

UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR



Grado en Ingeniería Informática

## TRABAJO FIN DE GRADO

SISTEMA DE RESPUESTA INMEDIATA CON CLICKERS  
PARA USO EDUCATIVO: APLICACIÓN WEB

Óscar Arroyo Nogales  
Tutora: Estrella Pulido Cañabate  
Junio 2018



# SISTEMA DE RESPUESTA INMEDIATA CON CLICKERS PARA USO EDUCATIVO: APLICACIÓN WEB

AUTOR: Óscar Arroyo Nogales  
TUTORA: Estrella Pulido Cañabate

Dpto. de Ingeniería Informática  
Escuela Politécnica Superior  
Universidad Autónoma de Madrid  
Junio de 2018







## Resumen

La finalidad de este Trabajo de Fin de Grado es ampliar la funcionalidad de la plataforma web y aplicación móvil de ClicEPS, resultados de un TFG defendido el año pasado, siguiendo las mismas tecnologías que se aplicaron en el momento de su desarrollo: Node.js como entorno *Javascript* para el servidor principal, el *framework* AngularJs y la base de datos MySQL. Los cambios realizados en este trabajo han conseguido mejoras en la aplicación Android sin olvidar su función principal: que los profesores puedan elaborar preguntas en el ámbito educativo y que éstas sean contestadas por los estudiantes, aumentando la participación y haciendo más interactivas las clases presenciales.

Las mejoras en la plataforma web han estado centradas principalmente en la interfaz de administración incluyendo funcionalidades para la matriculación/anulación de matrícula del estudiante en los distintos grupos de asignaturas, su registro mediante sus datos personales, la creación de nuevas asignaturas, y otras opciones que iremos viendo a lo largo de esta memoria.

A su vez, las mejoras que se aplicarán a la plataforma web y aplicación móvil ayudarán a mejorar el feedback entre el profesor y los alumnos (mediante la participación en clase con el uso del *Smartphone*), el seguimiento del rendimiento de cada alumno en la asignatura (mediante estadísticas por pregunta o por estudiante), la participación presencial en la clase (mediante preguntas que verifican la ubicación del estudiante en la misma) y la comunicación entre los usuarios y los administradores (mediante los F.A.Q de la plataforma, la página de Twitter dedicada al soporte, las encuestas, y las medidas que avisan cuándo la aplicación móvil o la web se encuentra en mantenimiento).

Por último, y sirviendo a modo de resumen, se mejorarán las dos aplicaciones que existen actualmente en ClicEPS (web y Android), utilizando el desarrollo en cascada para su implementación. Mediante pruebas, se garantizará el cumplimiento de los requisitos funcionales y no funcionales, y a través de encuestas intermedias se corregirán y mejorarán los aspectos que se vayan desarrollando.

## Palabras clave

Android, MySQL, AngularJs, Node.js, Firebase, *framework*, *clickers*, *geofences*, Heroku, Github



## Abstract

The objective of this Final Degree Project is to increase the functionality of the web platform and mobile application of ClickEPS, results of a FDP defended last year, following the same technologies that were applied at the time of its development: Node.js as Javascript environment for the main server, the AngularJs framework and the MySQL database. The changes made to this work have achieved improvements in the Android application without forgetting its main function: that teachers can develop questions in the educational field and that these can be answered by the students, increasing the participation and making the face-to-face classes more interactive.

Improvements to the web platform have been mainly focused on the administration interface including functionalities for the enrolment/de-registration of students in the different groups of subjects, their registration with their personal data, the creation of new subjects, and other options that we will see throughout this memory.

In turn, the improvements that will be applied to the web platform and mobile application will help to improve the feedback between teacher and students (by their participation in class with the use of the Smartphone), monitoring the performance of each student in the subject (through statistics by question or by student), face-to-face participation in the class (through questions that verify the student's location in the lecture room) and communication between users and administrators (with the platform's FAQs, the Twitter page dedicated to support, surveys, and measures that notify when the mobile application or the web are in maintenance).

Lastly, as a summary, the two applications that currently exist in ClickEPS (web and Android) will be improved, by using the waterfall model for its implementation. Via tests, compliance with functional and non-functional requirements will be guaranteed, and through intermediate surveys, the aspects that are developed will be corrected and improved.

## Keywords

Android, MySQL, AngularJs, Node.js, Firebase, *framework*, *clickers*, *geofences*, Heroku, Github





## ***Agradecimientos***

*A mi tutora Estrella Pulido Cañabate, por su ayuda constante y seguimiento con el trabajo realizado.*

*A mis compañeros de la Universidad, por los consejos y experiencias que me han servido para mejorar este proyecto.*

*A mi familia, por darme fuerzas cada vez que me sentía perdido.*

*A mi pareja, por estar siempre a pesar de la distancia.*

*Óscar Arroyo Nogales*

*Junio 2018*



## INDICE DE CONTENIDOS

1	Introducción.....	1
1.1	Motivación.....	1
1.2	Objetivos.....	2
1.3	Organización de la memoria.....	3
2	Estado del arte .....	5
2.1	Quizalize.....	5
2.2	Quizziz.....	6
2.3	Socrative .....	7
2.4	Conclusiones.....	8
3	Análisis y Diseño.....	9
3.1	Análisis de Requisitos .....	9
3.1.1	Requisitos Funcionales .....	9
3.1.2	Requisitos no Funcionales .....	16
3.2	Modelo de datos .....	17
4	Desarrollo .....	19
4.1	Aplicación Web .....	19
4.1.1	Conexión.....	19
4.1.2	Seguridad .....	20
4.1.3	Interfaz de administración .....	21
4.1.4	Interfaz de profesorado .....	22
4.1.4.1	Exportar pregunta .....	22
4.1.4.2	Estadísticas y pase de lista.....	23
4.1.4.3	Actualizar la contraseña.....	24
4.1.4.4	Crear tema.....	24
4.2	Aplicación Móvil.....	24
4.2.1	Preguntas presenciales o de tarea .....	25
4.2.1.1	MapsActivity .....	25
4.2.1.2	Firebase.....	25
4.2.1.3	Mock Locations .....	26
4.2.2	ClickEPSAdministracion.....	27
4.2.3	Dificultad de la pregunta .....	27
4.2.4	Optimizar la lista de preguntas .....	27
5	Integración, pruebas y resultados .....	29
5.1	Integración en producción .....	29
5.2	Pruebas funcionales .....	29
5.2.1	Aplicación móvil .....	29
5.2.2	Aplicación web .....	30
5.3	Pruebas unitarias.....	30
5.4	Mejoras en el proyecto .....	34
5.4.1	Recordar correo y contraseña para iniciar sesión .....	34
5.4.2	Actualizar la contraseña desde la aplicación móvil.....	35
5.4.3	No mostrar la respuesta hasta que no finalice la pregunta .....	35
5.4.4	Preguntas con un número variable de respuestas .....	36
5.4.5	Añadir a las preguntas una explicación de la respuesta.....	36
5.5	Resultados del proyecto.....	37
6	Conclusiones y trabajo futuro.....	39
6.1	Conclusiones.....	39

6.2 Trabajo futuro .....	39
Referencias .....	40
Glosario .....	41
Anexos.....	I
A Manual de Usuario.....	I
B Resultados de la encuesta.....	III
C Historial de versiones e incidencias resueltas .....	XV
D Fragmentos de código.....	XVII
E Capturas de pantalla de ClickEPS.....	XLIII

## INDICE DE FIGURAS

Figura 1: Captura de proyector en Quizalize.....	5
Figura 2: Captura de un menú de cuestionario en Quizziz.....	6
Figura 3: Captura del panel de control de Socrative .....	7
Figura 4: Modelo de datos .....	18
Figura 5: Gráfico estadístico de actividad en vivo MySQL .....	20
Figura 6: Captura de pantalla de exportar pregunta .....	23
Figura 7: Captura de la base de datos en Firebase.....	26
Figura 8: Logo de ClickEPS v.1.0.3 .....	I
Figura 9: Perfil de Twitter ClickEPS.....	II
Figura 10: F.A.Q de administrador.....	II
Figura 11: Encuesta sobre el filtrado por grupos.....	III
Figura 12: Encuesta sobre el detalle de preguntas.....	III
Figura 13: Encuesta sobre el listado de preguntas.....	IV
Figura 14: Encuesta sobre el filtrado por temas .....	IV
Figura 15: Encuesta sobre el login .....	IV
Figura 16: Encuesta sobre el GPS .....	V
Figura 17: Parte de la encuesta realizada a los estudiantes .....	VI
Figura 18: Encuesta sobre el gráfico circular en las estadísticas por estudiante .....	VII
Figura 19: Encuesta sobre las estadísticas por estudiante .....	VII
Figura 20: Encuesta sobre las estadísticas por pregunta.....	VIII
Figura 21: Encuesta sobre el pase de lista .....	VIII
Figura 22: Encuesta sobre el formulario de nueva pregunta .....	IX
Figura 23: Encuesta sobre nueva pregunta .....	IX
Figura 24: Encuesta sobre el formulario de exportar pregunta .....	X
Figura 25: Encuesta sobre exportar pregunta .....	X
Figura 26: Encuesta sobre el formulario de editar pregunta.....	XI
Figura 27: Encuesta sobre editar pregunta .....	XI
Figura 28: Encuesta sobre tema.....	XII
Figura 29: Encuesta sobre editar perfil.....	XII
Figura 30: Parte de la encuesta realizada a los profesores y administradores .....	XIII
Figura 31: Página principal de la interfaz de administración con welcome.....	XLIII
Figura 32: Edición de Admin (editAdmin) .....	XLIV
Figura 33: Editar Perfil de Admin (editProfileAdmin) .....	XLIV
Figura 34: Editar Grupo (editGroup).....	XLV
Figura 35: Editar Tema (editSection) .....	XLV
Figura 36: Editar Asignatura (editSubject).....	XLVI

Figura 37: Editar Usuario (editUser) .....	XLVI
Figura 38: Cargar CSV (manageCSV) .....	XLVII
Figura 39: Parte 1 de manageEnrollment .....	XLVII
Figura 40: Parte 2 de manageEnrollment .....	XLVIII
Figura 41: Grupo (manageGroup) .....	XLIX
Figura 42: Asignatura (manageSubjectSection) .....	L
Figura 43: Tema (manageSubjectSection) .....	LI
Figura 44: Parte 1 de User/Admin (manageUserAdmin) .....	LII
Figura 45: Parte 2 de User/Admin (manageUserAdmin) .....	LIII
Figura 46: Parte 3 de User/Admin (manageUserAdmin) .....	LIII
Figura 47: Captura de pantalla de Estadísticas por pregunta.....	LIV
Figura 48: Captura de pantalla de crear tema (profesorado) .....	LIV
Figura 49: Comparativa visual de versiones (Android).....	LV
Figura 50: Iniciar sesión (Android) .....	LVI
Figura 51: Listado de preguntas (Android) .....	LVII
Figura 52: Detalle de pregunta (Android) .....	LVIII
Figura 53: MockLocation funcionando (Android) .....	LIX
Figura 54: Filtrado por temas (Android) .....	LX
Figura 55: GPS (Android) .....	LXI

## INDICE DE TABLAS

Tabla 1: Comparativa de aplicaciones .....	8
Tabla 2: Pruebas unitarias de la plataforma web (Grupo) .....	31
Tabla 3: Pruebas unitarias de la plataforma web (Estudiante) .....	32
Tabla 4: Pruebas unitarias de la plataforma web (Asignatura).....	32
Tabla 5: Pruebas unitarias de la plataforma web (Tema) .....	32
Tabla 6: Pruebas unitarias de la plataforma web (Administrador/Profesor) .....	33
Tabla 7: Pruebas unitarias de la plataforma web (Pregunta) .....	34
Tabla 8: Recordar correo y contraseña para iniciar sesión .....	34
Tabla 9: Actualizar la contraseña desde la aplicación móvil.....	35
Tabla 10: No mostrar la respuesta hasta que no finalice la pregunta .....	35
Tabla 11: Preguntas con un número variable de respuestas .....	36
Tabla 12: Añadir a las preguntas una explicación de la respuesta .....	37
Tabla 13: Dispositivos Android utilizados durante las pruebas .....	XIV
Tabla 14: Historial de versiones .....	XV
Tabla 15: Historial de incidencias .....	XV

