo un prototipo, en futuras versiones de la aplicación se podría almacenar mas datos como la dirección, el teléfono, etc.

4.3.2. Diseño de la interfaz

Una vez explicado todo lo referente a la base de datos donde almacenaremos toda la información que maneje la aplicación, paso a definir la siguiente etapa del proceso de Ingeniería del Software que es el diseño de la interfaz [\[23\]](#).

En los sistemas que dependen del usuario para llevar a cabo una acción tiene un enorme valor el diálogo con el usuario. Por lo tanto la interfaz es una pieza la hora de desarrollar un producto y debe estar presente desde el comienzo del desarrollo. A el usuario final de la aplicación le interesa como interactuar con ella y que esta interacción sea fácil e intuitiva, toda la implementación o complejidad que haya detrás de la interfaz es irrelevante para el usuario.

El diseño de la interfaz es especialmente importante si este producto va destinado a usuarios con unos conocimientos básicos de informática y del uso de las nuevas tecnología.

En relación a esta aplicación lo más importante es que sea fácil de utilizar por el cliente y por los monitores, el tiempo de aprendizaje debe ser el mínimo posible. Los monitores tienen que poder ver con un simple golpe de vista cuales son las clases que tienen que realizar, y los usuarios ver a que clases se pueden apuntar y cuales son las clases a las que ya se han apuntado. Para conseguir esto usaremos elementos con los que el usuario esté familiarizado como son “botones” muy visibles y con una función fácilmente reconocible para el usuario, también usaremos “listas” que es otro elemento presente en la mayoría de aplicaciones móviles con lo que el usuario podrá utilizarlas sin ninguna dificultad.

Para poder mostrar el funcionamiento de la interfaz de la aplicación voy a utilizar los Storyboard.

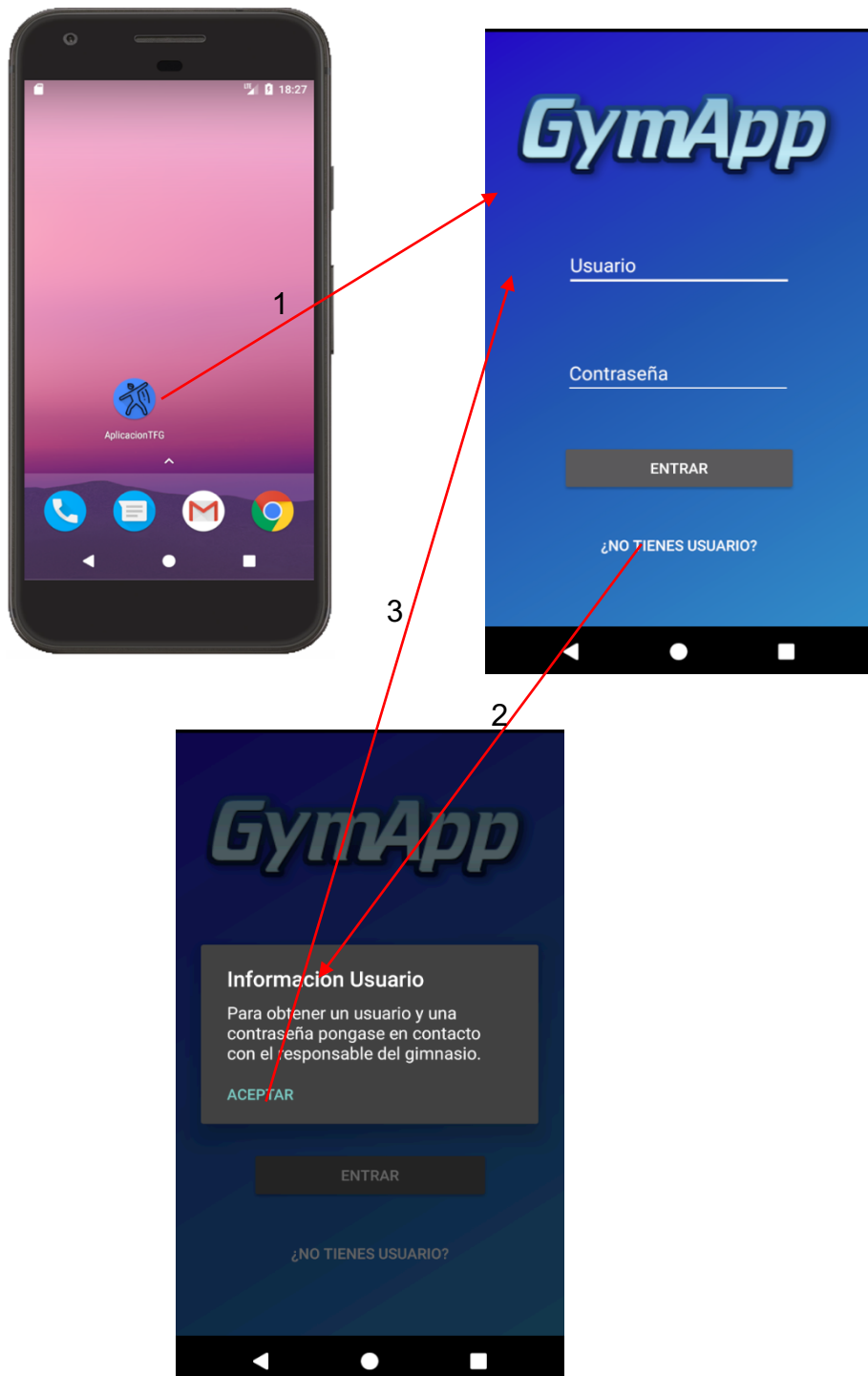
Storyboard

Los storyboard son utilizados para mostrar de manera gráfica una secuencia de acciones realizadas por el usuario. Nos permite indicar los enlaces a diferentes pantallas a partir de los resultados de estas acciones realizadas por el usuario.

A continuación mostraré los diferentes storyboard en función de la acción que cada tipo de usuario puede realizar.

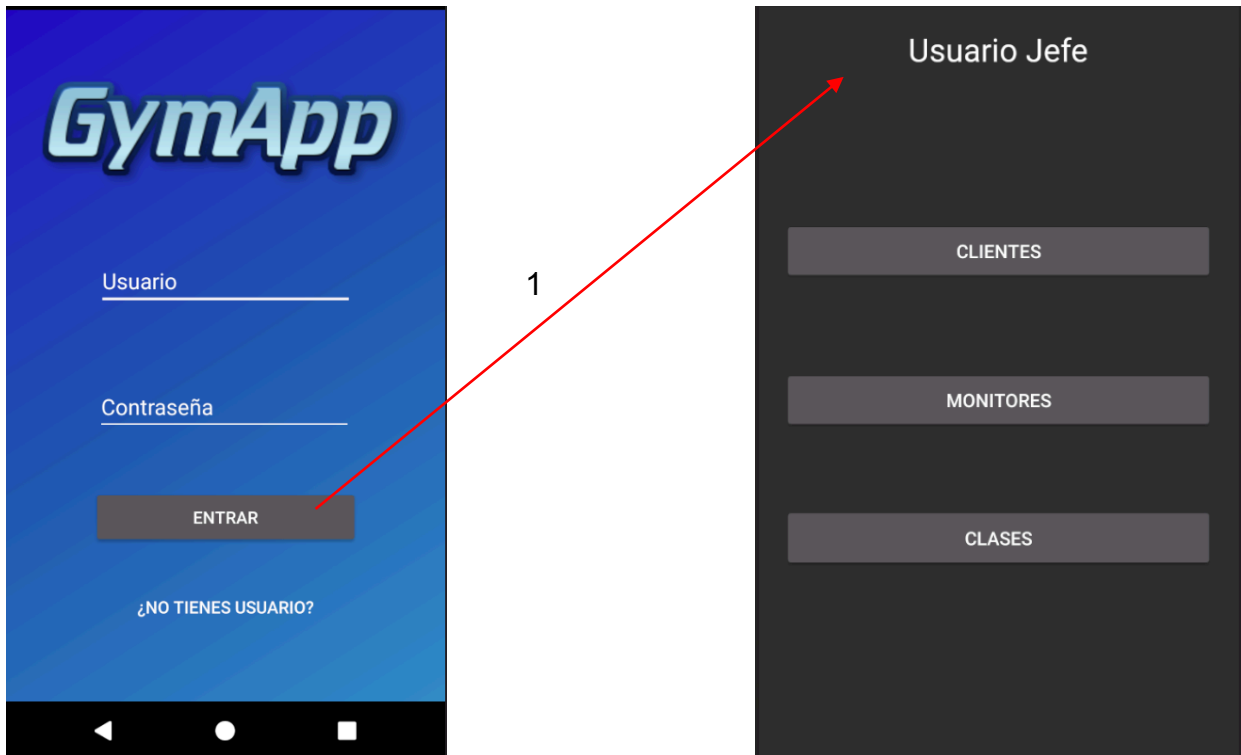
- **Autenticación e Información.**

El siguiente stroyboard muestra el proceso que tienen que seguir todos los usuarios para acceder a la pantalla de Autenticación para iniciar sesión.

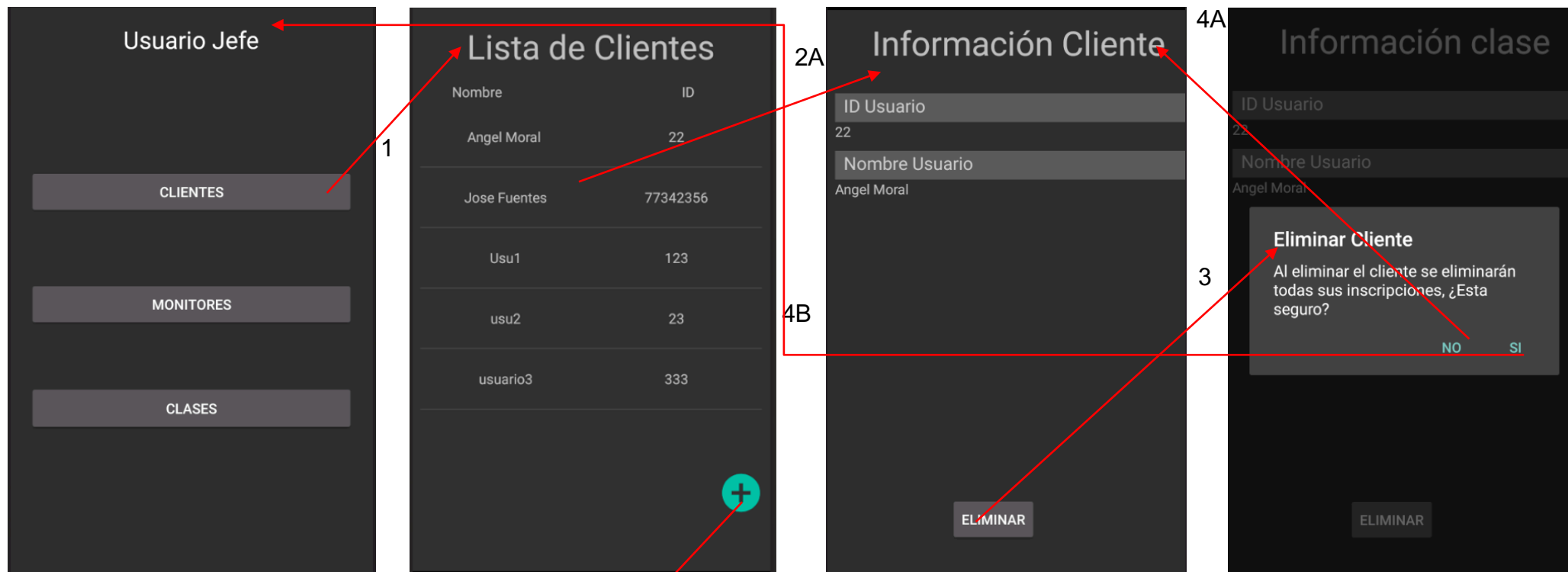


- **Autenticación Propietario**

El siguiente Storyboard muestra la pantalla que se mostrará en el caso de iniciar sesión como propietario del gimnasio.



Propietario: Gestión de clientes



2B

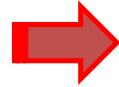
Nuevo Usuario

Nombre Usuario

DNI Usuario

Password

CREAR USUARIO



Nuevo Usuario

Jose Fuentes

77342356

.....

