UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA DEL SOFTWARE

EDUKKA 2.0: EXTENSION DE LA PLATAFORMA EDUKKA

EDUKKA 2.0: EXTENSION OF THE PLATFORM EDUKKA

Realizado por
Alberto Rodríguez Torres

Tutorizado por
Eduardo Guzmán de los Riscos

Departamento
Lenguaje y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA

MÁLAGA, JUNIO DE 2018

Fecha defensa:

Fdo. El/la Secretario/a del Tribunal

RESUMEN: Este Trabajo de Fin de Grado (TFG) se dedica a la mejora e implementación de nuevas funcionalidades a una plataforma ya existente llamada Edukka. Esta plataforma fue desarrollada para la enseñanza en dispositivos móviles siguiendo el concepto de e-learning, el cual permite al usuario interaccionar con el material a través de diversas herramientas informáticas. Edukka está dividida en dos módulos: la aplicación Web que funciona como servidor y que envía los datos que sean necesarios a la segunda parte, que es la aplicación Android con la que interactúan los usuarios. La integración de estos dos módulos independientes genera la plataforma final. Las mejoras realizadas se centran en la experiencia de juego de los alumnos y el uso que pueden dar los profesores a la plataforma, los cuales ahora pueden generar todo el contenido que los alumnos de sus respectivas clases van a consumir. Esta última parte es de vital importancia, ya que un gran número de fracasos escolares se deben al ritmo excesivamente lento de las clases y la perdida de interés en las mismas, y con esta plataforma a través de la gamificación, el profesor puede ofrecer desafíos para mantener la atención de los niños en la enseñanza.

PALABRAS CLAVE: Android, E-Learning, Gamificación.

ABSTRACT: This Final Degree Project (TFG) is dedicated to the improvement and implementation of new functionalities to an already existing platform called Edukka. This platform was developed for teaching in mobile devices following the e-learning concept, which allows the user to interact with the material through various computer tools. Edukka is divided into two modules: the web application that Works like a server and send the necessary data to the second part, which is the Android application which users interact with. The integration of these two independent modules generates the final platform. The improvements made are focused in the students' game experience and the usage the teachers can have on the platform, which now can generate all the content that the students of their respective classes is going to consume. This last part is of vital importance, since a large number of school failures are due to the excessively slow pace of the lessons and the lost of interest in them, and with this platform through the gamification, the teacher can offer challenges to keep the attention of the children in the education.

KEYWORDS: Android, E-Learning, Gamification.

Índice General

Capítulo 1. Introducción

1.1 Motivación

El principal problema que queremos tratar con este proyecto es el gran número de fracasos escolares que se dan en cierto tipo de estudiantes de Primaria que no son capaces de seguir el ritmo de la clase, no porque sea demasiado rápido, sino porque es demasiado lento. Los llamados niños de altas capacidades son aquellos que destacan por encima de la media en una o más asignaturas o áreas de estudio, y para ellos ir al mismo paso que el resto de alumnos se hace muy aburrido, perdiendo el interés por completo en la enseñanza y generando así dicho fracaso escolar, cerrándole muchas puertas hacia un futuro brillante.

A día de hoy, el e-learning es una herramienta muy poderosa de enseñanza, ya que cualquiera dispone de Internet y de todos los conocimientos que desee a dos clics de distancia, ya no es necesario un estatus social elevado para acceder al conocimiento ni tener acceso a una biblioteca, está en manos de todos. Estamos en una época dorada de la información con la que los eruditos de la antigüedad soñaban.

Sin embargo, aun estando este conocimiento tan a la mano de todos, sin un aliciente que le dé cierto atractivo la gente lo ve como una masa de información desmesurada, por ello la gamificación es algo tan importante. A través de ciertas herramientas informáticas le podemos dar el formato de un juego al hecho de aprender, de experimentar cosas nuevas, y como ya he comentado anteriormente, nuestro objetivo principal son los niños, y no hay niño al que no le guste jugar, por ello nos basamos fundamentalmente en esta técnica para transmitir conocimientos. Queremos aportarles una plataforma dedicada para ellos, donde puedan avanzar a su propio ritmo y aprender de forma ágil y divertida lo que a ellos le apetezca.

1.2 Objetivos

Ya contamos con una estructura dada, ya que este proyecto es la continuación o mejora de un proyecto ya existente de la plataforma denominada Edukka, creada por Javier Rosales Bañón en su TFG, y nuestro principal objetivo es añadirle nuevas funcionalidades y retocar otras para darles a los usuarios una experiencia más completa y personalizada. Esta plataforma está dividida en dos partes que se complementan entre sí que serán descritas en el siguiente capítulo de esta memoria.

Las funcionalidades ya existentes incluyen para los alumnos el inicio de sesión a través del nombre y contraseña, ver los perfiles de los demás y editar el perfil propio, jugar a los juegos disponibles y observar el progreso personal, mientras que los profesores disponen de las mismas herramientas que los usuarios además de poder crear y editar una clase en la que pueden entrar alumnos, ver los juegos y sus soluciones y el progreso de todos los alumnos de su clase.

De lo que no dispone la aplicación Android es de una forma de crear nuevos juegos para los alumnos, por lo que nuestros esfuerzos se van a dedicar en crear un editor de juegos que esté en manos del profesor de la clase para crear contenido de forma simple y cómoda dentro de la misma. También se realizarán cambios en la estructura de datos ya que esta solo permite que cada una de las diez asignaturas solo pueda acceder a un único tipo de juego o quiz, por lo que se refactorizarán algunas partes para darle más versatilidad a los juegos que puedan crear los profesores. Los juegos a partir de ahora contarán con un temporizador que marca el tiempo límite del que dispone el alumno para completar el juego, dándoles a estos mucho más dinamismo.

Por último, también se le dedicará una parte del tiempo en mejorar la experiencia de los alumnos para añadir una modalidad multijugador básica a través del envío de mensajes en tiempo real para crear juegos competitivos con otros miembros de su clase, dándole más interacción entre usuarios a la plataforma y un motivo más a los niños para divertirse aprendiendo. Para ello los alumnos crearán salas de juego en las que esperarán a algún otro alumno de la clase para jugar contra él en un duelo por puntos. Los profesores podrán adornar las salas creadas por los alumnos para hacer que doblen la puntuación obtenida y evitar que algún alumno no encuentre compañero con el que jugar y, por último, modificar las puntuaciones para hacer de las clasificaciones algo más interesante y evitar una ventaja excesiva e insalvable por parte de algún usuario.

1.3 Contenido de la memoria

Se va a exponer de forma breve la estructura de este documento y un breve resumen de todos los capítulos que este contiene.

- **Capítulo 1. Introducción**
  En este capítulo se presentan los motivos y la necesidad de la que surge el proyecto, se declaran los objetivos y también se aporta un breve resumen del contenido de este documento.
- **Capítulo 2. Tecnologías y recursos**
  Aquí se detallarán las tecnologías con las que contamos en el proyecto base y las añadidas durante el desarrollo. También se explicarán las dos partes que forman la plataforma.
- **Capítulo 3. Análisis**
  En este capítulo se mostrarán los requisitos que la aplicación debe cumplir (tanto funcionales como no funcionales) y se describirán a los usuarios que usarán la plataforma.
- **Capítulo 4. Diseño**
  Se presentarán los casos de uso y los nuevos diagramas del diseño de la plataforma y distribución de clases, además del diagrama de diseño de la base de datos.
- **Capítulo 5. Implementación**
  Aquí se detallará todo el desarrollo del trabajo, entrando en detalle sobre la refactorización realizada sobre la plataforma dada y la integración de nuevas funcionalidades.
- **Capítulo 6. Conclusiones y líneas futuras**
  Por último, en este capítulo se concluirá la memoria exponiendo lo aprendido durante el proceso y se añadirán algunas propuestas para mejorar este trabajo en un futuro.

Capítulo 2. Tecnologías y recursos

2.1 Aplicación Web

Esta es la primera de las dos partes que forman la plataforma Edukka, el back-end. Está basado en el lenguaje PHP y se despliega en un servidor web que accede a una base de datos SQL. Esta estructura envía las respuestas a las peticiones de la segunda parte (el front-end) en forma de texto ligero JSON para facilitar su posterior tratado. Cuenta con una estructura bastante robusta y ágil gracias a la extensión PDO de PHP, siendo esta una capa de abstracción para acceder a distintos tipos de bases de datos ya sean conexiones, recuperaciones o modificaciones de datos de las mismas. También se utilizó en su creación un micro framework denominado Slim que crea APIs de forma rápida y bastante sencilla. A dicha API se le han añadido nuevas funcionalidades y también se han incluido otras fuera de la misma por falta de compatibilidad, entre las cuales están la subida al servidor de archivos de imagen, de archivos de audio y la gestión de la mensajería en tiempo real.

2.2 Aplicación Android

La segunda parte de la plataforma es el front-end, una aplicación Android desarrollada en Android Studio. Se encarga de enviar peticiones al servidor y dar formato a las respuestas JSON para mostrar la información al usuario de forma simple y lo más atractiva posible. Su estilo visual esta basado en el estilo de Material Design de Google con la intención de presentarles a los niños una interfaz limpia y llamativa gracias a las guías de diseño que dicho patrón ofrece.

Para estilizar la aplicación o simplificar tareas se han utilizado las siguientes librerías externas:

- Retrofit: realiza el envío de peticiones HTTP para comunicarse con el servidor web de manera simple, gestiona diferentes tipos de parámetros y parsea automáticamente la respuesta a un objeto Java.
- StatusBarUtil: configura el estilo de la barra de estado.

- CircleImageView: muestra las imágenes de forma circular.
- MaterialEditText: muestra los campos de texto con un estilo distinto, más cercano al Material Design.
- MaterialBetterSpinner: muestra los desplegables con un estilo distinto, más cercano al Material Design.
- RingProgressBar: crea un contador en tiempo real adaptado al estilo Material Design.
- EasyPermissions: maneja de forma sencilla los permisos de la aplicación, entre los que incluye el de leer una imagen para posteriormente subirla al servidor.
- Picasso: librería para cargar imágenes de manera simple y rápida.
- Lottie: analiza animaciones de Adobe After Effects exportadas como JSON y las renderiza de forma nativa en los dispositivos móviles.
- FirebaseCore: se encarga de enviar y recibir mensajes de otros dispositivos en tiempo real.

2.3 Firebase

En la fase final de desarrollo del proyecto en la que se buscaban implementar funcionalidades multijugador se investigaron varios métodos por los que se podría introducir mensajería en tiempo real, siendo Firebase el más interesante de utilizar.

Firebase es una plataforma de Google que utiliza su infraestructura y escala junto a tu aplicación, y en mi caso funciona como un servidor aparte para gestionar la mensajería entre usuarios de forma autónoma a través de su API para manejar peticiones HTTP. Esta plataforma también cuenta con herramientas para desplegar tus aplicaciones web y el manejo de bases de datos no relacionales en tiempo real, dando lugar a una posible línea de trabajo de este proyecto para un futuro. Esto se explicará con más detalle al final del documento.

2.4 Herramientas utilizadas

- **GitKraken:** interfaz gráfica dedicada a la gestión de repositorios Git. Permite llevar de forma muy simple e intuitiva un seguimiento completo de nuestros repositorios con todas sus funcionalidades características.
- **MagicDraw:** herramienta CASE usada para el modelado de los diagramas UML, lenguaje gráfico usado para visualizar, especificar, construir y documentar un sistema.
- **HeidiSQL:** software libre y de código abierto que permite conectarse a servidores MySQL y administrar bases de datos de forma más intuitiva.
- **Balsamiq:** herramienta para crear maquetas, usadas como guía visual en este caso para el desarrollo de la aplicación Android.
- **WAMP:** sistema de infraestructura de internet que utiliza Windows, Apache, PHP y MySQL para generar un servidor local en el que desplegar tu aplicación.
-

2.5 Repositorio de código fuente

Los repositorios del proyecto están alojados de forma pública en GitHub, uno de los servidores de alojamiento de repositorios más utilizados, que además conforma una gran comunidad de desarrolladores. Esto les permite a otras personas interesadas en este proyecto acceder al código de forma gratuita.

Aplicación Web: https://github.com/AldresGit/EdukkaPHP

Aplicación Android: https://github.com/AldresGit/Edukka

Capítulo 3. Análisis

3.1 Requisitos funcionales

En este apartado se van a describir los requisitos funcionales que debe cumplir el proyecto, entre los cuales no se incluirán los que se desarrollaron anteriormente, sino los que pertenecen a nuevas funcionalidades. Los requisitos funcionales identifican las acciones que el sistema debe proporcionar en su fase final, y están enumerados en la Tabla 1.

ID	Requisito	Descripción
RF01	Crear Juego	El sistema debe permitir al profesor dueño de una clase crear un juego nuevo accesible para los alumnos de la misma. Es necesario rellenar de forma correcta todos los campos.
RF02	Modificar Juego	El sistema debe permitir al profesor que ha creado un juego modificarlo posteriormente, ya sea para modificar los datos o para añadir/eliminar preguntas.
RF03	Borrar Juego	El sistema debe permitir la eliminación de un juego por parte del profesor que lo ha creado.
RF04	Crear Pregunta	El sistema debe permitirle al profesor añadir preguntas tipo quiz de cualquiera de los diez tipos existentes en la plataforma a un juego que le pertenezca. Debe introducir todos los campos correctamente y, en caso de subir archivos, deben tener el formato correcto.
RF05	Modificar Pregunta	El sistema debe permitir la modificación de una pregunta tipo quiz al profesor que la creó, siempre que los campos modificados sean correctos en formato.
RF06	Eliminar Pregunta	El sistema debe permitir la eliminación de preguntas tipo quiz por parte del creador.
RF07	Subir Imagen	El sistema debe permitir la subida de archivos de imagen al servidor.
RF08	Subir Audio	El sistema debe permitir la subida de archivos de voz al servidor.
RF09	Iniciar Juego	El sistema debe mostrar las preguntas, ahora de cualquier tipo en cualquier asignatura, a los alumnos. Esta es la característica principal de la plataforma a través de la que los niños aprenden.

RF10	Crear Sala Multijugador	El sistema debe permitir crear una sala multijugador a los alumnos para buscar otro jugador contra el que competir por puntos.
RF11	Buscar Sala Multijugador	El sistema debe mostrar todas las salas creadas por un alumno y que tengan plaza disponible para unirse.
RF12	Unirse a Sala Multijugador	El sistema debe permitir a un alumno unirse a una sala multijugador creada por otro alumno que disponga de alguna plaza libre.
RF13	Salir Sala Multijugador	El sistema debe permitir tanto al creador como a otro alumno que se haya unido abandonar la sala. Si es el creador el que abandona la sala, esta se borrará.
RF14	Adornar Sala Multijugador	El sistema debe permitir a un profesor adornar una sala multijugador, haciendo que la misma doble la puntuación que otorga.
RF15	Modificar Puntos	El sistema debe permitir al profesor modificar la tabla de puntuaciones de los alumnos de su clase para balancearla.

Tabla 1: Requisitos Funcionales

3.2 Requisitos no funcionales

Los requisitos no funcionales son las características que el sistema debe cumplir y las condiciones que el cliente impone al producto final. En la Tabla 2 se describirán dichos requisitos.

ID	Requisito	Descripción
RNF1	**Requisitos de Fiabilidad**	
RNF1.1	Disponibilidad	El sistema debe estar disponible para todo usuario conectado a él, excepto durante labores de mantenimiento de la plataforma.
RNF1.2	Tolerancia a fallos	El sistema debe tener controladas las situaciones que causarían fallos graves durante el uso normal de la aplicación.
RNF2	**Requisitos de Usabilidad**	
RNF2.1	Facilidad de Aprendizaje	El sistema debe asegurar que las nuevas funcionalidades sean intuitivas y que cada elemento tenga iconos ilustrativos que aclaren la función de los mismos.
RNF2.2	Facilidad de Uso	El sistema debe asegurar un flujo de control cómodo dentro de la aplicación para facilitar su navegación.
RNF2.3	Atractividad	El sistema debe seguir los patrones de diseño de Google Material Design para hacer la interfaz lo más atractiva posible.
RNF3	**Requisitos de Eficiencia**	
RNF3.1	Velocidad	El sistema debe garantizar máxima fluidez durante su funcionamiento, asegurando que la duración entre petición y respuesta sea menor de 5 segundos.
RNF3.2	Capacidad	El sistema debe permitir la subida de archivos al servidor. La capacidad actual es limitada debido al servidor gratuito en el que se ha localizado la aplicación.
RNF4	**Requisitos de Mantenibilidad**	
RNF4.1	Mantenimiento de la plataforma	El mantenimiento está dividido en cuatro partes: El mantenimiento del servidor PHP, el mantenimiento de la base de datos, el mantenimiento de la aplicación Android y por último el del servidor Firebase encargado de los mensajes en tiempo real.
RNF5	**Requisitos de Portabilidad**	
RNF5.1	Portabilidad	La aplicación de momento solo está disponible para dispositivos Android con versión igual o superior a la 5.0 (Lollipop). Su portabilidad a iOS dependerá de las futuras extensiones del proyecto.
RNF6	**Requisitos de Seguridad**	
RNF6.1	Protección de datos	El sistema debe garantizar la protección de los datos sensibles de todos los usuarios.
RNF6.2	Integridad de Ficheros	Los usuarios deben poder acceder a la información actualizada si están conectados a Internet.

RNF7	Requisitos de Documentación	
RNF7.1	Guía de uso	El sistema debe aportar información sobre las acciones disponibles a realizar en pantalla para facilitar el uso de los usuarios.

Tabla 2: Requisitos No Funcionales

3.3 Descripción de Participantes y Usuarios

3.3.1 Participantes

Los participantes son las personas o asociaciones que influyen al desarrollo del proyecto. En este caso son los siguientes:

- Gerencia: Encargado de dirigir, gestionar y administrar el proyecto para que cumpla los objetivos establecidos además de las leyes, reglamentos u otras disposiciones generales establecidas.

Representante	**Eduardo Guzmán de los Riscos**
Tipo	Tutor del proyecto
Responsabilidades	Supervisar el desarrollo de la plataforma
Criterios de Éxito	El proyecto finalizado cumple los objetivos
Entregables	Idea del proyecto

- Desarrollador: Persona que se encarga del desarrollo del proyecto y de la entrega del mismo, además de realizar la documentación necesaria.

Representante	**Alberto Rodríguez Torres**
Tipo	Estudiante de Ingeniería del Software
Responsabilidades	Refactorizar la ya existente plataforma Edukka y añadir nuevas funcionalidades al proyecto.
Criterios de Éxito	Finalización de la plataforma cumpliendo todos los requisitos acordados.
Entregables	Proyecto finalizado.

3.3.2 Usuarios

Aquí se describirán los tipos de usuarios que utilizarán la plataforma final, de los cuales existen dos tipos:

- Alumno: Usuario que utilizará la aplicación para jugar con los juegos creados en la plataforma y para retar a otros alumnos de su clase a juegos en línea, con el propósito de divertirse y aprender.

Descripción	Persona que utiliza la aplicación para divertirse
Tipo	Cliente
Responsabilidades*	Iniciar Juego Crear Sala Multijugador Buscar y Unirse a Sala Multijugador Salir Sala Multijugador
Criterios de Éxito	El usuario aprende y se divierte usando la plataforma

*Solo se describirán las responsabilidades añadidas en esta ampliación del proyecto

- Profesor: Usuario encargado de gestionar y añadir los juegos a los que tendrán acceso los alumnos de su clase, de controlar la tabla de puntuaciones de la misma y monitorizar el desarrollo de sus alumnos en la aplicación.

Descripción	Persona que utiliza la aplicación para añadir contenido y monitorizar a sus alumnos
Tipo	Cliente
Responsabilidades*	Crear, Modificar y Borrar Juego Crear, Modificar y Borrar Preguntas de cualquiera de los 10 tipos Subir archivos de Audio y Video Buscar y Adornar Salas Multijugador Balancear Puntuaciones
Criterios de Éxito	Tener la capacidad de aportar a los alumnos de su clase una fuente de conocimiento que sea divertida de usar

*Solo se describirán las responsabilidades añadidas en esta ampliación del proyecto

Capítulo 4. Diseño

4.1 Diagrama de casos de uso

Un diagrama de casos de uso se utiliza para especificar comportamientos que un sistema puede tener a través de la interacción de sus usuarios. Cada funcionalidad del sistema forma un caso de uso en el diagrama con el que los actores pueden interactuar.

Como partimos de una plataforma ya desarrollada sobre la que vamos a introducir funcionalidades solo vamos a describir los casos de uso nuevos o que han recibido cambios significativos, los cuales están destacados en verde en el siguiente diagrama (Figura 1).