## INDICE DE FRAGMENTOS DE CÓDIGO

Fragmento de código 1: Conexión MySQL mediante un pool.....	XVII
Fragmento de código 2: Ejemplo de consulta MySQL .....	XVII
Fragmento de código 3: Implementación del cifrado HMAC-SHA1 (web) .....	XVII
Fragmento de código 4: Implementación del cifrado HMAC-SHA1 (Android).....	XVIII
Fragmento de código 5: Función hmacDigest para cifrado HMAC-SHA1 (Android) .	XVIII
Fragmento de código 6: Parte del fichero welcome .....	XIX
Fragmento de código 7: Función utilizada en editAdmin .....	XIX
Fragmento de código 8: Función utilizada en editProfileAdmin .....	XX
Fragmento de código 9: Función utilizada en editGroup .....	XX

Fragmento de código 10: Función utilizada en editSection .....	XX
Fragmento de código 11: Función utilizada en editSubject .....	XXI
Fragmento de código 12: Función utilizada en editUser .....	XXI
Fragmento de código 13: Parte de la consulta de manageCSV .....	XXII
Fragmento de código 14: Parte de la funcionalidad de manageEnrollment .....	XXIII
Fragmento de código 15: Parte de la funcionalidad de manageGroup .....	XXIV
Fragmento de código 16: Parte de la funcionalidad de manageSubjectSection .....	XXV
Fragmento de código 17: Parte de la funcionalidad de manageUserAdmin .....	XXVI
Fragmento de código 18: Parte de la funcionalidad de Resetear DB .....	XXVII
Fragmento de código 19: Ejemplo función en service adminDataSer.js .....	XXVII
Fragmento de código 20: Parte del formulario de exportar pregunta .....	XXVIII
Fragmento de código 21: Carga de todas las preguntas en el objeto vm.allQuestions .	XXIX
Fragmento de código 22: Función para conseguir los alumnos ausentes .....	XXIX
Fragmento de código 23: Función para conseguir el gráfico estadístico .....	XXX
Fragmento de código 24: div que genera el gráfico estadístico .....	XXX
Fragmento de código 25: div que genera la lista de estudiantes ausentes .....	XXX
Fragmento de código 26: updatePassword.HTML .....	XXXI
Fragmento de código 27: updatePassword en Android .....	XXXII
Fragmento de código 28: Función para crear tema .....	XXXII
Fragmento de código 29: Función para borrar tema .....	XXXIII
Fragmento de código 30: Implementación de un geofence para la facultad .....	XXXIII
Fragmento de código 31: Conexión al PlayServices de Google .....	XXXIV
Fragmento de código 32: Conexión a la API de Google .....	XXXIV
Fragmento de código 33: Objeto geoFire y referencia a Firebase .....	XXXIV
Fragmento de código 34: Reglas de la base de datos de Firebase .....	XXXV
Fragmento de código 35: Actualizaciones a base de datos de Firebase .....	XXXV
Fragmento de código 36: Implementación de Callback .....	XXXVI
Fragmento de código 37: Implementación de Mock Locations .....	XXXVI
Fragmento de código 38: isMockSettingsOn .....	XXXVII
Fragmento de código 39: areThereMockPermissionApps .....	XXXVII
Fragmento de código 40: Implementación de Shared Preferences .....	XXXVIII
Fragmento de código 41: Implementación de métodos de Shared Preferences .....	XXXIX
Fragmento de código 42: Implementación sencilla de ClickEPSAdministración .....	XL
Fragmento de código 43: Parte de ocultar respuesta hasta expiración .....	XL
Fragmento de código 44: Parte de la implementación de numAnswers .....	XLI
Fragmento de código 45: Implementación de la explicación de las preguntas .....	XLI
Fragmento de código 46: Implementación del evento refresh .....	XLII
Fragmento de código 47: Notificar disponibles .....	XLII





# 1 Introducción

---

En este capítulo se expondrán las motivaciones para realizar el proyecto, los objetivos que se pretenden alcanzar con su elaboración y la organización de la memoria.

## 1.1 Motivación

Vivimos en una época totalmente ligada a la tecnología. Los avances en este campo se aplican a la sanidad, la educación o incluso a temas tan concretos como la declaración de la renta. La tecnología ha beneficiado a todos ellos, y en este Trabajo de Fin de Grado veremos cómo ha llevado el ámbito educativo a estar directamente relacionado con el uso de aplicaciones web o con el uso de los dispositivos inteligentes.

Uno de los métodos de evaluación más novedosos se basa en el uso de *clickers* [\[1\]](#), un sistema de aprendizaje interactivo en el que el alumno tiene un papel participativo muy importante. Estos métodos de evaluación le permiten contestar a las preguntas que se plantean en clase de una forma fácil y efectiva.

También se han desarrollado aplicaciones web con el mismo objetivo que complementan el uso del *Smartphone*. Sin embargo y, como veremos en el capítulo de Estado del Arte, muchas de las aplicaciones en la actualidad tienen limitaciones relacionadas con el uso de proyectores o con la necesidad de evitar que los alumnos contesten desde fuera de clase durante un examen.

Por otro lado, parte de ellas se basan en cuestionarios y en ningún momento almacenan la información real de los alumnos, ya que no es necesario se registren. Esto es lo que motivó a la realización de este Trabajo de Fin de Grado, el crear un sistema de administración capaz de almacenar la información real de todos los alumnos, profesores, asignaturas, grupos, temas, preguntas y respuestas para así reutilizar datos entre asignaturas y profesores.

El hecho de que no existan aún aplicaciones que dispongan de un sistema de preguntas presenciales (que sólo puedan ser contestadas en clase) es lo que motivó a desarrollar un sistema basado en ubicaciones en tiempo real que mantendrá a los profesores al tanto de los alumnos que asisten y no asisten a clase, como veremos en el capítulo de Desarrollo.

Por todo esto se continuará el desarrollo de las dos aplicaciones que forman ClickEPS desde su versión v1.0.1: una desarrollada para dispositivos móviles Android [\[2\]](#) con una interfaz sencilla y una aplicación web implementada con el *framework* AngularJs [\[3\]](#), el entorno *Javascript* del lado del servidor Node.js [\[4\]](#), y la base de datos de MySQL [\[5\]](#).

## 1.2 Objetivos

El objetivo inicial de este TFG era la actualización de la aplicación ClickEPS, dotándola de una interfaz que administrara todos los datos desde la plataforma web.

Sin embargo, a lo largo de las fases de análisis, diseño y desarrollo del proyecto, se definieron nuevos objetivos que ayudarían a mejorar la plataforma de profesorado. La sección de preguntas, la seguridad y la conexión a la base de datos son algunos de los aspectos que se mejoraron durante el proyecto.

Un conjunto de objetivos específicos tienen que ver con una estabilidad mayor en las aplicaciones, que consisten en mejorar la conexión, las consultas a la base de datos y la seguridad de las contraseñas. Por otra parte, necesitamos crear un nuevo rol en la base de datos de administrador, que se encargará de manejar todos los datos de la aplicación. Junto con este rol se desarrollará una interfaz de administración similar a la de profesorado que permitirá importar datos de cualquier entidad mediante ficheros en formato CSV y crear, modificar o eliminar datos de cualquiera de ellas a través de formularios web. También se podrá matricular a estudiantes en distintos grupos y anular su matrícula en cualquier momento.

Otro objetivo de este trabajo es el desarrollo de una aplicación en Android que permita acceder de manera cómoda a la interfaz de administración o profesorado sin necesidad de utilizar un navegador. En relación a los datos personales, se permitirá actualizar la contraseña de los estudiantes desde la aplicación Android y al resto de usuarios desde la plataforma de administración y profesorado.

Se ha trabajado también sobre las preguntas que pueden plantearse a los estudiantes. Se ofertarán dos tipos posibles: presenciales o de tarea para casa. En caso de las preguntas presenciales, se implementará una base de datos en tiempo real de Firebase [\[6\]](#) para manejar la ubicación de los estudiantes. Para definir el área presencial de la Escuela, implementaremos un *geofence* [\[7\]](#) junto con una función capaz de detectar simuladores de ubicaciones. Con esta mejora, añadiremos en la visualización de las estadísticas a los estudiantes que han faltado a clase. Por último, extenderemos la diversidad de respuestas de una pregunta, que actualmente está siempre a cuatro. Permitiremos crear preguntas con dos y tres respuestas, y añadiremos un campo llamado explicación que utilizarán los estudiantes para consultar por qué han acertado o fallado la pregunta. Además, permitiremos al profesor poder exportar cualquiera de ellas que se encuentre disponible en la aplicación a su grupo y tema que desee.

Un último objetivo ha sido realizar pruebas en entorno real tanto de la aplicación web como de la aplicación Android. Todas ellas han sido realizadas en la Escuela Politécnica Superior de la Universidad Autónoma de Madrid.

### **1.3 Organización de la memoria**

Además de este capítulo de Introducción, la memoria se compone de los siguientes apartados:

- **Capítulo 2 (Estado del Arte):** En este capítulo estudiaremos aplicaciones muy similares a nuestro proyecto que existen en la actualidad para cubrir las mismas necesidades en el ámbito educativo. Analizaremos sus ventajas y desventajas con respecto a nuestra aplicación y finalizaremos con una tabla comparativa para destacar aquellas características que permiten mejorar nuestro proyecto.
- **Capítulo 3 (Análisis y Diseño):** En esta sección describiremos los nuevos requisitos funcionales y no funcionales para el desarrollo de las aplicaciones y finalizaremos mostrando el modelo de datos actualizado, explicando los nuevos campos de la base de datos.
- **Capítulo 4 (Desarrollo):** Este bloque describirá detalladamente cómo se ha desarrollado tanto la aplicación Android como la plataforma web destacando las implementaciones más importantes y explicando cómo se han llevado a cabo.
- **Capítulo 5 (Integración, pruebas y resultados):** En este capítulo explicaremos cómo integrar ambas aplicaciones en un entorno real, las pruebas unitarias y funcionales que se realizaron junto con las mejoras que se llevaron a cabo tras estas pruebas.
- **Capítulo 6 (Conclusiones y trabajo futuro):** Es el capítulo final del trabajo, detallaremos las conclusiones finales que resumen el trabajo realizado y propondremos desarrollos y mejoras para el futuro de ClickEPS.
- **Glosario:** En este apartado mencionaremos los tecnicismos utilizados durante el trabajo y explicaremos el significado de algunas palabras técnicas.
- **Anexos:** En esta sección adjuntaremos el manual del usuario, los resultados de las encuestas realizadas, un historial de incidencias de la aplicación, fragmentos de código y capturas de pantalla de las aplicaciones.



## 2 Estado del arte

---

En este capítulo estudiaremos algunas aplicaciones que existen en la actualidad con una funcionalidad similar a nuestro proyecto, que intentan cubrir las mismas necesidades en el ámbito de la docencia mediante técnicas de gamificación [8].

### 2.1 Quizalize

Quizalize [9] es una aplicación web cuya finalidad consiste en fomentar la participación en las clases a partir de pruebas divertidas o tareas interactivas. Esta aplicación no necesita de descargas para poder ser utilizada desde cualquier dispositivo. Actualmente está disponible en inglés, japonés y árabe, pero ya está ofreciendo una versión beta en español de América Latina.

Como docente podemos acceder a una interfaz privada con algunas funcionalidades interesantes, como compartir los cuestionarios con los demás docentes, añadir música durante el cuestionario, poner un cronómetro o imágenes a las preguntas y crear respuestas de opción múltiple o única. Sin embargo, de todas las características que ofrece Quizalize, destaca la posibilidad de añadir una explicación cuando un estudiante selecciona una respuesta incorrecta, haciéndoles conocedores del por qué han fallado sin necesidad de interactuar con el docente.

Quizalize tiene varios planes de suscripción, desde uno totalmente gratuito que ofrece la creación de exámenes y juegos por equipos en tiempo real, hasta otros de pago destinados a colegios, donde pueden alojar en la aplicación a un número de estudiantes y grupos ilimitado, resultados imprimibles o paneles estadísticos avanzados para realizar un seguimiento continuo de los alumnos y así poder ayudar a tiempo a quien lo necesite.

