UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería en Informática

TFG del Grado en Ingeniería Informática

Aplicación para el Acceso a Plataformas de ELearning desde Dispositivos Móviles

Presentado por Daniel Puente Gabarri
en Universidad de Burgos — 16 de mayo de 2017
Tutor: María Belén Vaquerizo García y Bruno Baruque Zanón

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería en Informática

D. nombre tutor, profesor del departamento de nombre departamento, área de nombre área.

Expone:

Que el alumno D. Daniel Puente Gabarri, con DNI 71347273-P, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado título de TFG.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 16 de mayo de 2017

Vº. Bº. del Tutor:	Vº. Bº. del co-tutor:
D. nombre tutor	D. nombre co-tutor

Resumen

En este primer apartado se hace una **breve** presentación del tema que se aborda en el proyecto.

Descriptores

Palabras separadas por comas que identifiquen el contenido del proyecto Ej: servidor web, buscador de vuelos, android ...

Abstract

A **brief** presentation of the topic addressed in the project.

Keywords

keywords separated by commas.

Índice general

Índice general III

Índice de figuras V

Índice de tablas VI

Introducción 1

Objetivos del proyecto 2
2.1. Objetivos funcionales . . . 2
2.2. Objetivos de carácter técnico . . . 3
2.3. Objetivos personales . . . 3

Conceptos teóricos 4
3.1. Web API . . . 4
3.2. LTI . . . 5
3.3. SOAP . . . 5
3.4. REST . . . 6

Técnicas y herramientas 8
4.1. Lenguajes . . . 8
4.2. Herramientas . . . 9
4.3. Técnicas . . . 11

Aspectos relevantes del desarrollo del proyecto 12
5.1. Inicio del proyecto . . . 12
5.2. Metodología . . . 12
5.3. Formación . . . 13
5.4. Desarrollo del API Rest . . . 13
5.5. Comunicación con el Web Service de Moodle . . . 14

5.6. Desarrollo de la app 14
5.7. Testing 15

Trabajos relacionados **16**

Conclusiones y Líneas de trabajo futuras **17**

Bibliografía **18**

Índice de figuras

3.1. Estructura de un mensaje SOAP [17] [3] 6

Índice de tablas

Introducción

Este trabajo de final de grado nace de la base de un proyecto anterior,debido a la necesidad de ampliar la funcionalidad de dicho proyecto a una aplicación móvil que permita llevar a cabo las mismas funcionalidades y acceder a los mismos contenidos a disposición del alumno por medio del sistema E-Learning de la Universidad de Burgos.

Objetivos del proyecto

A continuación, se va a llevar a cabo una citación y argumentación de los distintos objetivos a realizar en este proyecto con el objetivo de transmitir la finalidad del mismo.

2.1. Objetivos funcionales

Partiendo del proyecto de partida identificamos los siguientes nuevos objetivos.

Docentes

- Llevarán a cabo las tareas de crear, editar, publicar o duplicar un cuestionario en Moodle.
- Estos cuestionarios permitirán a los docentes poder llevar a cabo una evaluación de los conocimientos adquiridos por su alumnado.
- Al duplicar los cuestionarios estos podrán reutilizarse por los docentes para los distintos grupos o asignaturas que dicho docente imparta.
- Además, gracias a esta aplicación el docente podrá llevar a cabo todas estas funcionalidades con total comodidad desde su smartphone.

Alumnos

- Ofrecer la posibilidad de resolver los distintos cuestionarios a los que tenga que enfrentarse desde su smartphone.
- Al finalizar el mismo, el sistema notificará al alumno de la calificación obtenido junto con una retroalimentación de las diferentes preguntas.

- Estas calificaciones podrán variar en función de distintas recompensas a la hora de enfrentarse al cuestionario.
- Estas recompensas o comodines a partir de ahora, permitirán al alumnado enfrentarse a la prueba de una manera más amigable al enmascarar la verdadera finalidad del cuestionario.

2.2. Objetivos de carácter técnico

La aplicación móvil deberá de poder ser lo suficientemente amigable para los distintos usuarios para facilitar su correcta utilización y finalidad.

La aplicación móvil a desarrollar se llevará a cabo para Android.

Para llevar a cabo la creación de la aplicación se utilizará el entorno de desarrollo Android Studio.

Versión: de la aplicación

Como conectaremos Android con la Web

API APLICACIÓN, FUNCIONE EN UN DISPOSITIVOS. VERSION.