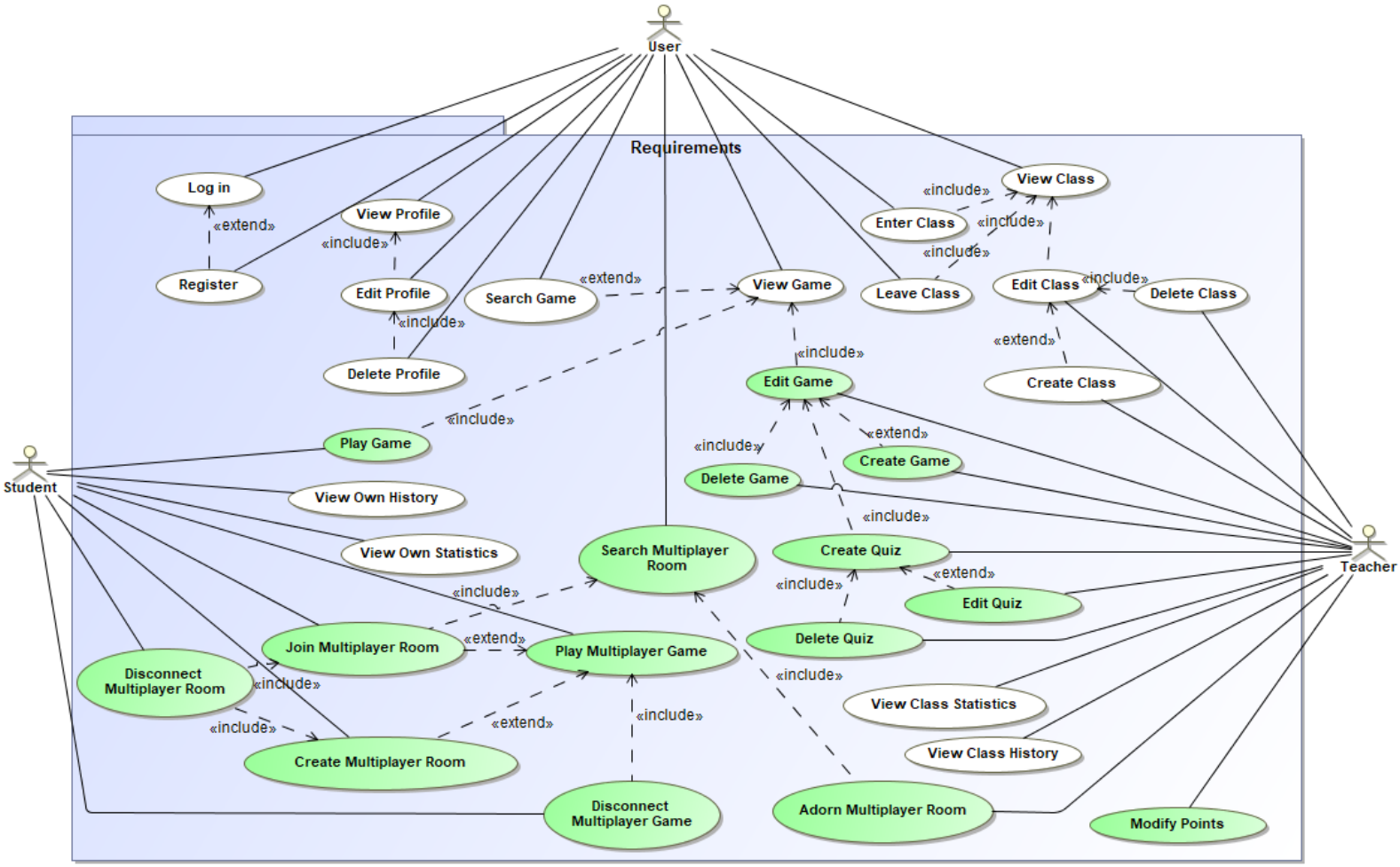

Figura 1 - Diagrama de casos de uso

- **CREAR JUEGO**

Caso de Uso	Crear Juego	**Identificador:** CU-01
Resumen	Creación de un nuevo juego para la plataforma	
Actores	Profesor	
Tipo	Primario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión y Crear Clase	
Precondición	El actor debe estar registrado y haber creado una clase propia en la aplicación	
Postcondición	El actor crea un juego nuevo en la plataforma	
Descripción	Caso de uso principal de este proyecto que permite añadir nuevo contenido a la plataforma para los alumnos. Los datos introducidos en los campos obligatorios deben ser correctos.	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar una de las diez asignaturas en el menú principal
2	Sistema	Muestra los juegos disponibles de esa asignatura
3	Actor	Pulsar el botón flotante de Crear Juego
4	Sistema	Muestra el formulario de datos para crear un juego
5	Actor	Rellenar los campos necesarios y pulsar el botón de guardar
6	Sistema	Comprobar los campos del formulario
7	Sistema	Crear juego en la base de datos con los valores introducidos
8	Sistema	Mostrar pantalla de edición de juego

Cursos Alternos

Nro.	**Descripción de acciones alternas**
4	El actor no puede crear juegos en la clase por defecto, recibiendo una notificación del sistema indicándole que debe crear una clase propia
5	El actor puede cancelar la creación de un juego y volver al menú principal
6	Los datos introducidos pueden ser inválidos, notificando al actor cuales necesitan corrección

- **EDITAR JUEGO**

Caso de Uso	Editar Juego	**Identificador:** CU-02
Resumen	Editar un juego ya creado en la plataforma	
Actores	Profesor	
Tipo	Secundario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión y Crear Juego	
Precondición	El actor debe estar registrado y haber creado el juego a editar en la aplicación	
Postcondición	El actor edita un juego en la plataforma y los datos se actualizan en la base de datos	
Descripción	Permite modificar los datos de un juego ya creado por el propio actor. Los datos introducidos en los campos deben ser correctos.	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar una de las diez asignaturas en el menú principal
2	Sistema	Muestra los juegos disponibles de esa asignatura
3	Actor	Pulsar en el recuadro perteneciente a un juego ya creado
4	Sistema	Muestra el formulario de edición de un juego con sus datos ya introducidos en cada campo.
5	Actor	Modificar los campos deseados y pulsar el botón de guardar
6	Sistema	Comprobar los campos del formulario
7	Sistema	Modificar los campos del juego en la base de datos con los valores introducidos
8	Sistema	Mostrar pantalla de edición de juego

Cursos Alternos

Nro.	**Descripción de acciones alternas**
4	El actor no puede editar los juegos de la clase por defecto, recibiendo una notificación del sistema indicándole que debe crear una clase propia
5	El actor puede cancelar la edición de un juego y volver al menú principal
6	Los datos introducidos pueden ser inválidos, notificando al actor cuales necesitan corrección

- **BORRAR JUEGO**

Caso de Uso	Borrar Juego	**Identificador:** CU-03
Resumen	Borrar un juego ya creado en la plataforma	
Actores	Profesor	
Tipo	Secundario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión y Crear Juego	
Precondición	El actor debe estar registrado y haber creado el juego a eliminar en la aplicación	
Postcondición	El actor elimina un juego de la plataforma y sus datos se borrarán de la base de datos	
Descripción	Permite borrar un juego ya creado por el propio actor. Las preguntas que pertenezcan al juego también se eliminarán de la base de datos	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar una de las diez asignaturas en el menú principal
2	Sistema	Muestra los juegos disponibles de esa asignatura
3	Actor	Pulsar en el recuadro perteneciente a un juego ya creado
4	Sistema	Muestra el formulario de edición de un juego con sus datos ya introducidos en cada campo
5	Actor	Pulsar el icono de borrar juego
6	Sistema	Muestra un dialogo que pide confirmación del usuario
7	Actor	El actor confirma la acción de borrar el juego
8	Sistema	El sistema elimina el juego y sus preguntas de la base de datos, llevando al actor al menú principal

Cursos Alternos

Nro.	**Descripción de acciones alternas**
7	El actor puede cancelar la eliminación del juego en este momento, con lo cual el sistema cerraría el dialogo y mostraría el formulario de edición del juego

- **CREAR PREGUNTA**

Caso de Uso	Crear Pregunta	**Identificador:** CU-04
Resumen	Creación de una nueva pregunta para un juego de la plataforma	
Actores	Profesor	
Tipo	Primario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión y Crear Juego	
Precondición	El actor debe estar registrado y haber creado un juego en la aplicación	
Postcondición	El actor crea una nueva pregunta de uno de los diez tipos en la plataforma, actualizando los datos de la base de datos	
Descripción	Permite agregar preguntas de cada uno de los diez tipos a los juegos de un profesor, las cuales posteriormente serán contestadas por los alumnos de la clase.	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar una de las diez asignaturas en el menú principal
2	Sistema	Muestra los juegos disponibles de esa asignatura
3	Actor	Pulsar en el recuadro perteneciente a un juego ya creado
4	Sistema	El sistema muestra el formulario de edición de un juego con sus datos ya introducidos en cada campo
5	Actor	Seleccionar el tipo de pregunta deseado en el desplegable y pulsar el botón de añadir pregunta
6	Sistema	El sistema añade una pregunta nueva vacía al juego
7	Sistema	El sistema actualiza los datos del juego

Cursos Alternos

Nro.	**Descripción de acciones alternas**
-	-

- **EDITAR PREGUNTA**

Caso de Uso	Editar Pregunta	**Identificador:** CU-05
Resumen	Editar una pregunta propia ya creada en la plataforma	
Actores	Profesor	
Tipo	Primario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión y Crear Pregunta	
Precondición	El actor debe estar registrado y haber creado la pregunta a editar en la aplicación	
Postcondición	El actor edita una pregunta de un juego propio y los datos se actualizan en la base de datos	
Descripción	Permite modificar las preguntas de los juegos ya creados por el propio actor. Los datos introducidos en los campos deben ser correctos. Cada tipo de pregunta usa un formulario de datos personalizado.	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar una de las diez asignaturas en el menú principal
2	Sistema	Muestra los juegos disponibles de esa asignatura
3	Actor	Pulsar en el recuadro perteneciente a un juego ya creado
4	Sistema	El sistema muestra el formulario de edición de un juego con sus datos ya introducidos en cada campo.
5	Actor	Pulsar el recuadro perteneciente a una pregunta ya creada
6	Sistema	El sistema muestra el formulario personalizado de dicha pregunta con sus datos actuales
7	Actor	Rellenar o modificar los campos del formulario y seguir los pasos necesarios del mismo y pulsar el botón de guardar
8	Sistema	El sistema comprueba los datos introducidos
9	Sistema	El sistema actualiza los datos en la base de datos y muestra la pantalla de edición de juego

Cursos Alternos

Nro.	**Descripción de acciones alternas**
7	El actor puede cancelar la edición de la pregunta y volver al menú principal
8	Los datos introducidos pueden ser inválidos, notificando al actor cuales necesitan corrección

- **BORRAR PREGUNTA**

Caso de Uso	Borrar Pregunta	**Identificador:** CU-06
Resumen	Borrar una pregunta ya creada de un juego propio de la plataforma	
Actores	Profesor	
Tipo	Secundario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión y Crear Pregunta	
Precondición	El actor debe estar registrado y haber creado la pregunta a eliminar en la aplicación	
Postcondición	El actor elimina una pregunta de un juego propio y sus datos se borrarán de la base de datos	
Descripción	Permite borrar una pregunta de un juego ya creado por el propio actor	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar una de las diez asignaturas en el menú principal
2	Sistema	Muestra los juegos disponibles de esa asignatura
3	Actor	Pulsar en el recuadro perteneciente a un juego ya creado
4	Sistema	Muestra el formulario de edición de un juego con sus datos ya introducidos en cada campo
5	Actor	Pulsar el recuadro perteneciente a una pregunta
6	Sistema	Mostrar el formulario personalizado de dicha pregunta con sus datos actuales
7	Actor	Pulsar el icono de borrar pregunta
8	Sistema	El sistema muestra un diálogo de confirmación para la eliminación de la pregunta
9	Actor	El actor confirma la eliminación de la pregunta
10	Sistema	El sistema elimina la pregunta de la base de datos y devuelve al actor a la pantalla de edición del juego al que pertenecía

Cursos Alternos

Nro.	**Descripción de acciones alternas**
9	El actor puede cancelar la eliminación de la pregunta en este momento, con lo cual el sistema cerraría el dialogo y mostraría el formulario de edición de la misma

- **MODIFICAR PUNTOS**

Caso de Uso	Modificar Puntos	**Identificador:** CU-07
Resumen	Modificar los puntos de los alumnos pertenecientes a una clase	
Actores	Profesor	
Tipo	Primario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión y Crear Clase	
Precondición	El actor debe estar registrado y haber creado una clase en la aplicación	
Postcondición	El actor añadirá o eliminará puntos de un alumno de su clase y el sistema actualizará los datos en la base de datos	
Descripción	Permite a un profesor modificar la tabla de puntuaciones de los alumnos pertenecientes a su clase	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar en el icono de modificar puntos en la sección multijugador
2	Sistema	Muestra los alumnos de la clase junto a sus puntuaciones
3	Actor	Pulsar en el recuadro perteneciente a un alumno
4	Sistema	Muestra un dialogo que permite la modificación de puntos
5	Actor	Rellenar el valor a modificar y pulsar el botón de añadir o eliminar puntos
6	Sistema	El sistema modifica la puntuación del alumno
7	Sistema	El sistema devuelve al actor a la lista de alumnos con los valores actualizados

Cursos Alternos

Nro.	**Descripción de acciones alternas**
5	El actor puede cancelar la modificación de puntos en este momento, con lo cual el sistema cerraría el dialogo y mostraría la lista de alumnos

- **BUSCAR SALA MULTIJUGADOR**

Caso de Uso	Buscar Sala Multijugador	**Identificador:** CU-08
Resumen	Busca una sala multijugador de la clase actual disponible en la plataforma	
Actores	Alumno y Profesor	
Tipo	Primario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión	
Precondición	El actor debe estar registrado y pertenecer a una clase	
Postcondición	El actor verá en pantalla una lista de las salas multijugador disponibles en ese momento	
Descripción	Permite a un alumno o profesor ver la lista de salas multijugador disponibles para realizar acciones sobre ella	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar en el icono de buscar sala en la sección multijugador
2	Sistema	Muestra al actor la lista de salas multijugador

Cursos Alternos

Nro.	**Descripción de acciones alternas**
2	Si el actor no pertenece a una clase, el sistema no le mostrará ninguna lista y notificará que debe unirse o crear una clase para acceder a esta funcionalidad

- **ADORNAR SALA MULTIJUGADOR**

Caso de Uso	Adornar Sala Multijugador	**Identificador:** CU-09
Resumen	Adornar una sala de juego creada por un alumno	
Actores	Profesor	
Tipo	Primario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión y Buscar Sala	
Precondición	El actor debe estar registrado y haber creado una clase en la aplicación	
Postcondición	El actor adornará una sala multijugador de un alumno de su clase	
Descripción	Permite a un profesor adornar una sala multijugador para que esta dé el doble de puntuación durante el juego posterior que se realizará en ella	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar en el icono de adornar sala en la sección multijugador
2	Sistema	Muestra la lista de salas multijugador de la clase
3	Actor	Pulsar en el recuadro perteneciente a una sala
4	Sistema	Adorna la sala, mostrando un icono que lo confirma

Cursos Alternos

Nro.	**Descripción de acciones alternas**
2	Si el actor no ha creado una clase propia el sistema no le mostrará la lista de salas multijugador y notificará de que es necesaria su creación

- **CREAR SALA MULTIJUGADOR**

Caso de Uso	Crear Sala Multijugador	**Identificador:** CU-10
Resumen	Crea una sala multijugador que permite jugar con otro alumno	
Actores	Alumno	
Tipo	Primario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión	
Precondición	El actor debe estar registrado y pertenecer a una clase	
Postcondición	El actor creará una sala multijugador y se mantendrá a la espera de recibir otro alumno o modificación de puntos del profesor	
Descripción	Permite a un alumno crear una sala de espera a la que puede unirse otro alumno para iniciar un juego competitivo multijugador	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar en el icono de crear sala en la sección multijugador
2	Sistema	Muestra al alumno la sala de espera y se prepara para recibir notificaciones de alumnos o el profesor

Cursos Alternos

Nro.	**Descripción de acciones alternas**
2	Si el alumno no pertenece a ninguna clase, en vez de crearse la sala, se notificará al alumno que debe unirse a una para acceder a esta funcionalidad

- **UNIRSE A SALA MULTIJUGADOR**

Caso de Uso	Unirse a Sala Multijugador	**Identificador:** CU-11
Resumen	Unirse a la sala multijugador creada por otro alumno	
Actores	Alumno	
Tipo	Primario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión	
Precondición	El actor debe estar registrado y pertenecer a una clase en la aplicación	
Postcondición	El actor se unirá a la sala multijugador de otro alumno y se guardan los datos para mantener comunicaciones con el creador de la sala	
Descripción	Permite a un alumno unirse a una sala multijugador que ha creado otro alumno de su misma clase, preparándose el sistema para iniciar el juego multijugador cuando reciba la notificación	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar en el icono de buscar sala en la sección multijugador
2	Sistema	Muestra la lista de salas multijugador disponibles en la clase
3	Actor	Pulsar en el recuadro perteneciente a una sala
4	Sistema	Realiza una comunicación con el creador de la sala para comprobar su disponibilidad e intercambiar datos de conexión
5	Sistema	Recibe un mensaje de confirmación de disponibilidad de la sala proveniente del sistema del creador de la sala
6	Sistema	El sistema mostrará la sala multijugador y guardará los datos de la conexión

Cursos Alternos

Nro.	**Descripción de acciones alternas**
3	El actor puede volver al menú principal en este momento y cancelar el proceso
5	La sala puede encontrarse llena al momento de enviar la solicitud, por lo que el sistema recibirá un mensaje que lo confirme y se le mostrará al actor un mensaje en pantalla indicándolo

- **ABANDONAR SALA MULTIJUGADOR**

Caso de Uso	Abandonar Sala Multijugador	**Identificador:** CU-12
Resumen	Desconectarse de una sala multijugador propia o a la que te has unido	
Actores	Alumno	
Tipo	Secundario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión	
Precondición	El actor debe estar registrado y estar en una sala multijugador	
Postcondición	El actor desconectará de la sala	
Descripción	Permite a un alumno desconectarse de una sala multijugador a la que pertenezca. Si hay otro alumno en la sala se le notificará la desconexión.	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar en el icono de crear sala en la sección multijugador
2	Sistema	Muestra la sala multijugador
3	Actor	El actor pulsa el icono de salir
4	Sistema	El sistema mostrará un diálogo que pedirá confirmación del actor para abandonar la sala
5	Actor	El actor confirma la desconexión de la sala
6	Sistema	El sistema mostrará la pantalla anterior a la de la sala multijugador. Si había otro alumno en la sala se le notificará el abandono del actor

Cursos Alternos

Nro.	**Descripción de acciones alternas**
1	El actor puede pulsar el icono de buscar sala y seleccionar una sala disponible para unirse
5	El actor puede cancelar la desconexión de la sala en este momento, manteniéndose en la misma

- **REALIZAR JUEGO**

Caso de Uso	Realizar Juego	**Identificador:** CU-13
Resumen	Permite a un alumno realizar un juego de la plataforma base o creado por un profesor	
Actores	Alumno	
Tipo	Primario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión	
Precondición	El actor debe estar registrado en la plataforma	
Postcondición	El actor podrá realizar las preguntas pertenecientes al juego y obtener una puntuación al final del mismo con la que actualizar la base de datos	
Descripción	Permite a un alumno jugar a un juego ya sea base de la aplicación o creado por un profesor, los cuales ahora pueden contener preguntas de cualquiera de los diez tipos	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar una de las diez asignaturas en el menú principal
2	Sistema	Muestra los juegos disponibles de esa asignatura
3	Actor	Pulsar en el recuadro perteneciente a un juego ya creado
4	Sistema	Muestra los datos del mismo y el botón de realizar juego
5	Actor	El actor pulsa el botón de realizar juego
6	Sistema	El sistema preparará las preguntas del juego y se las mostrará en orden al alumno
7	Actor	El actor contesta a las preguntas
8	Sistema	Comprobar las respuestas y mostrar las puntuaciones. El sistema actualizará la puntuación del actor en la base de datos según los resultados obtenidos.

Cursos Alternos

Nro.	**Descripción de acciones alternas**
5	El actor puede cancelar la realización del juego pulsando el botón de volver. El sistema mostrará de nuevo la lista de juegos disponibles
7.a	El actor puede salir del juego durante la realización de las preguntas. El sistema mostrará de nuevo los datos del juego
7.b	El sistema llevará un tiempo límite del juego y cuando se acabe llevará al alumno a la pantalla de puntuaciones directamente

- **REALIZAR JUEGO MULTIJUGADOR**

Caso de Uso	Realizar Juego Multijugador	**Identificador:** CU-14
Resumen	Permite a un alumno creador de una sala realizar un juego multijugador	
Actores	Alumno	
Tipo	Primario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión y Crear Sala	
Precondición	El actor debe estar registrado en la plataforma, pertenecer a una clase y que otro alumno se haya unido a su sala multijugador	
Postcondición	El actor y el otro alumno podrán realizar las preguntas pertenecientes al juego y obtener una puntuación al final del mismo	
Descripción	Permite a dos alumnos jugar a un juego competitivo creado por la plataforma, el cual contendrá preguntas válidas al azar pertenecientes a juegos de su clase	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar botón de iniciar juego multijugador
2	Sistema	Notificar al otro alumno de que la partida ha empezado y mostrar las preguntas en orden para que el actor las responda
3	Actor	Contestar las preguntas de forma correcta sumando puntos y avanzando a las siguientes, notificando al otro alumno y haciéndole avanzar a la siguiente pregunta a la vez
4	Sistema	El sistema mostrará cuando se acaben las preguntas la pantalla de puntuaciones con los resultados
5	Sistema	El sistema actualizará la puntuación del alumno en la base de datos con los resultados obtenidos

Cursos Alternos

Nro.	**Descripción de acciones alternas**
3.a	El actor puede responder de forma errónea, con lo que el sistema no avanzará a la siguiente pregunta y se le restarán puntos. También se notificará de este cambio de puntuaciones al otro jugador.
3.b	El otro alumno puede desconectarse del juego durante su ejecución. El sistema recibirá una notificación si se da el caso y mostrará al actor la pantalla de puntuaciones directamente.

- **ABANDONAR JUEGO MULTIJUGADOR**

Caso de Uso	Abandonar Juego multijugador	**Identificador:** CU-15
Resumen	Desconectarse de un juego multijugador durante su realización	
Actores	Alumno	
Tipo	Secundario	
Referencias	Dependencia con el caso de Uso Iniciar Sesión y Realizar Juego Multijugador	
Precondición	El actor debe estar registrado y realizando un juego multijugador	
Postcondición	El actor abandonará el juego multijugador	
Descripción	Permite a un alumno desconectarse de un juego multijugador al que esté jugando. El sistema notificará al otro alumno en la sala la desconexión.	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Actor	Pulsar en el icono de salir durante un juego multijugador
2	Sistema	Muestra un diálogo de confirmación al actor
3	Actor	Confirmar el abandono del juego multijugador
4	Sistema	El sistema mostrará la pantalla anterior y notificará al otro alumno la desconexión del actor

Cursos Alternos

Nro.	**Descripción de acciones alternas**
3	El actor puede cancelar la desconexión de la sala. En ese caso el juego continuaría con normalidad.

4.2 Diagrama de Clases

La Figura 2 corresponde al diagrama de clases, un tipo de diagrama UML de estructura estática usado para describir la estructura de un sistema través de la especificación de las clases, atributos y relaciones entre objetos.

A continuación de la figura se describirá de forma breve y simple la funcionalidad de cada clase y sus atributos dentro de la plataforma.