El funcionamiento de los juegos por equipos en tiempo real es sencillo: a medida que los alumnos van ingresando en la aplicación mediante el *link* y el código que les proporciona el docente, estos van dividiéndose en una serie de equipos, y entonces se les va presentando las preguntas por turnos que irán contestando desde sus dispositivos. Los cuestionarios funcionan del mismo modo, solo que en el proyector van apareciendo otro tipo de estadísticas o el enunciado de cada pregunta.



**Figura 1: Captura de proyector en Quizalize**

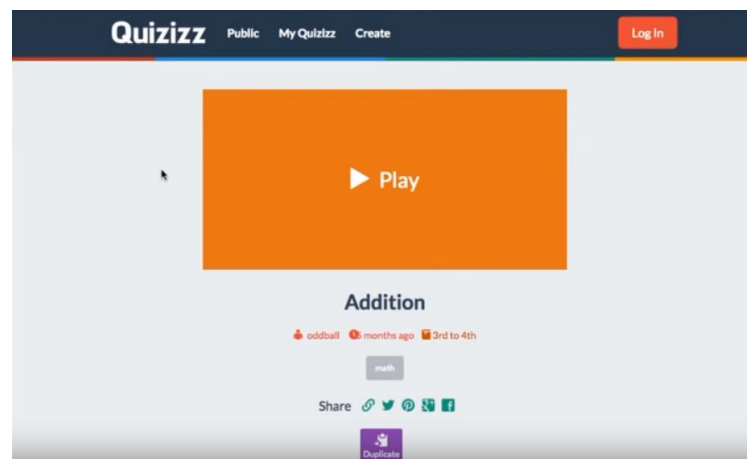
A pesar de ser una aplicación con un gran potencial, los estudiantes no pueden acceder a los resultados tras realizar el cuestionario, ni los docentes disponen de una interfaz de administración con muchas posibilidades, ya que lo único que pueden crear son clases y cuestionarios. Además, al necesitar un proyector, no permite que los alumnos puedan responder desde sus casas. Estas limitaciones no existen en nuestra aplicación porque ofrecemos una administración más compleja que no es sólo de clases o grupos, y permitimos a los estudiantes consultar preguntas realizadas y contestarlas desde cualquier lugar o únicamente de forma presencial.

## 2.2 Quizziz

Esta aplicación llamada Quizziz [10] es gratuita y puede ser utilizada en cualquier dispositivo, aunque también dispone de aplicaciones para IOS [11], Android y Chrome [12].

En Quizziz, los alumnos tampoco necesitan registrarse para contestar, sino que únicamente necesitan el código del juego o cuestionario. La diferencia con respecto a la aplicación anterior es que el proyector es opcional, lo que permite a los alumnos contestar también desde sus casas de forma independiente.

Aunque la funcionalidad de los cuestionarios es similar a Quizalize, en esta podemos añadir “memes” tras las respuestas, y disponemos de distintas opciones para las preguntas: que el orden de aparición cambie para cada alumno o que se altere el orden de las respuestas. También podemos premiar a los alumnos con más tiempo de respuesta a medida que obtengan más puntuación y mostrar la tabla de clasificación en la pantalla, entre otras.



**Figura 2: Captura de un menú de cuestionario en Quizziz**

Los estudiantes tendrán un avatar que podrán cambiar durante la espera del juego, así como el *background* de la pantalla en su dispositivo.

Tras haber realizado el cuestionario, el profesor podrá visualizar estadísticamente los resultados de cada estudiante por pregunta, y la puntuación que ha obtenido. También tendrá la posibilidad de exportar los resultados a una hoja de Excel, así como revisar las preguntas para poder repasarlas con los estudiantes.

Quizziz apuesta por la independencia de los alumnos, ofreciéndoles la posibilidad de contestar fuera de clase a través de sus dispositivos. Sin embargo, tampoco consigue que puedan consultar los resultados en cualquier momento, ni ofrece al docente más funcionalidad aparte de crear cuestionarios y juegos.

## 2.3 Socrative

Socrative [\[10\]](#) es una aplicación web compatible con cualquier dispositivo que como las anteriores ayuda a los estudiantes en clase con cuestionarios y torneos interactivos.

Al entrar en la aplicación web observamos que existen dos planes: el gratuito y el de pago, éste último llamado *pro*, donde ofrece una serie de características que no están disponibles en el primer plan.

Socrative puede ser utilizada sin necesidad de descargar nada, y como docente, ofrece crear cuestionarios, preguntas rápidas, “carrera espacial” y una encuesta final tras cada cuestionario creado.



**Figura 3: Captura del panel de control de Socrative**

Las preguntas son de tres tipos: elección única, respuesta corta o verdadero/falso. Éstas son las que se pueden utilizar para crear preguntas rápidas. Además, los cuestionarios pueden ser de tres tipos. El primero es *guiado por el estudiante con resultados inmediatos*, donde reciben inmediatamente la respuesta tras contestar a cada pregunta junto con la explicación en caso de haber fallado. No pueden modificar sus respuestas, y los resultados aparecen en directo en la pantalla del profesor. El segundo es *guiado por el estudiante pero con navegación del mismo*, donde pueden saltarse preguntas y volver hacia atrás para editar las respuestas. Una vez finalicen, envían el formulario. El profesor también puede visualizar los resultados en vivo según van enviando los resultados. Por último, tenemos el *guiado por el docente*, donde estos controlan el flujo de las preguntas y cada vez que envían una pregunta, pueden ir visualizando los resultados según van respondiendo los estudiantes.

La carrera espacial es muy similar a los juegos ya mencionados anteriormente: se elige un cuestionario, la elección y número de equipos (ya sea generada automáticamente por la aplicación o generada por el mismo profesor) y un avatar por equipo. A medida que los equipos vayan respondiendo a las preguntas, irán sumando puntos para avanzar en una carrera visual visible en el proyector.

Por último, ofrece encuestas que se utilizan para saber la opinión de los alumnos tras el cuestionario. Esto fomenta el *feedback* entre el docente y el alumno, pudiendo mejorar para futuros cuestionarios.

Esta aplicación es la más parecida a la propuesta en este trabajo debido a que ofrece un mayor control durante los cuestionarios. Sin embargo, los alumnos siguen sin poder visualizar los resultados desde sus dispositivos móviles una vez han enviado el cuestionario.

## 2.4 Conclusiones

La Tabla 1 muestra un resumen de las ventajas y desventajas de cada una de las aplicaciones analizadas en este capítulo. Aparecen subrayadas las características más llamativas que resultarán útiles para completar el proyecto:

Aplicación	¿Lo mejor?	¿Lo peor?
Quizalize	<ul style="list-style-type: none"> <li>• Multiplataforma</li> <li>• Varios idiomas disponibles</li> <li>• <u>Posibilidad de compartir cuestionarios</u></li> <li>• <u>Se pueden añadir explicaciones a las preguntas</u></li> </ul>	<ul style="list-style-type: none"> <li>• Necesita un proyector para visualizar las preguntas.</li> <li>• No permite a los estudiantes consultar los resultados tras enviar el cuestionario</li> </ul>
Quizziz	<ul style="list-style-type: none"> <li>• Tiene versión para móvil</li> <li>• <u>Sistema estadístico muy completo</u></li> <li>• Personalización de estudiante</li> <li>• Se puede modificar el orden de aparición de las preguntas y respuestas</li> </ul>	<ul style="list-style-type: none"> <li>• No tiene nada parecido a preguntas presenciales (realizables sólo en la facultad)</li> <li>• <u>No permite crear preguntas con un número de respuestas distinto de cuatro</u></li> </ul>
Socrative	<ul style="list-style-type: none"> <li>• Posibilidad de recorrer los cuestionarios</li> <li>• Juegos por turnos y equipos</li> <li>• <u>Permite realizar encuestas tras los cuestionarios</u></li> <li>• No necesita proyector</li> </ul>	<ul style="list-style-type: none"> <li>• Gratuita, aunque muy limitada al tener un modo <i>pro</i> que ofrece más funcionalidades</li> </ul>

**Tabla 1: Comparativa de aplicaciones**



## 3 Análisis y Diseño

---

En este capítulo presentamos los requisitos recogidos en la fase de análisis siguiendo el ciclo de vida en cascada [13], así como los aspectos de diseño estudiados para llevar a la aplicación a un nuevo nivel. Todos los requisitos forman en conjunto la interfaz de administración para la aplicación web, excepto los dos últimos requisitos funcionales, que pertenecen a mejoras de la interfaz de profesorado.

### 3.1 Análisis de Requisitos

En este análisis dividiremos los requisitos en funcionales y no funcionales con su descripción, entrada, proceso, y salida para comprender cómo y de qué manera tendría cada uno cabida en la aplicación.

#### 3.1.1 Requisitos Funcionales

A continuación expondremos los requisitos funcionales que se desarrollarán durante el ciclo de vida del proyecto. Los requisitos **RF21** y **RF22** corresponden a la interfaz de profesorado, mientras que el resto son de rol de administrador.

##### (RF1) Rol de administrador

**Descripción:** Se define un rol distinto al de usuario para la gestión de todos los datos de la aplicación. Este rol se identifica con un correo y contraseña, que puede coincidir con el correo institucional y contraseña utilizados como rol de usuario (Profesor).

##### (RF2) Acceso a la aplicación

**Descripción:** Es necesaria la autenticación mediante correo y contraseña.

**Entrada:** El administrador introduce su correo y contraseña en el formulario de iniciar sesión.

**Proceso:** La aplicación lanza una petición a la API, y ésta a su vez a la base de datos. Si los datos son válidos, pueden darse dos salidas posibles. Si son incorrectos, saldrá un mensaje de error.

**Salida:** Se contemplan dos salidas posibles: En caso de que las credenciales sean correctas, se comprobará si existen los mismos datos como rol de usuario (Profesor), y en caso afirmativo, permitirá que sea decisión del administrador acceder a la plataforma de profesorado o a la de administración. Si sólo existe como administrador, accederá automáticamente a la aplicación correspondiente.

### **(RF3) Crear grupo**

**Descripción:** Se podrán crear nuevos grupos asociados a una asignatura que serán impartidos por un profesor.

**Entrada:** En primer lugar, se introducen en un formulario las siglas del grupo, que deben seguir la nomenclatura habitual de la EPS. Por ejemplo, G121 correspondería de izquierda a derecha a un grupo del primer semestre, que pertenece al segundo curso de la carrera y número de grupo uno. También se tiene que seleccionar la asignatura y el profesor que la impartirá.

**Proceso:** La aplicación comprueba que los datos han sido introducidos correctamente y lanza una consulta a la base de datos para crear el nuevo grupo.

**Salida:** La aplicación lanza un mensaje avisando de la creación del grupo y redirige al administrador a la página principal, refrescando los datos para contemplar el nuevo grupo en futuras operaciones. Si existe el grupo, lanzará un mensaje de error.

### **(RF4) Editar grupo**

**Descripción:** Se podrán editar los grupos creados modificando algunos de sus datos.

**Entrada:** Seleccionando el grupo a editar, se cargarán las siglas. Se podrán editar a continuación dichas siglas, seleccionar una nueva asignatura o mantener la misma, y asociar un nuevo profesor o continuar con el mismo.

**Proceso:** La aplicación comprueba que han sido introducidos los datos y lanza una consulta a la base de datos para actualizar los datos del grupo seleccionado.

**Salida:** Se lanza un mensaje avisando de la actualización del grupo y se redirige al administrador a la página principal refrescando los datos para contemplar el grupo actualizado en futuras operaciones. Si ya existe el grupo, es decir, es igual al editado, la aplicación lanzará un mensaje de error.

### **(RF5) Eliminar datos de grupo**

**Descripción:** Se podrán eliminar los grupos creados junto con los datos dependientes.

**Entrada:** El administrador tendrá que seleccionar únicamente el grupo a eliminar entre todos los disponibles.

**Proceso:** La aplicación comprueba que ha sido seleccionado un grupo y lanza una consulta a la base de datos para eliminar tanto el grupo como los datos de las tablas que dependen del mismo: *Answer*, *Question*, *Tuition* y *Section*.

**Salida:** Se lanza un mensaje avisando de la eliminación de los datos del grupo y dependientes, y redirige al administrador a la página principal refrescando los datos para contemplar el grupo eliminado en futuras operaciones.