CREAR USUARIO

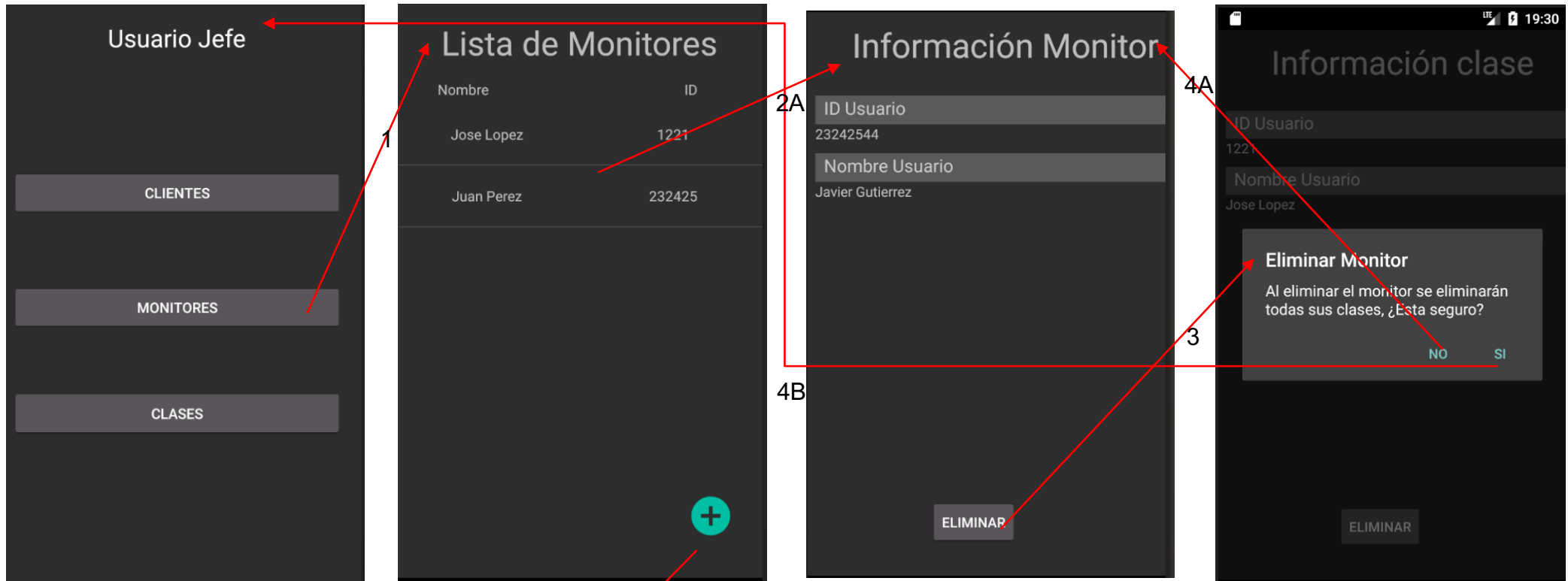
5

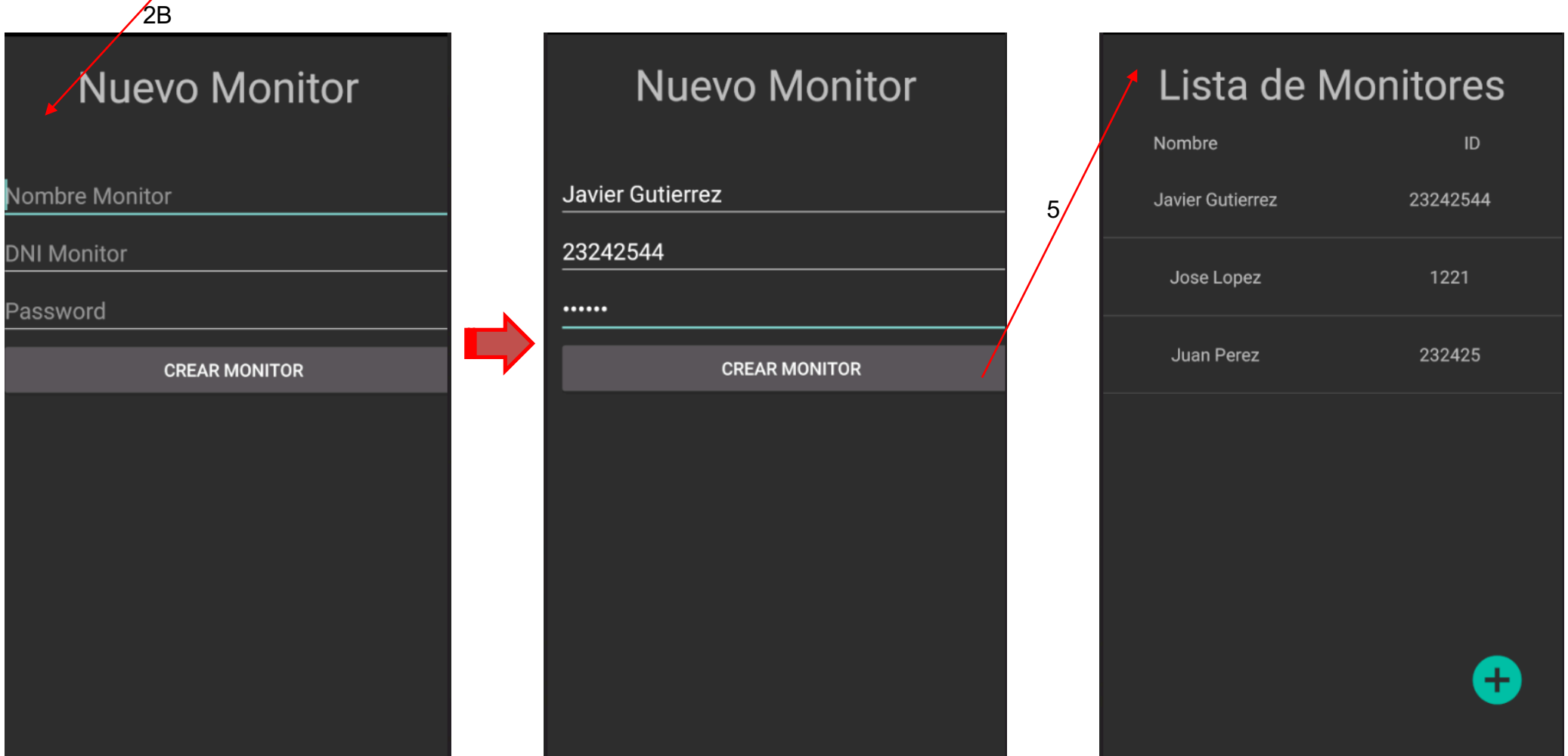
Lista de Clientes

Nombre	ID
Angel Moral	22
Jose Fuentes	77342356
Usu1	123
usu2	23
usuario3	333

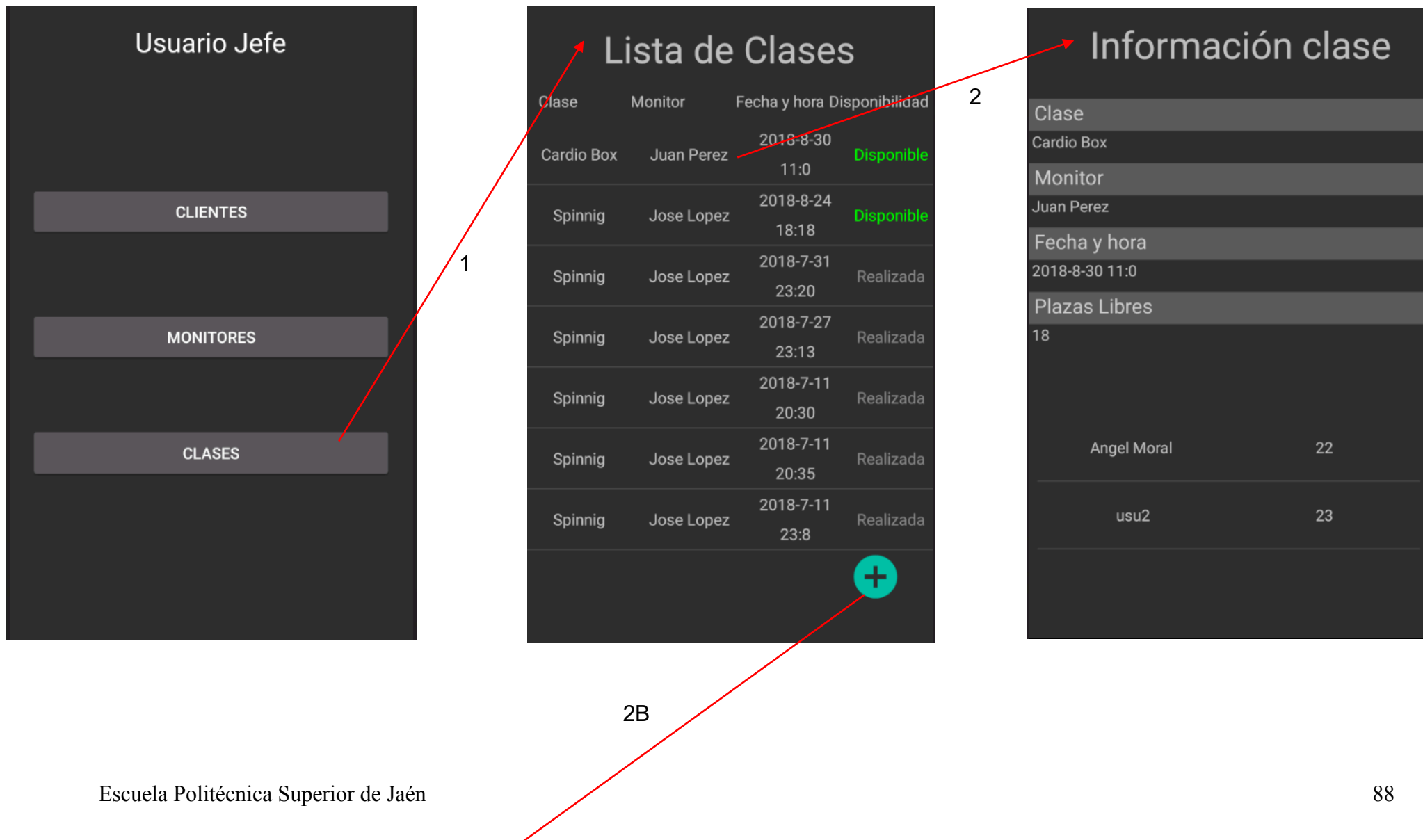


- Propietario: Gestión de Monitores





- Propietario: Gestión de clases



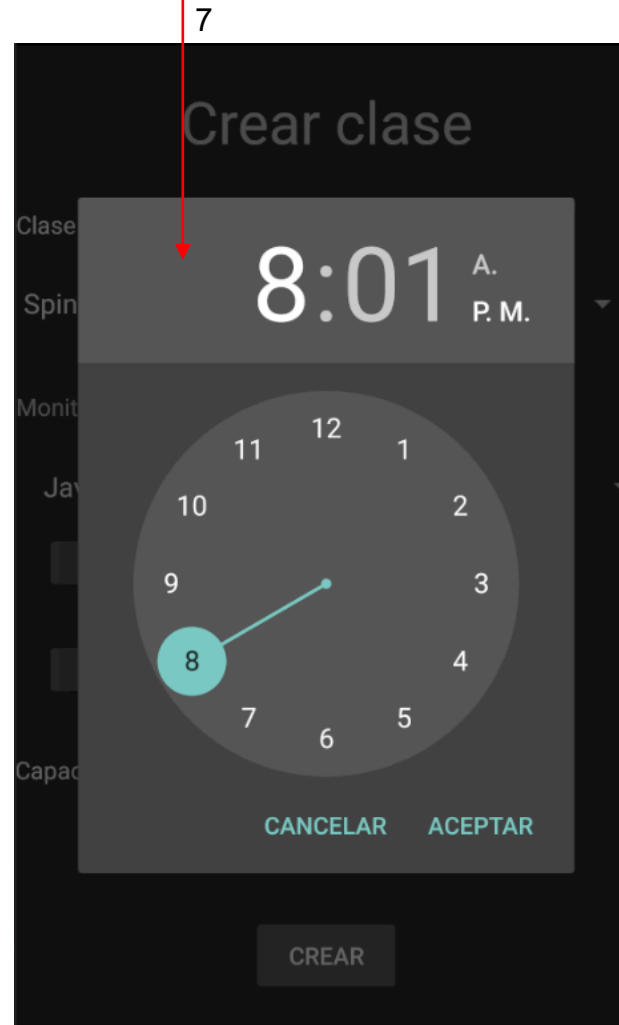
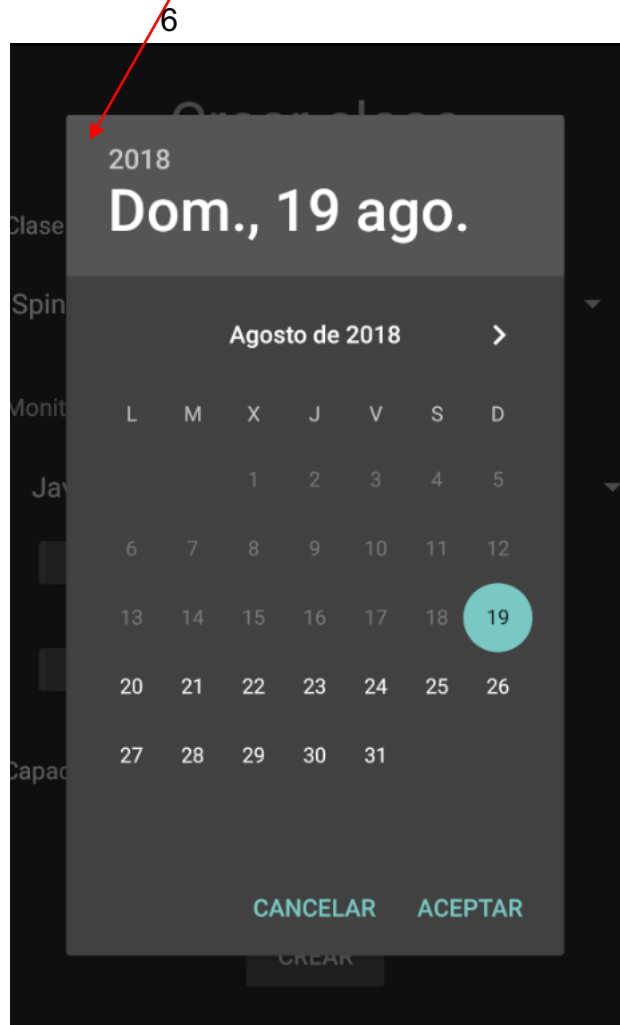
2B

The image displays three sequential screenshots of a mobile application interface titled "Crear clase". The interface is dark-themed and contains the following elements:

- Clase:** A dropdown menu with options "Spinnig", "Zumba", and "Cardio Box".
- Monitor:** A dropdown menu with options "Javier Gutierrez", "Jose Lopez", and "Juan Perez".
- FECHA:** A date input field.
- HORA:** A time input field.
- Capacidad:** A numeric input field.
- CREAR:** A button at the bottom right.

Red arrows and numbers indicate the sequence of steps:

- 2B:** Points to the "Clase" dropdown menu in the first screenshot.
- 4:** Points to the "Clase" dropdown menu in the second screenshot, where the menu is open showing the options.
- 5:** Points to the "Monitor" dropdown menu in the third screenshot, where the menu is open showing the options.
- 6:** Points to the "FECHA" input field in the first screenshot.
- 7:** Points to the "HORA" input field in the second screenshot.



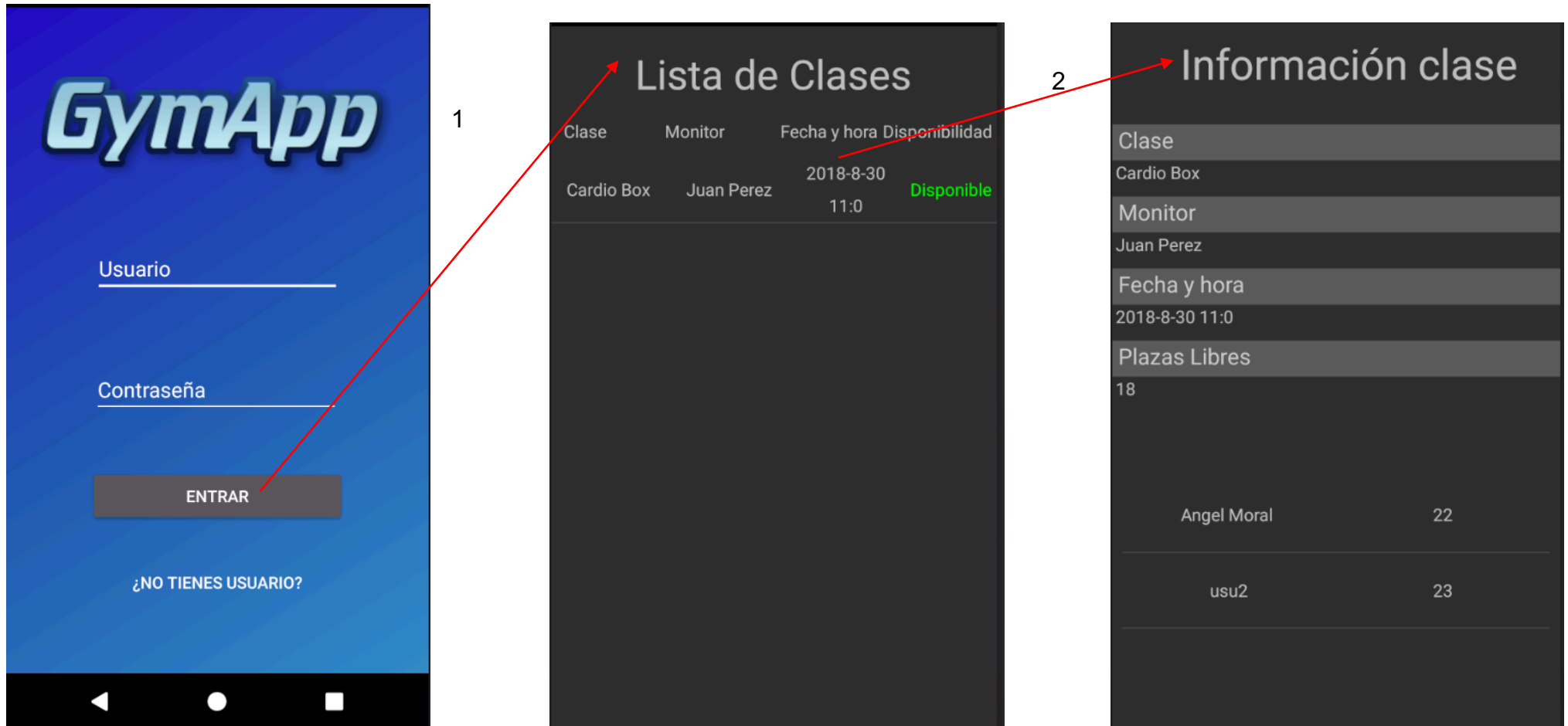
This screenshot shows the 'Crear clase' (Create class) form. It has a dark background with white text. The form includes the following fields and buttons:

- Clase:** A dropdown menu with 'Spinnig' selected.
- Monitor:** A dropdown menu with 'Javier Gutierrez' selected.
- FECHA:** A text input field containing '2018-8-19'.
- HORA:** A text input field containing '20:1'.
- Capacidad:** A text input field containing '10'.
- CREAR:** A red button at the bottom right.