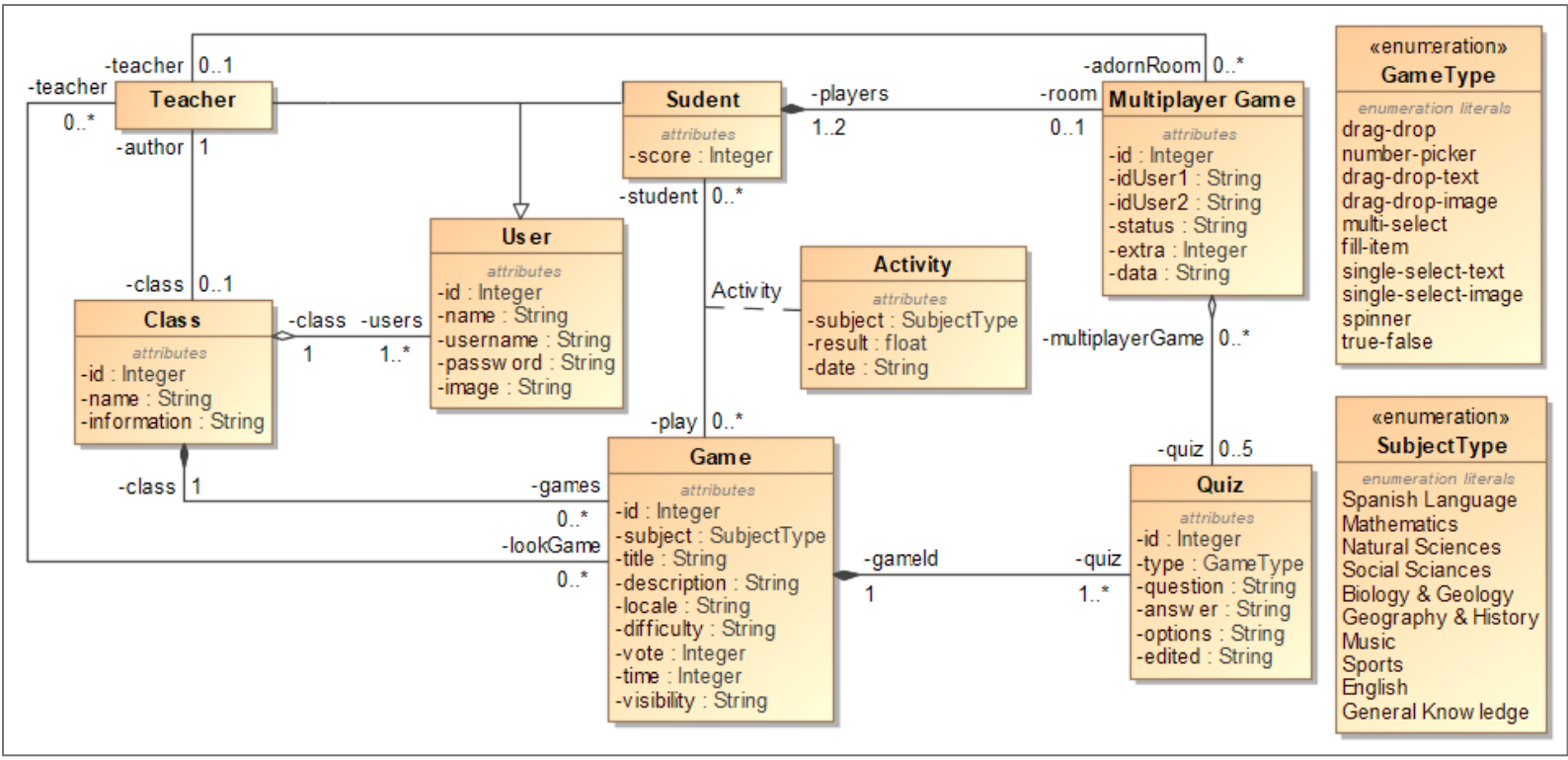

Figura 2 - Diagrama de clases

- **Usuario**

Persona con el objetivo de utilizar la aplicación. Se dividen en dos tipos según su rol: Profesor o Alumno. El alumno en esta versión de la plataforma no podrá ver las puntuaciones de los demás, siendo el profesor el único capaz. Las funcionalidades a las que pueden acceder cada uno de los roles también varían, entre ellas el acceso al editor de juegos o los juegos multijugador.

Atributo	**Descripción**	**Tipo**
Id	Identificador único del usuario	Integer
Name	Nombre completo del usuario	String
Username	Nombre de usuario único para iniciar sesión	String
Password	Contraseña para iniciar sesión. Se guardará cifrada en la base de datos	String
Image	Identificador del icono de usuario	String

- **Clase**

Objeto creado por profesores al que pueden unirse posteriormente los alumnos. Permite a los usuarios que pertenezcan a este objeto acceder a los elementos multijugador y a la creación y edición de juegos dependiendo de su rol. Los usuarios que no estén unidos a una clase pertenecerán a la clase por defecto, la cual impedirá acceder a las funciones mencionadas anteriormente.

Atributo	**Descripción**	**Tipo**
Id	Identificador único de la clase	Integer
Name	Nombre de la clase definido por el profesor	String
Information	Descripción de la clase definida por el profesor	String

- **Juego**

Clase que almacena la información necesaria para que los alumnos puedan realizar juegos. Está directamente relacionado con las preguntas que le pertenecen, y solo será visible para los alumnos que pertenezcan a la clase en la que se creó. Un juego puede no estar visible para su clase, por ejemplo, cuando está en proceso de edición.

Atributo	**Descripción**	**Tipo**
Id	Identificador único del juego	Integer
Subject	Asignatura a la que pertenece el juego	SubjectType
Title	Título del juego dado por el profesor	String
Description	Descripción del juego dada por el profesor	String
Locale	Idioma al que pertenece el juego. Solo será visible para dispositivos con el mismo idioma.	String
Difficulty	Dificultad del juego. Cuanto mayor sea, más puntuación dará completarlo con éxito	String
Vote	Puntuación que los alumnos le dan al juego	Integer
Time	Tiempo límite para completar el juego	Integer
Visibility	Permite que el juego sea o no visible según su estado	String

- **Pregunta**

Cada uno de los ejercicios que el alumno debe responder durante la realización de un juego. Dependiendo del tipo de juego cada atributo se utilizará de una forma u otra para almacenar los datos de la pregunta. El campo editado es fundamental, hace que un juego no pueda ser visible si contiene preguntas sin editar, ya que están vacías en el momento de creación.

Atributo	**Descripción**	**Tipo**
Id	Identificador único de la pregunta	Integer
Type	Tipo de la pregunta, cambio importante en la refactorización de la plataforma que permite añadir preguntas de cualquier tipo a los juegos	GameType
Question	Contiene la pregunta hacia el alumno	String
Answer	Contiene la respuesta correcta	String
Options	Contiene las opciones disponibles para el alumno	String
Edited	Estado actual de la pregunta	String

- **Actividad**

Acciones de los alumnos sobre los juegos. Cada vez que un alumno completa un juego se guarda su actividad, permitiendo la monitorización por parte del profesor.

Atributo	Descripción	Tipo
Subject	Asignatura a la que pertenece	SubjectType
Result	Nota del alumno tras la realización del juego	Float
Date	Fecha en la que se creó la actividad	String

- **Juego Multijugador**

Elemento que guarda la información que permite la comunicación entre dispositivos y el estado de la sala multijugador. Los identificadores únicos de Firebase se generan de forma automática por el propio servidor para cada dispositivo Android.

Atributo	Descripción	Tipo
Id	Identificador único de la sala	Integer
IdUser1	Identificador del creador de la sala en Firebase	String
IdUUser2	Identificador del segundo jugador en Firebase	String
Status	Estado de la sala actual. Puede estar libre o llena	String
Extra	Muestra si la sala está adornada o no	Integer
Data	Almacena datos del creador de la sala como su nombre e icono para mostrarlos cuando se buscan salas multijugador	String

4.3 Diagrama de Navegación

El diagrama de la Figura 3 representa el conjunto de estados de la aplicación de Android y las transiciones entre ellos según las acciones que realice el usuario, siendo el estado inicial la pantalla de inicio de sesión Log In y la pantalla principal el Main Menu, donde se encuentran la mayoría de funcionalidades de la aplicación.

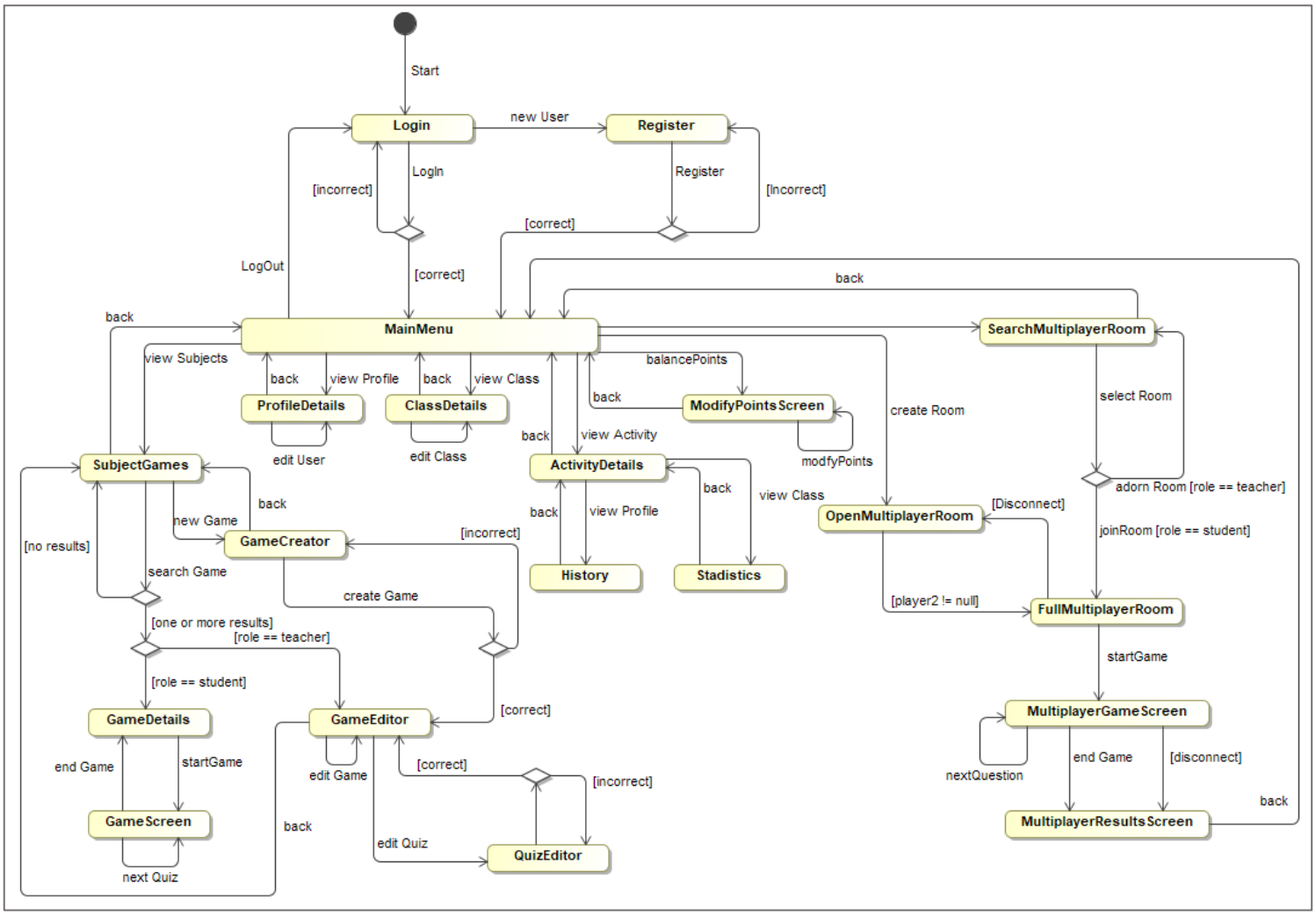

Figura 3 - Diagrama de estados

En total hay 20 estados en el diagrama, los cuales son:

- **Login:** Estado inicial donde el usuario puede iniciar sesión.
- **Register:** Pantalla que permite crear un nuevo usuario en la aplicación.
- **MainMenu:** Pantalla principal de la aplicación a la que los usuarios registrados pueden acceder.
- **SubjectGames:** Lista de juegos de una asignatura concreta.
- **GameDetails:** Pantalla en la que se muestran los datos de un juego.
- **GameScreen:** Muestra las preguntas de un juego en orden y posteriormente los resultados.
- **GameCreator:** Formulario que permite a un profesor crear un juego.
- **GameEditor:** Formulario que permite a un profesor editar un juego y, además, crear preguntas para dicho juego.
- **QuizEditor:** Pantalla en la que se pueden editar cada uno de los tipos de pregunta de la aplicación.
- **ProfileDetails:** Muestra los datos del usuario y permite la edición de los mismos.
- **ClassDetails:** Pantalla en la que se muestran los datos de la clase.
- **ActivityDetails:** Muestra la actividad del usuario en la aplicación.
- **History:** Pantalla que muestra el historial del usuario en cuanto a juegos realizados.
- **Stadistics:** Pantalla que puede mostrar tanto las estadísticas del alumno como los de la clase.
- **ModifyPointsScreen:** Pantalla que muestra los alumnos de la clase y sus puntuaciones permitiendo modificarlas.
- **SearchMultiplayerRoom:** Muestra la lista de salas multijugador disponibles.
- **OpenMultiplayerRoom:** Pantalla que muestra los datos de la sala a su creador y se mantiene a la espera de recibir notificaciones.
- **FullMultiplayerRoom:** Sala que cuenta con dos jugadores y permite iniciar el juego.
- **MultiplayerGameScreen:** Pantalla que muestra las preguntas del juego en orden.
- **MultiplayerResultScreen:** Pantalla que muestra los resultados del juego multijugador.

4.4 Diagrama de Servicios

La aplicación Android accede al servidor a través de una API creada a partir del micro framework Slim basada en lenguaje PHP que le proporciona una serie de servicios y funcionalidades. Para representar esta capa de abstracción vamos a utilizar un diagrama de clases con las cinco interfaces que la forman (Usuario, Juego, Quiz, Clase y Juego Multijugador) mostradas en la Figura 4. Cabe destacar que en este diagrama también aparecerán los servicios que no han sido creados por Slim, los cuales son la mensajería en tiempo real y la subida de archivos al servidor.

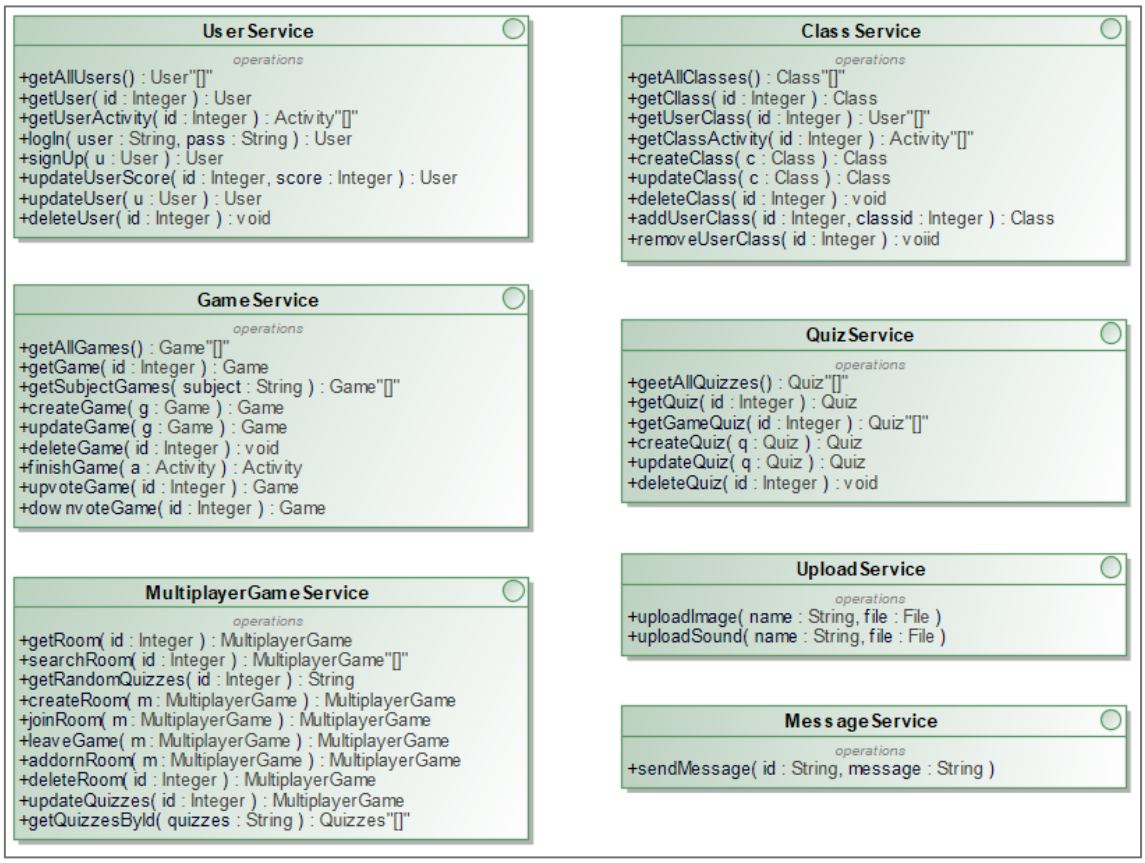

Figura 4 - Diagrama de servicios

4.5 Diagrama de Arquitectura

A continuación, se dispondrá el diagrama de despliegue, un tipo de diagrama UML utilizado para modelar la disposición física de los artefactos software en tiempo de ejecución de la plataforma repartidos en nodos (Figura 5). Hay tres nodos en esta versión de la plataforma: el servidor principal en el que está instalada la aplicación web junto a su conexión hacia la base de datos SQL, el dispositivo móvil Android en el que está instalada la aplicación móvil y, por último, el servidor Firebase al que accede el servidor principal a través de una API para enviar notificaciones a los dispositivos móviles de la plataforma.

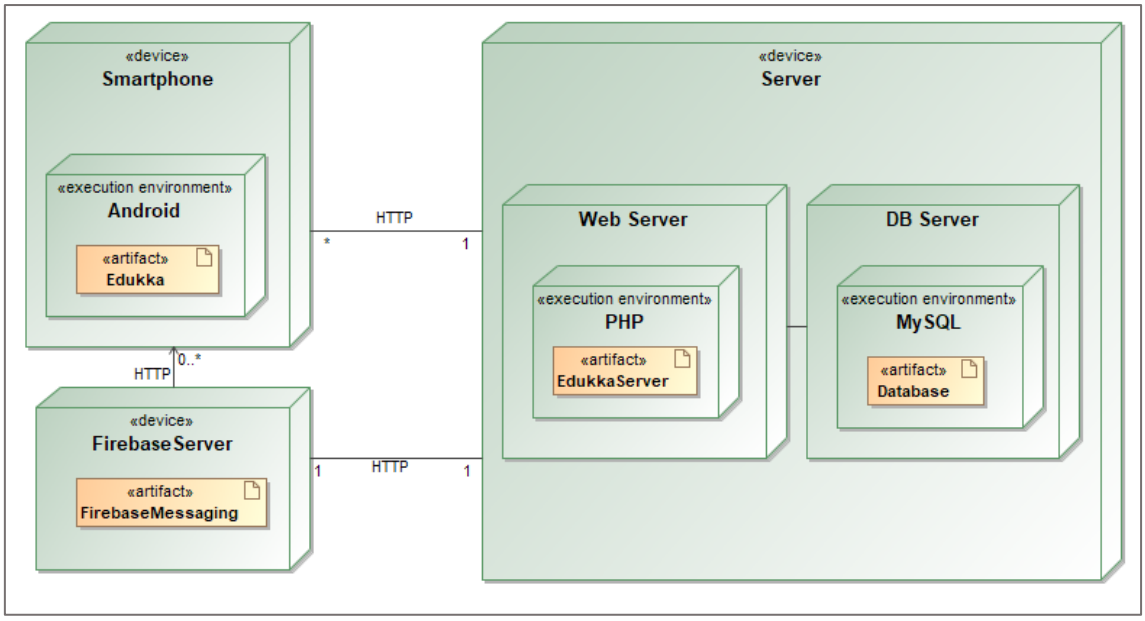

Figura 5 - Diagrama de arquitectura

Capítulo 5. Desarrollo e Implementación

5.1 Desarrollo del Back-end

En esta versión de la plataforma el back-end está formado por dos servidores, entre los cuales tenemos el principal basado en PHP desarrollado por nosotros y desplegado en la plataforma a nuestra elección. Este servidor principal está directamente conectado a la base de datos y accede a través de una API a la instancia Firebase desplegada en los servidores de Google para nuestra aplicación. Ahora se van a detallar las implementaciones de ambos.

5.1.1 Servidor Principal

Forma la aplicación Web como tal y lo componen un total de 12 ficheros PHP, además de los ficheros autogenerados por el micro framework Slim que están en su correspondiente carpeta. Los 12 ficheros PHP se dividen en 5 principales que forman la API que otorga los servicios a los que accede el front-end y están en la raíz del proyecto, un archivo index.php que recoge las funcionalidades de los 5 ficheros anteriores y luego contamos con las carpetas “images”, “sounds” y “firebase”.

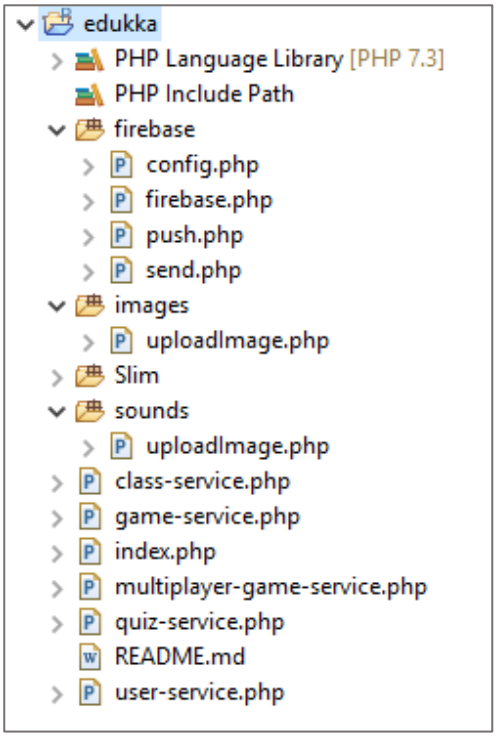

Figura 6 - Estructura Aplicación Web

Las carpetas de imágenes y sonidos cuentan cada una con un fichero PHP que permite la subida al servidor de archivos de audio y video en su correspondiente carpeta usando la siguiente función (Figura 7):

```php
$result = array("success" => $_FILES["file"]["name"]);
$file_path = basename( $_FILES['file']['name']);
if(move_uploaded_file($_FILES['file']['tmp_name'], $file_path)) {
    $result = array("success" => "File successfully uploaded");
} else{
    $result = array("success" => "error uploading file");
}
echo json_encode($result, JSON_PRETTY_PRINT);
```

Figura 7 - Código Subida Archivos

El control del tipo de archivos a subir al servidor se realiza en el front-end, del que hablaremos más adelante.

En la carpeta Firebase contamos con 4 ficheros, de los cuales el principal es Firebase.php, cuya función principal podréis ver en la Figura 8:

```php
private function sendPushNotification($fields) {

    //Access to the config data
    require_once __DIR__ . '/firebase/config.php';

    //Use the POST method of the Google's API
    $url = 'https://fcm.googleapis.com/fcm/send';

    //Set the Key from config
    $headers = array(
        'Authorization:' . FIREBASE_API_KEY,
        'Content-Type: application/json'
    );

    //Open a connection channel and set values
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, $url);
    curl_setopt($ch, CURLOPT_POST, true);
    curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
    curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($fields));

    //Execute the POST action
    $result = curl_exec($ch);
    if ($result === FALSE) {
        die('Curl failed: ' . curl_error($ch));
    }

    // Close connection
    curl_close($ch);
    return $result;
}
```

Figura 8 - Código Firebase.php

A través de este archivo, al cual la aplicación Web accederá por petición del front-end cuando un usuario quiera comunicarse con otro, se establece la conexión entre servidores a través de la petición POST 'https://fcm.googleapis.com/fcm/send', a la que responderá la API de Firebase alojada en los propios servidores de Google.