2.3. Objetivos personales

Destacar que uno de los objetivos principales de este proyecto es adquirir nuevos conocimientos dentro del desarrollo Android, junto con otros conocimientos necesarios para poder llevar a cabo la correcta integración del proyecto de partida a el proyecto a desarrollar.

Enfrentarme a nuevos retos que pongan a prueba todos mis conocimientos adquiridos a lo largo del grado.

Enfrentarme a un posible trabajo o proyecto que se asemeje a mi vida profesional.

Conceptos teóricos

En esta sección se va proceder a la explicación de ciertos conceptos teóricos necesarios para la correcta compresión de este trabajo. Además, dichos conceptos han sido necesarios para llevar a cabo la toma decisiones sobre cómo resolver el trabajo junto con la realización del mismo.

3.1. Web API

Antes de hablar de que es un Web API tendremos que explicar que es un API, ya que la funcionalidad de un Web API es similar a la de un API, pero orientada a la Web.

API

Una posible definición podría ser la siguiente: *Un API (siglas de Application Programming Interface) es un conjunto de reglas (código) y especificaciones que las aplicaciones pueden seguir para comunicarse entre ellas: sirviendo de interfaz entre programas diferentes de la misma manera en que la interfaz de usuario facilita la interacción humano-software"* [11].

Es decir, permite la comunicación entre distintos componentes software. La mayor ventaja es que permite reutilizar métodos escrito en un determinado lenguaje o software, de esta manera evitamos la existencia de duplicidad de una misma funcionalidad en los diferentes componentes. Además, estas funcionalidades se encuentran testeadas y funcionan de forma adecuada en un determinado componente software.

Web API

En este caso y relacionado con lo anteriormente explicado la lógica de un Web API es la misma que un API salvo que en este caso esta comunicación, es decir, el intercambio de información se realiza entre un servicio web y una

aplicación mediante una URL. Para llevar a cabo esta comunicación se utilizan peticiones HTTP o HTTPS y toda esta información se encuentra encapsulada generalmente en XML o JSON. Existen principalmente cuatro tipos de Web API.

SOAP

Es un protocolo estándar de intercambio de información y datos en XML entre dos objetos cuyas siglas son las siguientes Simple Object Access Protocol. Posteriormente en esta misma sección se dedicará un apartado donde se explicará en mayor.

XML-RPC

Es un protocolo que llama a un procedimiento remoto que utiliza XML para encapsular los datos y llamadas HTTP para llevar a cabo la comunicación.

JSON-RPC

Es un protocolo cuya lógica es igual que el protocolo explicado anteriormente, salvo que en este caso utiliza el formato JSON para encapsular los datos.

REST

En este caso es una arquitectura software para sistemas hipermedia en la World Wide Web. Además, esta arquitectura utiliza el protocolo HTTP para llevar a cabo la comunicación. No obstante, en esta misma sección se dedicará un apartado para llevar a cabo una explicación en mayor detalle. [1]

3.2. LTI

3.3. SOAP

Anteriormente se ha llevado a cabo una pequeña explicación sobre este protocolo de comunicación. Relacionado con esto definíamos SOAP como un protocolo que posibilita el intercambio de información, es decir, la comunicación mediante internet entre aplicaciones. Para llevar a cabo este intercambio de datos se utiliza el formato XML, gracias a este el intercambio de datos se puede realizar independientemente de la plataforma o lenguaje utilizado. Hoy en día, estos mensajes de tipo SOAP se envían generalmente mediante el protocolo HTTP, aunque también se pueden utilizar otros como SMTP, TCP o JMS entre otros. Otro aspecto importante es que es un pilar fundamental para los Web Services, ya que estos utilizan también el formato XML para

describir los servicios. Además, permite el intercambio de datos o llamadas a procedimientos remotos mediante RPC. Para poder llevar a cabo la correcta lectura de un mensaje SOAP el documento XML debe de tener una determinada estructura. Dicha estructura debe estar formada por las siguientes partes:

- **Envelope:** esta parte de la estructura del mensaje es obligatoria, ya que identifica el mensaje.
- **Header:** esta parte de la estructura permite enviar información adicional sobre cómo debe de procesarse el mensaje. Es por esto por lo que esta parte no es obligatoria, debido a que únicamente aporta cierta información relacionada con la lectura del mismo.
- **Body:** esta parte de la estructura es la encargada de almacenar toda la información del mensaje.
- **Fault:** esta parte de la estructura al igual que el Header es opcional, ya que aporte información relacionada con ciertos errores producidos durante el procesado del mensaje.