### **(RF6) Matricular estudiante**

**Descripción:** Se podrá asociar un estudiante con un grupo creado anteriormente. Mediante esta asociación, se determina que el estudiante ha sido matriculado en el grupo de la asignatura.

**Entrada:** El administrador seleccionará al estudiante de entre todos los disponibles, la opción de matricular, y el grupo en el que se matriculará de todos los creados.

**Proceso:** La aplicación comprueba que han sido seleccionados los datos de entrada y lanza una consulta a la base de datos para matricular al estudiante en el grupo seleccionado.

**Salida:** La aplicación lanza un mensaje avisando de la matriculación del estudiante. No admite mensajes de error debido al diseño, ya que en caso de estar el estudiante matriculado en un grupo, éste no será seleccionable.

### **(RF7) Anular matrícula de Estudiante**

**Descripción:** Se podrá anular la matrícula de un estudiante asociado a un grupo existente.

**Entrada:** El administrador seleccionará al estudiante de entre todos los disponibles, la opción de anular matrícula, y el grupo en el que se anulará de entre todos los que tiene matrícula.

**Proceso:** La aplicación comprueba que han sido seleccionados los datos de entrada y lanza una consulta a la base de datos para anular la matrícula del estudiante.

**Salida:** Lanza un mensaje avisando de la anulación. No admite mensajes de error debido a que en caso de que el estudiante no esté matriculado en el grupo, éste no será seleccionable.

### **(RF8) Registrar usuario o administrador**

**Descripción:** Mediante un formulario sencillo y distinto, dependiendo si se quiere registrar a un estudiante, a un profesor o a un administrador, se permitirá su registro en la aplicación para que posteriormente éste pueda utilizar su cuenta en el móvil o en la web, dependiendo del rol.

**Entrada:** Nombre, apellidos, correo institucional y contraseña. El correo debe ser institucional (@estudiante.uam.es o @uam.es) para que se valide el formulario, excepto en el rol de administrador.

**Proceso:** La aplicación comprueba que han sido introducidos los datos de entrada correctamente y realiza una consulta a la base de datos para registrar al usuario o administrador.

**Salida:** La aplicación lanza un mensaje avisando del registro en caso afirmativo. Si no ha cumplido con la norma del correo institucional en caso de ser obligatorio, o el correo introducido ya se encuentra en uso por parte de otra persona, el sistema lanzará un mensaje de error y devolverá al administrador a la página principal.

### **(RF9) Editar usuario o administrador**

**Descripción:** Al igual que en el requisito anterior, mediante un formulario sencillo que será distinto dependiendo de si es un estudiante, un profesor o un administrador, se permitirá la edición de sus datos para que posteriormente éste pueda utilizar su cuenta actualizada en el móvil o en la web, dependiendo del rol.

**Entrada:** El administrador selecciona a un usuario o administrador de entre todos los registrados. Una vez hecho, se rellenarán los campos que podrá cambiar: nombre, apellidos y correo institucional. El correo debe seguir siendo institucional (@estudiante.uam.es o @uam.es) en caso de ser un usuario para que se valide el formulario, y no puede estar ya en uso.

**Proceso:** La aplicación comprueba que han sido introducidos los datos de entrada correctamente y realiza una consulta a la base de datos para modificar los datos del usuario o administrador.

**Salida:** La aplicación lanza un mensaje avisando de la modificación en caso afirmativo. Si no ha cumplido con la norma del correo institucional, o el correo introducido ya se encuentra en uso por parte de otro usuario, el sistema enviará un mensaje de error y devolverá al administrador a la página principal.

### **(RF10) Actualizar contraseña de usuario o administrador**

**Descripción:** Mediante un formulario sencillo que será distinto dependiendo de si es un estudiante, un profesor o un administrador, se permitirá actualizar la contraseña del usuario o administrador.

**Entrada:** El administrador selecciona a un usuario o administrador de entre todos los registrados. Una vez hecho, podrá escribir la nueva contraseña.

**Proceso:** La aplicación comprueba que han sido introducidos los datos de entrada correctamente y realiza una consulta a la base de datos para modificar la contraseña del usuario o administrador, cifrándola antes de insertar los datos.

**Salida:** La aplicación lanza un mensaje avisando de la actualización en caso afirmativo. En caso contrario, el sistema enviará un mensaje de error y devolverá al administrador a la página principal.

### **(RF11) Eliminar datos de usuario o administrador**

**Descripción:** Se permitirá eliminar los datos del usuario o administrador de la aplicación.

**Entrada:** Usuario o administrador a eliminar.

**Proceso:** La aplicación verifica que se ha seleccionado un usuario o administrador y lanza una nueva consulta a la base de datos para eliminar todos los datos relacionados con él (*User o Admin, Answer y Tuition*).

**Salida:** El sistema lanza un mensaje avisando de la eliminación del mismo en caso afirmativo. En caso negativo, informará del error por pantalla.

### **(RF12) Crear asignatura**

**Descripción:** Mediante un formulario sencillo, se podrá crear una asignatura, introduciendo los datos de entrada correspondientes.

**Entrada:** El administrador deberá ingresar el código de la asignatura, que se puede encontrar en la guía docente; el nombre de la asignatura, las siglas (por ejemplo, Programación II es PROG2), el curso y el cuatrimestre.

**Proceso:** La aplicación comprueba los datos de entrada y manda una consulta a la base de datos para que ésta registre la nueva asignatura.

**Salida:** El sistema lanza un mensaje avisando de la creación de la asignatura en caso afirmativo. En caso de que el código de la asignatura ya exista, lanzará un mensaje de error y redigirá al administrador a la página principal.

### **(RF13) Editar asignatura**

**Descripción:** Se permitirá modificar los datos de una asignatura (excepto su código) mediante un formulario.

**Entrada:** Una vez que el administrador ha seleccionado qué asignatura editar, se cargarán sus datos en el formulario, de manera que podrá modificarlos a su gusto. Estos datos son el nombre de la asignatura, sus siglas, el curso y el cuatrimestre.

**Proceso:** La aplicación comprueba los datos de entrada y lanza una consulta a la base de datos para modificar los datos de la asignatura.

**Salida:** El sistema manda un mensaje avisando de la modificación de datos en caso afirmativo, y en caso de introducir mal alguno de los datos de entrada, informará mediante un error. Finalmente, redigirá al administrador a la página principal.

### **(RF14) Eliminar datos de una asignatura**

**Descripción:** Se permitirá eliminar los datos de una asignatura de la aplicación.

**Entrada:** Asignatura a eliminar.

**Proceso:** La aplicación verifica que se ha seleccionado una asignatura y lanza una nueva consulta a la base de datos para eliminar todos los datos relacionados con la asignatura (*Subject, Answer, Question, Tuition, Section y Group*).

**Salida:** El sistema lanza un mensaje avisando de la eliminación de la asignatura en caso afirmativo. En caso negativo, informará del error por pantalla.

### **(RF15) Crear tema**

**Descripción:** Mediante un formulario sencillo, se podrá crear un tema asociado a un grupo, introduciendo los datos de entrada correspondientes.

**Entrada:** El administrador deberá ingresar el nombre del tema y el grupo al que se le asignará de entre todos los disponibles.

**Proceso:** La aplicación comprueba los datos de entrada y manda una consulta a la base de datos para que ésta registre el nuevo tema.

**Salida:** El sistema lanza un mensaje avisando de la creación del tema en caso afirmativo. En caso negativo, lanzará un mensaje de error y redirigirá al administrador a la página principal.

#### **(RF16) Editar tema**

**Descripción:** Se permitirá modificar los datos de un tema (excepto su grupo) mediante un formulario.

**Entrada:** El administrador, una vez ha seleccionado qué tema editar, se cargará su nombre en el formulario, de manera que podrá modificarlo a su gusto. Debido a la facilidad de crear el temario, únicamente se puede modificar el nombre.

**Proceso:** La aplicación comprueba el dato de entrada y lanza una consulta a la base de datos para modificar los datos del tema.

**Salida:** El sistema manda un mensaje avisando de la modificación de datos en caso afirmativo, y en caso negativo, informará mediante un error. Finalmente, redigirá al administrador a la página principal.

#### **(RF17) Eliminar datos de tema**

**Descripción:** Se permitirá eliminar los datos de un tema de la aplicación.

**Entrada:** Tema a eliminar.

**Proceso:** La aplicación verifica que se ha seleccionado un tema y lanza una nueva consulta a la base de datos para eliminar todos los datos relacionados con el tema (*Section*, *Answer*, y *Question*).

**Salida:** El sistema lanza un mensaje avisando de la eliminación del tema en caso afirmativo. En caso negativo, informará del error por pantalla.

#### **(RF18) Restaurar base de datos**

**Descripción:** Se permitirá eliminar todos los datos de la base de datos, y restaurar las asignaturas que vienen por defecto en la aplicación.

**Entrada:** Botón en el panel lateral derecho que indica “Resetear DB”.

**Proceso:** El administrador pulsa la opción mencionada y el sistema le advierte de las consecuencias que conlleva la operación. En caso de aceptar, el sistema lanza una serie de

consultas en cascada para eliminar todos los datos excepto los de administradores, y restaura mediante otra consulta las asignaturas por defecto.

**Salida:** El sistema lanza un mensaje afirmativo en caso de haber restaurado la base de datos y devuelve al administrador a la página principal. En caso negativo, lanza un mensaje de error.

### **(RF19) Importar mediante ficheros CSV distintos tipos de datos**

**Descripción:** Se permitirá añadir múltiples datos mediante ficheros de tipo CSV siguiendo un formato adecuado.

**Entrada:** Fichero CSV a procesar.

**Proceso:** La aplicación comprueba que se ha seleccionado un fichero y lanza una consulta a la base de datos para importar todos los datos del mismo.

**Salida:** La aplicación lanza un mensaje avisando de la importación de los datos con el resultado de la consulta en caso afirmativo. En caso negativo, informará del error por pantalla.

### **(RF20) Cerrar sesión**

**Descripción:** Se permitirá que el administrador, al igual que en la plataforma de profesorado ya desarrollada, pueda cerrar sesión mediante un proceso *logout*.

**Entrada:** Botón en el panel derecho que indica “Cerrar sesión”.

**Proceso:** La aplicación elimina los datos de sesión del administrador y redirige al mismo a la página de inicio de sesión.

**Salida:** La aplicación cierra la sesión del usuario y le redirige a la pantalla de *login*.

### **(RF21) Exportar pregunta**

**Descripción:** En la plataforma de profesorado, el profesor únicamente puede crear una nueva pregunta, editarla, duplicarla o eliminarla, pero en ningún caso puede elegir una pregunta de entre todas las existentes en la aplicación. Con la opción de exportar pregunta, todas las preguntas se compartirán entre los profesores, de tal manera que estos podrán elegir qué preguntas quieren para sus grupos, y a qué tema asignárselos.

**Entrada:** Un formulario sencillo en el que el profesor únicamente seleccionará la pregunta a exportar. En cuanto la seleccione, todos los datos de la misma se reflejarán en el formulario.

**Proceso:** Una vez el profesor ha seleccionado la pregunta, tendrá que elegir a qué grupo y tema asignarle la nueva pregunta, con ayuda del panel izquierdo. Una vez seleccionado y pulsando el botón de exportar pregunta, el sistema lanza una consulta a la base de datos para crear la pregunta.

**Salida:** En caso de haber seleccionado un grupo y un tema, el sistema lanzará un mensaje de que se ha podido exportar la pregunta. En caso negativo, informará de un error.

### **(RF22) Preguntas presenciales o de tarea**

**Descripción:** En la plataforma de profesorado, el profesor puede actualmente crear preguntas que se contestan desde cualquier lugar en el que el estudiante se conecte con su móvil. Con esta nueva característica, se da la opción de crear preguntas de dos tipos, presenciales o como tarea. Las primeras comprueban mediante una base de datos en tiempo real que el alumno se encuentra en la facultad para poder contestar a la pregunta, mientras que las segundas se pueden contestar en cualquier momento.

**Entrada:** Un selector de tipo de pregunta, que da a elegir entre presencial o tarea. Éste estaría incluido en los formularios de nueva pregunta, editar pregunta y exportar pregunta.