8

Pantalla "Lista de Clases"

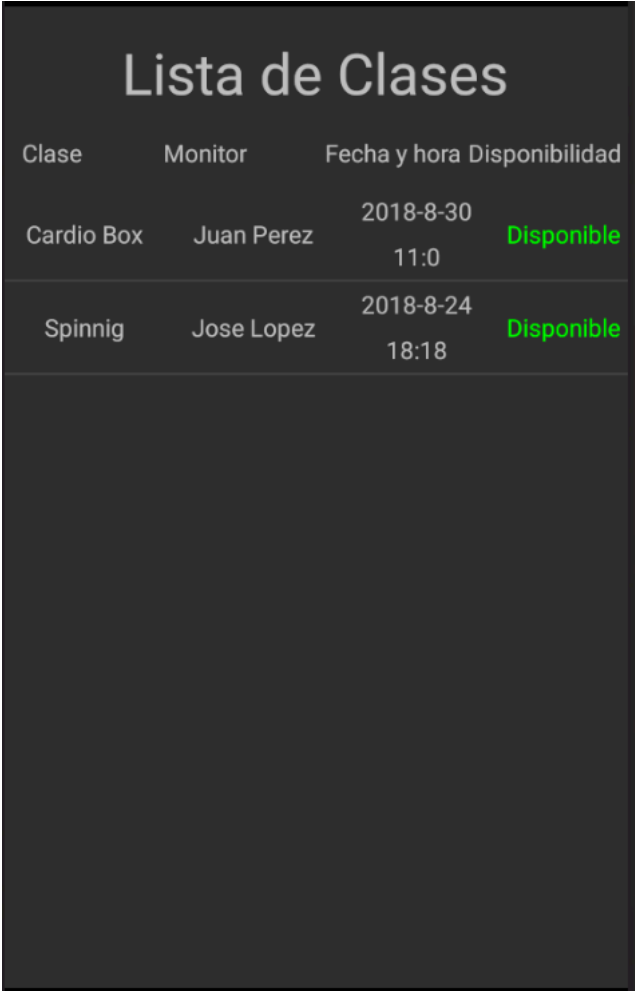
- **Monitor: Clases Asignadas**



- **Cliente**



2B



Clase	Monitor	Fecha y hora	Disponibilidad
Cardio Box	Juan Perez	2018-8-30 11:0	Disponible
Spinnig	Jose Lopez	2018-8-24 18:18	Disponible

Una vez realizado tanto el Análisis de Requerimiento, el Análisis del sistema y diseño del mismo tenemos establecidos los pilares básicos sobre los cuales tendremos que realizar el proyecto. Lo que tendremos que realizar a continuación será empezar a construir nuestro sistema con el uso de herramientas software e ir dándole forma basándonos en todos los aspectos estudiados en este capítulo hasta conseguir un producto final, este proceso se denomina Implementación y dará nombre al próximo capítulo de esta memoria.

CAPÍTULO 5

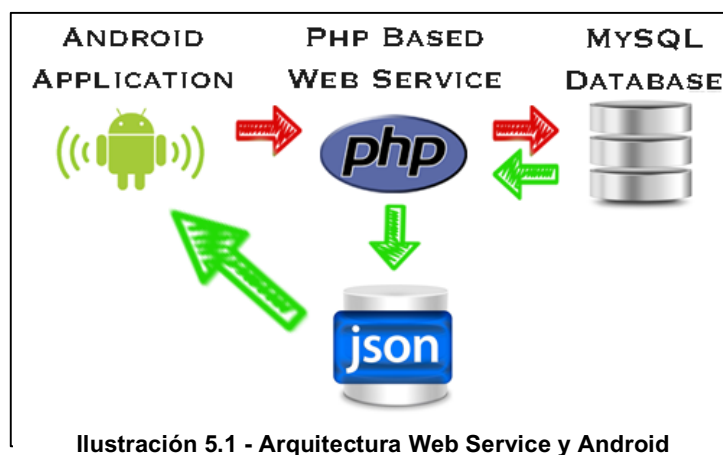
IMPLEMENTACIÓN

En el capítulo se presenta la fase de implementación del proyecto, una vez realizado el análisis y el diseño del proyecto tenemos que pasar todo lo anteriormente expuesto a algo tangible, para ello utilizaremos distintas herramientas software de desarrollo que mediante el uso de lenguajes de programación irán dando forma a nuestro proyecto hasta conseguir un producto final que el usuario pueda utilizar.

5.1. Arquitectura del Sistema

Este proyecto presenta dos tipos de arquitecturas ampliamente conocidas y desarrolladas, la arquitectura Cliente-Servidor que es la utilizada por el Servicio Web, y la arquitectura Cliente-Interfaz que es la que utiliza Android.

Para entender mejor cómo funcionarían estas arquitecturas vamos a explicarlo ayudándonos de una ilustración.



1. Un usuario al utilizar la aplicación realiza una petición al servicio Web.
2. La aplicación envía en formato PHP la petición al servicio Web.
3. El servicio Web realiza una consulta a su base de datos que es donde se encuentran almacenados los datos de la aplicación.
4. La base de datos devuelve una respuesta a esa consulta y el servicio Web, mediante PHP, convierte esta respuesta a un objeto JSON.
5. La aplicación recibe este objeto JSON del servicio Web que es la respuesta a la consulta que inicialmente había realizado el usuario.

Como podemos observar al utilizar esta arquitectura la aplicación móvil no tendrá un gran tamaño dentro del dispositivo del cliente ya que toda la información que utilizaremos se encuentra almacenada en la base de datos del servidor y el usuario accede a esta información mediante la aplicación.

Para poder entender mejor todos los aspectos referentes a la implementación del proyecto, previamente tenemos que definir cuales son los lenguajes de programación utilizados para dicha implementación.

5.2. Lenguajes de programación

Como hemos visto anteriormente y en los capítulos anteriores, nuestro proyecto esta dividido en dos partes una parte sería el Servidor Web y otra la aplicación Android, por lo tanto hemos utilizado diferentes lenguajes dependiendo de la parte que implementamos.

5.2.1. Servicio Web

Para la parte del servicio web hemos utilizado diferentes lenguajes de programación, por un lado PHP (Apartado [2.7.4](#)) es el lenguaje en él que el servicio web recibe las peticiones que se realizan en la aplicación Android, con él realizaremos la conexión a la base de datos y las consulta a esta en SQL. SQL es un lenguaje de consulta estructurado utilizado para administrar sistemas de gestión de bases de datos relacionales como es el caso de nuestro proyecto. Una vez realizada la consulta el servidor nos dará una respuesta a la misma, para poder devolver la respuesta a la aplicación Android y que esta sea manejable por la aplicación devolveremos un objeto en formato JSON (JavaScript Object Notation - Notación de Objetos de JavaScript).

JSON

JSON [\[44\]](#) no es un lenguaje propiamente dicho sino un formato ligero de intercambio de datos. JSON se compone de dos estructuras, una colección de pares de nombre/valor y una lista ordenada de valores. Un objeto JSON se puede representar de dos formas, como un Objeto (conjunto desordenado de pares nombre/valor) o un Array (colección de valores).

NetBeans

NetBeans es la herramienta que hemos utilizado para realizar los archivos PHP que nos permiten comunicarnos con el servicio Web.

NetBeans [\[26\]](#) es un entorno de desarrollo libre y multiplataforma, fue creado principalmente para el lenguaje de programación Java, aunque nos permite realizar proyectos en una gran variedad de lenguajes de programación como es el caso de C, C++, XML, HTML, PHP, etc.

NetBeans emplea módulos para desarrollar las aplicaciones. Un módulo es un archivo Java que contiene clases Java que permiten interaccionar con las APIs de NetBeans y un archivo especial que identifica a dicho módulo. Las aplicaciones realizadas a partir de módulos se le pueden añadir nuevos módulos independientes, con lo cual, las aplicaciones realizadas en NetBeans pueden ser extendidas por otros desarrolladores con facilidad.

Arquitectura de ficheros

Como expliqué en el capítulo 2 para la realización del servicio Web se ha empleado la herramienta XAMPP, este software tiene una serie de directorios donde incluiremos todos los archivos necesarios para utilizar y poner en funcionamiento el servicio. Los directorios de XAMPP necesarios para realizar este proyecto se organizan de la siguiente manera:

- XAMPP (directorio raíz de la herramienta XAMPP)
 - Htdocs
 - WebService

Dentro del directorio WebService se encuentra el proyecto NetBeans con los ficheros donde realizamos todas las peticiones. Como hemos visto anteriormente, cada petición es un fichero PHP, por lo tanto se han implementado los siguientes archivos:

- **addClases.php**: Es el fichero encargado de insertar una nueva Clase en la base de datos del sistema, deberemos recibir los atributos de la Clase (tipo, idmonitor, nombremonitor, fecha, hora y capacidad) para poder incluirla en la tabla correspondiente.
- **addInscripciones.php**: Es el fichero encargado de insertar una nueva inscripción en el sistema, recibiremos el identificador del usuario (idusu) que se quiere inscribir a una clase y el identificador de la clase (idclase) a la cual se quiere inscribir, comprobaremos la tabla Incripciones para verificar que el usuario no esta ya inscrito en esa clase, en el caso de no estarlo se realizará la inserción.
- **addUsuario.php**: Es el fichero encargado de insertar usuarios en la base de datos, ya sean de tipo Cliente como de tipo Monitor, para ello el fichero recibe el identificador del usuario (id), la contraseña (pass) y el tipo de usuario que es (roll), tras comprobar la tabla Usuarios y verificar que no existe ningún usuario con ese identificador se inserta el usuario en la tabla.

- **eliminarUsuario.php**: Es el fichero que se encarga de borrar un usuario del sistema ya sea un Cliente o un Monitor. Hay que tener en cuenta diferentes aspectos en función del tipo de usuario que queramos eliminar.
 - Eliminar un Cliente: Si borramos un cliente también tenemos que eliminar las inscripciones a clases que tenga asociadas dentro de la tabla Inscripciones y aumentar la capacidad de dichas las clases en una plaza más.
 - Eliminar un Monitor: Si borramos un monitor tenemos que eliminar las clases cuyo idmonitor coincida con el identificador del monitor a eliminar y las inscripciones a dichas clases.
- **obtenerClases.php**: Es el fichero encargado de devolver todas las clases del gimnasio, las clases se mostrarán con fecha descendiente, es decir, las clases que ya hayan pasado serán las últimas en aparecer.
- **obtenerClasesCliente.php**: Este fichero es el encargado de devolver las clases a las que está inscrito un cliente, tras recibir el identificador del cliente (id) comprobaremos sus inscripciones para saber el identificador de clase de cada inscripción para de esta manera obtener los datos referentes a cada clase que se encuentran dentro de la tabla Clases.
- **obtenerClasesMonitor.php**: Este fichero es el encargado de devolver las clases que tiene asignado un monitor en concreto, para ello recibe el identificador del monitor (id) y se devuelven aquellas clases tiene como idmonitor ese identificador.
- **obtenerNombre.php**: Este fichero es el encargado de devolver el nombre de un usuario independientemente del tipo que sea, recibe el identificador de usuario (id) y devuelve el nombre asociado a ese identificador.
- **obtenerUsuarios.php**: Este fichero devuelve una lista con los usuarios dependiendo del tipo que queramos para ello solicitamos el roll, para poder diferenciar entre cada uno de los tipos de usuarios.
- **obtenerUsuariosApuntados.php**: Este fichero se encarga de devolver un listado de los clientes que hay inscritos en una clase en concreto. Para ello necesita el identificador de la clase (idclase) de la cual queremos saber los Clientes inscritos, y tras comprobar la tabla de inscripciones podemos saber que Clientes se han inscrito a esa clase.

- **valida.php**: Este fichero es el encargado del proceso de identificación cuando un usuario quiere acceder a la aplicación tendrá que introducir su identificador de usuario (id) y su contraseña (pass), el fichero se encarga de comprobar las credenciales de ese usuario para darle o no acceso, devolveremos los datos relativos a ese usuario.

Mencionar que en todos los ficheros es necesario realizar la conexión pertinente con la base de datos del servidor para poder realizar las consultas y que estas devuelven un objeto en formato JSON para poder utilizarlo en la aplicación Android.

5.2.2. Aplicación Android

Java

Para la parte de la aplicación Android hemos utilizado el lenguaje de programación Java [\[24\]](#) que es el lenguaje nativo que se usa para la implementación de aplicaciones Android.

Java es un lenguaje de programación de propósito general, concurrente y orientado a objetos, su diseño estaba orientado a tener las mínimas dependencias de implementación. Su síntesis proviene de C y C++. En nuestro caso es esencial que sea concurrente ya que esto nos permite la ejecución de diferentes hilos o procesos que serán los encargados de realizar las peticiones al servicio Web.

Java es uno de los lenguajes más utilizados por los desarrolladores en la actualidad, especialmente para el desarrollo en dispositivos móviles gracias a la creación de la especificación J2ME (Java 2 Platform, Micro Edition), que es una versión del entorno de ejecución Java reducido y muy optimizado para dispositivos electrónicos de consumo (Tarjetas inteligentes, teléfonos móviles, sintonizadores de TV y pequeños electrodomésticos).

Java se creó con cinco características diferenciadoras:

- Usar el paradigma de programación orientada objetos.
- Poder ejecutarse en diferentes sistemas operativos,

- Soportar trabajo en red.
- Poder ejecutar código en sistemas remotos de forma segura.
- Fácil de usar e incorporar lo mejor de otros lenguajes orientados a objetos.

Android Studio

La plataforma que hemos utilizado para desarrollar la aplicación es Android Studio [25] que es el entorno de desarrollo oficial de Android. Dentro de las múltiples características de Android Studio cabría destacar que incorpora un dispositivo virtual Android que podemos utilizar para ejecutar y probar las aplicaciones con lo cual, no es necesario tener un dispositivo Android para poder probar nuestras aplicaciones.

Uno de los aspectos que es importante mencionar a la hora de hablar de Android Studio es diferenciar las distintas partes o distintos clases de archivos que se utilizan a la hora de realizar un proyecto con Android Studio:

- **Clases:** Es el concepto por excelencia dentro de la programación orientada a objetos. Una clase consiste en una abstracción que utilizamos para poder crear objetos y estos utilizarlos en la la aplicación. En nuestro proyecto tenemos tres clases, Global, Usuario y Clase, las explicaremos más adelante.
- **Actividades:** Las actividades [27] son los elementos más importantes dentro del desarrollo de aplicaciones móviles, ya que son las diferentes pantallas con las interactúa el usuario. A cada actividad que creemos se le asignará una vista para implementar la interfaz que usará el usuario. Una aplicación Android suele tener múltiples vistas conectadas mediante botones o diferentes acciones que realiza el usuario para poder ir navegando de unas a otras.
- **Adaptadores:** Un adaptador es un elemento muy utilizado a la hora de realizar aplicaciones Android ya que nos permite organizar diferentes objetos dentro de una vista de una manera predeterminada por el programador, en definitiva es un “puente” entre la vista y los datos de la vista. Los adaptadores se suelen utilizar para mostrar un Lista, Array o ArrayList por tanto es un recurso que utilizaremos a menudo dentro de

este proyecto ya que se realizan muchos listados de datos a la hora de presentarlos al usuario (Clases, Monitores, Clientes, etc).

Arquitectura de Ficheros

Una vez definido el lenguaje y la herramienta software que vamos a utilizar pasaremos a definir como se organizarán los ficheros que compondrá la aplicación. Para organizar estos ficheros los vamos a dividir según los tipo de ficheros que hemos visto anteriormente que existe en Android Studio.

- Clases:
 - **Glovals.java:** Esta clases está destinada a almacenar variables globales, las variables globales son variables accesibles desde cualquier parte de la aplicación, es decir, independientemente de la Actividad, Clase o adaptador en el que estemos podemos acceder al valor de esta variable y usarlo. En el caso de nuestro proyecto tenemos la variable *Ip* en esta variable almacenaremos la Dirección IP del servidor al cual nos conectamos.
 - **Usuario.java:** Es una clase que se utiliza como modelo de objetos para la aplicación, en su código se describen los atributos junto con los métodos `get()` y `set()` y los constructores pertinentes para poder usar objetos de este tipo.
 - **Clase.java:** También es una clase que se utiliza como modelo de objetos encontraremos los métodos `get()` y `set()` y los constructores.