Para dar el formato JSON al mensaje y obtener los datos de configuración se accede a los otros archivos PHP de la carpeta, siendo Config.php el archivo que almacena datos de conexión como la clave de la API Firebase (Figura 9), Send.php el que obtiene las IDs de los dispositivos a los que se destinan los mensajes (Figura 10) y por último Push.php el archivo que da formato JSON al mensaje a enviar (Figura 11).

```php
require_once __DIR__ . '/firebase.php';
require_once __DIR__ . '/push.php';

//get the connection
$firebase = new Firebase();
//get the format
$push = new Push();

//get the message from Front-end
$payload = array();
$payload['type'] = 'Comunication';
$message = $_POST["message"];
$push->setTitle('Notification');
$push->setMessage($message);
$push->setIsBackground(FALSE);
$push->setPayload($payload);

//get the IID from Front-end
$json = '';
$json = $push->getPush();
$regId = $_POST["firebase_id"];
$response = $firebase->send($regId, $json);
echo $response;
```

Figura 10 - Código Send.php

```php
<?php

define('FIREBASE_API_KEY', 'AAAAEFuurP8:APA91bGDo...');

?>
```

Figura 9 - Código Config.php

```php
public function getPush() {
    $res = array();
    $res['data']['title'] = $this->title;
    $res['data']['is_background'] = $this->is_background;
    $res['data']['message'] = $this->message;
    $res['data']['image'] = $this->image;
    $res['data']['payload'] = $this->data;
    $res['data']['timestamp'] = date('Y-m-d G:i:s');
    return $res;
}
```

Figura 11 - Cóodigo Push.php

En la interfaz de Eclipse no se muestran algunos elementos, pero la aplicación también cuenta con un archivo .htaccess que permite la configuración de ciertas variables a la hora de desplegar nuestro servidor en el dominio deseado. Su código se muestra a continuación (Figura 12).

```
# HTID:4321906: DO NOT REMOVE OR MODIFY THIS LINE AND THE LINES BELOW
php_value display_errors 1
# DO NOT REMOVE OR MODIFY THIS LINE AND THE LINES ABOVE HTID:4321906:
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^ index.php [QSA,L]
```

Figura 12 - Código .htaccess

Por último, y no menos importante, hay que destacar la estructura y funcionamiento de los ficheros que están en la raíz del proyecto. En la primera imagen se va a mostrar un segmento del archivo PHP principal denominado index.php (Figura 13), el cual tiene acceso al resto de archivos que pertenecen a la API, importa el framework Slim y crea una instancia del mismo para obtener respuestas en formato JSON y, finalmente, crea una conexión a la base de datos usando la extensión PDO.

```php
//Set the route and access to the other classes
require 'C:\wamp64\www\edukka\Slim\Slim.php'; //require '.././Slim/Slim.php';
require_once 'user-service.php';
require_once 'class-service.php';
require_once 'game-service.php';
require_once 'quiz-service.php';
require_once 'multiplayer-game-service.php';
\Slim\Slim::registerAutoloader();

//Creates the Slim instance
$app = new \Slim\Slim();
$app->contentType('application/json');

$app->get('/', function () use ($app) {
    $app->response->redirect('index.html');
});

// Functions of the User Service
$app->get('/user/:id', 'getUser');
//...

//Executes the application
$app->run();

//Create the conection with the DB
function getDB() {
    $dbhost = 'localhost';
    $dbuser = 'root';       //$dbuser = 'id5255892_root';
    $dbpass = '';           //$dbpass = 'k4zGDiZJ6EqCKnkDOhAH';
    $dbname = 'edukka';     //$dbname = 'id5255892_edukka';

    $mysql_conn_string = "mysql:host=$dbhost;dbname=$dbname;charset=utf8mb4";
    $dbConnection = new PDO($mysql_conn_string, $dbuser, $dbpass);
    $dbConnection->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    return $dbConnection;
}
```

Figura 13 - Código Index.php

Las funciones de cada servicio se hayan en su respectiva clase. Podemos observar un ejemplo de funcionamiento en la Figura 14, las cuales crean una instancia SQL y le introducen valores obtenidos de la petición GET o POST para, a continuación, ejecutarla y enviar la respuesta en formato JSON al front-end.

```php
function getGame($id) {
    $sql = 'SELECT * FROM game WHERE id = ?';
    try {
        $db = getDB();
        $stmt = $db->prepare($sql);
        $stmt->bindValue(1, $id);
        $stmt->execute();
        $game = $stmt->fetchObject();
        if ($game === false) {
            $game = ['id'=>null];
        }
        $db = null;
        echo json_encode($game);
    } catch(PDOException $e) {
        echo json_encode($e->getMessage());
    }
}
```

Figura 14 - Código Game-Service.php

La aplicación cuenta ahora con 42 funciones, de entre las cuales solo describiremos las que han sufrido cambios en esta versión y las que se han añadido en dos tablas:

Método	URL	Descripción
POST	/user/score	Ahora actualiza la puntuación a partir del valor actual sumado a la puntuación obtenida
POST	/quiz/new	Ahora incluye los campos ‘type’ y ‘edited’ en la creación de preguntas
POST	/quiz/edit	Ahora modifica el campo ‘edited’ al editar la pregunta
POST	/game/new	Ahora incluye los campos ‘time’, ‘visibility’ y ‘classId’ en la creación de juegos
POST	/game/edit	Ahora permite modificar los campos ‘time’ y ‘visibility’ en la edición de juegos

Tabla 1 - Funciones cambiadas

Método	URL	Descripción
GET	/room/:id	Devuelve los datos de una sala
GET	/rooms/:id	Devuelve una lista con las salas libres de una clase
GET	/room /randomquizzes /:id	Obtiene hasta 5 preguntas aleatorias editadas pertenecientes a la clase del usuario que realizó la petición
POST	/room/new	Crea una sala multijugador
POST	/room/join	Permite a un alumno unirse a una sala multijugador
POST	/room/leave	Permite a un alumno abandonar la sala multijugador
POST	/room/adorn	Permite a un profesor adornar una sala
POST	/room/delete	Borra una sala multijugador
POST	/room /updatequizzes	Actualiza las preguntas aleatorias de una sala multijugador
POST	/room /getquizzesbyid	Obtiene la lista de preguntas de una sala a partir de sus IDs

Tabla 2 - Funciones añadidas

5.1.2 Servidor Firebase

La aplicación cuenta con una instancia en la plataforma Firebase de Google, que en este caso se ocupa únicamente de gestionar el tráfico de mensajes entre usuarios para mantener la comunicación multijugador.

Figura 15 - Icono Firebase

Esta no es la única función de la que se podría encargar esta plataforma, ya que permite el despliegue de aplicaciones web y el mantenimiento de bases de datos en tiempo real, que notifican a los dispositivos de los cambios de datos en cuanto se producen para tener siempre la información actualizada y mucho más.

La pantalla principal de configuración se va a mostrar a continuación en la siguiente imagen. Para que el servidor haga su función hace falta registrar la aplicación Android y añadir el fichero "*google-services.json*" al proyecto en Android Studio.

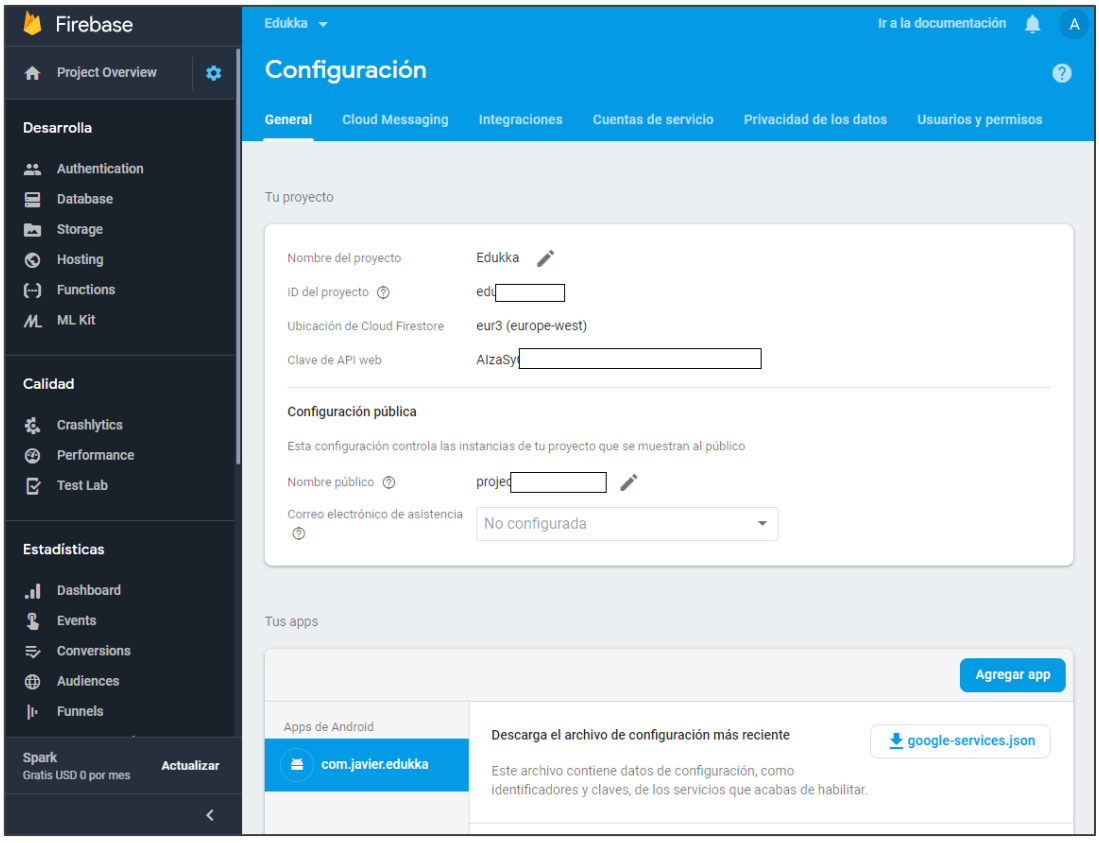

Figura 16 - Configuración Firebase

Como se ve en la Figura 16, esta plataforma es capaz de brindar a las aplicaciones desplegadas herramientas para manejar autenticación, gestión de bases de datos no relacionales, almacenamiento de archivos en línea, acceso y manejo de los archivos de la aplicación desplegada en línea y mucho más, como mostrar estadísticas de la aplicación tales como la cantidad de usos diarios entre sus usuarios. Esta última funcionalidad se muestra en la Figura 17.

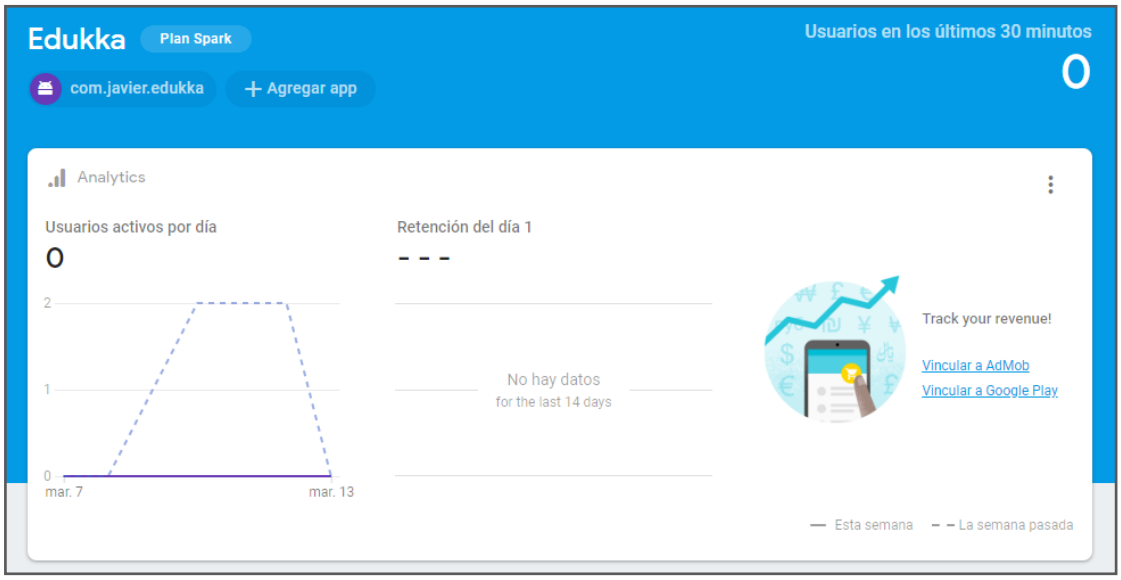

Figura 17 - Estadísticas en Firebase

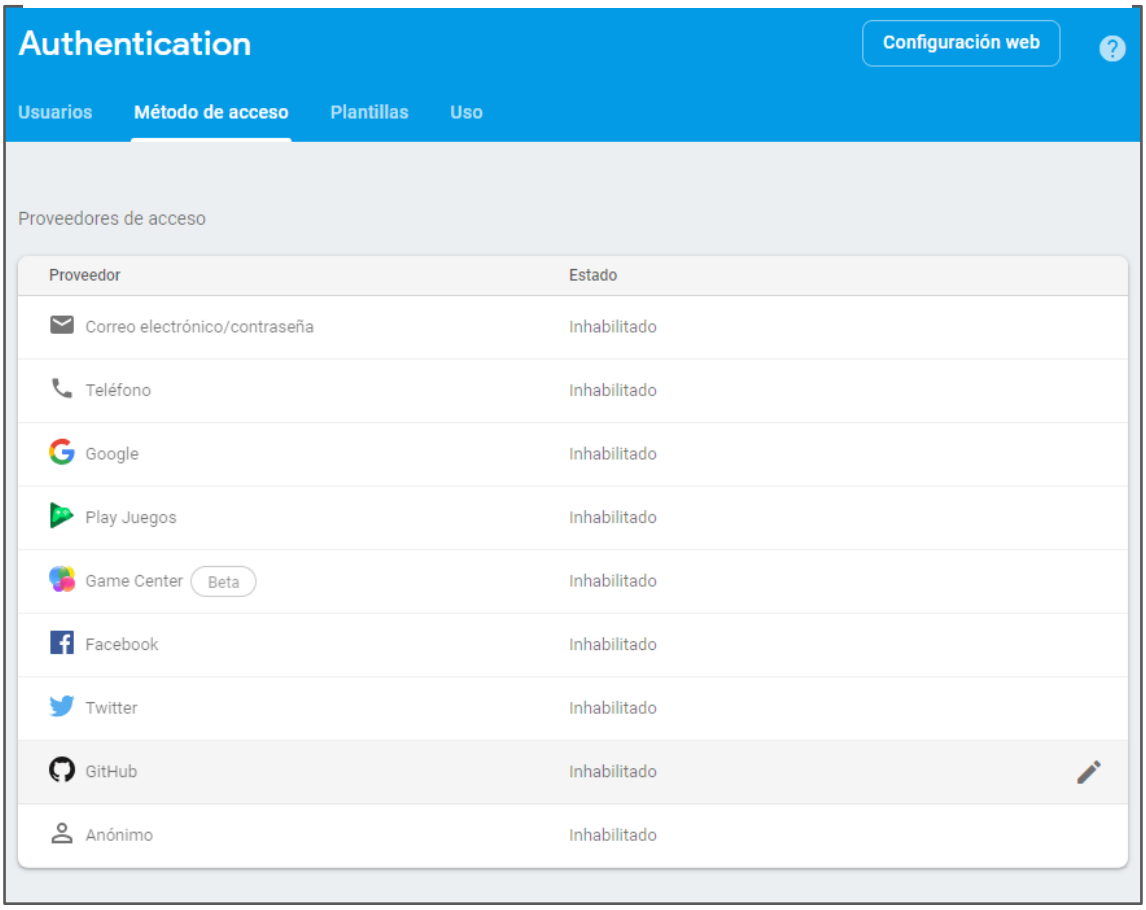

Figura 18 - Métodos Acceso Firebase

5.2 Desarrollo del Front-end

5.2.1 Estructura

En esta parte de la plataforma contamos con una aplicación Android que sigue el patrón MVC (Modelo Vista Controlador) con una estructura ordenada en paquetes según la funcionalidad de las clases, dando bastante importancia a la reutilización de código para facilitar la comprensión posterior de los posibles desarrolladores que quieran mejorar la plataforma. Además, se mantiene la estética de Google Material Design en cada uno de los layouts y se ha utilizado el patrón ViewHolder en las listas de datos para asegurar un mayor rendimiento en cualquier dispositivo. La prioridad de las modificaciones realizadas en el front-end ha sido mantener estas características intactas, adaptándose en todo momento a ellas durante la implementación.

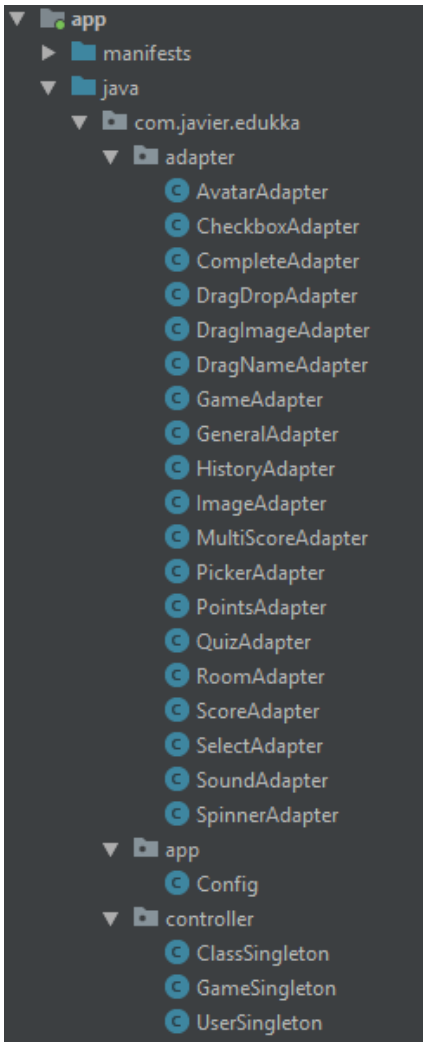

Figura 19 - Estructura Android 1

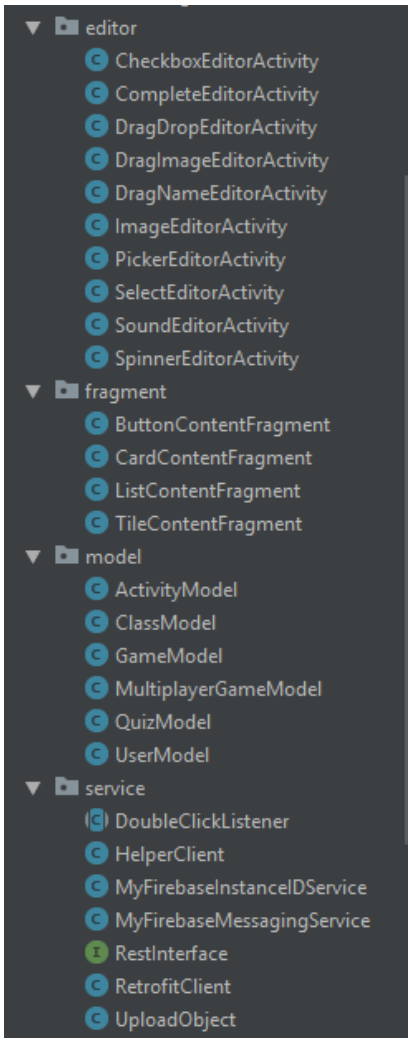

Figura 20 - Estructura Android 2

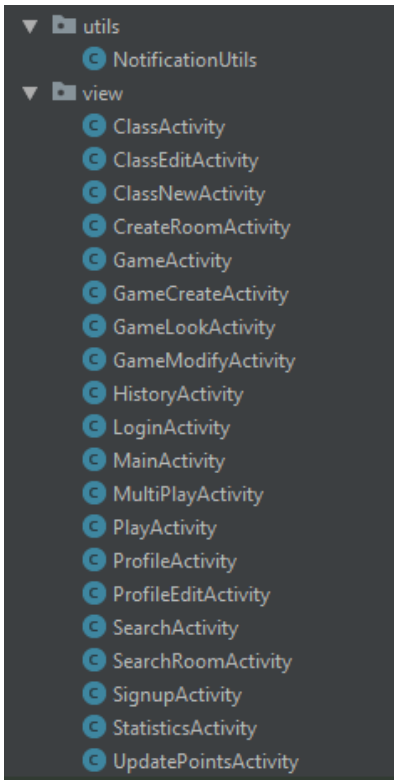

Figura 21 - Estructura Android 3

En las 3 figuras anteriores se muestra la estructura final del proyecto Android en el que se ha respetado el orden ya existente y se han añadido nuevos paquetes en caso de necesitarse. En esta versión la aplicación se divide en 9 nodos o carpetas:

- **Adapter:** Clases dedicadas a mostrar las listas de datos obtenidas del back-end en un formato agradable y comprensible para el usuario.
    - AvatarAdapter: Adaptador dedicado a mostrar los usuarios y sus imágenes.
    - CheckboxAdapter: Adaptador para mostrar las preguntas tipo Checkbox con sus opciones y que permite enviar la respuesta del alumno.
    - CompleteAdapter: Adaptador para mostrar las preguntas tipo Complete con su imagen y que permite enviar la respuesta del alumno.
    - DragDropAdapter: Adaptador para mostrar las preguntas tipo DragDrop con sus recuadros de palabras y que permite enviar la respuesta del alumno.
    - DragImageAdapter: Adaptador para mostrar las preguntas tipo DragImage con sus recuadros de imágenes y que permite enviar la respuesta del alumno.
    - DragNameAdapter: Adaptador para mostrar las preguntas tipo DragName con sus imágenes y palabras y que permite enviar la respuesta del alumno.
    - GameAdapter: Adaptador dedicado a mostrar la lista de juegos de una asignatura.
    - GeneralAdapter: Adaptador que gestiona el tipo de adaptador a utilizar según el tipo de pregunta que debe mostrar en pantalla.
    - HistoryAdapter: Adaptador para mostrar las actividades realizadas por un usuario en forma de historial.
    - ImageAdapter: Adaptador para mostrar las preguntas tipo Image con sus 4 imágenes y que permite enviar la respuesta del alumno.
    - MultiScoreAdapter: Adaptador para mostrar la pantalla final de puntuaciones en una partida multijugador.
    - PickerAdapter: Adaptador para mostrar las preguntas tipo Picker junto a una barra numérica y que permite enviar la respuesta del alumno.
    - PointsAdapter: Adaptador para mostrar la lista de usuarios de una clase junto a su puntuación.
    - QuizAdapter: Adaptador para mostrar en el editor de juegos la lista de preguntas que contiene.
    - RoomAdapter: Adaptador para mostrar las salas multijugador disponibles.