**Proceso:** Una vez el profesor ha creado, editado o exportado una pregunta, este nuevo campo se almacena con la consulta del sistema en la base de datos, concretamente en un nuevo campo de la tabla *Question* llamada *homework*, donde un valor 0 indica que será presencial, mientras que un valor 1 indica que será no presencial o de deberes.

**Salida:** En caso de haber seleccionado un grupo y un tema, el sistema lanzará un mensaje de que se ha podido crear la pregunta. En caso negativo, informará de un error.

Este requisito quizá sea el más ambicioso del proyecto, porque obliga a desarrollar una serie de funcionalidades nuevas que garanticen la comprobación presencial de los alumnos a la hora de contestar a las preguntas. La primera de ellas consiste en la **actualización de las estadísticas por pregunta**, donde los alumnos que no asistan a clase tendrán su porcentaje en el gráfico circular de la sección estadísticas por pregunta, disponible en la interfaz de profesorado de la aplicación web. Por otro lado tenemos el **pase de lista**, en el que se mostrará a los alumnos que no registren una respuesta, con el objetivo de llevar el recuento de los que no han asistido. Otra funcionalidad sería la **comprobación de que el alumno se encuentra en la facultad**. Para ello se desarrollará un *geofence* en Android y mediante consultas a una base de datos en tiempo real, comprobará que las ubicaciones se encuentren en la facultad. Por último, necesitaremos **que el alumno no pueda simular una ubicación**. Esto es importante porque hoy en día existen multitud de aplicaciones Android capaces de simular ubicaciones, haciendo pensar al sistema que el alumno se encuentra en la facultad. Por ello, se va a desarrollar una comprobación de que el alumno no utilice este tipo de aplicaciones.

### **3.1.2 Requisitos no Funcionales**

A continuación se exponen los requisitos no funcionales de las aplicaciones, algunos de los cuales son actualizaciones de los implementados en la versión de partida v1.0.0.

#### **(RNF1) Requisitos de Disponibilidad**

**(RNF 1.1)** Durante el mantenimiento programado de ambas aplicaciones, el sistema debe avisar a los usuarios de ello en caso de que intenten acceder a la web o a la aplicación móvil. La versión actual sólo avisa en la aplicación móvil que no hay red.



(RNF 1.2) Para atender a los usuarios y darles soporte, el sistema debe indicar y redirigir a páginas que sirvan para comunicarse directamente con ellos, así como proporcionar tutoriales de uso.

#### **(RNF2) Requisitos de Seguridad**

(RNF 2.1) La aplicación será accesible mediante correo y contraseña para todos los roles, incluido el de administrador.

(RNF 2.2) Las contraseñas se encriptarán con un nuevo algoritmo de encriptación llamado HMAC-SHA1 [14], que a diferencia del que se implementó anteriormente (SHA1), evita posibles ataques masivos al codificarse junto a una clave de cifrado, y serán almacenadas de esta manera en la base de datos.

(RNF 2.3) La sesión del administrador se acaba tras finalizar sesión, al igual que la del usuario.

#### **(RNF3) Requisitos de Rendimiento**

(RNF 3.1) La aplicación mantiene los tiempos de respuesta menores a los 10 segundos.

(RNF 3.2) Para las consultas a la base de datos, se mejorará la estabilidad y escalabilidad agrupando las conexiones en lo que se conocen como *pool*, logrando una alta estabilidad con hasta 500 conexiones simultáneas.

#### **(RNF4) Requisitos de Testeabilidad**

(RNF 4.1) El proyecto se dividirá en tres versiones y en las dos últimas se permitirá a los usuarios testear las aplicaciones para detectar incidencias. También se proporcionarán vídeo tutoriales a los profesores para probar todos los aspectos de la plataforma web.

#### **(RNF5) Requisitos de Extensibilidad**

(RNF 5.1) El proyecto mantendrá las características de *frameworks* que hacen posible su sencilla extensibilidad.

#### **(RNF6) Requisitos de Plataformas**

(RNF 6.1) La aplicación móvil continuará siendo accesible desde cualquier dispositivo Android con versión mayor o igual a la 4.4.2.

(RNF 6.2) La aplicación web seguirá siendo accesible desde cualquier navegador web.

(RNF 6.3) No se realizarán cambios sobre la versión de la API, que reside en el servidor Node.js ya implementado.

(RNF 6.4) La base de datos utilizada en general será MySQL excepto para las preguntas de tipo presencial, que se harán mediante consultas a la base de datos en tiempo real de Firebase.

### **3.2 Modelo de datos**

En esta sección se describe el modelo de datos utilizado en el proyecto. El número de entidades ha aumentado en uno con respecto a la versión original. Esto se debe a la nueva tabla llamada *admin*, que se encarga de recopilar todos los datos del nuevo rol de administrador.

La tabla tiene los mismos datos de entrada que su análoga *user*, excepto la clave primaria que es *idAdmin*. Como se puede observar en la Figura 4, no es necesario establecer ninguna relación de esta entidad con ninguna otra.

El resto de cambios realizados en el modelo se corresponden únicamente con nuevos campos en la tabla *Question*, que describiremos a continuación:

*numAnswers*: Campo utilizado para el **RF21** del proyecto que corresponde a permitir preguntas con número de respuestas variable. Almacena el número de respuestas que contiene la pregunta.

*homework*: Campo utilizado para saber si una pregunta es de tipo presencial (*homework* = 0) o de tarea para casa (*homework* = 1). Se utiliza para el **RF22** del proyecto que corresponde a las preguntas presenciales o de tarea.

*explanation*: Este campo se introdujo durante la fase de pruebas. Es utilizado para almacenar la explicación de la respuesta que da el profesor.

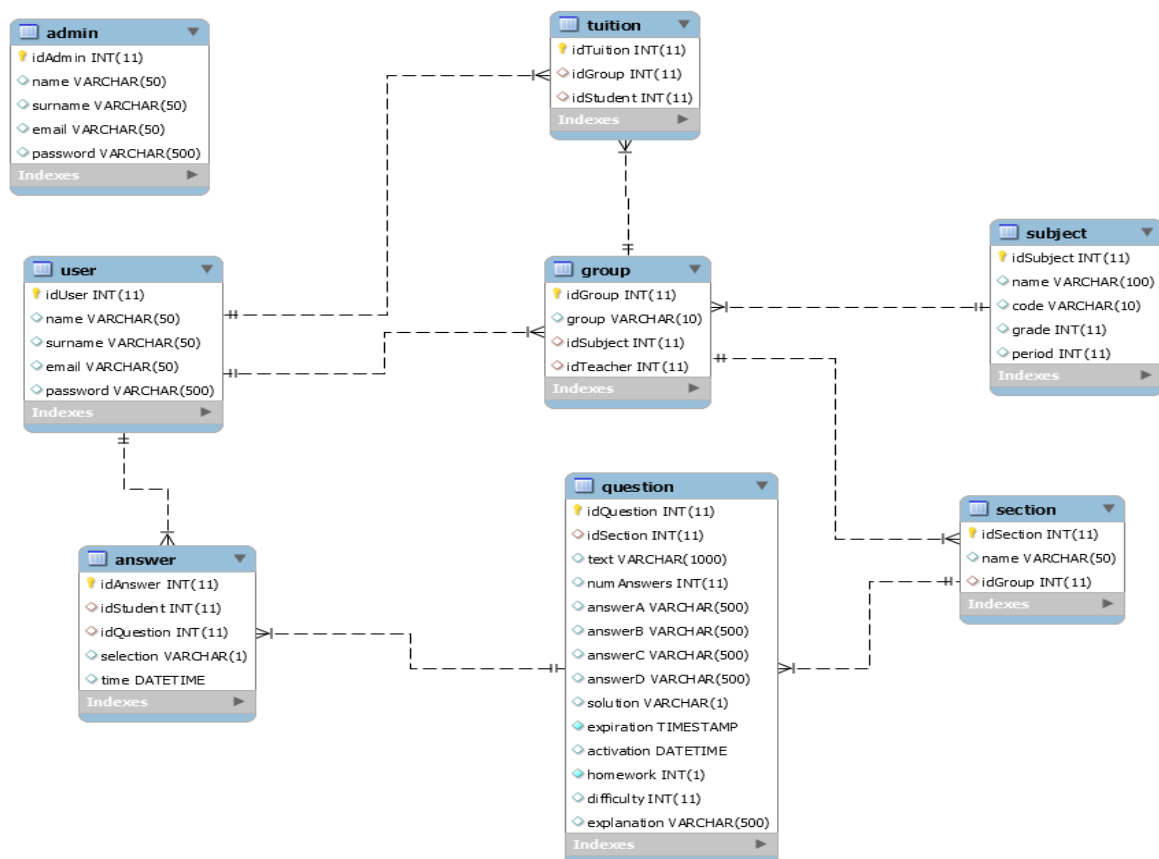


Figura 4: Modelo de datos

## 4 Desarrollo

---

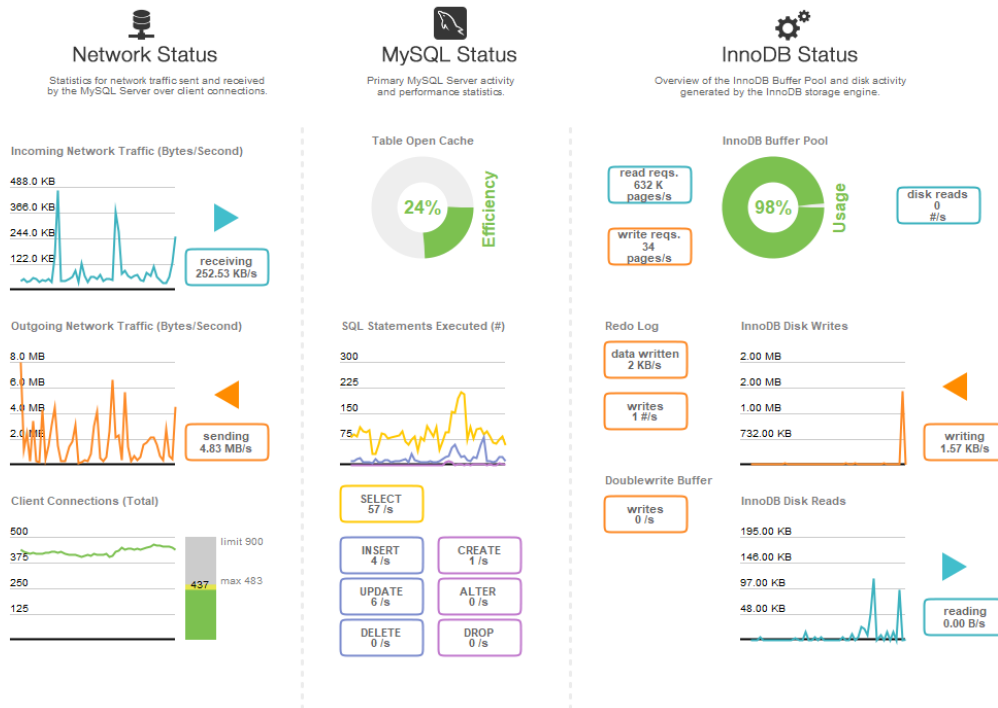
En esta sección describiremos los distintos desarrollos que se han llevado en ambas aplicaciones, así como la estructura de las mismas. Es importante recordar que las dos aplicaciones parten de una primera versión implementada en un TFG ya defendido, por lo que solo se mencionarán los detalles de estructura necesarios para la nueva versión. La aplicación móvil se desarrolló en un IDE llamado Android Studio, y seguiremos utilizándolo para este trabajo. Para la web, también seguiremos utilizando Sublime Text, una herramienta de texto muy útil para lectura y escritura de código.

### 4.1 Aplicación Web

#### 4.1.1 Conexión

Lo primero que se observó a la hora de realizar pruebas de la versión de partida (v1.0.1), es que las consultas que realizaba a la base de datos MySQL resultaban lentas y cuando accedía más de un profesor, solía cerrar la conexión por error. Esto se debía a la implementación de MySQL en el API ubicado en *server.js* por defecto, que no daba soporte efectivo a múltiples conexiones. La conexión se realiza ahora mediante la sentencia que podemos ver en el **Anexo D – Fragmento de código 1**.