Tanto la clase “Usuario” como la clase “Clase”, están destinadas a crear objetos que podamos usar con los adaptadores para poder mostrar datos relacionados con los clientes y las clases.

- Actividades: Como hemos explicado anteriormente cada actividad se compone de dos partes, la clase que es donde se implementan todas las acciones de la actividad y la vista que es donde se implementa la interfaz

que se presentará al usuario. Voy a definir de manera más detallada la implementación de cada clase y la vista que tiene asociada.

- **MainActivity.java:** Es la actividad correspondiente con la vista de inicial de la aplicación, en nuestro caso es la vista donde se realiza la autenticación de usuario, debemos introducir el identificador de usuario junto con la contraseña para poder acceder a las funciones de la aplicación dependiendo de nuestro roll. En este fichero encontramos los diferentes campos de texto y los diferentes botones que la componen. También encontramos cuatro funciones:
 - `String enviarPost(String id, String pass):` Esta función es la encargada de enviar el id y el pass del usuario al fichero `valida.php` para comprobar si esos datos se corresponde con algún usuario y puede acceder a sus funcionalidades.
 - `Int objJSON(string rspta):` Esta función es la encargada de convertir el objeto JSON que devuelve la consulta del fichero `valida.php`. En este caso recibiríamos todos los datos del usuario y nos quedaríamos con el roll para saber que funcionalidades tiene disponibles.
 - `String obtenerClasesMonitor(string id):` Una vez comprobado que el usuario existe y obtenido su roll, en caso de que sea monitor, utilizamos esta función para enviar el identificador de ese monitor al fichero `obtenerClasesMonitor.php` para obtener las clases que tiene ese asignadas monitor.
 - `ArrayList<Clase> objJSON3(string rspta):` Esta función es la encargada de convertir el objeto JSON que devuelve el fichero `obtenerClasesMonitor.php`. En este caso recibiríamos todos los datos de las clases y los almacenaremos en un array de tipo Clase, este array se lo enviaremos a la actividad `MostrarClases.java`.

Sus vista asociada es `activity_crear_monitor.xml`.

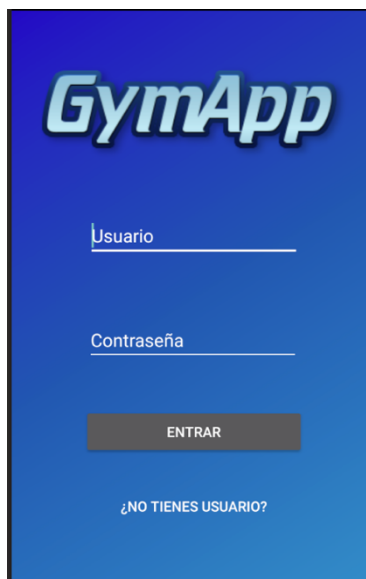


Ilustración 5.2 – `activity_main.xml`

- **Jefe.java:** Es la actividad correspondiente con la vista del Propietario de la aplicación, encontraremos definidos tres botones para las tres funcionalidades principales que puede realizar el propietario “Clientes”, “Monitores” y “Clases”. Pulsando en cada uno de ellos accederemos a la gestión de los mismos. Caben destacar cuatro funciones que implementa esta actividad:
 - `String obtenerDatosUsuarios(String roll)`: Esta función es la encargada de enviar el roll, dependiendo del tipo de usuario del cual queramos conocer los datos, al fichero `obtenerUsuario.php`.
 - `ArrayList<Usuario> objJSON2(string rspta)`: Esta función es la encargada de convertir el objeto JSON que devuelve el fichero `obtenerUsuarios.php`. En este caso recibiríamos todos los datos del usuario pero nos quedamos con el id y con el nombre y lo almacenaremos en un array de tipo `Usuario`, este array se lo enviaremos a la actividad `MostrarUsuarios.java`.
 - `String obtenerClases()`: Esta función es la encargada de solicitar la lista de las clases al fichero `obtenerClases.php`.

- `ArrayList<Clase> objJSON3(string rspta)`: Esta función es la encargada de convertir el objeto JSON que devuelve el fichero `obtenerClases.php`. En este caso recibiríamos todos los datos de las clases y lo almacenaremos en un array de tipo `Clase`, este array se lo enviaremos a la actividad `MostrarClases.java`.

Su vista asociada es `activity_jefe.xml`.

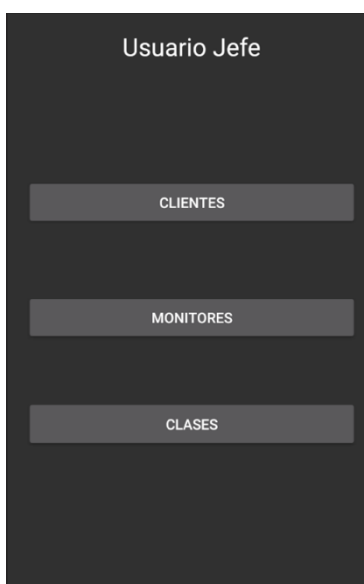


Ilustración 5.3 – `Activity_jefe.xml`

- **Cliente.java**: Es la actividad correspondiente con la vista inicial de los Clientes de la aplicación, encontraremos definidos dos botones, uno para ver las clases disponibles y otro para ver las clases a las que esta apuntado ese cliente. Caben destacar dos funciones:
 - `String obtenerClasesCliente(String id)`: Esta función es la encargada de solicitar la lista de las clases a las que está apuntado ese cliente al fichero `obtenerClasesCliente.php` enviado el id del cliente.
 - `ArrayList<Clase> objJSON4(string rspta)`: Esta función es la encargada de convertir el objeto JSON que devuelve el fichero `obtenerClases.php`. En este caso recibiríamos todos los datos de las clases a las que está apuntado ese cliente

y lo almacenaremos en un array de tipo Clase, este array se lo enviaremos a la actividad MostrarClases.java.

Su vista asociada es activity_cliente.xml

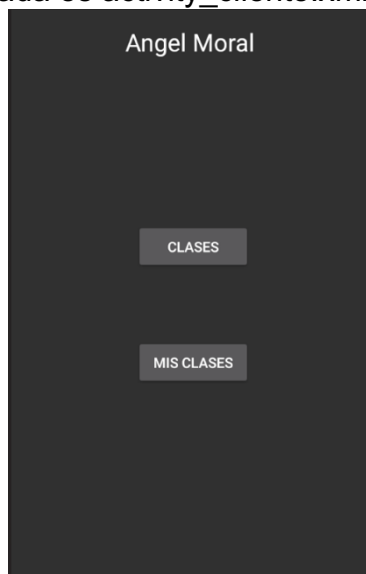


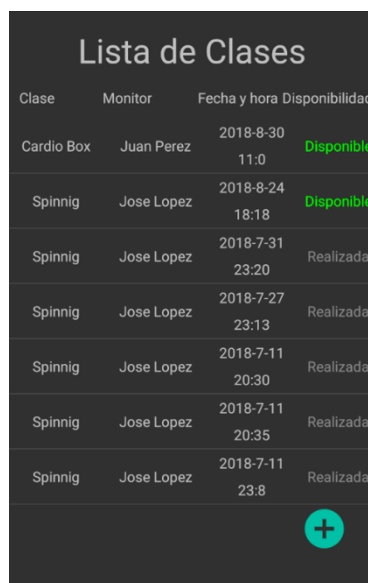
Ilustración 5.4 – activity_cliente.xml

- **MostrarClases.java:** Es la actividad correspondiente con la vista en la que aparece un listado con clases, este listado contendrá diferentes clases si la persona que accede a la vista es el propietario, un cliente que quiere ver sus clases, un cliente que quiere ver todas las clases disponibles o un monitor que quiere consultar las clases que tiene asignadas. Encontraremos definido un botón para añadir una nueva clase que solo estará disponible si el usuario que accede a esta actividad es el propietario. Destacar cuatro funciones:
 - `String obtenerDatosMonitor(String roll):` Esta función es la encargada de solicitar los datos de los monitores que hay en el sistema para ello envía el roll "1" al fichero `obtenerUsuario.php`.
 - `ArrayList<Clase> objJSON(string rspta):` Esta función es la encargada de convertir el objeto JSON que devuelve el fichero `obtenerUsuario.php`. En este caso recibiríamos todos los datos de los monitores que hay en el sistema y

guardaremos el id y el nombre del monitor, nos servirán a la hora de crear una clase nueva y asignarle un monitor.

- `String obtenerUsuariosApuntados(String idclase)`: Esta función es la encargada de solicitar los datos de los usuarios que están apuntados en una clase en concreto al fichero `obtenerUsuariosApuntados.php`.
- `ArrayList<Usuario> objJSON2(String rspta)`: Esta función es la encargada de convertir el objeto JSON que devuelve el fichero `obtenerUsuariosApuntados.php`. Esta función se utiliza en el caso de que el propietario o un monitor quieran ver los detalles de esa clase, que se mostrarán los usuarios que están apuntados a esa clase.

Su vista asociada es `activity_mostrar_clases.xml`.



Clase	Monitor	Fecha y hora	Disponibilidad
Cardio Box	Juan Perez	2018-8-30 11:0	Disponible
Spinnig	Jose Lopez	2018-8-24 18:18	Disponible
Spinnig	Jose Lopez	2018-7-31 23:20	Realizada
Spinnig	Jose Lopez	2018-7-27 23:13	Realizada
Spinnig	Jose Lopez	2018-7-11 20:30	Realizada
Spinnig	Jose Lopez	2018-7-11 20:35	Realizada
Spinnig	Jose Lopez	2018-7-11 23:8	Realizada

Ilustración 5.5 – `activity_mostrar_clases.xml`

- **CrearClases.java**: Es la actividad con la el propietario crea una nueva clase. Esta actividad viene de la actividad `MostrarClases.java`. En esta actividad a demás de los monitores que pueden ser asignados a esa clase, encontramos los diferentes campos de texto para rellenar con los datos de una clase como son el tipo, la fecha, la hora y la capacidad y el botón para crear la clase. La función más destacable es:

- `String enviarPost(String clase, String idmonitor, String nombreMonitor, String fecha, String hora, int capacidad):` Esta es la función que envía los datos al fichero `addClase.php` para crear una nueva clase.

Su vista asociada es `activity_crear_clase.xml`.

Ilustración 5.6 – `activity_crear_clase.xml`

- **InformacionClase.Java:** Es la actividad donde cualquier usuario puede ver los datos de una clase aunque hay algunos detalles que si dependerán del usuario que acceda a esta actividad. Si el usuario que accede a esta actividad es el propietario o un monitor podrá ver los usuarios que hay apuntados a la clase, si el usuario que accede es un cliente le aparecerá un botón para apuntarse a la clase. También encontramos dos funciones:
 - `String enviarPost(String idusu, String idclase):` Es la función encargada de realizar la inscripción. Envía el identificador del usuario que se apunta y el identificador de la clase a la cual se apunta al fichero `addInscripciones.php`.

- **int objJSON:** Esta función es la encargada de convertir el objeto JSON que devuelve el fichero addInscripciones.php. Este archivo devolverá un 1 o un 0 en caso de que el usuario se haya podido inscribir a la clase o no.

Su vista asociada es `activity_informacion_clase.xml`.

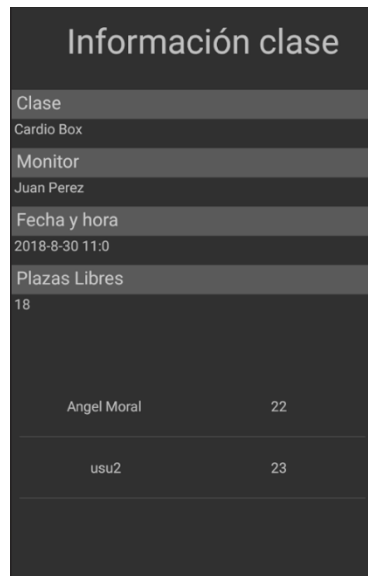


Ilustración 5.7 –
activity_informacion_clase
(Monitor Propietario)



Ilustración 5.8 –
activity_informacion_clase
(Cliente)

- **MostrarClientes.java:** Es la actividad correspondiente con la vista en la que aparece un listado con los cliente, a esta vista solo podrá acceder el propietario, será desde su actividad(Jefe.java) desde la cual se envíen los datos de los clientes que se mostraran. Encontraremos definido un botón para añadir un nuevo cliente. Su vista asociada es `activity_mostrar_clientes.xml`.

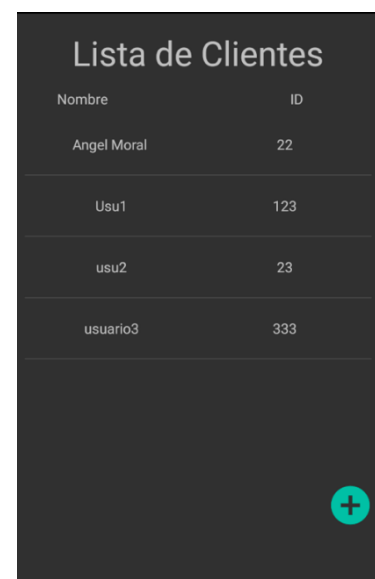


Ilustración 5.9 –
activity_mostrar_clientes.xml

- **CrearCliente.java:** Es la actividad con la cual el propietario crea un nuevo cliente. Esta actividad viene de la actividad `MostrarClientes.java`. En esta actividad encontramos los diferentes campos de texto para rellenar con los datos del cliente. Destacar dos funciones:
 - `String enviarPost(String id, string pass, String nombre):` Es la función encargada de enviar los datos del usuario al fichero `addUsuario.php`.
 - `Int objJSON(String rspta):` Esta función es la encargada de convertir el objeto JSON que devuelve el fichero `addUsuario.php`. Este archivo devolverá un 1 o un 0 en caso de que el usuario se haya podido crear o no.

Su vista asociada es `activity_crear_cliente.xml`

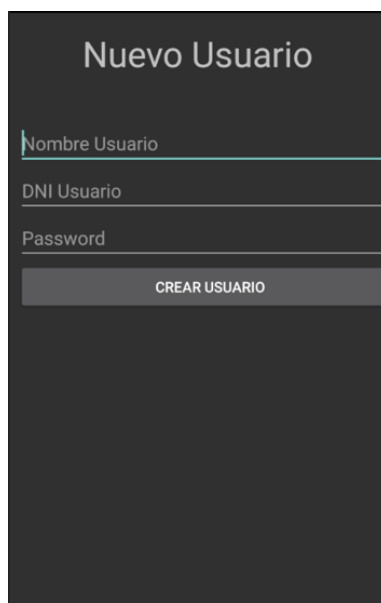


Ilustración 5.10 –
`activity_crear_clientes.xml`

- **MostrarMonitores.java:** Es la actividad correspondiente con la pantalla en la que aparece un listado con los monitores, a esta actividad solo podrá acceder el propietario, será desde su actividad (`Jefe.java`) desde la cual se envíen los datos de los monitores que se mostraran. Encontraremos definido un botón

para añadir un nuevo monitor. Su vista es activity_mostrar_clientes.xml.

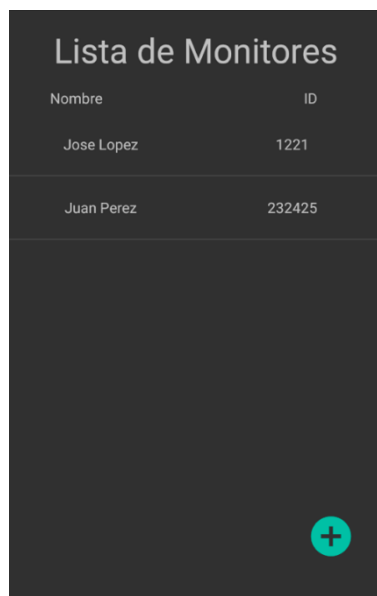


Ilustración 5.11 –
activity_mostrar_monitores.xml

- **CrearMonitor.java:** Es la actividad con la el propietario crea un nuevo monitor. Esta actividad viene de la actividad MostrarMonitores.java. En esta actividad encontramos los diferentes campos de texto para rellenar con los datos del monitor. Destacar dos funciones:
 - **String enviarPost(String id, string pass, String nombre):** Es la función encargada de enviar los datos del usuario al fichero addUsuario.php.
 - **Int onJSON(String rspta):** Esta función es la encargada de convertir el objeto JSON que devuelve el fichero addUsuario.php. Este archivo devolverá un 1 o un 0 en caso de que el usuario se haya podido crear o no.