    - ScoreAdapter: Adaptador usado para mostrar los resultados al realizar un juego con sus aciertos y fallos.
    - SelectAdapter: Adaptador para mostrar las preguntas tipo Select con un verdadero o falso y que permite enviar la respuesta del alumno.
    - SoundAdapter: Adaptador para mostrar las preguntas tipo Sound con un reproductor y sus opciones y que permite enviar la respuesta del alumno.
    - SpinnerAdapter: Adaptador para mostrar las preguntas tipo Spinner con un desplegable y que permite enviar la respuesta del alumno.

- **App:** Carpeta dedicada a almacenar clases destinadas a recoger claves o diversos elementos de configuración.
    - Config: Clase que contiene constantes para el manejo de notificaciones.

- **Controller:** Carpeta con controladores que almacenan información del usuario.
    - ClassSingleton: Clase que almacena la instancia de la clase del usuario actual.
    - GameSingleton: Clase que almacena la instancia del juego actual del usuario.
    - UserSingleton: Clase que almacena la instancia del usuario actual al iniciar sesión.

- **Editor:** Carpeta donde se guardan las clases destinadas a la gestión de formularios para la edición de preguntas. Todas estas clases sirven para mostrar su propio formulario personalizado que expondremos en más detalle más adelante.

- **Fragment:** Fragmentos destinados a crear partes de una interfaz, que en conjunto forman una interfaz multifuncional.
    - ButtonContentFragment: Fragmento con las funcionalidades multijugador.
    - CardContentFragment: Fragmento que muestra una lista de actividades.
    - ListContentFragment: Fragmento que muestra la lista de usuarios.
    - TileContentFragment: Fragmento que muestra la lista de asignaturas.

- **Model:** Clases que recogen y dan el formato de objetos a los datos en JSON recibidos del back-end.
    - ActivityModel: Modelo de objeto de las actividades.
    - ClassModel: Modelo de objeto de las clases.
    - GameModel: Modelo de objeto de los juegos.
    - MultiplayerGameModel: Modelo de objeto de las salas multijugador.
    - QuizModel: Modelo de objeto de las preguntas.
    - UserModel: Modelo de objeto de los usuarios.

- **Service:** Carpeta que recoge clases dedicadas a dar funcionalidades y servicios en segundo plano.
    - DoubleClickListener: Clase dedicada a detectar dos pulsaciones en la pantalla.
    - HelperClient: Clase con métodos que facilitan la programación, como el acceso a la dirección de despliegue o la traducción inglés-español.
    - MyFirebaseInstanceIDService: Clase que recoge la clave única del dispositivo Android en la plataforma Firebase.
    - MyFirebaseMessagingService: Clase que se encarga de recibir notificaciones del servidor Firebase y su posterior procesado.
    - RestInterface: Interfaz que contiene las funciones del servidor principal.
    - RetrofitClient: Clase que realiza la conexión a la base de datos.
    - UploadObject: Clase que le da formato a la respuesta al subir un archivo al servidor principal.

- **Utils:** Contiene clases para personalizar ciertas interacciones y darles distintos formatos visuales.
    - NotificationsUtils: Personaliza las acciones al recibir una notificación dependiendo de su tipo.

- **View:** Carpeta con las vistas principales de la aplicación.
    - ClassActivity: Vista que muestra la información de una clase.
    - ClassEditActivity: Vista con el formulario para editar los datos de una clase.
    - ClassNewActivity: Vista con el formulario para crear una nueva clase.

- CreateRoomActivity: Vista con los datos de la sala multijugador.
- GameActivity: Vista que muestra los datos de un juego.
- GameCreateActivity: Vista con el formulario para crear un juego nuevo.
- GameLookActivity: Vista que muestra las preguntas y soluciones de un juego.
- GameModifiActivity: Vista con el formulario para editar un juego y con la lista de preguntas que le pertenecen.
- HistoryActivity: Vista con las actividades de un alumno o de la clase.
- LoginActivity: Vista que permite el inicio de sesión en la aplicación.
- MainActivity: Pantalla principal de la aplicación que muestra el menú con sus 4 apartados correspondientes.
- MultiPlayActivity: Vista que muestra las preguntas en modo multijugador y las puntuaciones de ambos usuarios.
- PlayActivity: Vista que muestra las preguntas de un juego en orden y permite la interacción por parte del usuario.
- ProfileActivity: Vista que muestra los datos del usuario.
- ProfileEditActivity: Vista que permite editar los datos de usuario.
- SearchActivity: Vista que muestra la lista de juegos de una asignatura.
- SearchRoomActivity: Vista que muestra las salas multijugador disponibles.
- SignupActivity: Vista que muestra el formulario para registrarse en la plataforma.
- StatisticsActivity: Vista que muestra las estadísticas del usuario.
- UpdatePointsActivity: Vista que muestra los usuarios de una clase y permite modificar sus puntos.

5.2.2 Modificaciones propuestas

Esta fue la primera fase del desarrollo de la aplicación Android, que consistía en construir una pantalla de creación y edición de juegos con sus correspondientes preguntas, pero encontramos una pequeña pared: la plataforma base solo permitía añadir un tipo de pregunta por asignatura. Es decir, la asignatura de Matemáticas tenía acceso único y exclusivo a las preguntas de tipo Picker (La barra desplazable con valores numéricos), lo mismo que la asignatura de Conocimiento General con las preguntas de verdadero y falso.

Para refactorizar este diseño y darles mayor versatilidad a los juegos con objetivo de hacerlos más entretenidos y variados para los niños se realizaron cambios en la base de datos, declarando el tipo de pregunta en su propia instancia y se modificó el código de la aplicación añadiendo un adaptador general de preguntas, el cual crea adaptadores de cada tipo y los va utilizando según el tipo de pregunta que siga. De esta forma accede a los métodos propios de forma simple y permite una refactorización posterior mucho más fácil sin reutilizar código. La función mostrada en la Figura 22 utiliza el método propio de cada adaptador para recopilar la pregunta dada por el usuario.

```
public Object getItem(int i) {
    String str = "";
    switch(types.get(i)) {
        case "dragdrop":
            str = dragDropAdapter.getItem(i).toString();
            break;
        case "picker":
            str = pickerAdapter.getItem(i).toString();
            break;
        case "dragname":
            str = dragNameAdapter.getItem(i).toString();
            break;
        case "dragimage":
            str = dragImageAdapter.getItem(i).toString();
            break;
        case "checkbox":
            str = checkboxAdapter.getItem(i).toString();
            break;
        case "complete":
            str = completeAdapter.getItem(i).toString();
            break;
        case "sound":
            str = soundAdapter.getItem(i).toString();
            break;
        case "image":
            str = imageAdapter.getItem(i).toString();
            break;
        case "spinner":
            str = spinnerAdapter.getItem(i).toString();
            break;
        case "select":
            str = selectAdapter.getItem(i).toString();
            break;
    }
    return str;
```

Figura 22 - Adaptador General

Para darle más dinamismo a los juegos, en esta versión se ha añadido un temporizador, el cual marca el tiempo límite disponible para realizar el juego. Este campo es editable desde el editor de juegos y será visible por los alumnos en los datos del juego antes de iniciarlo. Para crear el contador se ha implementado la librería externa RingProgressBar, dándole un aspecto más llamativo. En la Figura 23 se puede observar cómo se ve el contador durante un juego.

Si durante el juego el contador de tiempo llega a cero, el juego terminará de forma automática y pondrán como erróneas las preguntas que falten por responder, como se puede observar en la Figura 24.

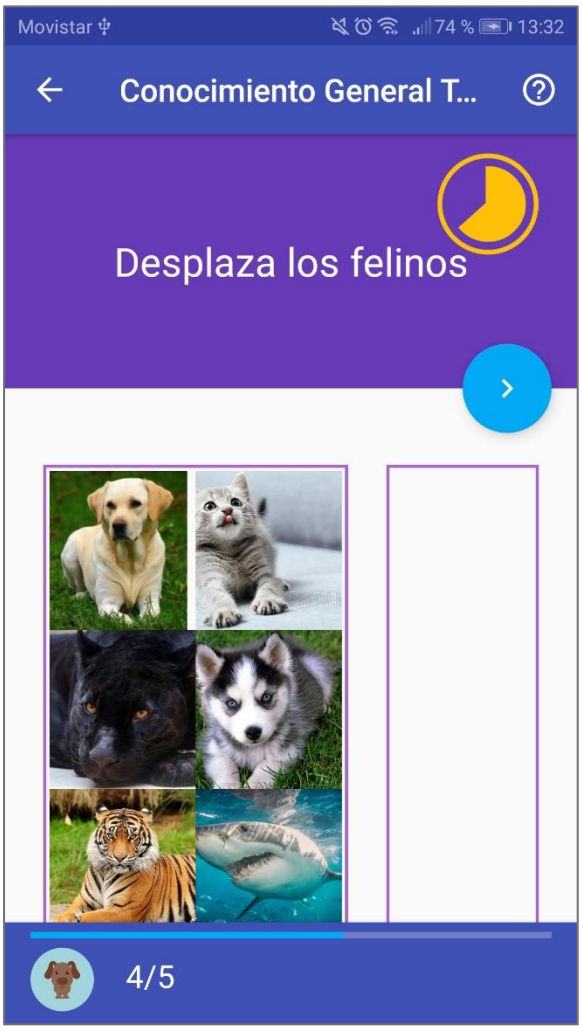

Figura 23 - Contador de la App

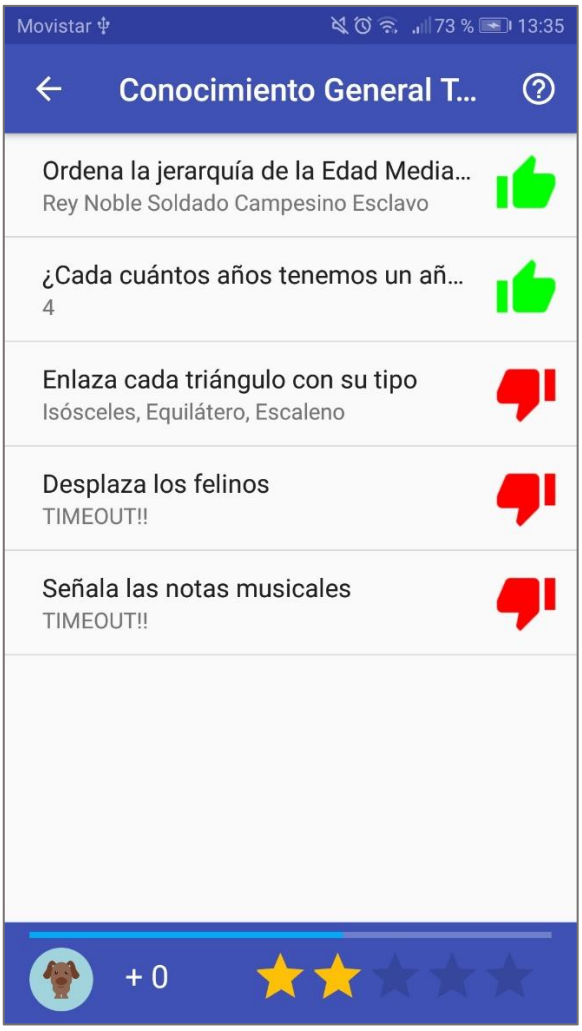

Figura 24 - Contador a cero

Ahora los alumnos no pueden ver las puntuaciones de la clase, mientras que el profesor sí que podrá verlas. Lo que se persigue con este cambio es que los niños no pierdan la motivación de llegar arriba porque haya mucha diferencia de puntos entre puestos ni se frustren con las puntuaciones, ya que el principal objetivo de la aplicación es todo lo contrario, buscamos que se diviertan aprendiendo. Sin embargo, se siguen manteniendo las medallas para los 3 primeros como forma de recompensar la constancia en la aplicación como se muestra en la Figura 25.

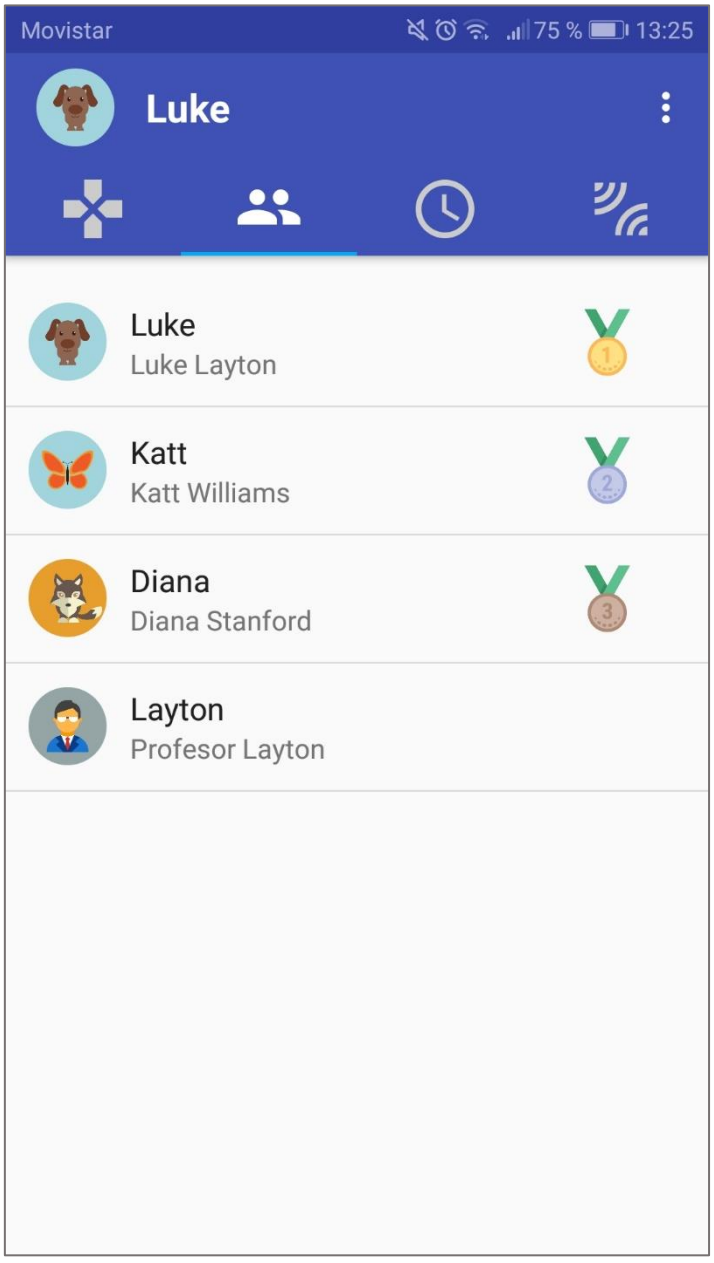

Figura 25 - Cambios en la tabla de puntuaciones

También y, como ya hemos mencionado anteriormente, ahora la pantalla en la que se visualizan los datos de juego también mostrará el tiempo límite del que dispone el alumno para realizar el juego junto a los demás datos de relevante importancia, como el nombre, la descripción del mismo y los votos recibidos.

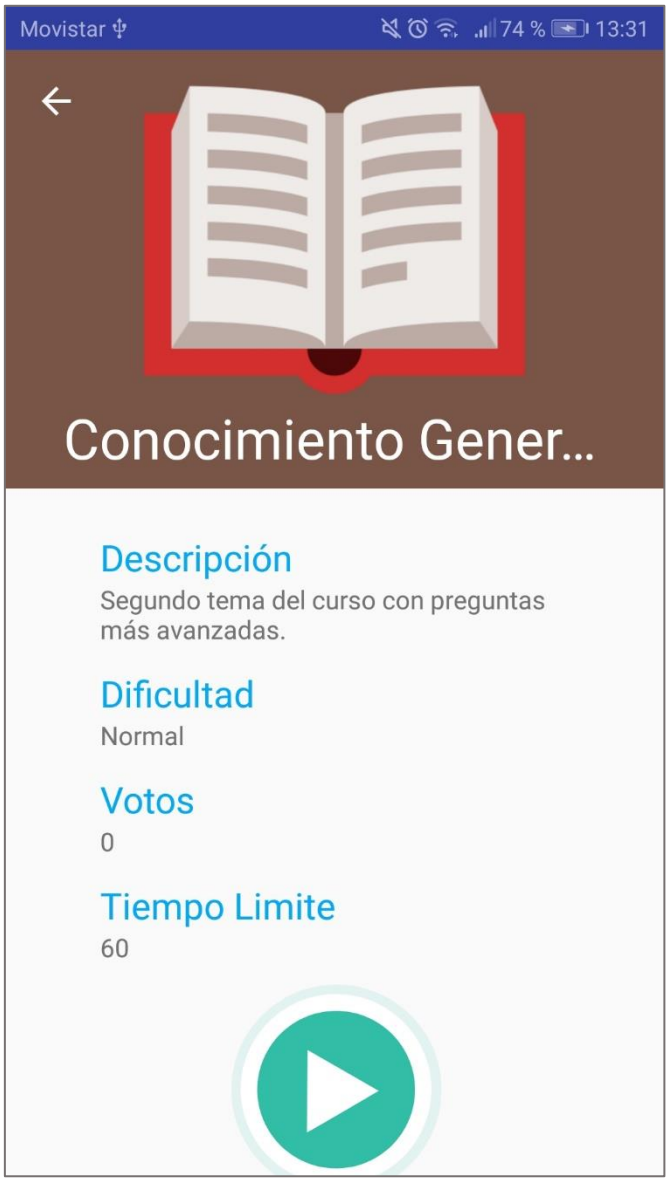

Figura 26 - Datos del juego

5.2.3 Editor de juegos y preguntas

En esta segunda fase del desarrollo de la aplicación para móviles se implementó una herramienta para crear juegos personalizados dentro de la aplicación disponible para el perfil de usuario de los profesores de una clase. Esta herramienta solo será accesible para profesores que hayan creado una clase propia, impidiendo así la modificación de la clase por defecto. La implementación de esta parte tiene en cuenta que el acceso usual de los profesores suele ser en tablets, por lo cual se han desarrollado dos layouts para cada vista, uno para móviles y otro para tablets grandes, aportando en este último un diseño más amplio y cómodo. En las figuras 27 y 28 se va a ver la pantalla de creación de juegos, en la que el profesor debe introducir el nombre del juego, su descripción, su tiempo límite y su dificultad, marcando el tamaño máximo de los campos de texto. El tiempo límite no puede ser inferior a 15 segundos para evitar juegos imposibles de resolver por falta de tiempo, de lo cual se notificará al profesor para que lo cambie y mostrando el error en pantalla. El área de las preguntas es inaccesible en este momento, y se notificará de ello en el layout para tablets.

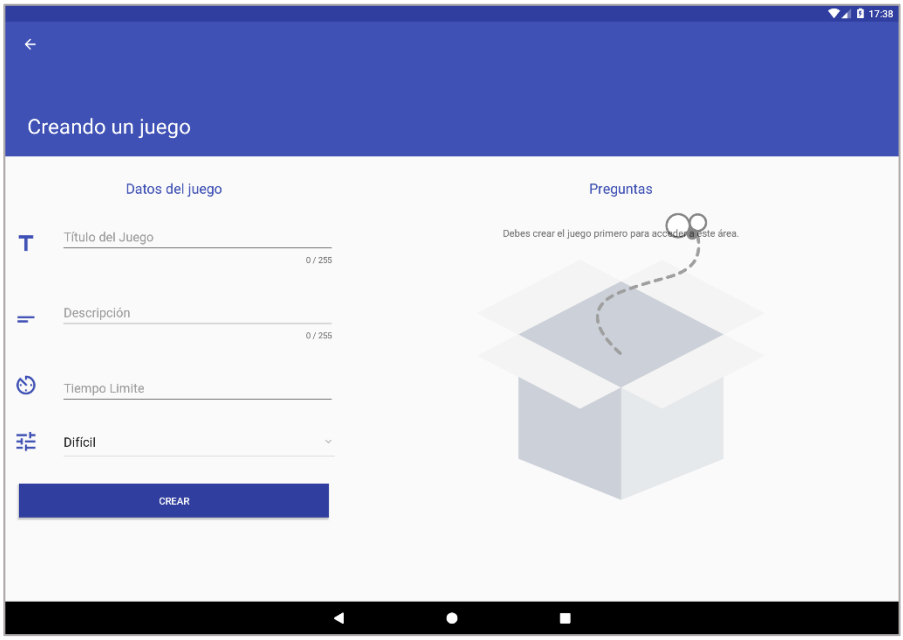

Figura 27 - Crear juego (Tablets)

Figura 28 - Crear juego (Móviles)

La pantalla de edición de juego cuenta con más campos accesibles, entre ellos el de la visibilidad y la sección de preguntas con un desplegable y un botón de añadir. La visibilidad marca si el juego en cuestión será o no accesible por los alumnos, y al añadir preguntas y al crear el propio juego se mantiene desactivado, solo permite su activación cuando todas las preguntas están correctamente editadas y que su número sea mayor a cero. Por otro lado, el desplegable cuenta con 10 opciones, una por cada tipo de pregunta en la plataforma, y el botón de añadir crea directamente una pregunta vacía del tipo seleccionado en el desplegable. Las preguntas vacías mostrarán un mensaje en rojo informando de la necesidad de editarlas, mientras que las editadas correctamente mostrarán un mensaje en verde.

En la Figura 29 se muestra la sección de las preguntas en móviles y en la Figura 30 la interfaz completa, ya que al ser una Tablet cabe por completo y permite una edición mucho más cómoda. En las figuras 31 y 33 la sección de preguntas ya cuenta con datos, de entre los cuales cada pregunta tiene un color según su tipo, haciendo de la interfaz en conjunto algo bastante vistoso. Por último, en la Figura 32 se muestra el desplegable abierto.