El método *mysql.createPool* establece un grupo de conexiones simultáneas entre los clientes y la base de datos. Originalmente se utilizaba el método *mysql.createConnection*, cuya función se basaba en abrir una conexión por cliente, y cerrarla tras la consulta. Con esta nueva implementación, se establece un grupo que va dejando entrar a los clientes para ser atendidos y a su vez mantiene en espera a quienes aún no pueden consultar. Además, el límite de conexión se establece en 500, y en lugar de cerrar la conexión del cliente, la libera, de manera que la agrupación recupera la conexión para que pueda ser reutilizada más tarde. Con esta implementación, la aplicación ha mostrado mayor rapidez, estabilidad y rendimiento, como se puede observar en el gráfico estadístico que proporciona MySQL Workbench (Figura 5):



**Figura 5: Gráfico estadístico de actividad en vivo MySQL**

Por último, y antes de entrar en el siguiente apartado, vamos a observar el código de una consulta simple, ya que debido al cambio anterior, hay que añadir alguna corrección (**ver Anexo D – Fragmento de código 2**).

La principal característica de estas consultas es el método *connection.release*, cuyo objetivo es liberar la conexión que tiene un cliente en el momento en que ya ha realizado la consulta, de manera que esta se reutiliza volviendo a estar disponible para otro cliente en el grupo de conexiones.

#### 4.1.2 Seguridad

En este aspecto la aplicación no contenía fallas de seguridad graves a priori, ya que el servidor que alojaba las aplicaciones llamado Heroku [\[15\]](#) sólo admitía conexiones seguras mediante el protocolo HTTPS.

Sin embargo, ante un posible robo de la base de datos, las contraseñas de los usuarios serían fácilmente descifrables debido al algoritmo utilizado, SHA-1, sobre todo tras el 23 de Febrero de 2017, cuando un equipo formado por Google y CWI Ámsterdam anunciaron la primera colisión del algoritmo, llamada SHAttered [\[16\]](#).

Por ello, se optó por modificar el algoritmo de cifrado para las contraseñas, y se cambió a HMAC-SHA1, cuya definición general HMAC es la siguiente:

$$HMAC(K, m) = H\left((K \oplus opad) \parallel H((K \oplus ipad) \parallel m)\right)$$

Para no entrar en detalles de criptografía, vamos a centrarnos en que la fuerza criptográfica de este algoritmo depende del tamaño de clave de cifrado ( $K$ ) que se utilice, es decir, ante un hipotético robo de datos, los piratas informáticos tendrían que conocer la clave de cifrado para poder conseguir las contraseñas, y es la elección de la clave lo que hace más o menos seguro al sistema. **(Ver Anexo D – Fragmentos de código 3, 4 y 5 como ejemplo de implementación del cifrado HMAC-SHA1 tanto en web como en móvil).**

### 4.1.3 Interfaz de administración

Es uno de los gruesos del trabajo por la cantidad de código, y aún en conjunto el total de los requisitos funcionales descritos en la fase de análisis y diseño, a excepción de los dos últimos, que explicaremos en apartados separados debido a su complejidad.

**El primer objetivo** fue crear un nuevo módulo similar a *mainMenu*, llamado *mainMenuAdmin* (reciclando los estilos LESS [\[17\]](#) ya desarrollados), cuyo controlador de nombre *controller.js* se encarga de manejar el comportamiento de la página principal *mainMenu.html* y de cada una de las páginas subyacentes contenidas en la carpeta *childViews*; y *main.js* de estructurar los diferentes estados. Fragmentos de los ficheros en formato HTML junto a su funcionalidad que se describirán a continuación aparecen en el **Anexo D, desde el 6 al 18**, y en el **Anexo E como capturas de pantalla de la aplicación**. Los nombres de estos ficheros no son los mismos que el nombre de los enlaces que se utilizan en la aplicación.

A modo de bienvenida a la aplicación web tenemos *welcome*, independientemente de la plataforma (profesorado o administración) a la que accedamos. En ella hay un F.A.Q con las preguntas más frecuentes, enlaces a videotutoriales y publicación de las últimas novedades cuyo objetivo es mantener informado al usuario. Pero quien se encarga de la interfaz principal de administración es *mainMenu*, ya que contiene el diseño de los paneles laterales y el panel horizontal para ir desplazándose por la aplicación. Si nos adentramos en el **panel lateral izquierdo**, observamos distintos enlaces que nos llevan a páginas de edición rápida como *editAdmin*, dedicada a modificar los datos de un administrador, como el nombre, apellidos, correo y contraseña. Para editar el grupo tenemos *editGroup*, utilizada para actualizar los datos de un grupo, como las siglas, la asignatura que se imparte, y su profesor. También podemos editar asignaturas con *editSubject*, que sirve para modificar el nombre, las siglas, el curso y el cuatrimestre; y modificar los temas de un grupo mediante *editSection*, donde sólo es editable el nombre. Por último, existe la página análoga a *editAdmin* llamada *editUser*, utilizada para modificar los datos de un profesor o estudiante (*User*).

En el **panel lateral derecho** también tenemos enlaces a páginas con otras funcionalidades interesantes, como *editProfileAdmin*, utilizada para que el administrador que ha iniciado sesión pueda actualizar sus datos cómodamente. Otra opción muy interesante es *manageCSV*, que podemos usar para importar datos a partir de ficheros en formato CSV. Los datos a importar contemplan todas las entidades de la base de datos.

Por último, en el **panel horizontal o barra de navegación**, la funcionalidad se distribuye entre las pestañas. Cada una de ellas enlaza a una página destinada a una función diferente. Empezamos con *manageEnrollment*, sección dedicada al estudiante, donde se lleva a cabo la matriculación y anulación de matrícula del mismo, su registro en la plataforma, modificación de sus datos siguiendo el modelo de *editUser* y eliminación de los datos del estudiante.

Para grupos tenemos *manageGroup*, donde se puede crear un grupo a partir de las siglas, asignatura y profesor. También permite modificar los datos de uno existente o eliminarlo por completo. Después, tenemos *manageSubjectSection*, que contiene registro, edición y eliminación de asignaturas y temas. Sin embargo, en la web están divididos con ayuda de la clase *ng-show*, mostrando un formulario HTML distinto según la pestaña que se seleccione. Por último, tenemos *manageUserAdmin*, que además de juntar la funcionalidad de *editAdmin* y *editUser*, permite registros y eliminaciones de ambos.

Aunque no se haya creado una página específica en formato HTML para la siguiente funcionalidad, es importante tenerla en cuenta. En este caso se hace únicamente uso del fichero *controller.js*, y en la interfaz es un enlace llamado **Resetear DB**, y se encarga de eliminar todos los datos de la aplicación y restaurar únicamente las asignaturas que existen en la carrera por defecto.

**El segundo objetivo** es lograr el comportamiento diseñado en la interfaz descrita mediante las consultas a la base de datos. Todas ellas se encuentran en el fichero *server.js*, que reúne un total de 91 consultas partiendo de las 23 consultas originales, todas ellas mediante los métodos GET y POST del protocolo HTTP.

Todas las consultas se han ido declarando en la carpeta *services*, desde donde se lanzaban las llamadas HTTP (**ver Anexo D Fragmento 19 para ejemplo de llamada**) en cada uno de los modelos de las entidades.

#### 4.1.4 Interfaz de profesorado

Además de desarrollar una interfaz de administración con las características ya descritas, también se ha mejorado la plataforma de profesorado, añadiendo nuevas funcionalidades, resolviendo incidencias (**ver Anexo C – Historial de versiones e incidencias resueltas**) y mejorando las condiciones de los formularios existentes. Vamos a describir las características más destacables que se han implementado.

##### 4.1.4.1 Exportar pregunta

Cuando estudiamos en el estado del arte las características de las aplicaciones similares a ClicEPS, se repetía continuamente una funcionalidad muy interesante: poder compartir los cuestionarios con el resto de profesores.

De ahí surgió la característica de proyecto de exportar pregunta, cuyo funcionamiento es simple: los profesores, desde la sección “Nueva pregunta” pueden acceder a una subsección llamada “Exportar Pregunta”. En ella, estos podrán seleccionar una pregunta de entre todas las creadas en la aplicación, filtradas como se aprecia en la Figura 6.

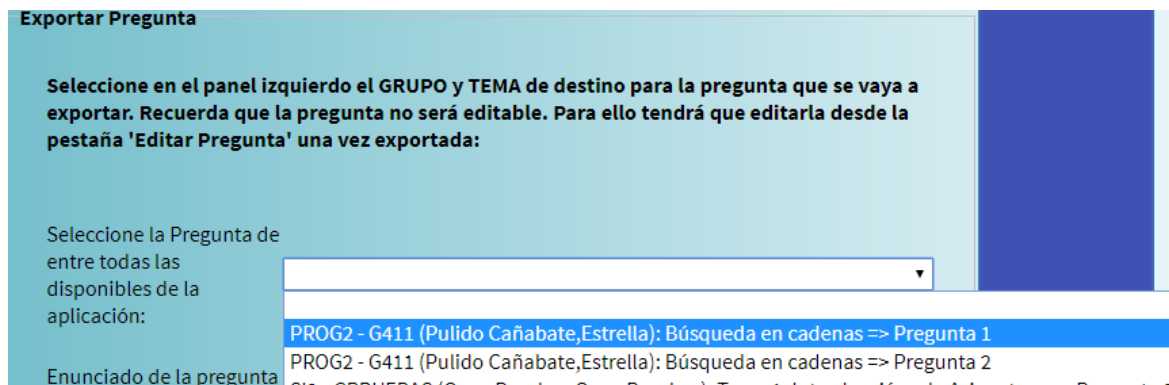


Figura 6: Captura de pantalla de exportar pregunta

Como se observa en la figura, el selector de preguntas las filtra con la estructura “*ASIGNATURA – GRUPO (PROFESOR): TEMA => PREGUNTA NÚMERO*”

Para realizar el formulario, se ha seguido la estructura utilizada en *editQuestion*, haciendo uso de la clase *ng-model* para poder representar los datos de la pregunta a exportar a partir de un objeto en Javascript declarado en *controller.js*. En el **Anexo D** aparece a modo de ejemplo el **fragmento de código 20** correspondiente a la carga de los datos para ser representados en el formulario una vez elegida la pregunta.

Centrándonos en el *div* de clase *new-question-select* podemos observar cómo las preguntas se filtran por el objeto *vm.allQuestions* mediante la clase *ng-options* y el campo de cada pregunta *title*. Este filtrado lo utilizamos en todos los formularios de la interfaz de administración que utilizan algún *select* para la carga de datos.

Por otra parte, si observamos el *div* de clase *new-question-text-content* que corresponde con el enunciado de la pregunta a exportar, vemos como el objeto que muestra mediante la clase *ng-model* es *vm.exportQuestion.text*. Esto es porque el objeto *vm.exportQuestion* es el objeto que representa la pregunta seleccionada en el *select*, que contiene todos los datos de la pregunta, los cuales se cargarán y mostrarán en el formulario para que el profesor finalmente decida si quiere exportar la pregunta seleccionada o no.

Para acabar con esta sección y sirviendo a modo de ejemplo para la funcionalidad descrita en la interfaz de administración (ya que hace uso de funciones de carga similares, solo que con distinta entidad) el **Anexo D (fragmento de código 21)** muestra cómo cargan las preguntas al objeto *vm.allQuestions*.

Si analizamos el código, concluiremos que a partir del servicio *QuestionDataSer* se hace una consulta a la base de datos (*getQuestionsData*) para recibir posteriormente en *vm.allQuestions* todas ellas y poder mapearlas de una en una en la variable *q* mediante el modelo *QuestionDataModel.QuestionData(question)*. Una vez modelada la pregunta y guardada en dicha variable, se devuelve para insertarla en el Array.

#### 4.1.4.2 Estadísticas y pase de lista

Se ha modificado la sección de “Estadísticas por pregunta” para añadir una subsección que muestre los estudiantes que no han contestado a la pregunta, y modificar el gráfico circular para representar ese porcentaje y separarlo del resto de opciones.