Su vista asociada es `activity_mostrar_monitores.xml`.

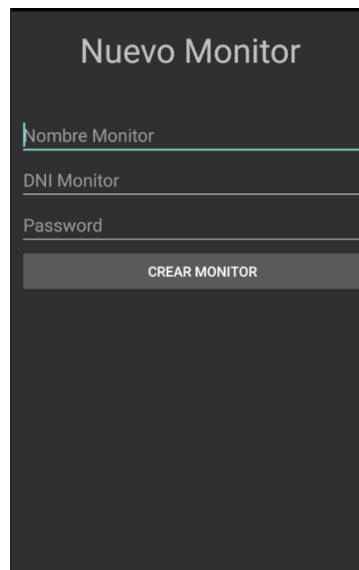


Ilustración 5.12 –
`activity_crear_monitor.xml`

- **InformacionUsuario.java:** Es la actividad correspondiente con la vista en la que aparece la información relativa a un usuario que seleccionemos, a esta actividad solo podrá acceder el propietario, será desde la vista `MostrarClientes.java` o `MostrarMonitores.java` desde las cuales se envíen los datos de los usuarios que se mostraran. Encontraremos definido un botón para eliminar al usuario. Destaca la función:
 - `String eliminarUsuario(String id, String roll):` Es la función encargada de enviar los datos al fichero `eliminarUsuario.php`.

Su vista asociada es `activity_informacion_usuario.xml`.

activity_informacion_usuario.xml

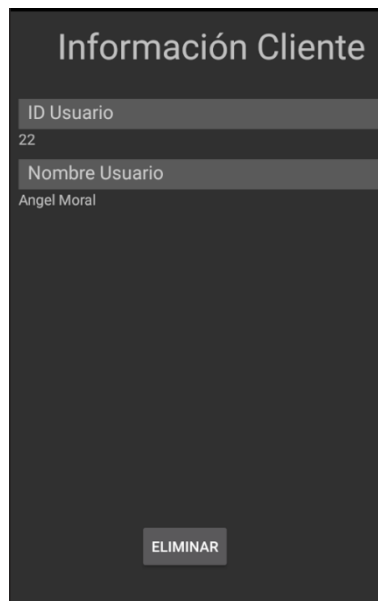


Ilustración 5.13 –
activity_informacion_usuario.xml

- Adaptadores:
 - userListAdapter: Es el adaptador que se utiliza para mostrar las lista de usuarios, mostrando el id del usuario y el nombre. Su vista asociada es adapter_view_layout.xml.
 - clasesListAdapter: Es el adaptador que se utiliza para mostrar las listas de las clases, mostrando todos los datos relativos a una clase. Su vista asociada es adapter_view_layout2.xml.

5.3. Pruebas y validación

En este apartado realizaremos una serie de pruebas para comprobar el correcto funcionamiento del sistema, para de esta manera detectar si los hubiera los errores para subsanarlos y poder tener un software de la mayor calidad posible.

Para la realización de la pruebas se utilizará una herramienta de “Testing” facilitada por el navegador Google Chrome que nos permite realizar peticiones y obtener respuestas en un sistema con arquitectura Cliente/Servidor.

Esta herramienta se denomina “Advanced REST client” y lo podemos encontrar en el catálogo de aplicaciones de Google Chrome en el siguiente enlace: <https://chrome.google.com/webstore/detail/advanced-rest-client/hgmloofddfnphfgcellkdfbfbjeloo>

Su instalación es muy sencilla únicamente tendremos que añadirla a nuestro navegador y esta aparecerá en la pantalla de aplicaciones.

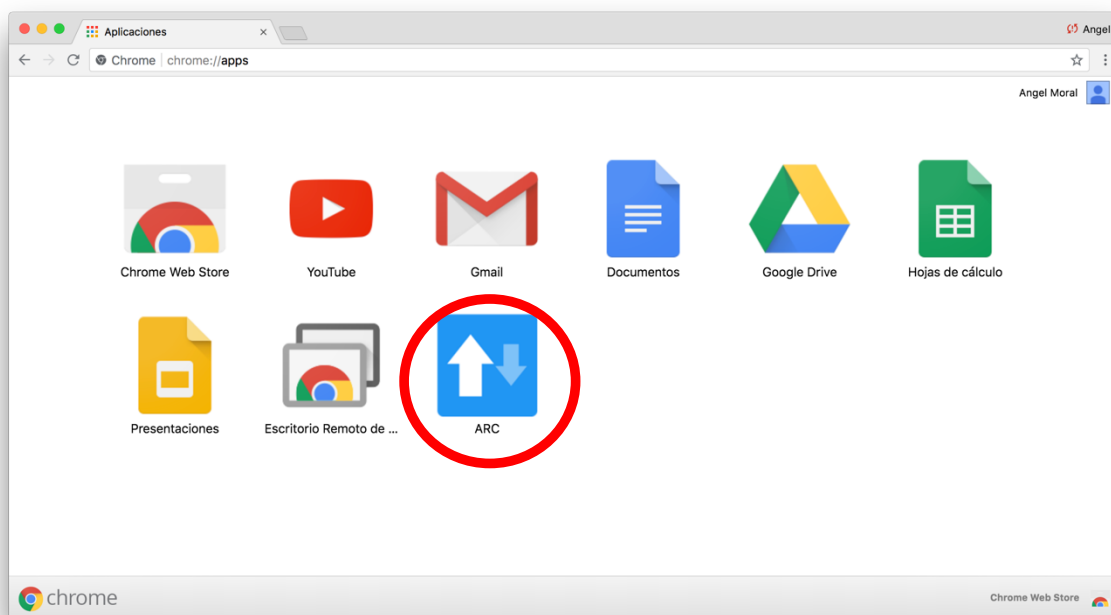


Ilustración 5.14 - Aplicaciones Google Chrome

Una vez pulsemos sobre su icono se ejecutará la aplicación y podremos comenzar a utilizarla.

Su utilización es muy sencilla, únicamente hay que especificar la URL a la que se realizará la petición, el tipo de petición e introducir los parámetros necesarios para poder llevar a cabo la petición. Una vez realizado esto se enviará la petición y en la parte inferior aparecerá la respuesta del servidor.

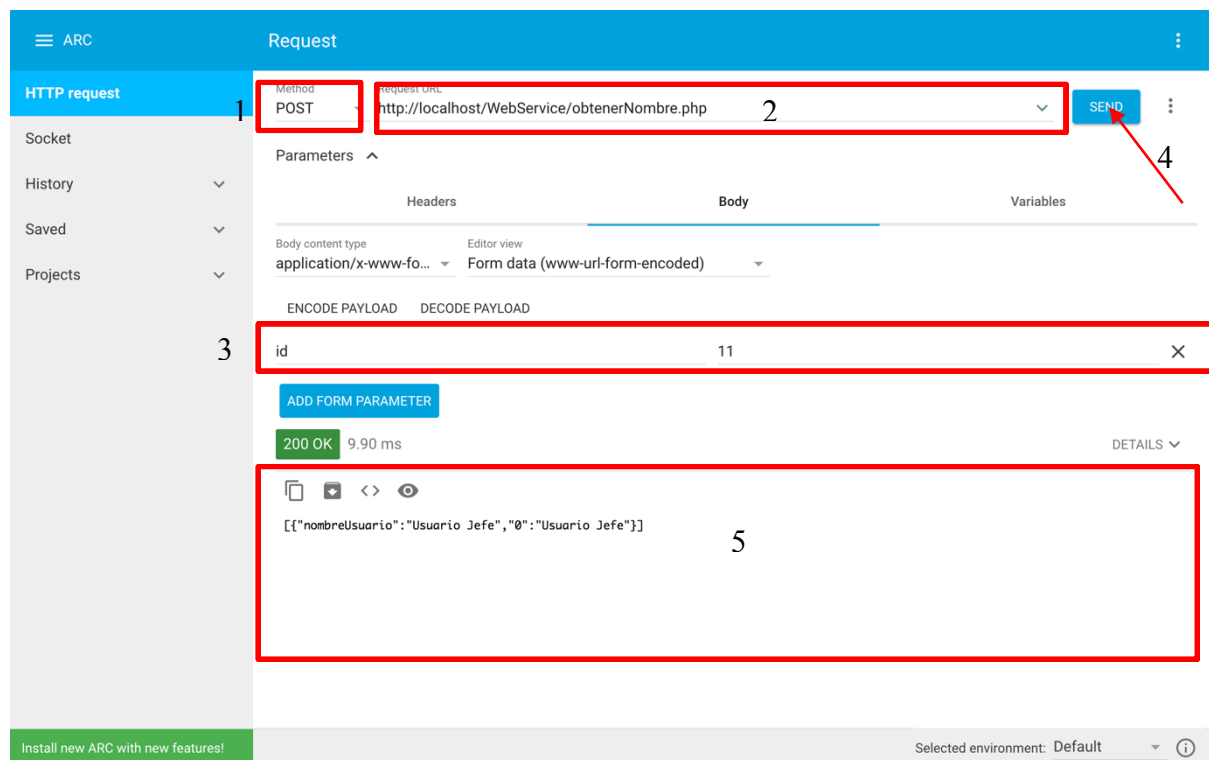


Ilustración 5.15 – Advanced REST cliente

Los pasos recomendados para realizar una petición son los siguientes:

1. Seleccionar el tipo de petición a realizar.
2. Introducir la URL a la cual se va a realizar la petición.
3. Introducir los parámetros necesarios para llevar a cabo la petición.
4. Enviar la petición.
5. Comprobar la respuesta obtenida.

A continuación se definirán una serie de casos de prueba y los resultados obtenidos.

Casos de prueba

- **Test 1.**

Acción	Iniciar sesión
Entrada	Identificador y contraseña
Salida	Datos del usuario
Rol	Propietario, Monitor, Cliente

Tabla 5.1 - Test 1

- **Test 2.**

Acción	Ver lista de Clientes
Entrada	Roll
Salida	Listado con los identificadores y el nombre de los clientes.
Rol	Propietario

Tabla 5.2 - Test 2

- **Test 3.**

Acción	Crear cliente
Entrada	Datos del nuevo cliente
Salida	"[]"(Creado correctamente) o Datos del cliente si ya existe
Rol	Propietario

Tabla 5.3 - Test 3

- **Test 4.**

Acción	Ver lista de monitores
Entrada	Roll
Salida	Listado con los identificadores y el nombre de los monitores.
Rol	Propietario

Tabla 5.4 - Test 4

- **Test 5.**

Acción	Crear monitor
Entrada	Datos del nuevo monitor
Salida	“[]”(Creado correctamente) o Datos del monitor si ya existe
Rol	Propietario

Tabla 5.5 - Test 5

- **Test 6.**

Acción	Ver listado de clases
Entrada	
Salida	Listado con los datos de todas las clases.
Rol	Propietario, Cliente

Tabla 5.6 - Test 6

- **Test 7.**

Acción	Crear nueva clase
Entrada	Datos datos de la nueva clase
Salida	“0” o “Error”
Rol	Propietario

Tabla 5.7 - Test 7

- **Test 8.**

Acción	Ver clases asignadas a un monitor
Entrada	Identificador del monitor
Salida	Listado con los datos de las clases que tiene que realizar el monitor con ese identificador.
Rol	Monitor

Tabla 5.8 - Test 7

- **Test 9.**

Acción	Obtener nombre Usuario
Entrada	Identificador de Usuario.
Salida	Nombre del usuario.
Rol	Propietario, Monitor, Cliente.

Tabla 5.9 - Test 9

- **Test 10.**

Acción	Apuntarse a una clase
Entrada	Identificador de Usuario e Identificador de Clase
Salida	"[]"(Inscripción realizada) o datos inscripción (Si el usuario ya estaba apuntado).
Rol	Cliente

Tabla 5.10 - Test 10

- **Test 11.**

Acción	Ver clases a las que un cliente se ha apuntado.
Entrada	Identificador de Usuario.
Salida	Datos de las clases a las que e ha apuntado.
Rol	Cliente

Tabla 5.11 - Test 11

- **Test 12.**

Acción	Eliminar Cliente o Monitor
Entrada	Identificador de usuario.
Salida	"[]"(Eliminación Realizada) o Error
Rol	Propietario

Tabla 5.12 - Test 12

- **Test 13.**

Acción	Clientes apuntados a una clase
Entrada	Identificador de Clase
Salida	Identificador y nombre de los Clientes apuntados.
Rol	Propietario, Monitor.

Tabla 5.13 - Test 13

Con los resultados de estos test finaliza el capítulo donde se han expuesto todos los aspecto relacionados con la implementación de este proyecto tanto el el apartado del Servicio Web como el del desarrollo de la aplicación Android.

Resultado de los casos de prueba

TEST	RESULTADO
Test 1	[{"id":"11","0":"11","pass":"11","1":"11","roll":"0","2":"0","nombreUsuario":"Usuario Jefe","3":"Usuario Jefe"}]
Test 2	[{"id":"22","0":"22","pass":"22","1":"22","roll":"2","2":"2","nombreUsuario":"Angel Moral ","3":"Angel Moral "},{ "id":"24232556","0":"24232556","pass":"2222222","1":"2222222","roll":"2","2":"2","nombreUsuario":"Jorge Moral","3":"Jorge Moral"}, {"id":"77342356","0":"77342356","pass":"123456","1":"123456","roll":"2","2":"2","nombreUsuario":"Jose Fuentes","3":"Jose Fuentes"}, {"id":"123","0":"123","pass":"56","1":"56","roll":"2","2":"2","nombreUsuario":"Usu1","3":"Usu1"}, {"id":"23","0":"23","pass":"23","1":"23","roll":"2","2":"2","nombreUsuario":"usu2","3":"usu2"}, {"id":"333","0":"333","pass":"333","1":"333","roll":"2","2":"2","nombreUsuario":"usuario3","3":"usuario3"}, {"id":"22333","0":"22333","pass":"50","1":"50","roll":"2","2":"2","nombreUsuario":"usuPrueba","3":"usuPrueba"}]
Test 3	[]
Test 4	[{"id":"23242544","0":"23242544","pass":"123456","1":"123456","roll":"1","2":"1","nombreUsuario":"Javier Gutierrez","3":"Javier Gutierrez"}, {"id":"1221","0":"1221","pass":"1221","1":"1221","roll":"1","2":"1","nombreUsuario":"Jose Lopez","3":"Jose Lopez"}, {"id":"232425","0":"232425","pass":"232425","1":"232425","roll":"1","2":"1","nombreUsuario":"Juan Perez","3":"Juan Perez"}]
Test 5	[]
Test 6	[{"id":"48","0":"48","clase":"Cardio Box","1":"Cardio Box","idmonitor":"232425","2":"232425","nombremonitor":"Juan Perez","3":"Juan Perez","fecha":"2018-08-30","4":"2018-08-30","hora":"11:0","5":"11:0","capacidad":"18","6":"18"}, {"id":"47","0":"47","clase":"Spinnig","1":"Spinnig","idmonitor":"1221","2":"1221","nombremonitor":"Jose Lopez","3":"Jose Lopez","fecha":"2018-08-24","4":"2018-08-24","hora":"18:18","5":"18:18","capacidad":"17","6":"17"}, {"id":"50","0":"50","clase":"Spinnig","1":"Spinnig","idmonitor":"23242544","2":"23242544","nombremonitor":"Javier Gutierrez","3":"Javier Gutierrez","fecha":"2018-08-21","4":"2018-08-21","hora":"21:30","5":"21:30","capacidad":"2","6":"2"}, {"id":"49","0":"49","clase":"Spinnig","1":"Spinnig","idmonitor":"23242544","2":"23242544","nombremonitor":"Javier Gutierrez","3":"Javier Gutierrez","fecha":"2018-08-19","4":"2018-08-19","hora":"20:18","5":"20:18","capacidad":"10","6":"10"}, {"id":"45","0":"45","clase":"Spinnig","1":"Spinnig","idmonitor":"1221","2":"1221","nombremonitor":"Jose Lopez","3":"Jose Lopez","fecha":"2018-07-31","4":"2018-07-31","hora":"23:20","5":"23:20","capacidad":"10","6":"10"}]