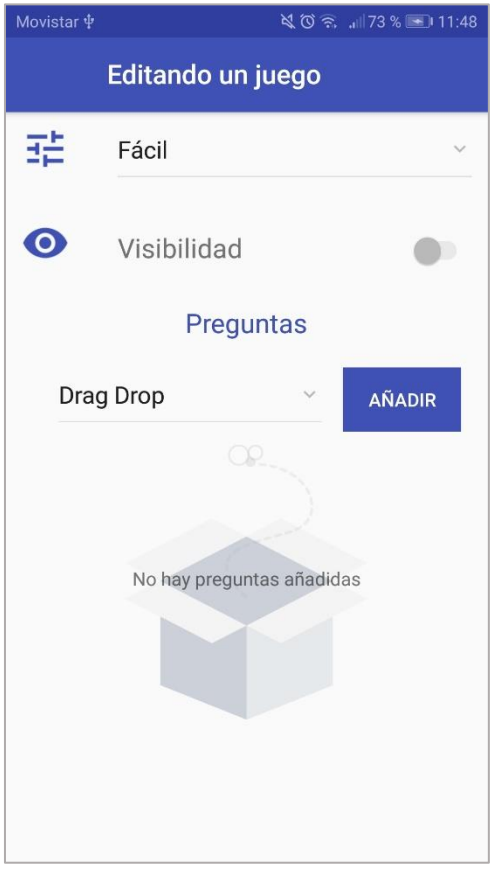

Figura 29 - Sección preguntas vacía (Moviles)

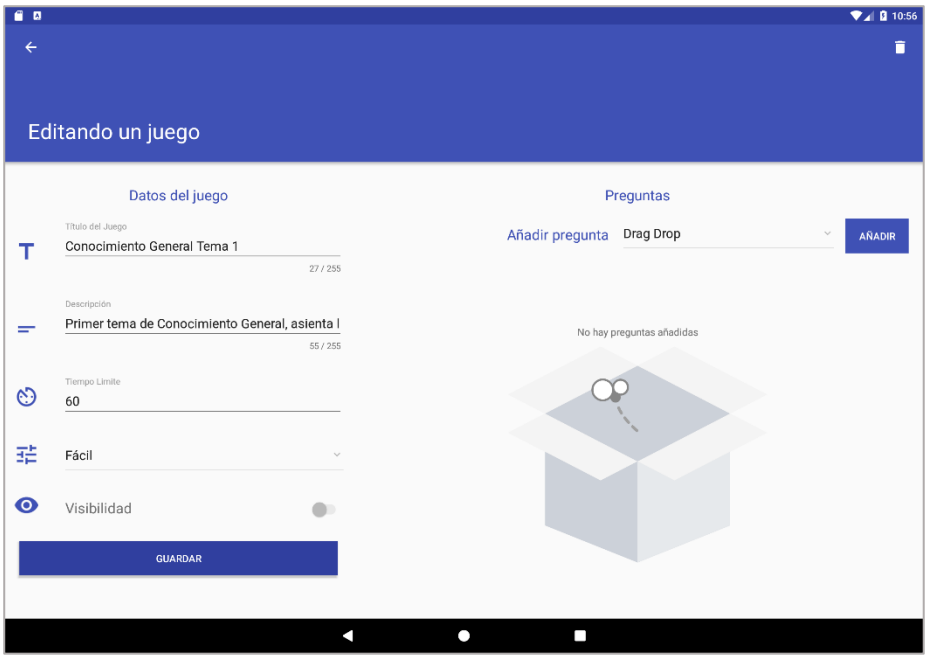

Figura 30 - Sección preguntas vacía (Tablets)

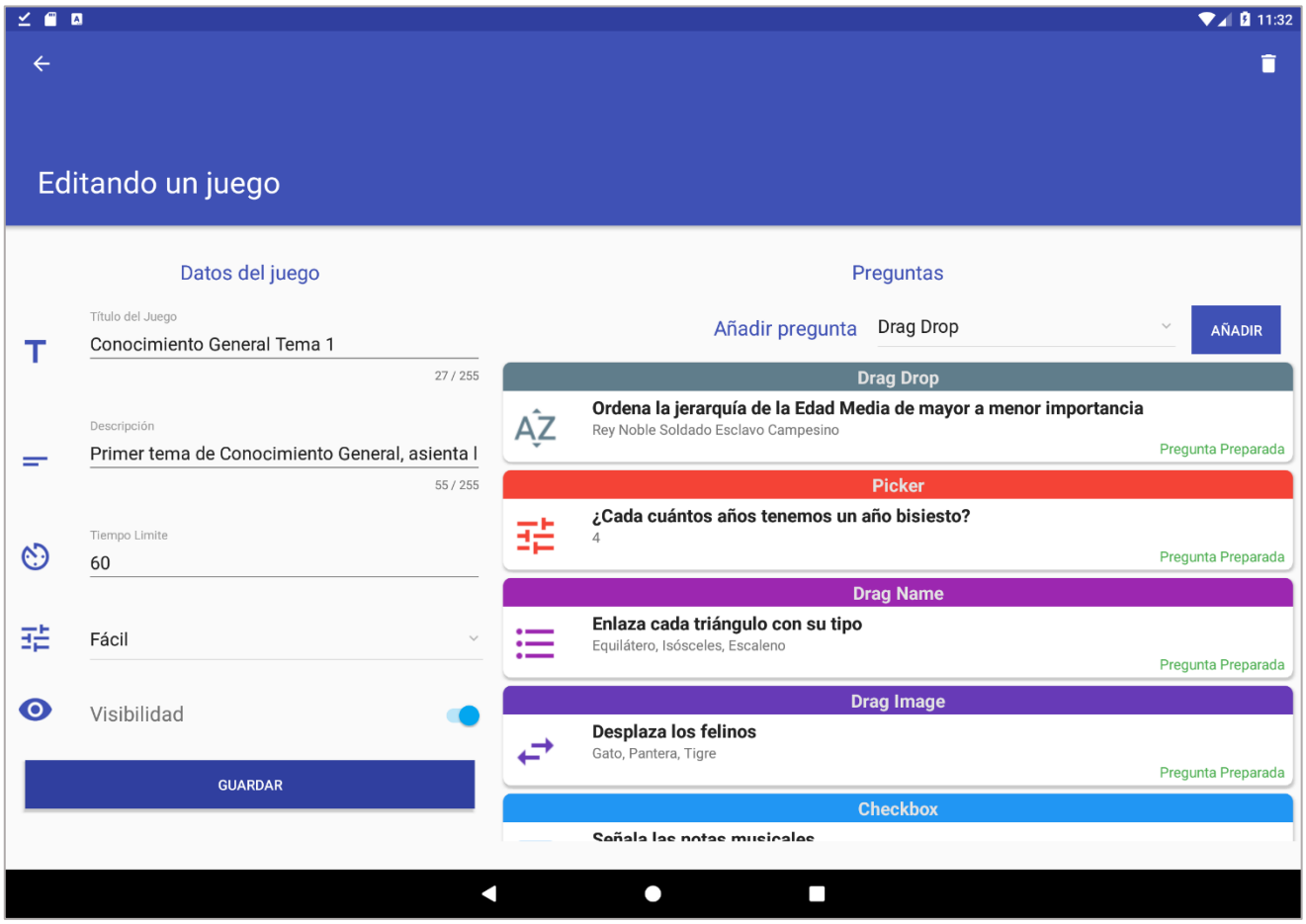

Figura 31 - Preguntas editadas correctamente

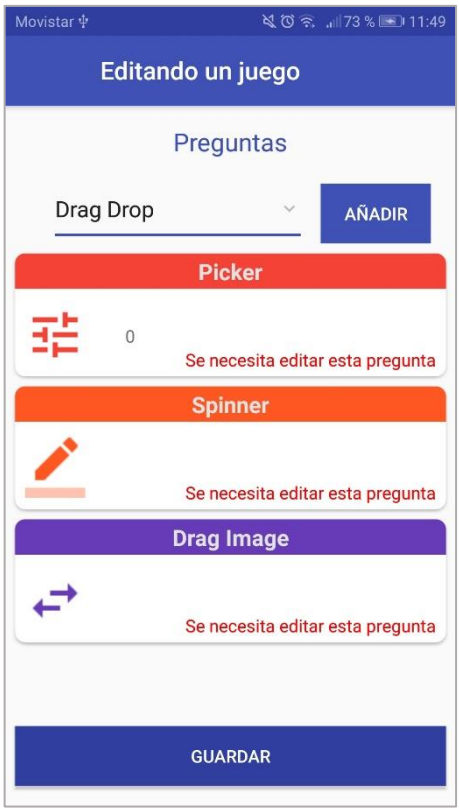

Figura 33 - Preguntas sin editar

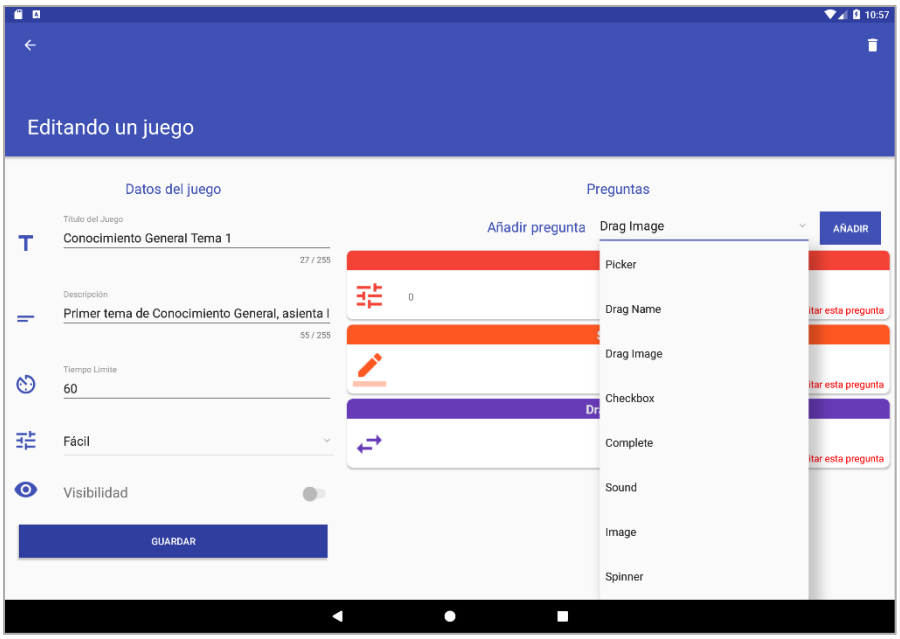

Figura 32 - Desplegable abierto

Una vez creado el juego la sección de preguntas está disponible, y eso permite añadir preguntas según el tipo seleccionado en el desplegable pulsando el botón de añadir. Lo siguiente es pulsar en el recuadro de una pregunta ya creada, acción que te llevará al editor de pregunta correspondiente al tipo de la seleccionada. Por lo tanto, hay 10 editores de preguntas distintos, y cada uno de ellos tiene interfaces tanto para móviles como para tablets grandes. Para aprovechar el diseño alargado de las tablets, se han agregado en cada editor cuadros de ayuda para aclarar la funcionalidad de los formularios. Cabe destacar que todos los textos de la aplicación están tanto en español como en inglés, adaptándose al lenguaje del dispositivo.

Los editores de preguntas tipo DragDrop, Picker, Checkbox y Spinner cuentan con un botón intermedio que comprueban la entrada y hacen al profesor seleccionar las respuestas correctas como lo haría un alumno para actualizar los datos de la base de datos.

Figura 34 - Editor DragDrop Móvil

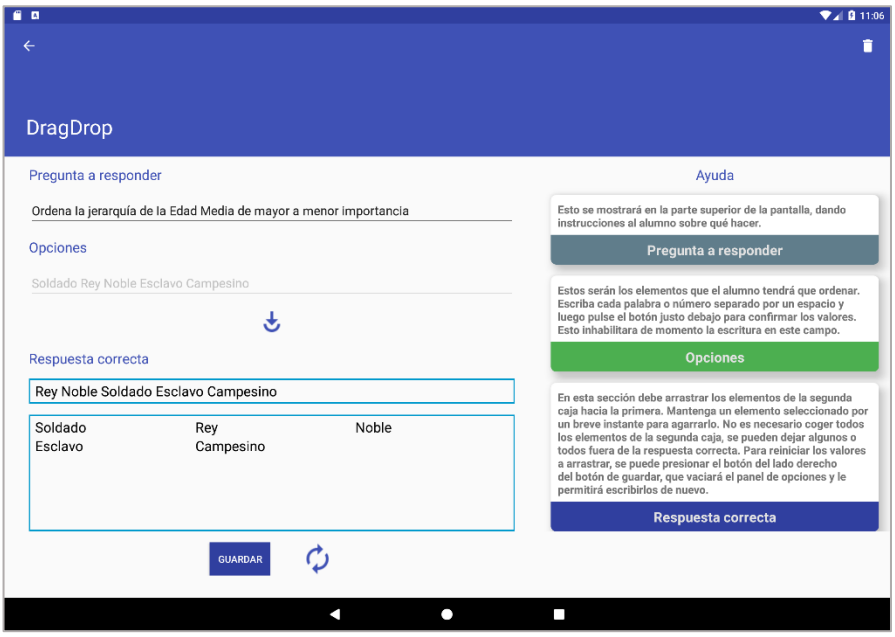

Figura 35 - Editor DragDrop Tablet

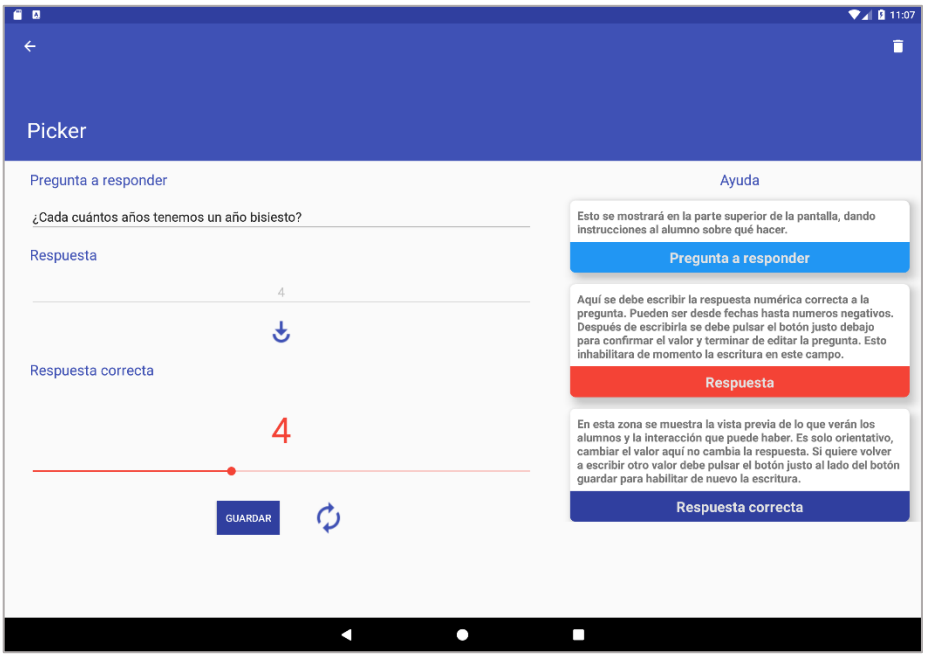

Figura 36 - Editor Picker Tablet

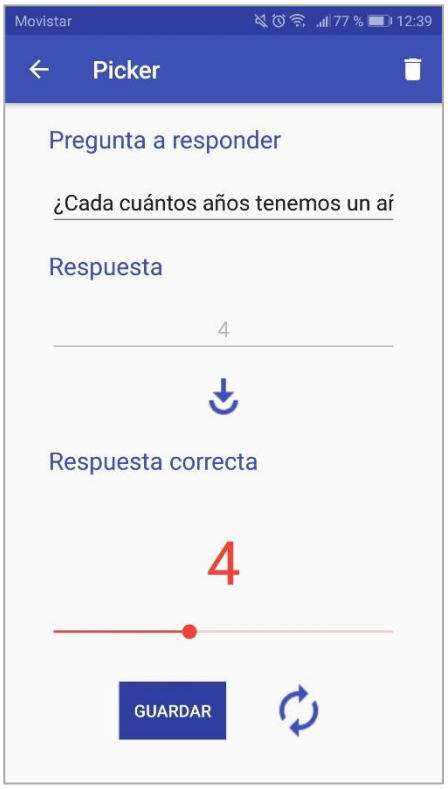

Figura 37- Editor Picker Móvil

Figura 38 - Editor Checkbox Móvil

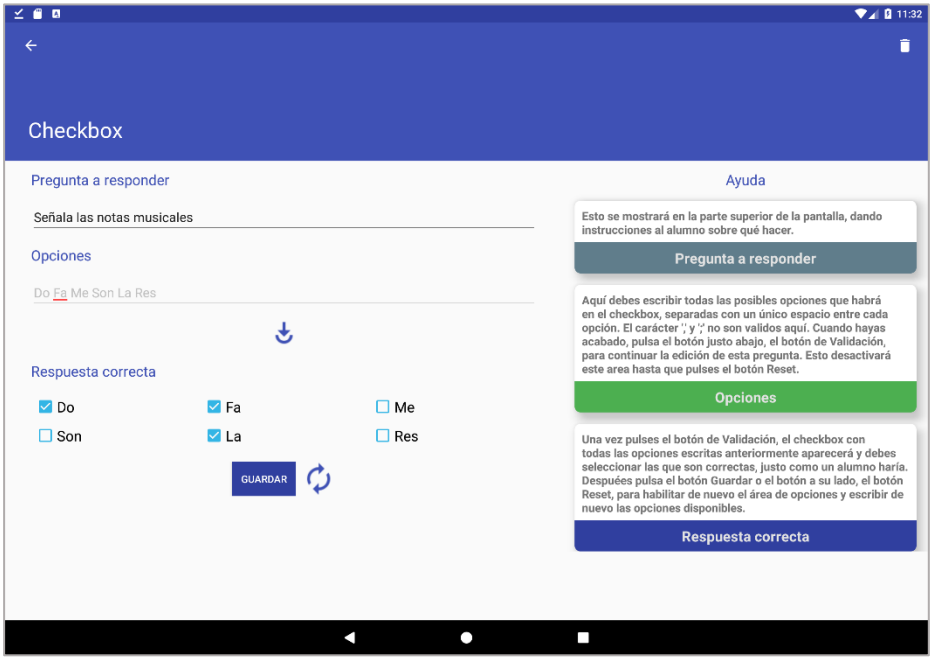

Figura 39 - Editor Checkbox Tablet

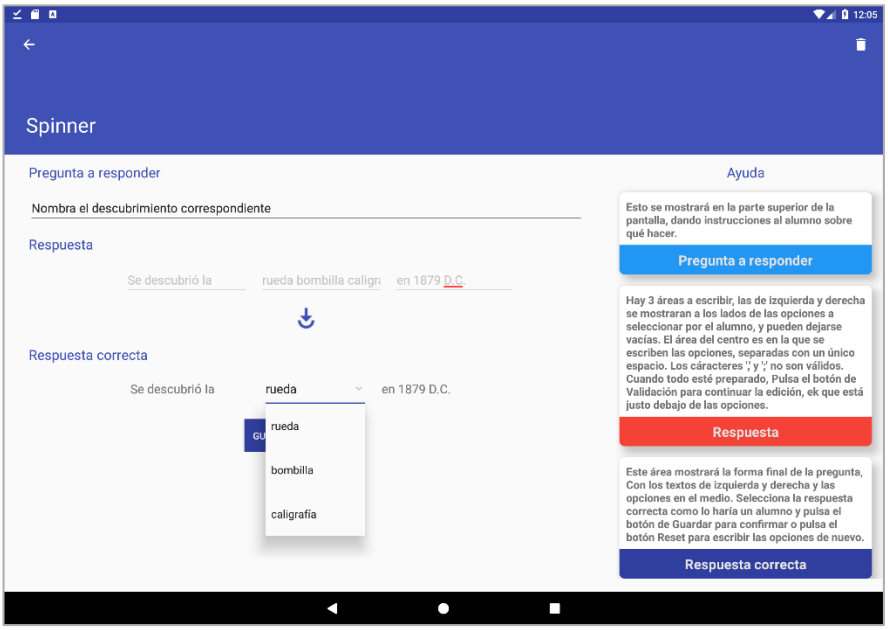

Figura 41 - Editor Spinner Tablet

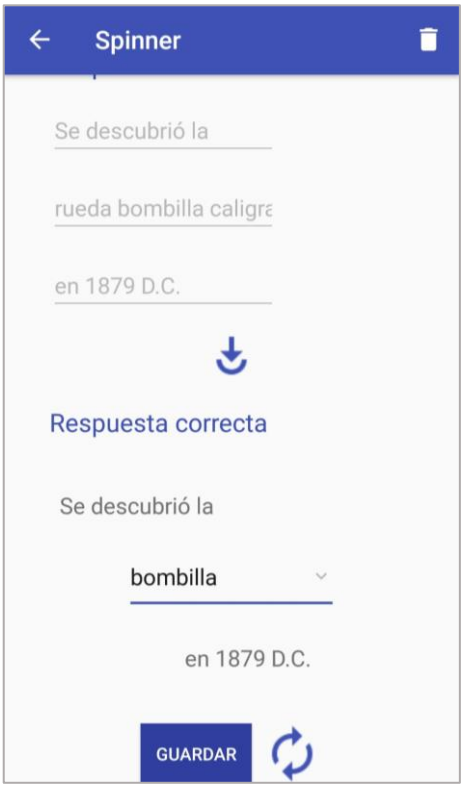

Figura 40 - Editor Spinner Móvil

Los siguientes editores, entre los cuales incluimos los de tipo DragName, DragImage, Complete, Sound e Image acceden al servicio de subida de archivos del servidor principal, y además piden los permisos necesarios sobre el dispositivo para abrir archivos como para grabar audio (Figura 44). Para ello utilizan la librería externa EasyPermissions, la cual se encarga de gestionar de forma muy simple y rápida las peticiones de permisos en aplicaciones Android (Figura 42). El archivo se envía al servidor con una petición Multipart como se muestra en la Figura 43.

```
@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    EasyPermissions.onRequestPermissionsResult(requestCode, permissions, grantResults, object: CompleteEditorActivity.this);
}
```

Figura 42 - Código Acceso EasyPermissions

```
RequestBody mFile = RequestBody.create(MediaType.parse("image/*"), file);
MultipartBody.Part fileToUpload = MultipartBody.Part.createFormData( name: "file", file.getName(), mFile);
RequestBody filename = RequestBody.create(MediaType.parse("text/plain"), file.getName());
RestInterface restInterface = RetrofitClient.getInstance();
Call<UploadObject> fileUpload = restInterface.postImage(fileToUpload, filename);
fileUpload.enqueue(new Callback<UploadObject>() {...});
```

Figura 43 - Código Petición Multipart

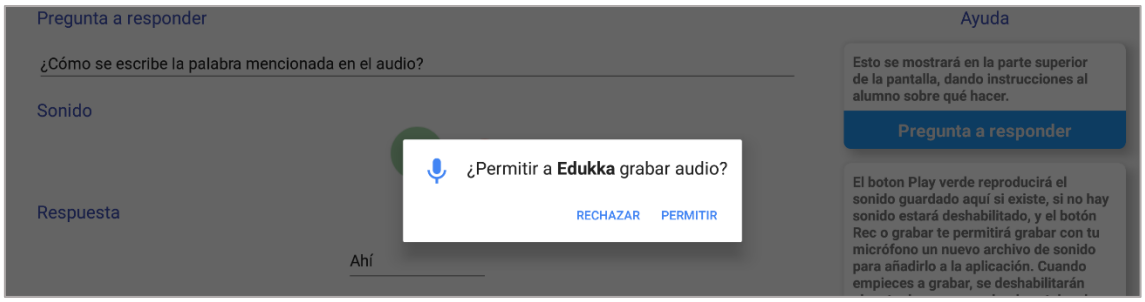

Figura 44 - Pedir Permisos en la App

Para realizar la búsqueda de imágenes después de pedir permisos, la aplicación pregunta por la aplicación que quiere utilizar el usuario para buscar la imagen en cuestión a subir en el servidor. Un ejemplo de ello es el de la Figura 45, y para que esta petición ocurra y la búsqueda esté limitada a imágenes para evitar errores se utilizan las instrucciones que aparecen en Figura 46.