En el **Anexo E – Figura 47** podemos contemplar el resultado del gráfico circular con el nuevo porcentaje. Ahora muestra el tanto por ciento de estudiantes que no han respondido a la pregunta. Para implementar esta nueva característica hemos hecho uso de *controller.js* de nuevo, mediante la variable *vm.afkStudents* que utilizamos en la función *getAfkStudentsOfQuestion* (ver **Anexo D – Fragmento de código 22**) para luego actualizar el *chart* mediante el método *calculateQuestionRates* (ver **Anexo D – Fragmento de código 23**) y posteriormente plasmar este en el fichero HTML *questionStats* (ver **Anexo D – Fragmento de código 24**)

Por último, resulta sencillo después de esto mostrar una lista con los nombres de los alumnos que faltasen a clase, ya que tenemos la variable *vm.afkStudents* con toda su información. En el **Anexo D – Fragmento de código 25**, se observa el *div* que genera la lista de estudiantes ausentes y los filtra por nombre y apellidos (contenido en variable *title*)

#### 4.1.4.3 Actualizar la contraseña

Este cambio resultó muy sencillo de añadir también en la aplicación Android gracias al uso de los *webView* [18] que ofrece Android Studio. Los *webView* son complementos que se añaden a la interfaz y que muestran el contenido de una página web.

En la aplicación web añadimos un nuevo módulo llamado *updatePassword* que contiene un *controller.js* capaz de manejar dos páginas HTML, *updatePassword* y *updatePasswordAdmin* (ver **Anexo D – Fragmento de código 26 y 27**).

La diferencia con el resto de módulos es la forma de acceder a esta sección de la web, ya que se puede entrar a ella sin necesidad de iniciar sesión en la plataforma web, de ahí el poder utilizarlo para Android. La dirección en el servidor Heroku de ejemplo es <http://serene-caverns-11982.herokuapp.com/#/updatePassword>

Por último, se añadió el complemento *webView* en Android durante la fase de pruebas como mejora para permitir actualizar las contraseñas desde los dispositivos móviles en la actividad de iniciar sesión.

#### 4.1.4.4 Crear tema

Consiste en añadir una página equivalente a *manageSubjectSection* implementada en la interfaz de administración. Con esto se quiere permitir al profesor poder crear o borrar el temario de su grupo sin depender de un administrador. En el **Anexo D – Fragmentos de código 28 y 29**, podemos ver un ejemplo de cómo crear y borrar un tema desde *controller.js*, y en el **Anexo E – Figura 48**, podemos ver una captura resultado de la sección “Tema” en la plataforma de profesorado.

Las funciones de crear y borrar tema son las mismas que utiliza el *controller.js* de la interfaz de administración para su particular *manageSubjectSection*.

## 4.2 Aplicación Móvil

En esta sección describiremos las implementaciones llevadas a cabo en la aplicación móvil, recogidas en los requisitos durante la fase de análisis y diseño. Tras las pruebas



también se realizaron mejoras sobre la aplicación que pueden consultarse en el siguiente capítulo. (Para ver capturas de pantalla de la aplicación móvil, consultar **Anexo E**).

### 4.2.1 Preguntas presenciales o de tarea

Teniendo en cuenta que en la plataforma web se tuvieron que realizar modificaciones en los formularios de *Question* para incorporar un nuevo campo llamado *homework* que determina si una pregunta es presencial o no, el mayor cambio se llevó a cabo en la aplicación Android. Vamos las modificaciones más destacables y que más tiempo han llevado.

#### 4.2.1.1 MapsActivity

*MapsActivity* es la actividad que maneja toda la lógica del estudiante y las actividades presenciales que éste tiene. Se encarga de cargar un mapa en la pantalla, en el cual se señala en cada momento la ubicación del estudiante y, en caso de estar en la zona presencial, se acaba cerrando la misma para devolver un resultado a la actividad *QuestionActivity* que comprobará mediante el método *onActivityResult*.

¿Cómo detecta que el estudiante está en la zona presencial? Mediante lo que se conoce como *geofences*. Un *geofence* en Android es un área del mapa sensible a eventos en el momento en que se interactúa con ella (**ver Anexo D – Fragmento de código 30**).

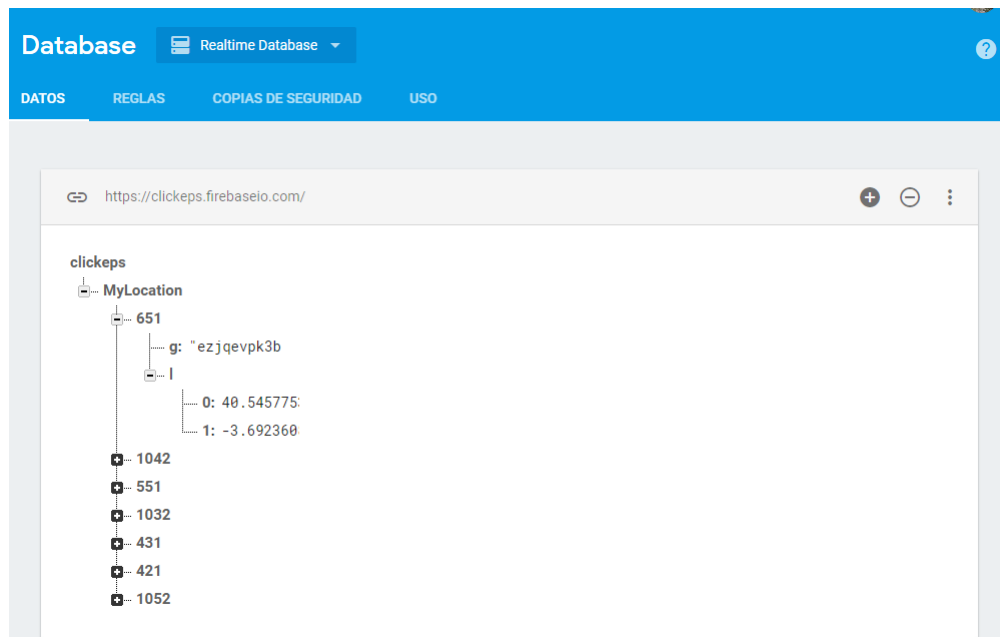
Observando el fragmento de código anterior, podemos ver cómo en la actividad se implementan métodos que tienen que ver con eventos del mapa, como cuando está disponible (*OnMapReady* que corresponde al momento en que dibujamos el área en el mapa), cuando el estudiante accede al interior del área (*onKeyEntered*), cuando el estudiante sale del interior del área (*onKeyExited*), o cuando el estudiante se mueve (*onKeyMoved*).

Sin embargo y como último paso, para utilizar los *geofences* necesitamos conectarnos a *GooglePlayServices* (**ver Anexo D – Fragmento de código 31**), acceder a la API de Google (**ver Anexo D – Fragmento de código 32**), y así crear ubicaciones y desplegarlas con el uso de Firebase que veremos en el siguiente apartado.

#### 4.2.1.2 Firebase

Firebase es un conjunto de funciones y servicios que ayudan a crear mejores apps para dispositivos móviles permitiendo rápidas compilaciones y ofrecer bases de datos en tiempo real, consultando fácilmente los datos desde la consola del sitio web.

En *MapsActivity*, se creó un objeto de tipo *geoFire*, complemento que utiliza Firebase para conseguir las ubicaciones en tiempo real, haciendo referencia a nuestro objeto raíz en la base de datos global llamado *MyLocation* (**ver Anexo D – Fragmento de código 33**). Este nodo raíz guardará las ubicaciones de los usuarios que vayan accediendo. En la Figura 7 cada hijo de *MyLocation* corresponde al *id* del estudiante, el nodo hoja *g*, una variable de filtro aleatoria, y el nodo *l* contiene latitud (*0*) y longitud (*1*).



**Figura 7: Captura de la base de datos en Firebase**

Toda base de datos se rige por unas reglas, y las reglas de ésta se pueden consultar en el **Anexo D – Fragmento de código 34**.

La mecánica es sencilla: en el momento en que un estudiante entra en una pregunta presencial se inicia *MapsActivity*, se declara el objeto *geoFire*, un objeto hijo con el id del estudiante donde meteremos la ubicación, y se comienzan a registrar ubicaciones mediante la API de Google. Estas ubicaciones se suben en tiempo real a la base de datos Firebase mediante el método *geoFire.setLocation* (ver **Anexo D – Fragmento de código 35**).

Por último, el método utilizado para comprobar si el estudiante está en el área definida por el *geofence* es mediante un *Callback*, concretamente un *LocationCallback* que podemos ver implementado en el **Anexo D – Fragmento de código 36**. Este *callback* es llamado a partir de la sentencia: *geoFire.getLocation (userId, comprobarLocation)*. Cuando se activa, el método *onLocationResult* recoge la ubicación actual y la clave del estudiante. Haciendo una comprobación de distancia respecto al centro del *geofence*, determinamos finalmente si el estudiante se encuentra en el área presencial, en cuyo caso mandamos un *Activity.RESULT\_OK* a *QuestionActivity*. En caso contrario, mantenemos el mapa abierto hasta recibir nuevas ubicaciones.

#### 4.2.1.3 Mock Locations

Esta característica que tienen los dispositivos Android permite al usuario falsear la información de ubicación que registra el GPS de su dispositivo, es decir, simula una ubicación falsa.

Lo que se ha desarrollado para que los estudiantes no puedan falsificar la ubicación y así no dar a entender que están en clase es un detector de aplicaciones ajenas a ClickEPS que utilizan esta característica. En el **Anexo D – Fragmento de código 37**, vemos la manera en que podemos detectar este tipo de acción fraudulenta. En el código se observa que

utilizamos un método llamado *onActivityResult* que, como indica el nombre, se ejecuta en cuanto recibimos el resultado de una actividad, en concreto *MapsActivity*. Si todo el proceso descrito en dicha actividad junto con Firebase se realiza de forma correcta, se llega al método en *QuestionActivity*.

Existen dos variables booleanas que recogen el resultado de dos métodos: *isMockSettingsON*, cuya implementación aparece en el **Anexo D – Fragmento de código 38** y *areThereMockPermissionApps* (**ver Anexo D – Fragmento de código 39**). El primer método se encarga de comprobar si la opción de “disfrazar ubicación” para desarrolladores está activada en el dispositivo. El segundo se encarga de ver si existe alguna aplicación ajena a ClickEPS que tenga permisos para falsear o simular la ubicación del GPS, mediante el permiso de Android *android.permission.ACCESS\_mock\_location*.

Finalmente, si una de las dos variables es *false*, la aplicación advertirá al estudiante de que tiene aplicaciones que simulan la ubicación y que debe eliminarlas para volver a utilizar ClickEPS.

#### 4.2.2 ClickEPSAdministracion

Se considera en la memoria como un cambio a nivel de aplicación móvil, pero ClickEPSAdministracion es una APK completamente independiente a ClickEPS, que pueden utilizar profesores y administradores para acceder a las plataformas web de profesorado y administración a partir de sus dispositivos Android. La aplicación utiliza un único recurso, que es un *webView* que emula el contenido web de ClickEPS. Además, tiene algunas mejoras para girar el dispositivo y hacer zoom (**ver Anexo D – Fragmento de código 42**).

#### 4.2.3 Dificultad de la pregunta

Un detalle que observamos en los resultados de las encuestas de la versión original es que los profesores no entendían el campo *difficulty* de la entidad *Question*, ya que no aportaba ningún tipo de información. Lo que hicimos fue darle una cabida visual en la aplicación móvil (**ver Anexo E – Figura 52**) para que así los estudiantes pudiesen ver qué dificultad tiene cada una de las preguntas. De esta manera, los profesores podrán evaluar con la puntuación que consideren oportuno las respuestas dependiendo de la dificultad que hayan escogido. Este campo se mantuvo en el formulario de crear y editar pregunta. Además se añadió en las estadísticas para que el profesor lo tenga siempre en cuenta.

#### 4.2.4 Optimizar la lista de preguntas

Por último y para completar esta fase, se optimizó la lista de preguntas presentadas en la actividad *QuestionListActivity*. Para ello, se añadió un botón que permitiera refrescar la lista de preguntas para evitar reiniciar la aplicación. Dicho cambio se puede ver en el **Anexo D – Fragmento de código 46**.

También se añadieron una serie de notificaciones para contemplar preguntas disponibles que no fueran visibles en la aplicación debido a que existiera un gran número y hubiera que hacer *scroll* (**ver Anexo D – Fragmento de código 47**).