	<pre> ": "30", "6": "30"}, {"id": "44", "0": "44", "clase": "Spinnig", "1": "Spinnig", "idmonitor": "1221", "2": "1221", "nombremonitor": "Jose Lopez", "3": "Jose Lopez", "fecha": "2018-07-27", "4": "2018-07-27", "hora": "23:13", "5": "23:13", "capacidad": "20", "6": "20"}, {"id": "17", "0": "17", "clase": "Spinnig", "1": "Spinnig", "idmonitor": "1221", "2": "1221", "nombremonitor": "Jose Lopez", "3": "Jose Lopez", "fecha": "2018-07-11", "4": "2018-07-11", "hora": "20:30", "5": "20:30", "capacidad": "10", "6": "10"}, {"id": "19", "0": "19", "clase": "Spinnig", "1": "Spinnig", "idmonitor": "1221", "2": "1221", "nombremonitor": "Jose Lopez", "3": "Jose Lopez", "fecha": "2018-07-11", "4": "2018-07-11", "hora": "20:35", "5": "20:35", "capacidad": "20", "6": "20"}, {"id": "43", "0": "43", "clase": "Spinnig", "1": "Spinnig", "idmonitor": "1221", "2": "1221", "nombremonitor": "Jose Lopez", "3": "Jose Lopez", "fecha": "2018-07-11", "4": "2018-07-11", "hora": "23:8", "5": "23:8", "capacidad": "1", "6": "1"}] </pre>
Test 7	0
Test 8	<pre> [{"id": "50", "0": "50", "clase": "Spinnig", "1": "Spinnig", "idmonitor": "23242544", "2": "23242544", "nombremonitor": "Javier Gutierrez", "3": "Javier Gutierrez", "fecha": "2018-08-21", "4": "2018-08-21", "hora": "21:30", "5": "21:30", "capacidad": "2", "6": "2"}, {"id": "49", "0": "49", "clase": "Spinnig", "1": "Spinnig", "idmonitor": "23242544", "2": "23242544", "nombremonitor": "Javier Gutierrez", "3": "Javier Gutierrez", "fecha": "2018-08-19", "4": "2018-08-19", "hora": "20:18", "5": "20:18", "capacidad": "10", "6": "10"}] </pre>
Test 9	<pre> [{"nombreUsuario": "Javier Gutierrez", "0": "Javier Gutierrez"}] </pre>
Test 10	[]
Test 11	<pre> [{"id": "48", "0": "48", "clase": "Cardio Box", "1": "Cardio Box", "idmonitor": "232425", "2": "232425", "nombremonitor": "Juan Perez", "3": "Juan Perez", "fecha": "2018-08-30", "4": "2018-08-30", "hora": "11:0", "5": "11:0", "capacidad": "18", "6": "18"}] </pre>
Test 12	[]
Test 13	<pre> [{"id": "22", "0": "22", "nombreUsuario": "Angel Moral ", "1": "Angel Moral"}, {"id": "23", "0": "23", "nombreUsuario": "usu2", "1": "usu2"}] </pre>

Tabla 5.14 - Resultados casos de prueba

CAPÍTULO 6

CONCLUSIÓN Y TRABAJO FUTUROS

6.1. Conclusión

El presente proyecto consiste en la gestión y acceso a los servicios de una organización o empresa mediante una aplicación móvil, es un trabajo genérico que en nuestro caso se ha orientado a la gestión y acceso de los servicios de un gimnasio con vista a que el propietario del mismo pudiera tener toda la información referente al gimnasio en su móvil, como son la gestión de Clientes, personal de gimnasio y clases que se desarrolla y que tanto el personal de gimnasio como los usuarios del mismo pudieran utilizar la aplicación en beneficio propio.

Me gustaría destacar que el desarrollo del proyecto ha presentado distintas dificultades, especialmente en el desarrollo de la aplicación Android al no haber trabajado nunca con la herramienta de desarrollo Android Studio, sin embargo y como ya mencioné en el capítulo 3 Android posee una gran comunidad de desarrolladores con lo que es fácil encontrar información especialmente de los aspectos básicos para poder a comenzar el desarrollo.

Por otro lado el desarrollo del servicio web ha resultado más sencillo al conocer de antemano la herramienta XAMPP gracias a su utilización en la asignatura del grado “Tecnologías basadas en web” y a su interfaz muy intuitiva no ha presentado grandes dificultades.

Para concluir me gustaría mencionar que el desarrollo de aplicaciones móviles es uno de los apartados de la ingeniería informática que más me apasionan por el enorme potencial que tienen tanto en el ámbito del entretenimiento como en el ámbito empresarial, por lo que ha supuesto una motivación extra para realizar este proyecto.

6.2. Trabajos futuros

Puedo afirmar que los objetivos redactados en el capítulo 1 de esta memoria han sido desarrollados y todas las funcionalidades del sistema se encuentran operativas, aunque me gustaría mencionar que el sistema puede evolucionar e incorporar nuevas funcionalidades muy interesantes, pero que por la falta de tiempo lógica en este tipo de proyecto no se han podido incorporar, como podrían ser:

- Reserva de espacios deportivos en distintas franjas horarias que formen parte del gimnasio por parte de los usuario. Por ejemplo reserva de piscas de futbol, tenis, pádel etc.
- Gestión por parte del propietario de estos espacios deportivos.
- Que los monitores puedan realizar grupos de clientes a los que impartir sus clases.
- Incluir notificaciones que avisen al usuario cuando este disponible una nueva clase.
- Incluir notificaciones que recuerden al usuario que próximamente tiene una clase.

A demás de una mejora en la interfaz de usuario con la inclusión de fotos de los clientes, más datos personales como incluir la dirección el numero de teléfono, etc por que recordamos que esto es simplemente un prototipo.

Bibliografía

Servicio Web [Fecha última consulta: 17/07/2018]

[1] <http://camoralesma.googlepages.com/articulo2.pdf>

Página World Wide Web Consortium España [Fecha última consulta: 17/07/2018]

[2] <https://www.w3c.es/>

Página OASIS Web [Fecha última consulta: 17/07/2018]

[3] <https://www.oasis-open.org>

Página WS-I [Fecha última consulta: 17/07/2018]

[4] <http://www.ws-i.org/>

Que podemos hacer con los Servicios Web [Fecha última consulta: 17/07/2018]

[5] <https://www.ibm.com/developerworks/ssa/webservices/newto/service.html>

Ventajas de los Servicios Web 1 [Fecha última consulta: 17/07/2018]

[6] https://es.wikipedia.org/wiki/Servicio_web

Ventajas de los Servicios Web 2 [Fecha última consulta: 17/07/2018]

[7] http://www.eumed.net/tesis-doctorales/2007/cavl/Beneficios_de_los_servicios_Web.htm

XAMPP [Fecha última consulta: 19/07/2018]

[8] <https://es.wikipedia.org/wiki/XAMPP>

Licencia GPL [Fecha última consulta: 19/07/2018]

[9] <https://www.gnu.org/licenses/gpl-faq.es.html>

Servidor HTTP Apache [Fecha última consulta: 19/07/2018]

[10] <http://www.ibrugor.com/blog/apache-http-server-que-es-como-funciona-y-para-que-sirve/>

Apache Software Foundation [Fecha última consulta: 19/07/2018]

[11] <https://httpd.apache.org/>

MySql [Fecha última consulta: 19/07/2018]

[12] <https://www.oracle.com/es/mysql/>

Qué es y para que sirve MySQL [Fecha última consulta: 19/07/2018]

[13] <http://culturacion.com/que-es-y-para-que-sirve-mysql/>

Que es MariaDB, ventajas y desventajas [Fecha última consulta: 19/07/2018]

[14] <http://www.tuprogramacion.com/bases-de-datos/mariadb-vs-mysql/>

PHP [Fecha última consulta: 23/07/2018]

[15] <https://es.wikipedia.org/wiki/PHP>

REST, Características y ventajas [Fecha última consulta: 23/07/2018]

[16] <https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos>

Android y sus características [Fecha última consulta: 24/07/2018]

[17] “Android 100% - Ramón Invarato Menéndez” <https://jarroba.com/libro-android-100-gratis>

Arquitectura Android [Fecha última consulta: 24/07/2018]

[18] https://es.wikipedia.org/wiki/Android_-_Arquitectura

Características y tabla comparativa de los sistemas operativos móviles más usados [Fecha última consulta: 24/07/2018]

[19] <https://jmacuna.tecnoblog.guru/2017/03/sistemas-operativos-moviles.html>

Android Nougat [Fecha última consulta: 24/07/2018]

[20] https://www.android.com/intl/es-419_mx/versions/nougat-7-0/

Escenarios [Fecha última consulta: 02/08/2018]

[21] Potts, C., Takahashi, K., Antón, A. I., “Inquiry-Based Requirements Analysis”, IEEE Software, Vol. 11, N° 2, 1994, pp. 21-32 . -
http://www.exa.unicen.edu.ar/catedras/ingrequi/index_archivos/Escenarios-07.pps

Diseño software

[22] Tema 9 de la teoría de la asignatura Fundamentos de Ingeniería del Software del grado en ingeniería Informática de la Universidad de Jaén. Año 2015-2016.

Diseño de la interfaz

[23] Diseño gráfico de interfaces de Usuario- Teoría de la Asignatura Iteración Persona Ordenador del grado en ingeniería Informática de la Universidad de Jaén. Año 2016-2017.

Java [Fecha última consulta: 12/08/2018]

[24] https://java.com/es/about/whatis_java.jsp?bucket_value=desktop-safari11-osx10126&in_query=no

Android Studio [Fecha última consulta: 12/08/2018]

[25] <https://developer.android.com/studio/intro/?hl=es-419>

NetBeans [Fecha última consulta: 12/08/2018]

[26] https://netbeans.org/index_es.html

Activity – Actividades en Android Studio [Fecha última consulta: 12/08/2018]

[27] <https://developer.android.com/guide/components/activities?hl=es-419>

SOAP [Fecha última consulta: 20/08/2018]

[28] <https://searchmicroservices.techtarget.com/definition/SOAP-Simple-Object-Access-Protocol>

HTTP – Hypertext Transfer Protocol [Fecha última consulta: 20/08/2018]

[29] <https://searchwindevelopment.techtarget.com/definition/HTTP>

XML – Extensible Markup Language [Fecha última consulta: 20/08/2018]

[30] <https://searchmicroservices.techtarget.com/definition/XML-Extensible-Markup-Language>

SQL – Structured Query Language [Fecha última consulta: 20/08/2018]

[31] https://www.w3schools.com/sql/sql_intro.asp

MyISAM [Fecha última consulta: 20/08/2018]

[32] <https://dev.mysql.com/doc/refman/8.0/en/myisam-storage-engine.html>

InnoDB [Fecha última consulta: 20/08/2018]

[33] <https://dev.mysql.com/doc/refman/8.0/en/innodb-storage-engine.html>

SolidDB [Fecha última consulta: 20/08/2018]

[34] <https://teambblue.unicomsi.com/products/soliddb/>

memcache [Fecha última consulta: 20/08/2018]

[35] <https://memcached.org/>

Aria - MariaDB [Fecha última consulta: 20/08/2018]

[36] <https://mariadb.com/kb/en/library/aria-storage-engine/>

XtraDB - MariaDB [Fecha última consulta: 20/08/2018]

[37] <https://mariadb.com/kb/en/library/about-xtradb/>

HTML [Fecha última consulta: 20/08/2018]

[38] https://www.w3schools.com/html/html_intro.asp

Node.JS [Fecha última consulta: 20/08/2018]

[39] <https://nodejs.org/es/about/>

ASP.NET [Fecha última consulta: 20/08/2018]

[40] <https://www.asp.net/learn>

Open Handset Alliance [Fecha última consulta: 20/08/2018]

[41] https://www.openhandsetalliance.com/oha_faq.html

Informe Mobile en España y en el Mundo [Fecha última consulta: 20/08/2018]

[42] https://www.amic.media/media/files/file_352_1289.pdf

¿Por qué las empresas necesitan una App móvil? [Fecha última consulta: 20/08/2018]

[43] <http://www.europapress.es/comunicados/sociedad-00909/noticia-comunicado-empresas-necesitan-app-movil-20180320093849.html>

JSON [Fecha última consulta: 20/08/2018]

[44] <https://www.json.org/>

ANEXO I

MANUAL DE INSTALACIÓN DEL SERVIDOR WEB

Como se explico en el capítulo 2 con el análisis de las tecnología, para la realización de este proyecto se ha utilizado la herramienta XAMPP la cual nos proporciona todo lo necesario para poner en marcha un Servicio Web de una manera muy sencilla y con un alto rendimiento.

Su instalación no conlleva ninguna dificultad, a continuación se muestra la instalación en un equipo con sistema operativo MAC OS, en el caso de poseer otro sistema operativo el proceso de instalación será el mismo, habrá que seguir los siguiente pasos.

1. Acceder a la página oficial de XAMPP para proceder a descargarlo.
<https://www.apachefriends.org/es/download.html>
2. Seleccionar la versión de XAMPP en función de nuestro sistema operativo.
3. Realizar la instalación de XAMP con la ayuda del asistente de instalación.