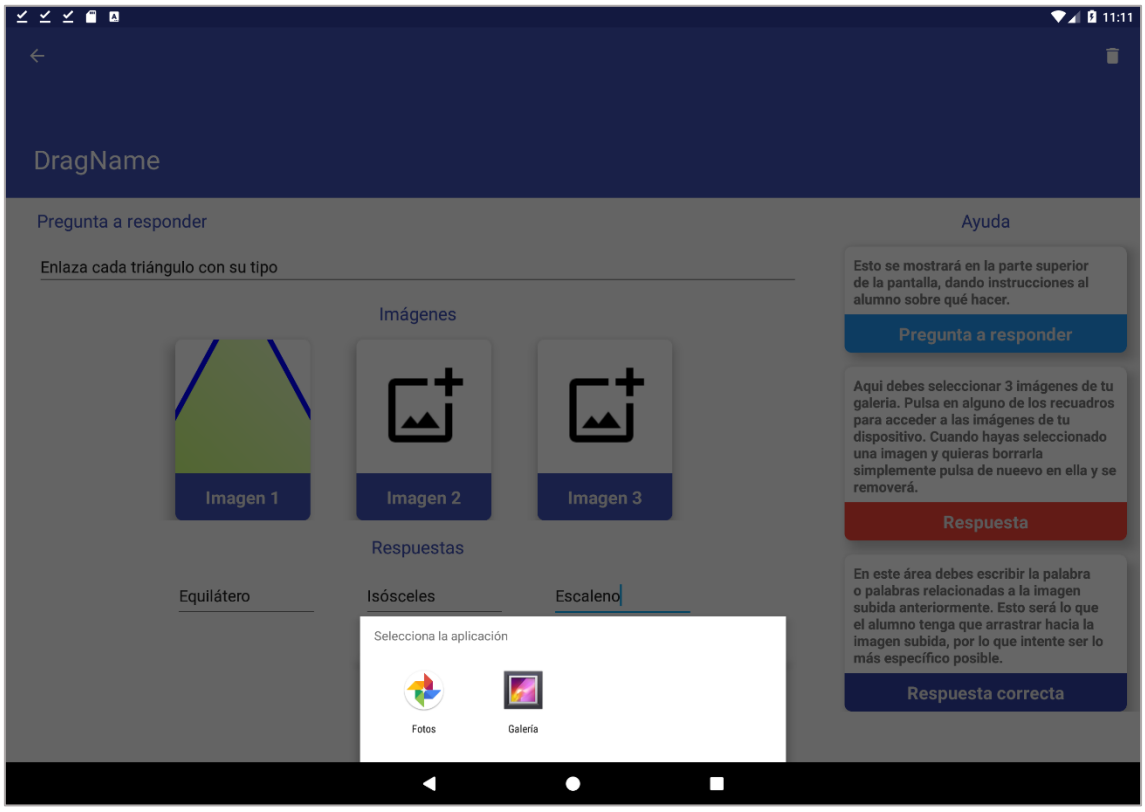

Figura 45 - Seleccionar Aplicación

```
Intent intent = new Intent(Intent.ACTION_PICK, MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
intent.setType("image/*");
startActivityForResult(Intent.createChooser(intent, "Select the application"), SEARCH_IMAGE);
```

Figura 46 - Código para buscar imagen

Para cambiar una imagen subida simplemente hay que pulsar en dicha imagen, haciendo que el recuadro se vacíe y permitiéndote subir otra imagen. Si no hay imágenes subidas en todos los recuadros la aplicación no te permitirá actualizar la base de datos.

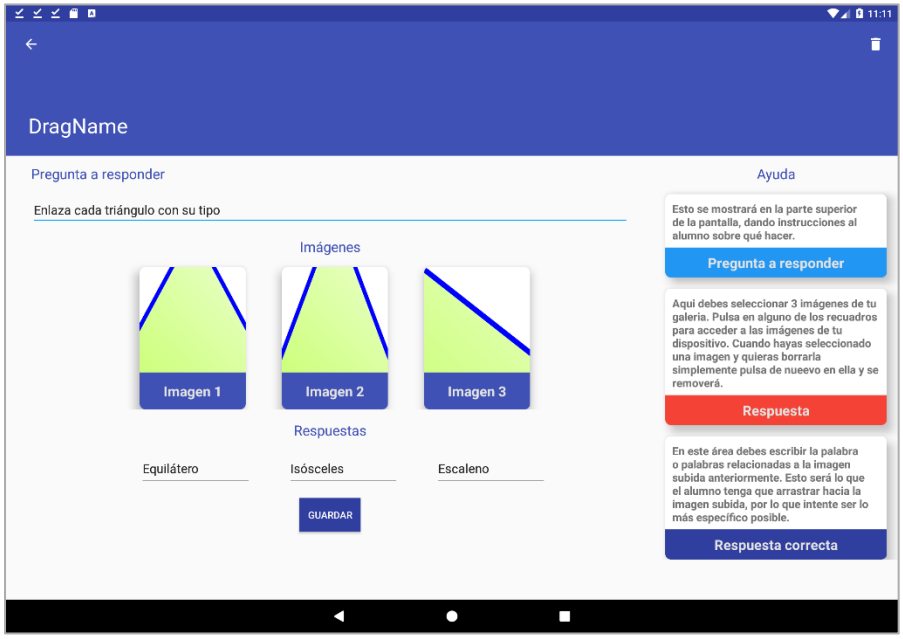

Figura 47 - Editor DragName

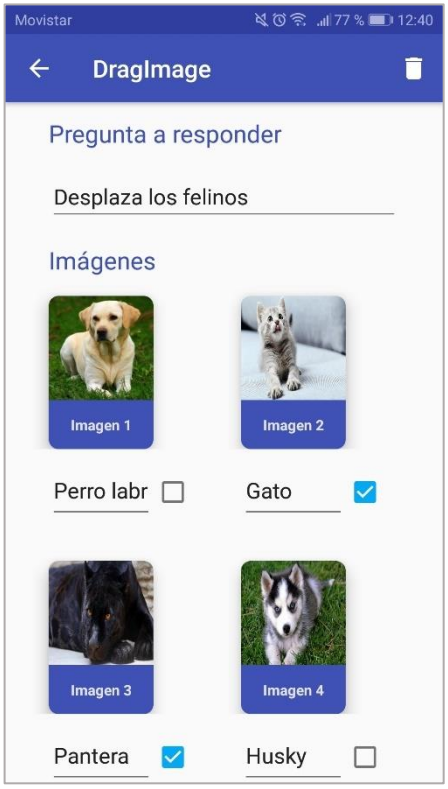

Figura 48 - Editor DragImage

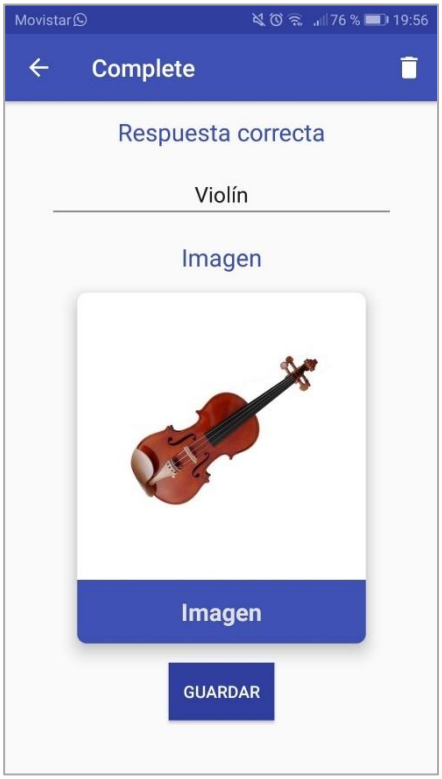

Figura 49 - Editor Complete

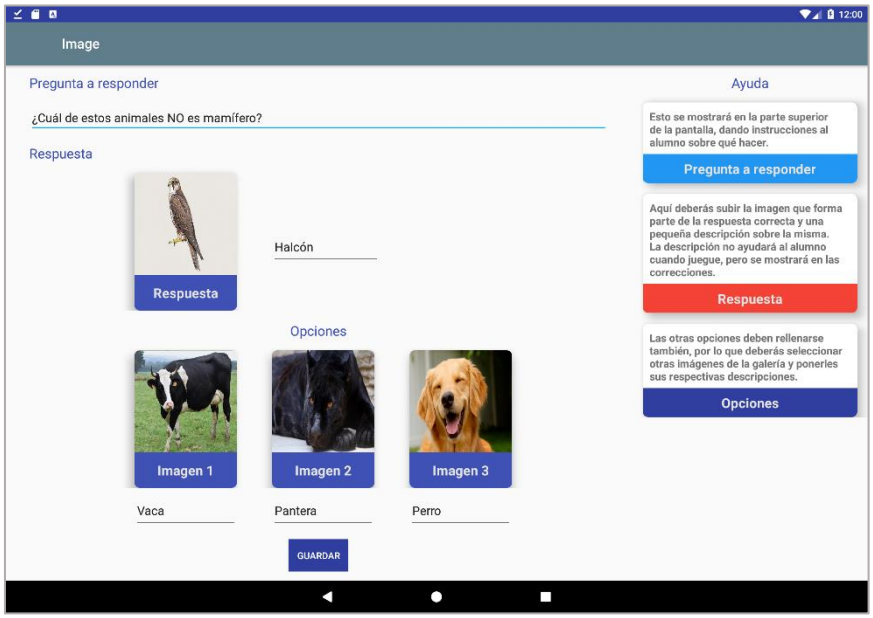

Figura 50 - Editor Image

El editor de las preguntas tipo Sound utiliza la grabadora del dispositivo Android para recordar nuestra voz o cualquier sonido al pulsar el botón de grabar hasta que pulsemos el botón de parar que lo sustituirá inmediatamente. Ese archivo lo podremos reproducir con el botón verde de Play para confirmar si es lo que deseamos.

Si no hay ninguna grabación o los campos son incorrectos no te dejará guardar la pregunta, al igual que con las imágenes y campos del resto de editores, si hay campos erróneos el editor no dejará actualizar los datos de la base de datos.

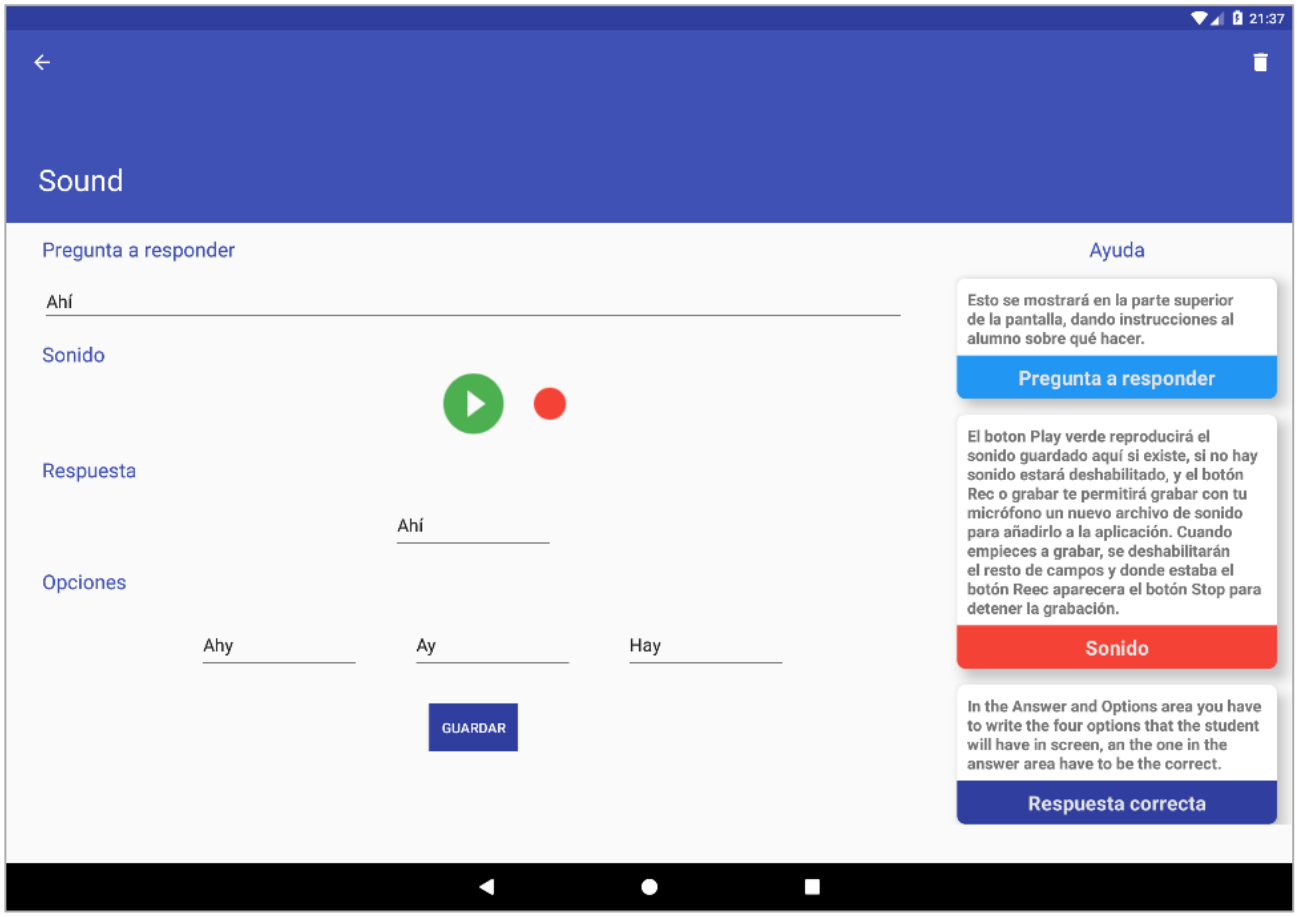

Figura 51 - Editor Sound

El último editor de la lista es el más simple de todos, que cuenta con la pregunta y un botón que cambia de verdadero a falso y viceversa cuando lo pulsas.

Figura 52 - Editor Select

Por último, destacar la opción de eliminar tanto juegos como preguntas. En el caso de los juegos, internamente se borrarán también todas las preguntas relacionadas con el mismo, mientras que eliminar una pregunta solo la eliminará a esta de la base de datos.

Al pulsar el icono de eliminar siempre se nos mostrará un diálogo que nos pida confirmación y nos informará de lo que sucederá al aceptar tal y como aparece en la Figura 53.

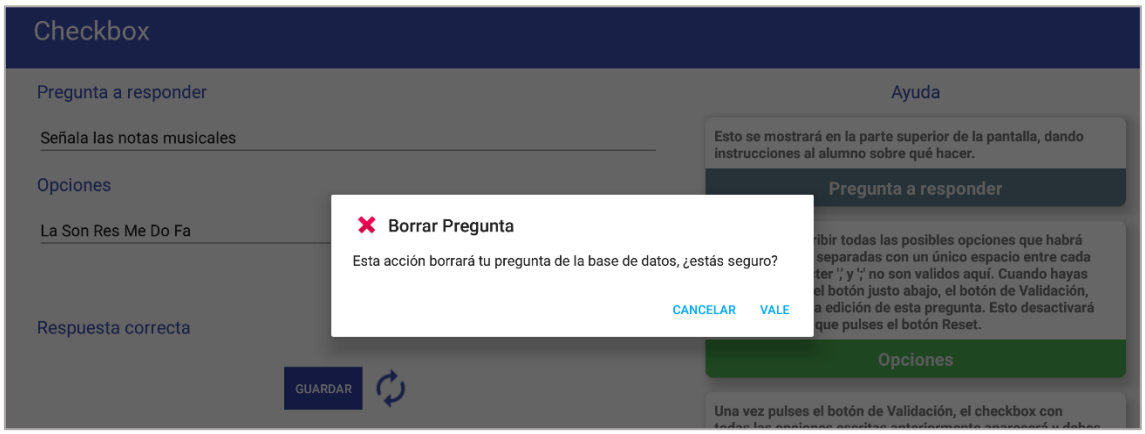

Figura 53 - Eliminar Pregunta

5.2.4 Interacción y Juego Multijugador

La última fase de desarrollo buscaba crear una forma de comunicarse entre usuarios para crear experiencias multijugador, y se realizó una investigación durante el propio desarrollo tras la cual, entre otras opciones, se eligió añadir un servidor Firebase a la plataforma para la gestión de mensajería en tiempo real. Estas características multijugador se ubican en la nueva cuarta pestaña del menú principal, diferente entre alumnos y profesores (Figuras 54 y 55) ya que presentan funcionalidades distintas.

Figura 54 - Vista del Alumno

Figura 55 - Vista del Profesor

Las funcionalidades del alumno se dividen en crear una sala multijugador y buscarla. Cuando un alumno crea una sala se guarda en la base de datos dicha sala con el identificador de comunicación de dicho usuario, y su dispositivo permanecerá a la espera de recibir notificaciones de otro usuario en la pantalla que muestra los datos de la sala, que serían tus datos y el estado de la sala, que puede estar adornada como en el caso de la Figura 56 por el icono de estrella a la derecha, cuya utilidad veremos más adelante.

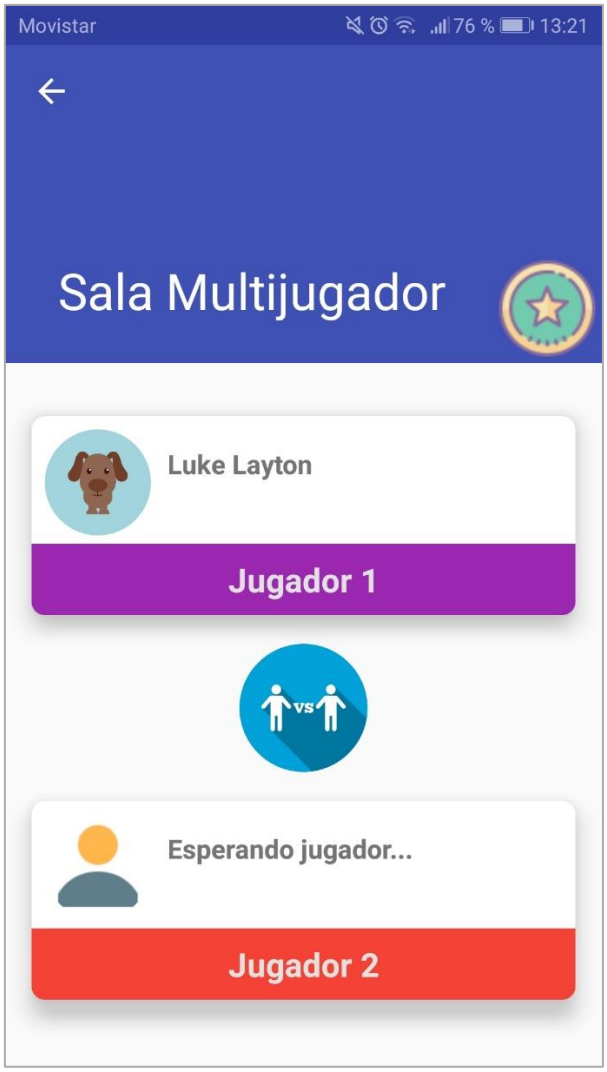

Figura 56 - Sala de Espera Libre

La otra acción que pueden realizar los alumnos es buscar una sala multijugador, lo cual los lleva a una lista de salas multijugador disponibles en ese momento en su clase (Figura 57). Al seleccionar una sala, se realiza una comunicación entre el creador y el buscador, ya que el buscador envía internamente un mensaje al creador con su nombre, su icono de perfil y su identificador de comunicación. Si la sala sigue libre en el momento en que el creador recibe el mensaje, responde con una confirmación, y el buscador finalmente entra en la sala. Si por el contrario la sala está llena al momento de recibir el mensaje, el creador notificará al buscador de que la sala está llena, que recibirá un aviso informándolo del estado de la sala. Cuando otro alumno se une a una sala, el creador actualiza la vista con los datos recibidos por su parte, es decir, su imagen e icono, y aparece en pantalla el botón para iniciar el juego (Figura 58).