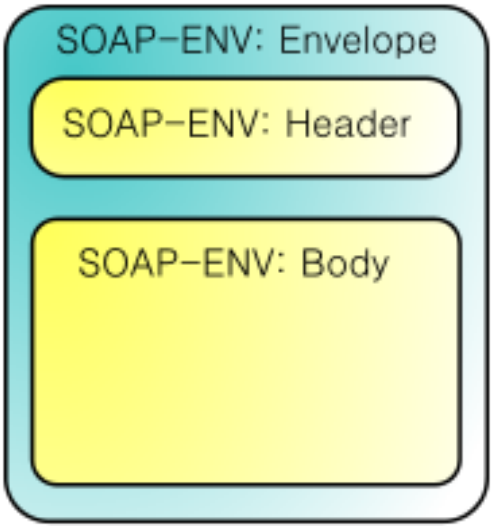

Figura 3.1: Estructura de un mensaje SOAP [17] [3]

3.4. REST

Como ya hemos explicado anteriormente, es un estilo de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web. [15]

cuyas siglas son las siguientes REpresentational State Transfer o en castellano Transferencia de Estado Representacional. En la actualidad, el termino REST se utiliza para aquellas interfaces que utilicen HTTP para el intercambio de información entre sistemas pudiendo utilizar cualquier formato para encapsular los datos, aunque generalmente los más utilizados son XML y JSON. Además, actualmente los sistemas que siguen las pautas o principios REST también se les suelen denominar RESTful. Para que un sistema se considere RESTful debe de cumplir con las siguientes pautas o principios.

- **Protocolo cliente/servidor sin estado:** de manera que cada petición HTTP debe contener toda aquella información necesaria para poder ejecutarse, es decir, de esta forma evitamos que tanto el cliente como el servidor tengan que almacenar información sobre el estado previo de la misma para poder llevarla a cabo.
- **Operaciones:** un sistema REST debe poder llevar cabo las siguientes operaciones: POST (crear), GET (leer o consultar), PUT (editar) y DELETE (borrar). Estas operaciones como podemos observar se asemejan en gran parte a las operaciones CRUD en bases de datos.
- **Sistema de capas:** el sistema deberá de utilizar una arquitectura jerárquica entre los distintos componentes que la formen, de esta manera garantizamos que cada una de estas capas se encargue de llevar a cabo una única funcionalidad.
- **Manipulación de recursos:** para realizar la manipulación de los objetos se lleva a cabo mediante la URI. Dicha URI se utiliza como el identificador único para cada recurso, un recurso es un elemento de información. De esta manera se simplifica el acceso a la información para su posterior manipulación.
- **Uso de hipermedios:** para las transiciones entre los distintos estados de la aplicación y para la información de la misma.

[1]

Técnicas y herramientas

En esta sección se llevará a cabo una mención y breve explicación sobre el conjunto técnicas y herramientas utilizadas durante el desarrollo del proyecto.

4.1. Lenguajes

PHP

PHP es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML [9] cuyas siglas es un acrónimo recursivo de PHP Hypertext Preprocesssor. Este lenguaje se utiliza principalmente en el lado del servidor y está enfocado en el desarrollo web para el contenido dinámico [14]. Este lenguaje se ha decidido utilizar frente a otras alternativas en el lado del servidor debido principalmente a que el proyecto de partida se basa en este lenguaje para el acceso a la base de datos. Además, cabe destacar que desde mi punto de vista me parece una gran decisión al ser uno de los lenguajes más utilizados en el lado del servidor, existe una gran documentación al respecto y la curva de aprendizaje es menos costosa al asemejarse a otros lenguajes orientados a objetos.

JSON

JSON es un formato ligero de intercambio de datos cuyas siglas es un acrónimo de JavaScript Object Notation, es decir, Notación de Objetos de JavaScript [7]. JSON es un formato de texto independientemente del lenguaje y se le considera una gran alternativa frente al formato de intercambio de datos XML al ser mucho más sencillo de parsear por un analizador sintáctico [13]. Este lenguaje se ha decidido utilizar para la devolución de los diferentes datos a la aplicación Android. De esta forma, la comunicación de la aplicación se podrá realizar de una forma más sencilla con la base de datos. Además,

las propias librerías de Android contienen ciertas clases para poder realizar de forma fácil un correcto tratamiento de los datos.