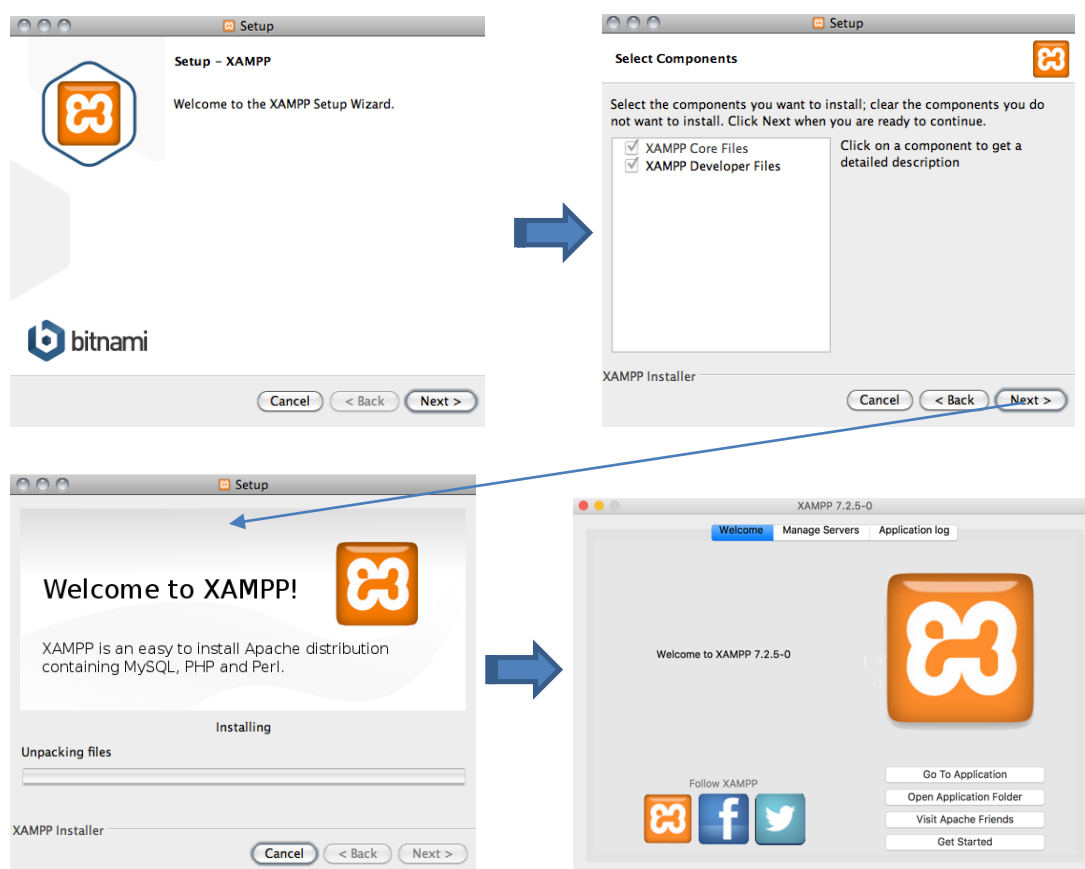


Ilustración 1 Anexo I - Instalación XAMPP

4. Iniciar los servicios que nos proporciona XAMPP.

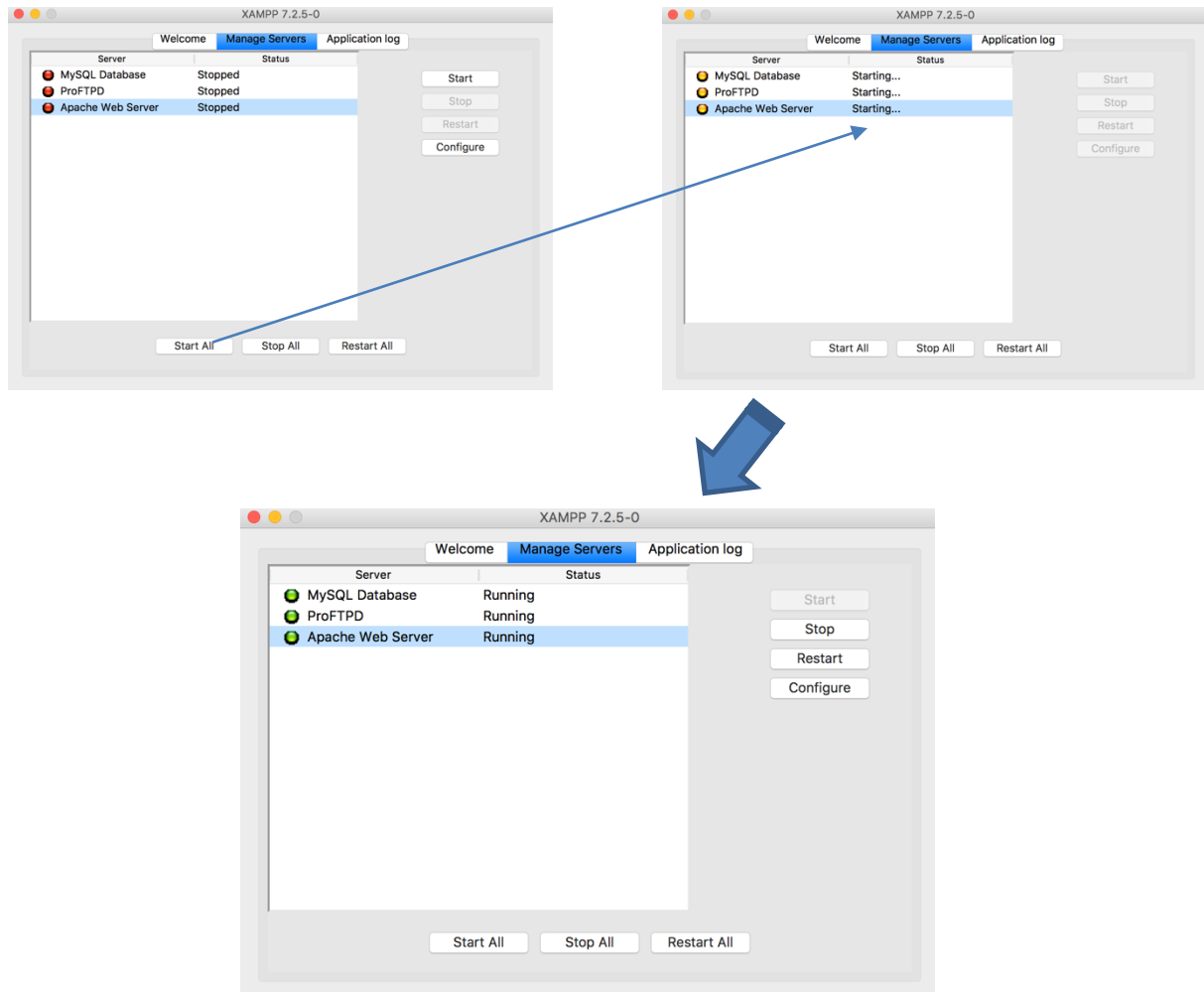
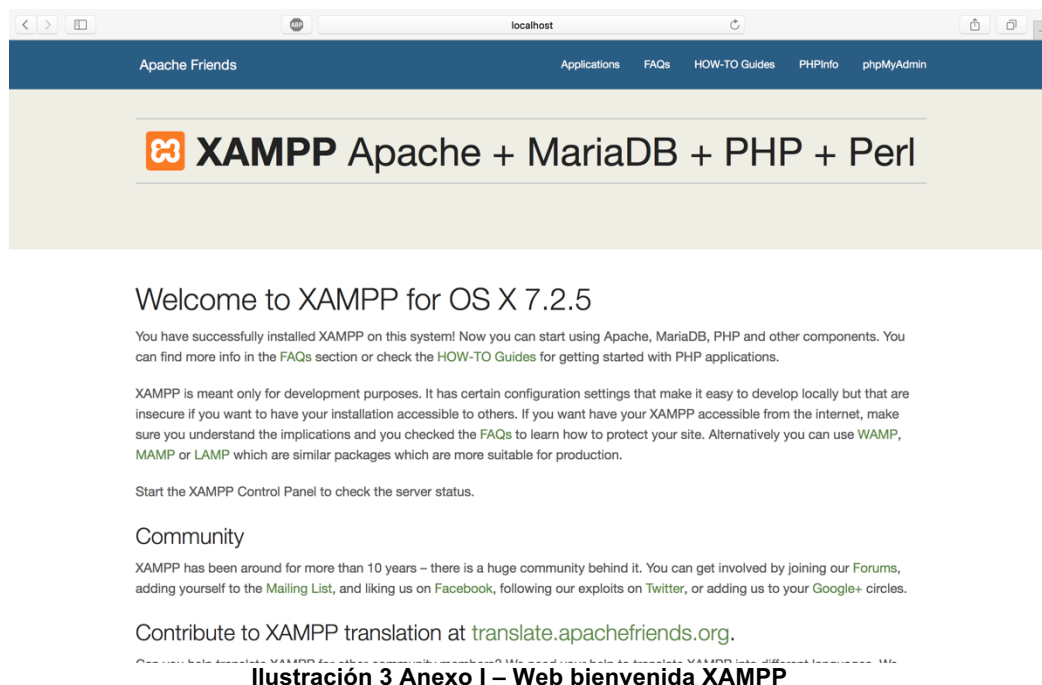


Ilustración 2 Anexo I – Puesta en funcionamiento Servicio Web

5. Comprobar que el servicio web funciona correctamente. Para ello nos dirigiremos al navegador web e introduciremos <http://localhost/dashboard/> en la URL del navegador. Si todo esta correcto aparecerá la siguiente pantalla.



Una vez puesto en marcha el Servicio Web habrá que importar la base de datos de nuestra aplicación e incluir los ficheros PHP encargados de realizar las consultas.

6. Añadir los ficheros PHP. Para ello nos dirigiremos a la carpeta raíz de XAMPP, una vez ahí accederemos al directorio htdocs donde copiaremos el archivo WebService.zip que se adjunta en este proyecto y lo descomprimiremos.
 - En el caso de Windows la ruta absoluta donde copiar el archivo WebService.zip sería: C:/XAMPP/htdocs/
 - En el caso de MAC OS: /Applications/XAMPP/xamppfiles/htdocs/

7. Crear la base de datos. Para la creación de la base de datos nos dirigiremos de nuevo a nuestro navegador web y escribiremos en la URL de nuevo <http://localhost/dashboard/> una vez aparezca la pantalla de bienvenida pulsaremos en la zona superior derecha “phpMyAdmin”.

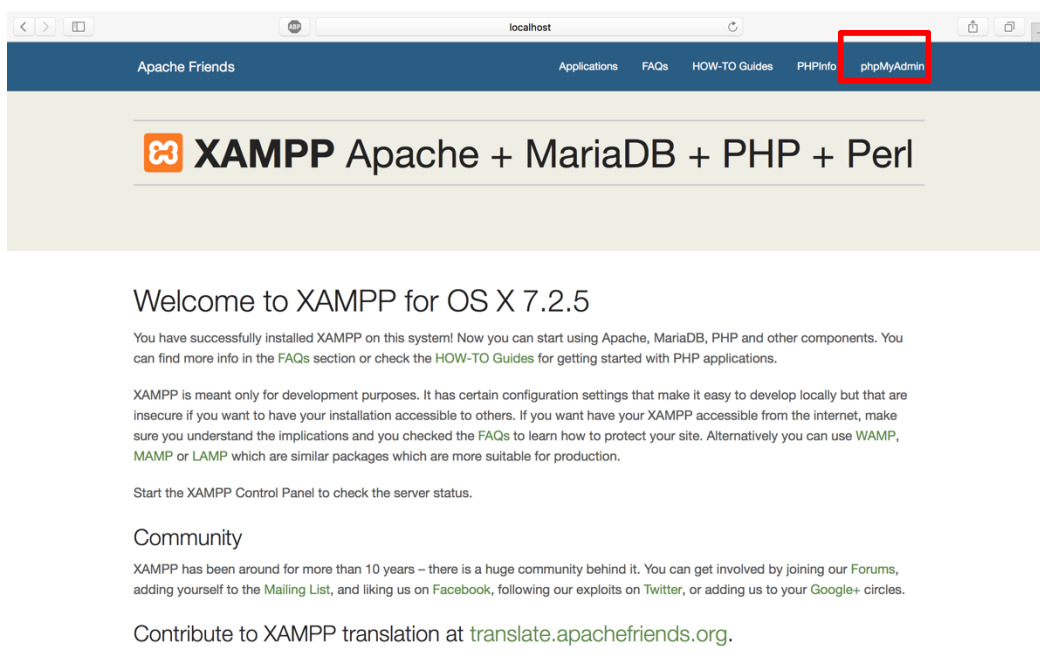


Ilustración 4 Anexo I – Web bienvenida XAMPP 2

Se mostrará la siguiente pantalla y pulsaremos sobre “Nueva” en el menú desplegable de la izquierda.

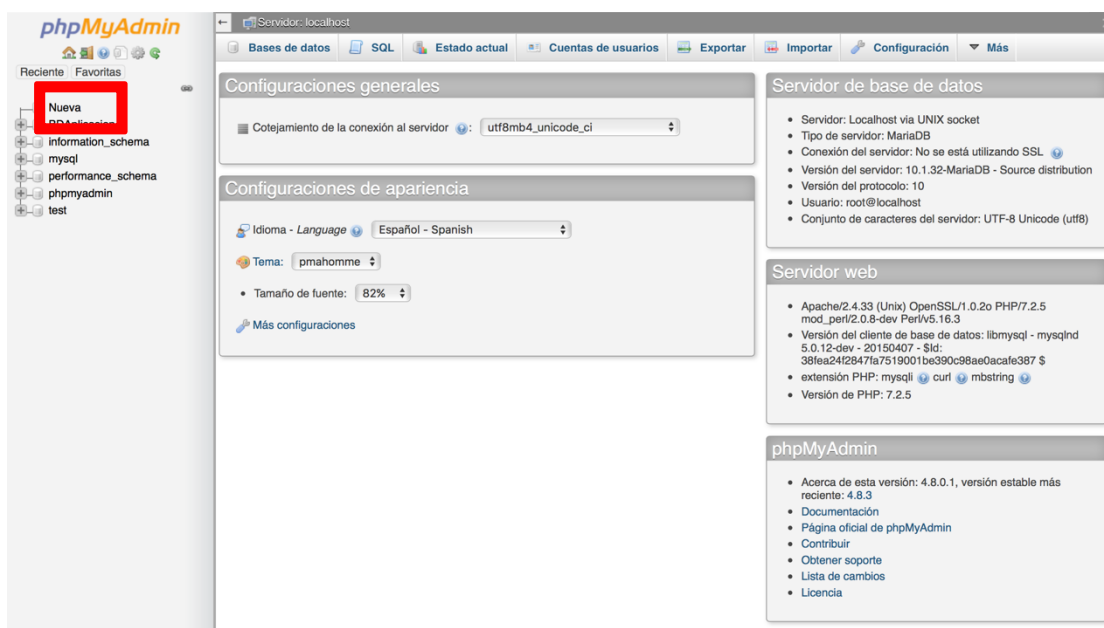


Ilustración 5 Anexo I – Web phpMyAdmin

A continuación, aparecerá la pantalla de creación de la Base de Datos donde introduciremos el nombre de la misma “BDAplicacion” de la siguiente manera y pulsaremos en “Crear”.

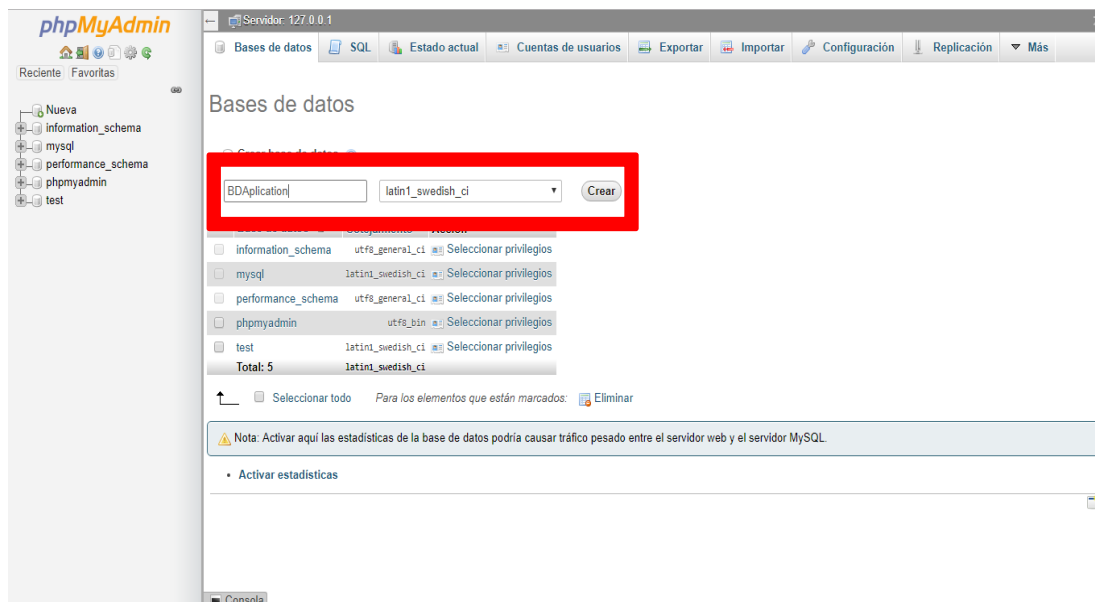


Ilustración 6 Anexo I – Creación Base de Datos

8. Importación de Tablas. El siguiente pasó será importar las tablas que componen nuestra base de datos para ello, volveremos a dirigirnos al menú de la izquierda seleccionaremos la base de datos (BDAplicacion) y en el menú horizontal superior seleccionaremos la opción “Importar”.