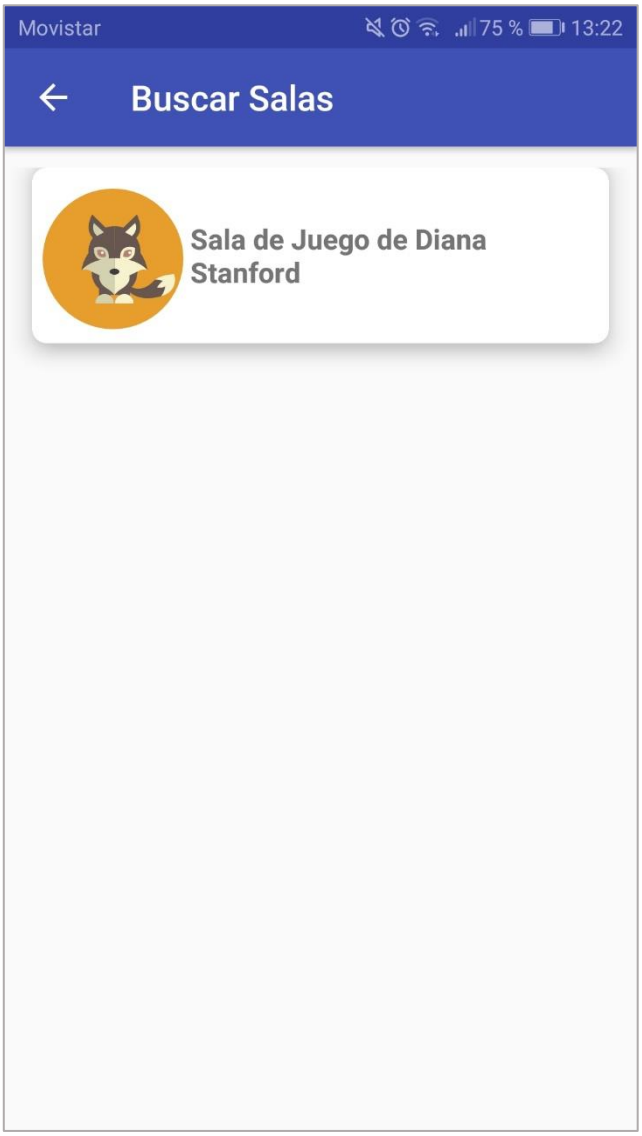

Figura 57 - Buscar Salas

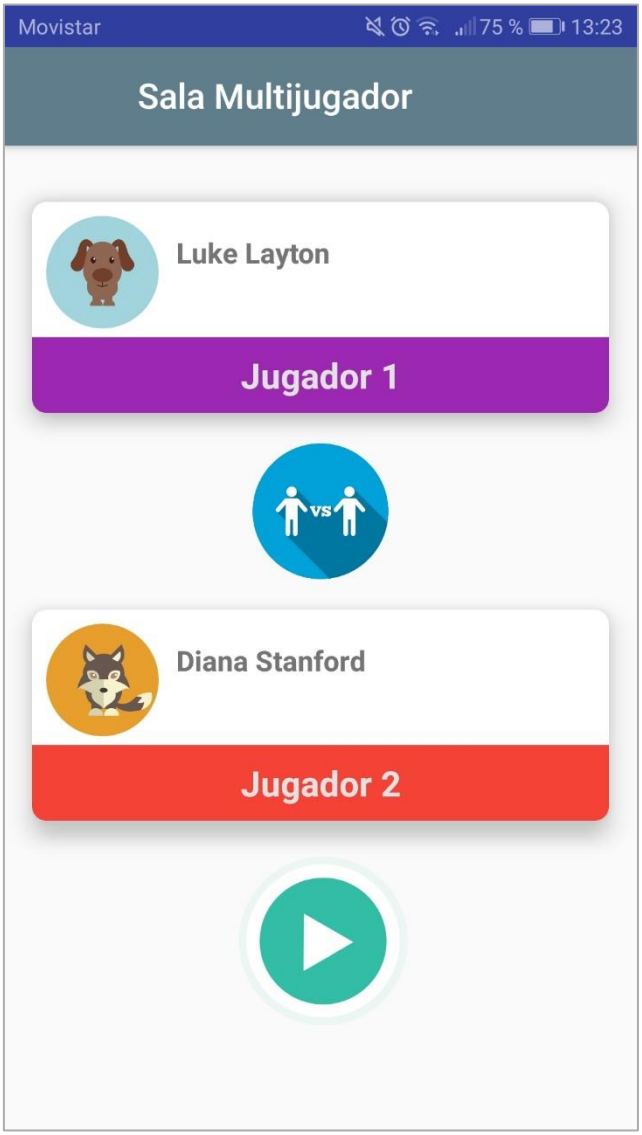

Figura 58 - Sala Llena

El juego multijugador se basa en un test de 5 preguntas en el que solo se puede avanzar al acertar preguntas. En la vista se mantienen las puntuaciones de ambos jugadores (Figura 60) y los aciertos suman puntos mientras que los fallos los restan, pudiendo llegar a puntuaciones negativas si no se va con cuidado. Durante el transcurso del juego ambos usuarios conocen el identificador de comunicación del otro y están enviándose información en cada evento, siendo estos eventos el fallar una pregunta, acertarla o desconectarse. Al fallar se actualizan las puntuaciones en ambos dispositivos, al acertar avanzan ambos jugadores a la siguiente pregunta hasta que no queden más y se finalice el juego y cuando un usuario se desconecta también se termina el juego dándole puntuación de compensación al jugador que permanece en la sala. Al terminar el juego se mostrarán los resultados obtenidos, variando la parte superior entre rojo y verde según se haya ganado o perdido el juego.

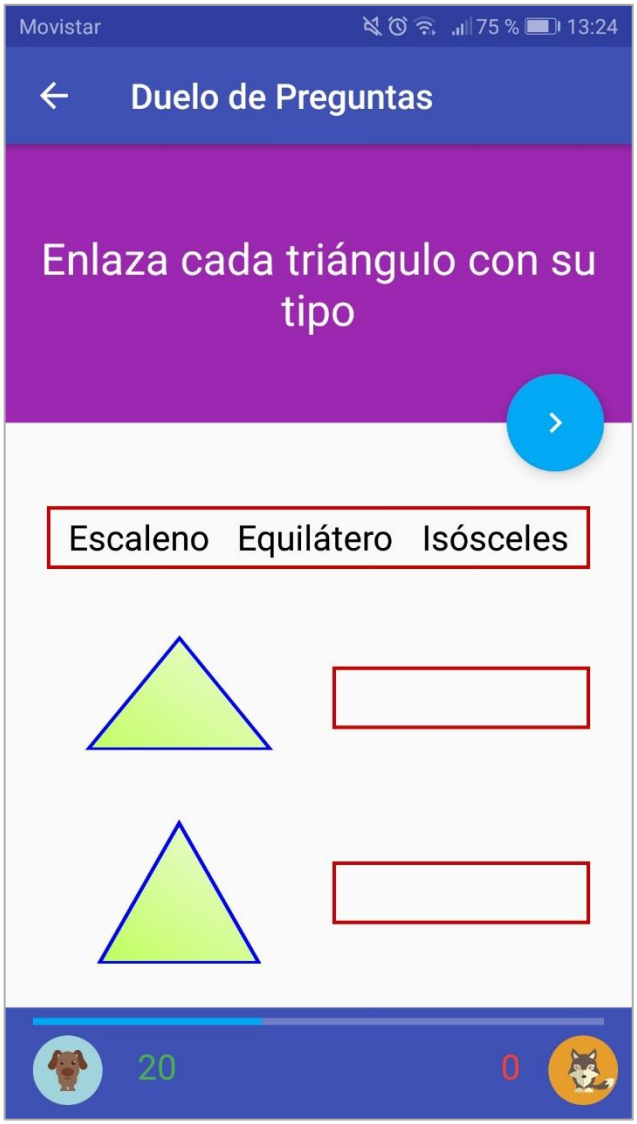

Figura 60 - Juego Multijugador

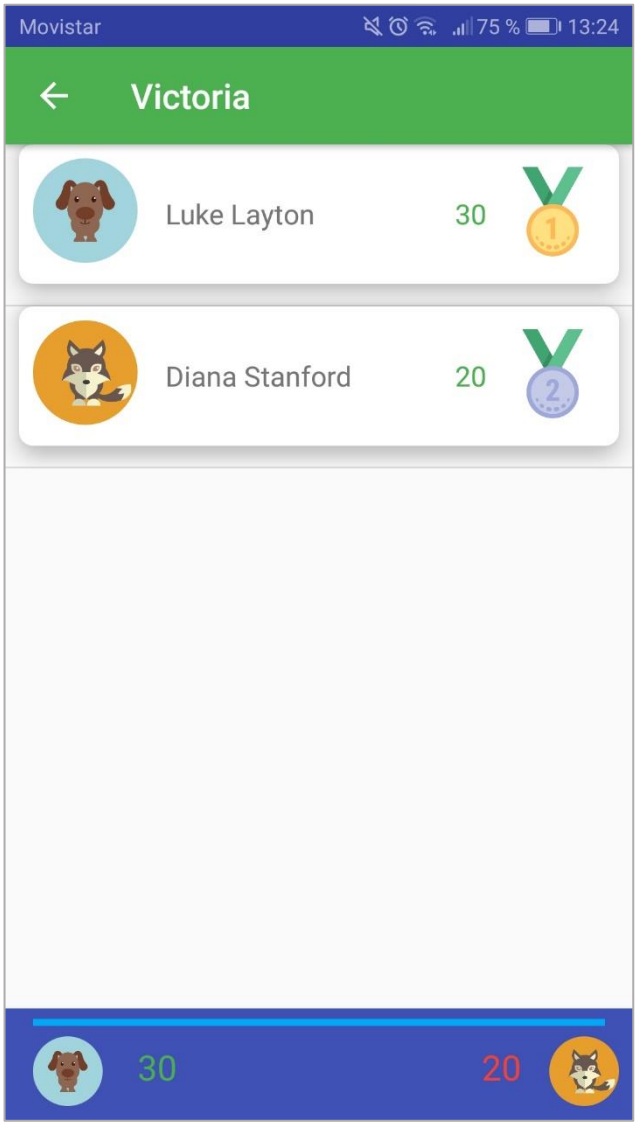

Figura 59 - Resultados del Juego

Un usuario se puede desconectar tanto de las salas como del juego en curso, y en ambos casos se le pedirá confirmación en un diálogo (Figura 61). En caso afirmativo, se notificará al otro usuario del abandono y su aplicación realizará los procesos requeridos. Cuando el creador de una sala se desconecta, la sala se borra de la base de datos y expulsa al otro alumno si es que se había unido, mientras que si el que se desconecta es un usuario unido a posteriori simplemente se pasará la sala al estado libre otra vez, esperando la unión de otro usuario.

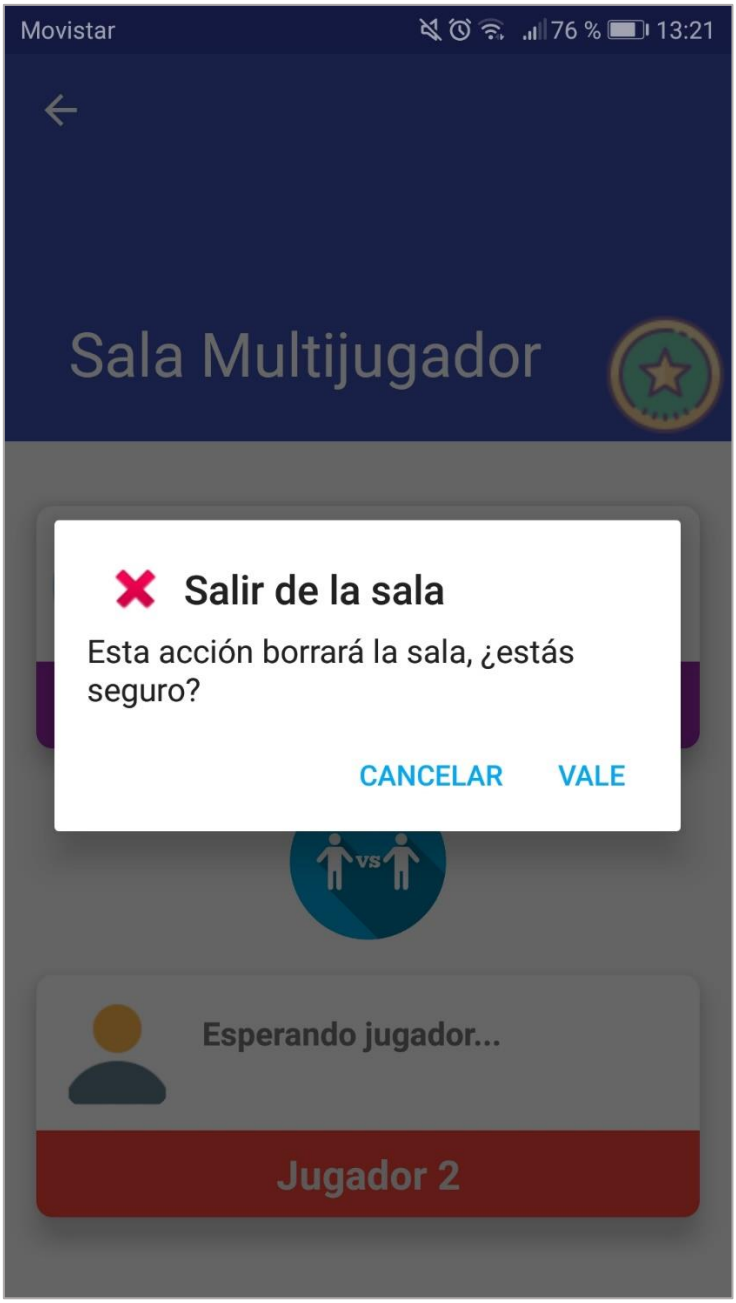

Figura 61 - Desconexión

Los profesores pueden realizar dos funciones, que son adornar una sala y modificar la puntuación de los alumnos de su clase.

Adornar una sala hace que durante el juego multijugador las puntuaciones se doblen, haciendo que sea mucho más interesante unirse a una sala y aumentando el trafico de juego, evitando que algunos alumnos no encuentren a nadie con quien jugar. Cuando una sala está adornada aparece un icono de estrella en ella como indicativo (Figura 62) y se le indica al profesor la acción realizada. Los usuarios que estén en esa sala también serán notificados y les aparecerá una estrella como en la Figura 56.

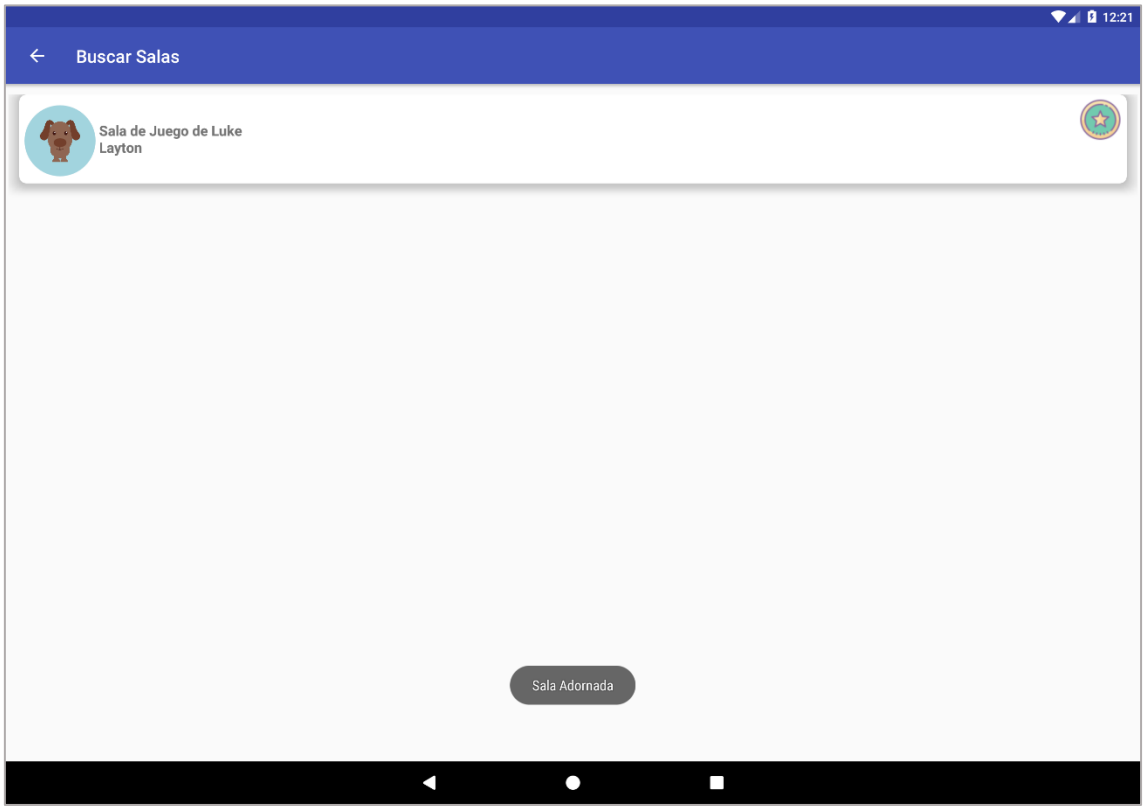

Figura 62 - Adornar Sala

La última funcionalidad de los profesores es modificar la puntuación de los alumnos, principalmente para evitar que se creen desequilibrios y que los primeros puestos de la clase tengan una diferencia de puntos demasiado marcada con el resto, haciendo que conseguir estar en el podio sea algo accesible para todos los alumnos y los motive a seguir.

La pantalla de modificación de puntos muestra los usuarios de la clase y sus puntuaciones, y al pulsar uno de ellos que no sea el profesor se abrirá un dialogo que permitirá sumar o restar puntos al alumno seleccionado (Figura 63).

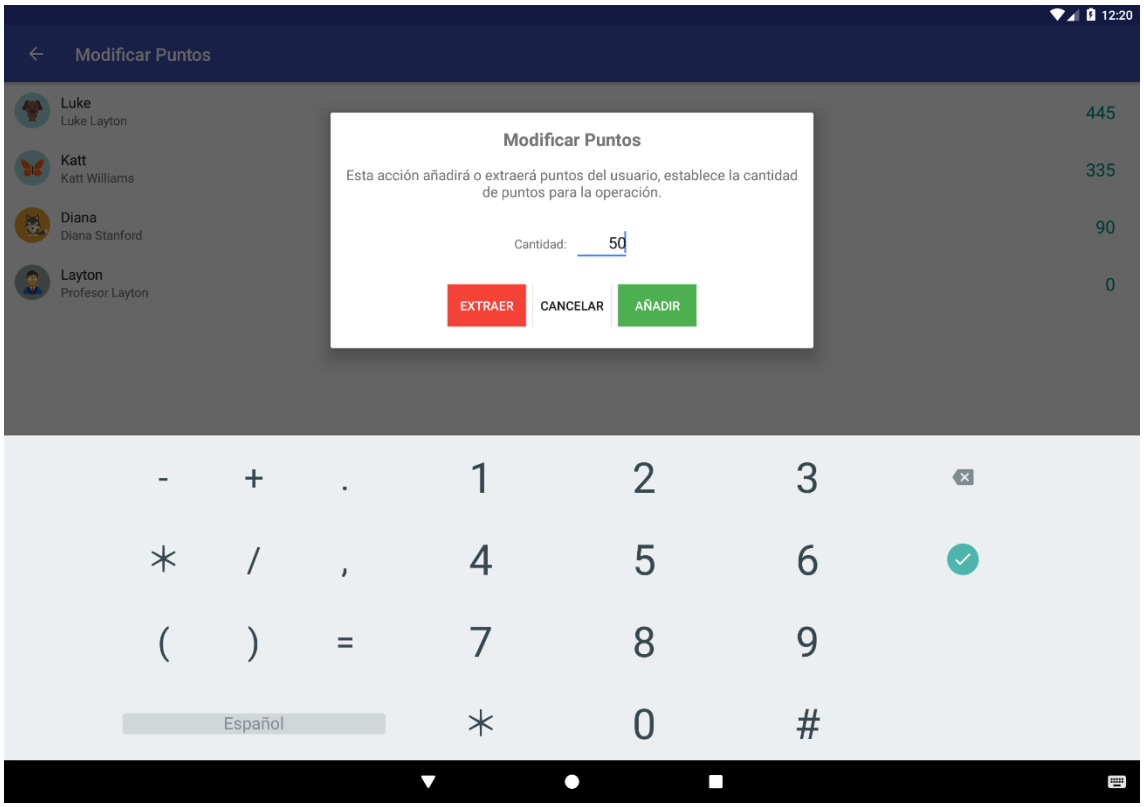

Figura 63 - Modificar Puntos

Capítulo 6. Conclusiones y líneas futuras

Con el desarrollo de esta versión de la plataforma hemos conseguido darle uno de los objetivos principales en los que se pensó durante su creación: darles a los niños de Primaria una plataforma con contenido dedicado para ellos, y con la introducción del editor de juegos para profesores pueden tener una cantidad ilimitada y personalizada. También se han dedicado esfuerzos en hacer la plataforma más divertida para asegurar la constancia de los niños en ella, con modificaciones como el temporizador, la variación en las puntuaciones y los juegos multijugador.

La realización de este TFG me ha dado la oportunidad de aprender a programar aplicaciones Android desde una aplicación funcional y eficiente dándome un montón de pautas viables a seguir en muchas situaciones, y hay que añadir que también he podido aprender a programar un servidor basado en PHP. Además, he descubierto Firebase, una plataforma potente y gratuita para desplegar aplicaciones dotándolas de funciones muy útiles. Todas estas herramientas me han parecido interesantes y es probable que se encuentren en mi carrera profesional por su comodidad y posibilidades prácticas.

Por último, me gustaría exponer las posibles líneas futuras que personalmente veo interesantes para este proyecto:

- **Añadir más tipos de preguntas a la plataforma:** El tipo de preguntas existente no tiene por qué limitarse a 10, siendo una posibilidad muy interesante aumentar su número para darle mucha más versatilidad a la aplicación, tanto para los alumnos al jugar como para los profesores al crear.
- **Crear más modalidades de juego multijugador:** Actualmente solo existe un modo multijugador, el Duelo de preguntas en el que se recogen 5 preguntas aleatorias de la clase y se intentan resolver antes que el otro alumno. Sería muy interesante añadir más modalidades a la ecuación, ya sean competitivas o incluso cooperativas.

- **Realizar el despliegue completo en Firebase:** Uno de los descubrimientos que menos esperaba en este proyecto era encontrar una plataforma gratuita como esta. El problema al desplegar aplicaciones en ella es que precisa de que la base de datos sea no relacional, siendo incompatible con el proyecto actual sin realizar muchos cambios. Por ello desplegar Edukka en Firebase puede ser una línea interesante por la cantidad de beneficios que ofrece.
- **Crear un servicio de chat entre alumnos y profesores:** En esta versión se ha añadido mensajería en tiempo real a través de notificaciones para manejar eventos, pero esto no es lo único que puede realizarse con esto. Se puede crear un chat dentro de la aplicación para alumnos entre ellos y con el profesor para preguntar dudas, realizar peticiones, organizar juegos…
- **Añadir notificaciones a la aplicación:** Las notificaciones se manejan en forma de mensajes sin activar sonidos del dispositivo ni ningún tipo de alerta fuera de la aplicación, pero puede ser muy práctico notificar cuando un profesor cambia un juego a visible por poner un ejemplo.

Bibliografía

Ian F. Darwin, *Android Cookbook: Problems and Solutions for Android Developers*, 1 mayo 2017

Rick Boyer, Kyle Mew, *Android Application Development Cookbook*, 31 marzo 2017

Avanzo, ¿Qué es el e-Learning?, 24 febrero 2017: *https://www.avanzo.com/que-es-el-elearning/*

Nuria Vallejo, Como diseñar e-Learning, 20 febrero 2017: *https://ojulearning.es/2017/02/disenar-elearning-para-ninos-y-adolescentes/*

Slim, Documentación Slim: https://www.slimframework.com/

Android Developers, Documentación Android: *https://developer.android.com/*

Material Design, Documentación Material Design: *https://material.io/design/*

Google Firebase, Documentación Firebase: *https://firebase.google.com/docs/cloud-messaging/*

Heroku, Sitio Oficial Heroku: *https://www.heroku.com/*

Android Developers, Información MediaRecorder: *https://developer.android.com/guide/topics/media/mediarecorder*

GitHub, Repositorio RingProgressBar: *https://github.com/HotBitmapGG/RingProgressBar*

GitHub, Repositorio EasyPerrmissions: *https://github.com/googlesamples/easypermissions*

Android Example, Subir Archivo a Servidor PHP: *https://androidexample.com/Upload_File_To_Server_-_Android_Example/index.php?view=article_discription&aid=83&aaid=106*

Ravi Tamada, Notificaciones Push Android, 12 julio 2017: *https://www.androidhive.info/2012/10/android-push-notifications-using-google-cloud-messaging-gcm-php-and-mysql/*