Android

4.2. Herramientas

XAMPP

XAMPP es un paquete de instalación independiente de la plataforma y de software libre. Dicho paquete contiene el sistema de gestión de bases de datos MySQL¸ el servidor web Apache y los interpretes para lenguajes de script: PHP y Perl [19]. Su nombre proviene del acrónimo X indicando que es compatible para cualquier sistema operativo y el resto de siglas AMPP hace referencia a cada uno de los elementos que contiene el paquete, es decir, Apache, MySQL, PHP, Perl respectivamente. Esta herramienta se ha decidido utilizar principalmente al ser utilizada en el proyecto de partida. Además, cabe destacar que me parece una gran decisión ya que el propio paquete contiene todo lo necesario para poder crear un servidor web.

GitHub

GitHub es una plataforma de desarrollo colaborativo para alojar proyectos utilizando un sistema de control de versiones Git [12]. Se ha decidido optar por esta herramienta al haberla utilizado en ciertas ocasiones, al ser una de las más utilizadas para el desarrollo colaborativo y al existir un plugin ZenHub para la gestión desde el propio repositorio de las distintas tareas a realizar en un panel de trabajo. Además, cabe destacar que el repositorio permite dos tipos de formato para el repositorio: público y privado. En nuestro caso se encuentra público. `https://github.com/danielpuente-dpg/GII14.K.QUICKTEST`

ZenHub

TortoiseSVN

TortoiseSVN es un software libre utilizado para llevar a cabo el control de versiones del proyecto sobre GitHub [18]. Implementa una extensión del shell de Windows, por lo que el control de versiones no tiene por qué realizarse mediante línea de comandos, sino que se puede realizar mediante la interfaz gráfica que proporciona esta herramienta la cual es muy fácil de utilizar.

Moodle

PhpStorm

phpMyAdmin

PHPUnit

SonarQube

Advanced REST Client

StarUML

Librerías

A continuación, se incluye una pequeña explicación sobre las librerías utilizadas para el desarrollo del proyecto.

Consumir peticiones

Durante el desarrollo de la aplicación se han barajado ambas librerías para en consumo de las peticiones al propio API y al webservice de Moodle. Finalmente, se ha decido optar por la librería Retrofit al ser más fácil de utilizar, más intuitiva a la hora de crear y configurar las distintas peticiones, más flexible y suele tener mejores resultados que Volley

- **Retrofit:** Retrofit proporciona un framework para poder interactuar con APIs y enviar peticiones HTTP de forma fiable desde aplicaciones Android. Para ello proporciona un cliente HTTP encargado de interactuar con APIs y de gestionar las peticiones de manera transparente. Esta librería es utilizada para el tratamiento de las peticiones desde la aplicación android. Además, permite consumir peticiones de manera síncrona y asíncrona. [10]
- **Volley:** Al igual que la librería anterior, Volley es una librería desarrollada por Google para enviar peticiones HTTP desde aplicaciones Android. [6]
- **Gson:** Esta librería proporcionada por Google, permite trabajar con JSON a la hora de serializar y deserializar los objetos. Es utilizada para convertir las respuestas JSON de las APIs en objetos Java de manera muy sencilla, eliminado toda esta carga de conversión al programador. Además, también permite el paso contrario a la hora de enviar objetos a las APIs, al encargarse de transformar estos objetos en respuestas en formato JSON. [5]

4.3. Técnicas

LTI

MVC

SCRUM

SCRUM es un modelo de referencia que define un conjunto de práctica y roles, que pueden utilizarse como punto de partida para llevar a cabo la definición de como se elaborará el proceso de desarrollo del proyecto [16]. Cabe destacar que este modelo se encuentra dentro de los marcos de las metodologías ágiles y que es de los más utilizados actualmente. Los roles principales son:

- **Scrum Master:** se encarga de gestionar los cambios y procura facilitar la aplicación de esta metodología.
- **Product Owner:** en este grupo se encontrarán el personal interno o externo que representa al cliente. Este grupo se encargará de que el trabajo se realice de forma acorde con las necesidades y peticiones del cliente.
- **Team:** representa al equipo de desarrollo encargados de ejecutar el desarrollo y de entregar el producto deseado.
- **Product Owner:** en este grupo se encontrarán el personal interno o externo que representa al cliente. Este grupo se encargará de que el trabajo se realice de forma acorde con las necesidades y peticiones del cliente.