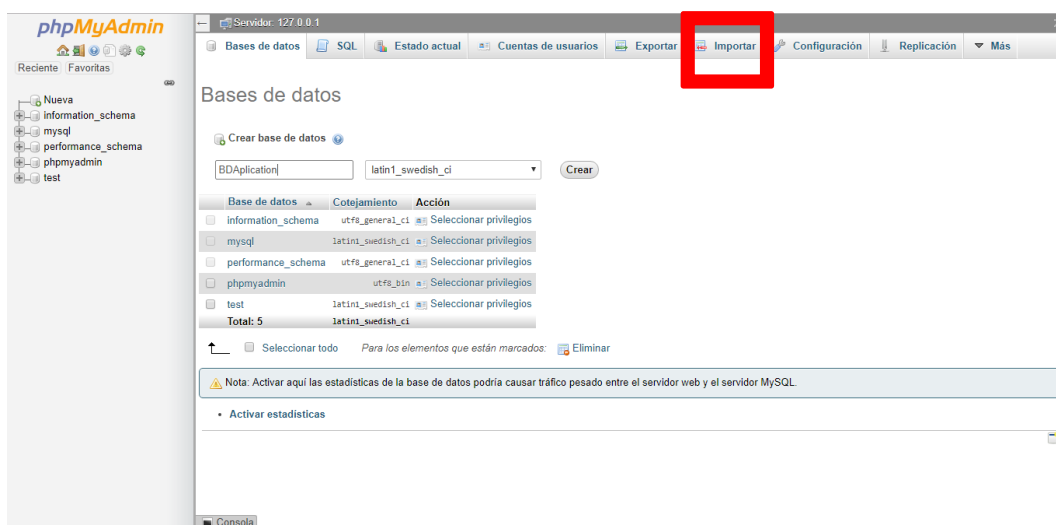


Ilustración 7 Anexo I – Importación Base de Datos

Una vez seleccionada esta opción, se mostrará la pantalla de la importación, para importar las tablas pulsaremos en la opción de “Seleccionar archivo” y se abrirá una pantalla donde encontraremos todos los directorios de nuestra computadora. Tendremos que buscar donde se encuentra almacenado el archivo “BDAplicacion.sql” que se adjunta en el proyecto y seleccionarlo. Una vez seleccionado pulsaremos el botón “Aceptar” de la parte inferior.

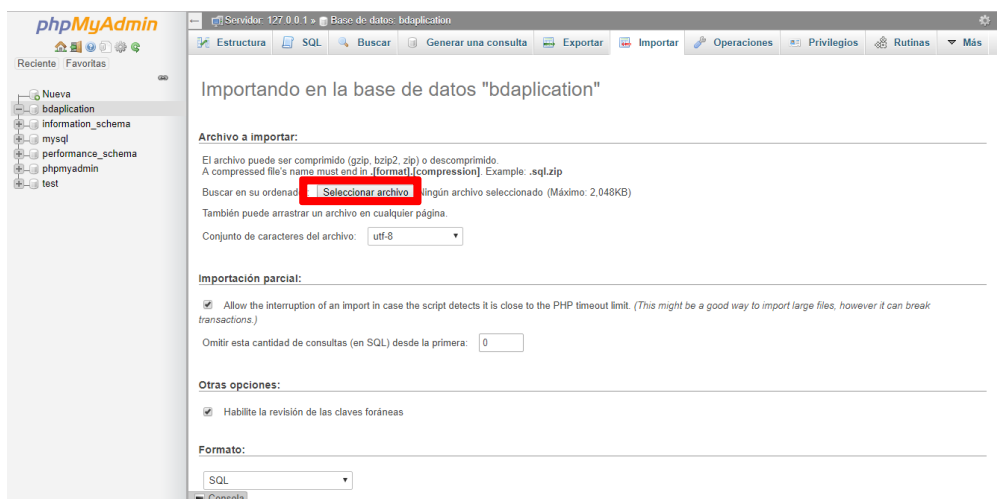


Ilustración 8 Anexo I – Importación Base de Datos 2

Una vez realizados todos estos pasos tendremos el servicio web instalado y ejecutándose de forma correcta.

ANEXO II

MANUAL DE INSTALACIÓN DE LA APLICACIÓN ANDROID

En este manual vamos a explicar como realizar la instalación de la aplicación Android. El proceso es muy sencillo y fácil de realizar.

Hay dos formas de poder utilizar la aplicación Android. Por un lado sería usando el emulador que nos proporciona Android Studio, la segunda instalarla en nuestro dispositivo Android.

Utilizar aplicación en Android Studio

Lo primero que tenemos que hacer es instalar Android Studio, es un proceso muy sencillo. Adjunto este tutorial de la web de Android Studio donde se explica de manera detallada: <https://developer.android.com/studio/install?hl=es-419>

También tendríamos que configurar el dispositivo virtual que nos proporciona Android Studio es importante que este dispositivo cumpla con los requisitos no funcionales especificados en esta memoria.

Una vez configurado Android Studio abriremos el proyecto que se adjunta en el CD de este proyecto AplicacionTFG.zip(previamente tendremos que descomprimirlo).

Nos dirigiremos a la clase java Global.java y cambiaremos la variable IP en función de la dirección IP que tenga el ordenador donde estamos utilizando Android Studio.

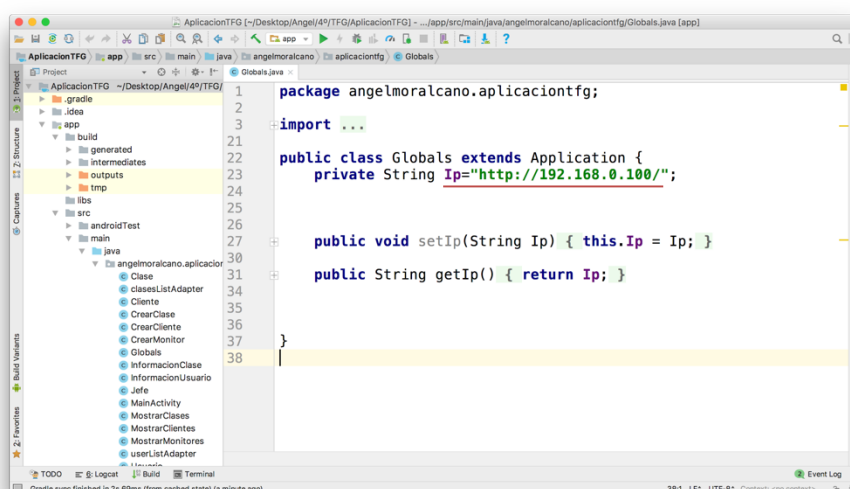


Ilustración 1 Anexo II - Instalación XAMPP

Por último ejecutaríamos el proyecto pulsando el botón “Play” y seleccionaremos el dispositivo previamente configurado y se abrirá el emulador de Android donde se ejecutará la aplicación.

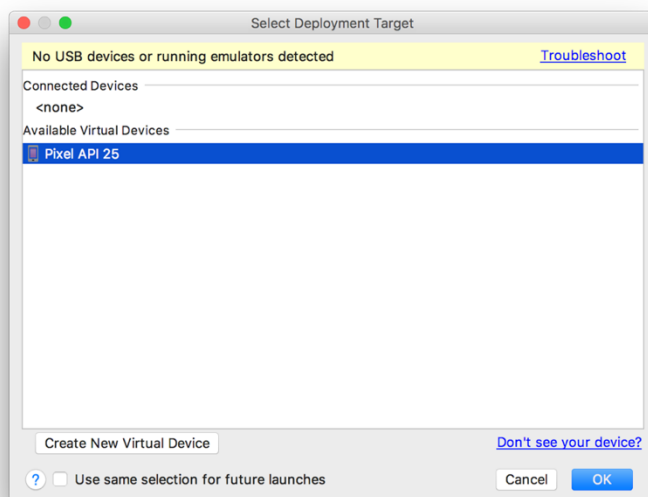


Ilustración 2 Anexo II - Instalación XAMPP

Instalar Aplicación Android en dispositivo móvil

Para instalar la aplicación en un dispositivo móvil tendremos que copiar (vía USB) el archivo AplicacionTFG.apk en alguna de las carpetas de nuestro dispositivo Android como por ejemplo “Descargas”, una vez copiada nos dirigiremos a este archivo desde nuestro dispositivo móvil y lo ejecutaremos, lo siguiente será realizar la instalación siguiendo los pasos del ayudante de instalación.

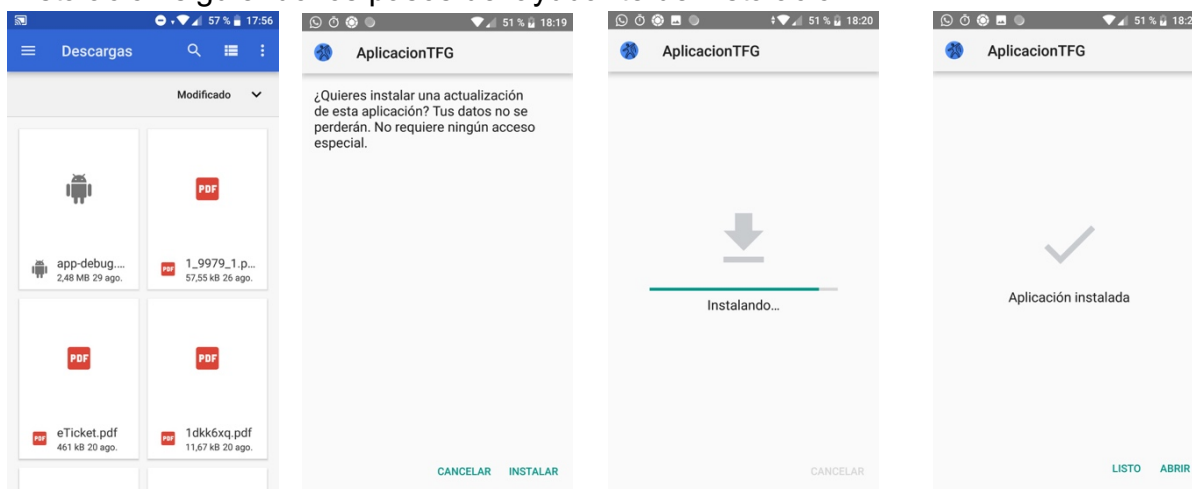


Ilustración 3 Anexo II - Instalación XAMPP

ANEXO III

MANUAL DE USUARIO

El siguiente manual mostrará las diferentes formas con las que el usuario puede interactuar con la aplicación Android.

La aplicación muestra la siguientes formas de interacción:

- Botones.
- Botones con imagen.
- Listas.
- Campos de texto.
- Diálogos
- Calendario y reloj

Botones

El usuario encontrará botones a lo largo de toda la aplicación, estos botones servirá para navegar entre las distintas vistas o para realizar acciones que se podrá identificar en el título del botón.

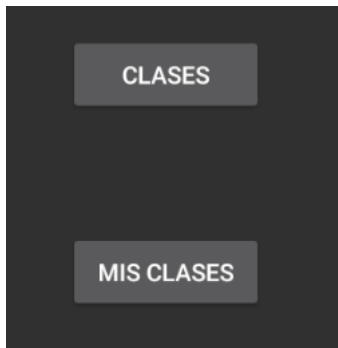


Ilustración 1 Anexo III - Botón

Botones con imagen

La función de estos botones son para acceder a las vistas de “creación”, se utilizan este tipo de botones en lugar de normales por se más fáciles de reconocer por el usuario y mejorar la estética de la aplicación.

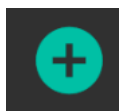
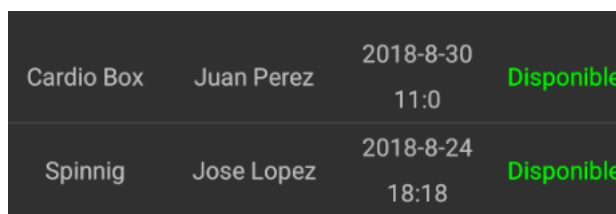


Ilustración 2 Anexo III - Botón con Imagen

Listas

Son un elemento esencial de la aplicación ya que es la forma con la que mostramos los datos que encontramos en la base de datos del sistema. Además es un elemento fácil de reconocer para el usuario y con el que interactuará seleccionando elementos de ella sin ninguna dificultad.

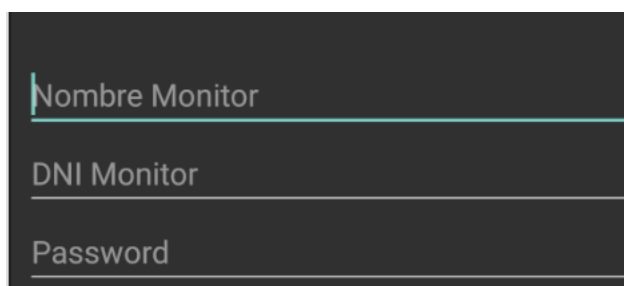


Cardio Box	Juan Perez	2018-8-30 11:0	Disponible
Spinnig	Jose Lopez	2018-8-24 18:18	Disponible

Ilustración 3 Anexo III - Lista

Campos de texto

Es el elemento con el cual podremos introducir datos a la aplicación.



Nombre Monitor

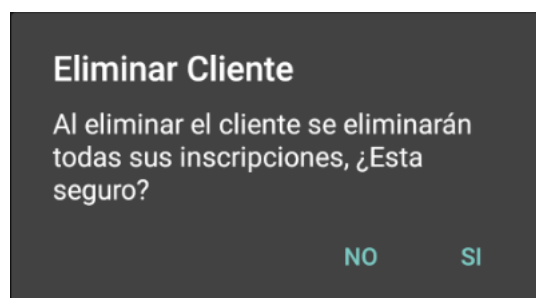
DNI Monitor

Password

Ilustración 4 Anexo III – Campos de texto

Diálogos

Es un elemento que utilizaremos para informar a usuario de un hecho o para confirmar acciones determinadas que conllevan una alta repercusión en el sistema como la eliminación de usuarios.



Eliminar Cliente

Al eliminar el cliente se eliminarán todas sus inscripciones, ¿Esta seguro?

NO SI

Ilustración 5 Anexo III – Diálogos

Calendario y reloj

Son dos elementos que utilizaremos para facilitar la introducción de la fecha y la hora.

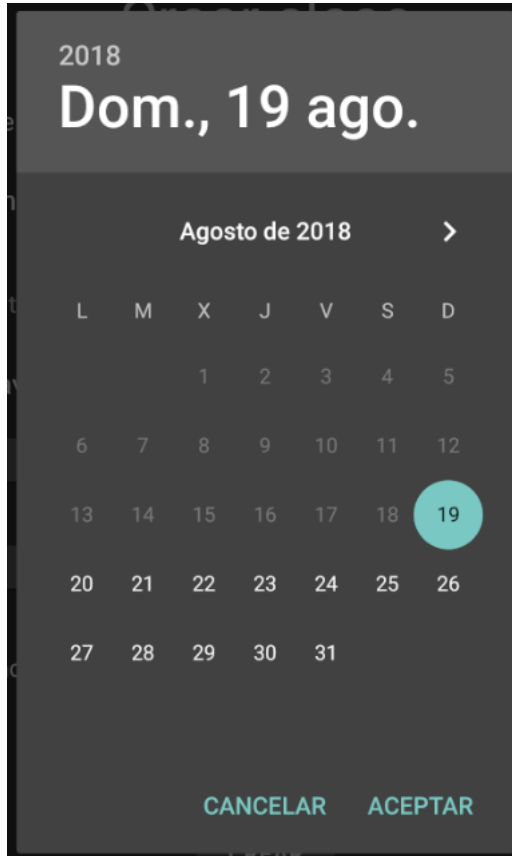


Ilustración 6 Anexo III - Calendario

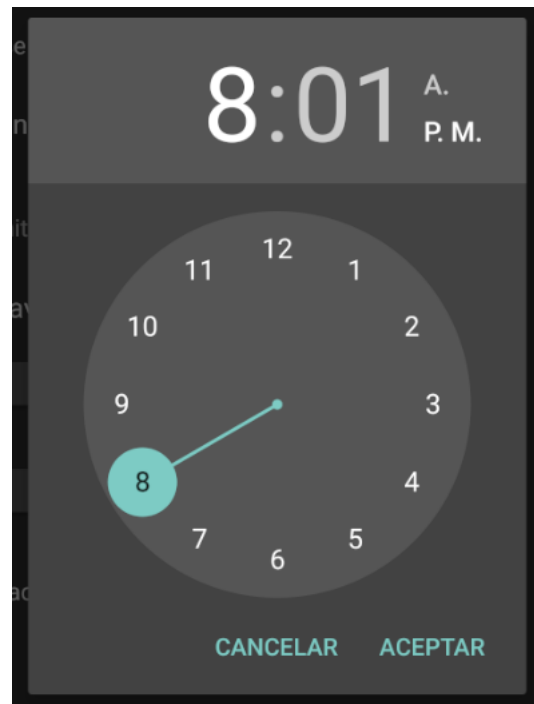


Ilustración 7 Anexo III - Reloj