[16]

Para realizar el desarrollo del producto se deben de definir un Product Backlog, el cual contendrá todas las historias de usuario a realizar. Dichas historias de usuario se asignarán a los distintos Sprints o iteraciones que se realicen a lo largo del desarrollo del producto hasta finalizar con todas las historias de usuario.

Aspectos relevantes del desarrollo del proyecto

En este apartado se recogen los aspectos mas relevantes que han surgido a lo largo del desarrollo del proyecto, es decir todos aquellos puntos de inflexión que han marcado los pasos a seguir o la toma de decisiones frente a diferentes problemáticas.

5.1. Inicio del proyecto

La selección de este trabajo final de grado, nace de la necesidad propia de la construcción de una aplicación en Android. Una vez publicados las distintas propuestas, busque aquellas que mas se acercarán a mis necesidades. Aquí, es donde este proyecto destacó sobre las demás.

Una vez realizada mi elección de proyecto, comunique a mis tutores mi interés en el mismo y que me explicarán en mayor profundidad las funcionalidades a desarrollar. Tras las explicaciones y aclaraciones pertinentes nos pusimos manos a la obra.

5.2. Metodología

Para la correcta gestión del proyecto se decidió utilizar una metodología ágil, en este caso Scrum 4.3. Destacar que no se siguió al pie de la letra, ya que el equipo de proyecto solamente estaba formado por 3 personas. No obstante, se siguieron las siguientes pautas:

- Desarrollo incremental del proyecto basado en iteraciones o sprints.
- La duración ideal de las distintas iteraciones fue de 15 días.

- Al finalizar cada iteración, se realiza una revisión del sprint anterior y al finalizar esta, se realiza la planificación de las nuevas tareas a entregar en la siguiente iteración.
- Para manipular estas tareas se utiliza un tablero Kanban. Este tablero se incluye en GitHub gracias a la utilizacion de ZenHub 4.2.
- Utilización de gráficos Burndown para monitorizar el progreso de proyecto.

5.3. Formación

La realización de este proyecto requería una serie de conocimientos que al comienzo del mismo no disponía. Para ello en las primeras iteraciones se realizarón estudios sobre estos conocimientos técnicos. Para ello se realizó:

- Para la formación en PHP:
  - Lectura de la documentación oficial que proporciona PHP. [8]
  - Curso de aprendizaje de PHP. [2]
- Para la formación en la plataforma Moodle:
  - Lecturas varias, del funcionamiento de la misma.
  - Documentación del proyecto de partida.
- Para la formación en Android:
  - Programación de Android desde Cero + 35 horas (Udemy).
  - Android. Guía De Desarrollo De Aplicaciones Para Smartphones Y Tabletas [4] Libro proporcionado por mis tutores.
  - ***Documentación que utilice para el Testing.***

5.4. Desarrollo del API Rest

Para poder comunicar la aplicación Android con las propias funcionalidades del proyecto de partida, se utiliza un API. Las principales funcionalidades de este API, han sido desarrolladas durante las primeras iteraciones del proyecto, no obstante otras han sido desarrolladas fruto de la necesidad de las mismas.

Se ha decidido utilizar una arquitectura Rest para este API, ya que:

- **Separación cliente/servidor:** de esta manera al ser independiente, la comunicación entre cliente y servidor se realiza mediante un lenguaje común de intercambio, en este caso JSON 4.1. Gracias a esta idea, podemos realizar la construcción de un API que funcione de adaptador entre las funcionalidades del proyecto de partida y la información que la propia aplicación Android necesita. Además, de esta manera no es necesario modificar ninguna funcionalidad del proyecto de partida.
- **Lenguajes independientes:** permite la comunicación de información entre distintos lenguajes. En nuestro caso, este API esta desarrollado en PHP y el cliente que lo consume en Java.
- **Flexibilidad:** el cambio de alguno de estos nexos no implica la transformación de los mismos, siempre y cuando la devolución de los datos se realice de forma correcta.
- **Sin estado:** al no tener estado, no requiere un almacenamiento de este y por lo tanto permite un mayor número de peticiones.

Es por esto que la visión que podemos tener de este, es la del BackEnd de la app, es decir es el proveedor de los datos a mostrar en la aplicación.

5.5. Comunicación con el Web Service de Moodle

La aplicación Android nos permitirá resolver los mismos cuestionarios del proyecto de partida. Para ello, es necesario conocer cierta información que el propio API desarrollado no tiene. Es por esto, que la aplicación tendrá que comunicarse de alguna con Moodle para conocer estos datos, es aquí donde entra en juego los Web Service de Moodle.

Para llevar a cabo esta comunicación se han tenido que realizar previamente la instalación y configuración de estos. Una vez realizado esto, la aplicación Android podrá comunicarse con Moodle.

5.6. Desarrollo de la app

Una vez desarrolladas las principales funcionalidades del API en las primeras iteraciones, se comenzó con el desarrollo de la aplicación Android. La versión utilizada en el proyecto es Android 4.1 (Jelly Bean) al ser actualmente la que mayor soporte cubre, con un 95,2 % al inicio del desarrollo de la app.

La primera problemática que surgió fue la del tratamiento de las peticiones desde la app. Para ello y después de mucha investigación y pruebas se decidio utilizar la libreria Retrofit de Square [10] entre otras alternativas como Volley [6] de Google, al permitir esta la utilización de peticiones síncronas y la libreria

Gson de Google [5] para facilitarla la transformación de los datos en formato JSON proporcionados por el propio API o el Web Service de Moodle, a objetos en Java.

5.7. Testing

Trabajos relacionados

Este apartado sería parecido a un estado del arte de una tesis o tesina. En un trabajo final grado no parece obligada su presencia, aunque se puede dejar a juicio del tutor el incluir un pequeño resumen comentado de los trabajos y proyectos ya realizados en el campo del proyecto en curso.

Conclusiones y Líneas de trabajo futuras

Todo proyecto debe incluir las conclusiones que se derivan de su desarrollo. Éstas pueden ser de diferente índole, dependiendo de la tipología del proyecto, pero normalmente van a estar presentes un conjunto de conclusiones relacionadas con los resultados del proyecto y un conjunto de conclusiones técnicas. Además, resulta muy útil realizar un informe crítico indicando cómo se puede mejorar el proyecto, o cómo se puede continuar trabajando en la línea del proyecto realizado.

Bibliografía

[1] BBVAopen4u. Api rest – qué es y cuáles son sus ventajas en el desarrollo de proyectos. [Internet; descargado 23-febrero-2017].

[2] Codejobs. Php — curso: Aprende php desde cero hd. [Internet; descargado 16-mayo-2017].

[3] DesarroloWeb. Soap – soap. simple object access protocol. [Internet; descargado 23-febrero-2017].

[4] Google. Android — android. guía de desarrollo de aplicaciones para smartphones y tabletas - 2ª edición. [Internet; descargado 2-marzo-2017].

[5] Google. Gson — google-gson. [Internet; descargado 4-mayo-2017].

[6] Google. Volley — volley is an http library. [Internet; descargado 4-mayo-2017].

[7] JSON. Json – introducción a json. [Internet; descargado 2-marzo-2017].

[8] PHP. Php — referencia del lenguaje. [Internet; descargado 16-mayo-2017].

[9] PHP. Php – ¿qué es php? [Internet; descargado 2-marzo-2017].

[10] SQUARE. Retrofit — a type-safe http client for android and java. [Internet; descargado 4-mayo-2017].

[11] Ticbeat. Api — ¿qué es una api y para qué sirve? [Internet; descargado 23-febrero-2017].

[12] Wikipedia. Github — wikipedia, la enciclopedia libre. [Internet; descargado 2-marzo-2017].

[13] Wikipedia. Json — wikipedia, la enciclopedia libre. [Internet; descargado 2-marzo-2017].

[14] Wikipedia. Php — wikipedia, la enciclopedia libre. [Internet; descargado 2-marzo-2017].

[15] Wikipedia. Rest — wikipedia, la enciclopedia libre. [Internet; descargado 23-febrero-2017].

[16] Wikipedia. Scrum — wikipedia, la enciclopedia libre. [Internet; descargado 2-marzo-2017].

[17] Wikipedia. Soap — wikipedia, la enciclopedia libre. [Internet; descargado 23-febrero-2017].

[18] Wikipedia. Tortoisesvn — wikipedia, la enciclopedia libre. [Internet; descargado 2-marzo-2017].

[19] Wikipedia. Xampp — wikipedia, la enciclopedia libre. [Internet; descargado 2-marzo-2017].