## 5 Integración, pruebas y resultados

---

En este capítulo detallaremos cómo se ha llevado a cabo la integración de las aplicaciones web y Android en un entorno real. También describiremos cómo se realizó la fase de pruebas y cuál ha sido el resultado final del proyecto.

### 5.1 Integración en producción

Una vez finalizado el desarrollo de ambas aplicaciones, se pasó a organizar ambos proyectos (web y Android) para que fueran accesibles para todos los usuarios.

Para el proyecto web se decidió utilizar Heroku como plataforma en la nube para alojar nuestra web y así desplegarla con su API para hacer accesible toda la funcionalidad a la aplicación móvil. Como hemos mencionado en capítulos anteriores, la plataforma web sigue disponible en <https://serene-caverns-11982.herokuapp.com/>. Para conectar la base de datos se utilizó el servicio de Heroku llamado ClearDB MySQL [19] y se creó una base de datos exactamente igual a la utilizada en local, solo que al tratarse de un servicio gratuito, la característica de importar datos desde ficheros CSV no estaría disponible por los privilegios que ofrece el servicio de ClearDB MySQL.

A continuación fue necesario configurar las consultas de *server.js* para que hicieran referencia a la nueva base de datos con la configuración de conexión del proyecto en Android Studio. Por último, hubo que exportar cada APK que se quería publicar firmando con las credenciales correspondientes. Todas las APK se renombraron con el formato *añomesdia\_ClickEPS\_versión* o *añomesdia\_ClickEPSAdministracion\_version* (por ejemplo, 20180423\_ClickEPS\_v1.0.3.apk) para llevar un control de versiones. Los profesores tenían que acceder a una carpeta pública en Dropbox para posteriormente copiar la APK y publicarla en su grupo de Moodle. Todas las versiones se pueden consultar en el **Anexo C**.

### 5.2 Pruebas funcionales

#### 5.2.1 Aplicación móvil

La mejor forma de probar la aplicación es con los usuarios en un entorno real. Para probar la aplicación móvil se realizaron un total de **3 pruebas** en distintas clases de Programación II, empezando el **23 de Abril**. Antes de esta primera prueba, el profesor detectó errores en la plataforma web mientras creaba las preguntas y estos fueron reportados y solventados para la siguiente prueba. Esta consistía en que los alumnos contestasen a dos preguntas presenciales. En cuanto a los errores detectados durante esta prueba en la aplicación móvil, todos ellos tuvieron que ver con los **datos en tiempo real de Firebase** (impedía a los alumnos contestar a las preguntas presenciales) y **Mock Locations** (detectaba aplicaciones que no simulaban ubicación como que sí). En el primer caso, lo que ocurría es que las

ubicaciones se estaban sobrescribiendo en una única variable de tiempo real llamada **MyLocation**. La solución consistió en cambiar la mecánica de las reglas de la base de datos Firebase y crear, para cada estudiante que se conectaba, una variable con su identificador de usuario como clave. Cada una de ellas tendría su ubicación, y sólo se editaría tal ubicación en caso de que el alumno abriera el mapa. Para el segundo error detectado se realizó un estudio de qué aplicaciones podrían verse alteradas con el permiso de Android *android.permission.ACCESS\_mock\_location*. Sin embargo, se concluyó que las aplicaciones que detectaba sí podían simular ubicación (como por ejemplo, *FakeGPS*). La segunda prueba se realizó el **30 de Abril** y se realizó con un número inferior de alumnos respecto a la anterior para no saturar el ancho de banda del servidor gratuito en el que se aloja la aplicación. Fueron 5 alumnos que tuvieron que contestar tanto a preguntas presenciales como no presenciales y únicamente se detectó en un primer intento de responder las presenciales que **Mock Locations** detectaba una aplicación simulando sin que el estudiante tuviese nada sospechoso instalado. Al final se debió a que su versión de Android no era muy actual y dio un falso positivo, permitiendo en un segundo intento contestar al estudiante. La última prueba se realizó el **8 de Mayo** no sucediendo ninguna incidencia y se solicitó a los estudiantes contestar una encuesta de evaluación de la aplicación.

### 5.2.2 Aplicación web

Para la realización de estas pruebas se crearon videotutoriales específicos para profesores y administradores, cuyos enlaces pueden consultarse en el **Anexo A**. Estos vídeos explican con detalle cada una de las funcionalidades de la plataforma, con ayuda de ejemplos prácticos.

Con estos vídeos se envió un correo a varios profesores indicándoles la importancia de verlos para probar la aplicación. A continuación, se les asignó un grupo desde la plataforma de administración y ellos crearon los temas y las preguntas, respondiendo finalmente a la encuesta destinada a los profesores, cuyos resultados pueden consultarse en el **Anexo B**.

Para la administración se llevaron a cabo reuniones para comprobar la usabilidad junto con la tutora que más se adaptara a las necesidades del proyecto.

## 5.3 Pruebas unitarias

La plataforma web está compuesta de numerosos formularios con entradas de datos para formar los objetos que sirven para realizar las consultas a la base de datos y así completar toda la información del sistema. A lo largo del control de versiones de ClicEPS que puede consultarse en el **Anexo C** se fueron realizando pruebas unitarias que servían para comprobar que cada entidad funciona correctamente por separado. A continuación podemos observar unas tablas que recogen todas las pruebas unitarias que se realizaron en la versión web.

Entidad	Formulario	Campos	Pruebas	Salida	Resultado
Grupo	Crear	* Siglas del grupo	* Crear un grupo que ya existe	* Error: El grupo que intenta crear ya existe con los datos	OK

		* Asignatura * Profesor	* Dejar uno de los campos vacío	introducidos * No permite completar el formulario	
	Editar	* Grupo a modificar * Siglas del grupo * Nueva asignatura * Nuevo profesor	* Modificar el grupo a otro que ya existe * Dejar uno de los campos vacío	* El grupo que intenta editar ya existe con los nuevos datos introducidos * No permite completar el formulario	OK
	Eliminar datos	* Grupo a eliminar	* Dejar el campo vacío	* No permite completar el formulario	OK

Tabla 2: Pruebas unitarias de la plataforma web (Grupo)

Entidad	Formulario	Campos	Pruebas	Salida	Resultado
Estudiante	Matricular	* Estudiante a matricular * Opción: Matricular * Grupo en el que se matriculará	* Intentar matricular al estudiante en un grupo en el que ya esté ya matriculado * Dejar uno de los campos vacío	* No permite matricularse en grupos ya matriculados * No permite completar el formulario	OK
	Anular matrícula	* Estudiante para anulación * Opción: Anular matrícula * Grupo en el que se anulará su matrícula	* Intentar anular la matrícula de un grupo en el que no esté matriculado * Dejar uno de los campos vacío	* No permite anular la matrícula de grupos en los que no esté matriculado * No permite completar el formulario	OK
	Registrar	* Nombre * Apellidos * Correo institucional * Contraseña	* Registrar un estudiante que ya existe * Registrar un estudiante con un correo que ya existe * Dejar uno de los campos vacío * Utilizar un correo no institucional	* Error: El nombre y apellidos ya están en uso * Error: La dirección de correo ya está en uso * No permite completar el formulario * Error: Para registrar un usuario (ya sea estudiante o profesor), el correo debe ser institucional (@estudiante.uam.es o @uam.es)	OK

	Editar	<ul style="list-style-type: none"> <li>* Estudiante a editar</li> <li>* Nombre</li> <li>* Apellidos</li> <li>* Correo institucional</li> <li>* Contraseña</li> </ul>	<ul style="list-style-type: none"> <li>* Editar un estudiante con los datos de otro que ya existe</li> <li>* Editar un estudiante con un correo que ya existe</li> <li>* Dejar uno de los campos vacío</li> <li>* Utilizar un correo no institucional</li> </ul>	<ul style="list-style-type: none"> <li>* Error: El nombre y apellidos ya están en uso</li> <li>* Error: La dirección de correo ya está en uso</li> <li>* No permite completar el formulario</li> <li>* Error: Para editar un usuario (ya sea estudiante o profesor), el correo debe ser institucional (@estudiante.uam.es o @uam.es)</li> </ul>	OK
	Actualizar contraseña	<ul style="list-style-type: none"> <li>* Estudiante a editar</li> <li>* Contraseña</li> </ul>	<ul style="list-style-type: none"> <li>* Dejar uno de los campos vacío</li> </ul>	<ul style="list-style-type: none"> <li>* No permite completar el formulario</li> </ul>	OK
	Eliminar datos	<ul style="list-style-type: none"> <li>* Estudiante a eliminar</li> </ul>	<ul style="list-style-type: none"> <li>* Dejar el campo vacío</li> </ul>	<ul style="list-style-type: none"> <li>* No permite completar el formulario</li> </ul>	OK

Tabla 3: Pruebas unitarias de la plataforma web (Estudiante)

Entidad	Formulario	Campos	Pruebas	Salida	Resultado
Asignatura	Crear	<ul style="list-style-type: none"> <li>* Código</li> <li>* Nombre</li> <li>* Siglas</li> <li>* Curso</li> <li>* Cuatrimestre</li> </ul>	<ul style="list-style-type: none"> <li>* Crear una asignatura con un código que ya existe</li> <li>* Dejar uno de los campos vacío</li> </ul>	<ul style="list-style-type: none"> <li>* Error: La asignatura que intenta crear ya existe con el código introducido</li> <li>* No permite completar el formulario</li> </ul>	OK
	Editar	<ul style="list-style-type: none"> <li>* Asignatura a editar</li> <li>* Nombre</li> <li>* Siglas</li> <li>* Curso</li> <li>* Cuatrimestre</li> </ul>	<ul style="list-style-type: none"> <li>* Dejar uno de los campos vacío</li> <li>* Escribir un curso distinto de 1,2,3 o 4</li> <li>* Escribir un semestre distinto de 1 o 2</li> </ul>	<ul style="list-style-type: none"> <li>* No permite completar el formulario</li> <li>* El curso de la asignatura debe estar contenido entre 1 y 4, ambos incluidos</li> <li>* El período de la asignatura debe estar contenido entre 1 y 2, ambos incluidos</li> </ul>	OK
	Eliminar datos	<ul style="list-style-type: none"> <li>* Asignatura a eliminar</li> </ul>	<ul style="list-style-type: none"> <li>* Dejar el campo vacío</li> </ul>	<ul style="list-style-type: none"> <li>* No permite completar el formulario</li> </ul>	OK

Tabla 4: Pruebas unitarias de la plataforma web (Asignatura)

Entidad	Formulario	Campos	Pruebas	Salida	Resultado
Tema	Crear	<ul style="list-style-type: none"> <li>* Nombre</li> <li>* Grupo al que se le asigna</li> </ul>	<ul style="list-style-type: none"> <li>* Dejar uno de los campos vacío</li> </ul>	<ul style="list-style-type: none"> <li>* No permite completar el formulario</li> </ul>	OK
	Editar	<ul style="list-style-type: none"> <li>* Tema a editar</li> <li>* Nombre</li> </ul>	<ul style="list-style-type: none"> <li>* Dejar uno de los campos vacío</li> </ul>	<ul style="list-style-type: none"> <li>* No permite completar el formulario</li> </ul>	OK
	Eliminar datos	<ul style="list-style-type: none"> <li>* Tema a eliminar</li> </ul>	<ul style="list-style-type: none"> <li>* Dejar el campo vacío</li> </ul>	<ul style="list-style-type: none"> <li>* No permite completar el formulario</li> </ul>	OK

Tabla 5: Pruebas unitarias de la plataforma web (Tema)



Entidad	Formulario	Campos	Pruebas	Salida	Resultado
Administrador/Profesor	Registrar	* Nombre * Apellidos * Correo institucional * Contraseña	* Registrar un profesor o administrador con un correo que ya existe * Dejar uno de los campos vacío	* Error: La dirección de correo ya está en uso  * No permite completar el formulario	OK
	Editar	* Administrador o profesor a editar * Nombre * Apellidos * Correo institucional	* Editar un profesor o administrador con un correo que ya existe * Dejar uno de los campos vacío	* Error: La dirección de correo ya está en uso  * No permite completar el formulario	OK
	Actualizar contraseña	* Administrador o profesor a editar * Contraseña	* Dejar uno de los campos vacío	* No permite completar el formulario	OK
	Eliminar datos	* Administrador o profesor a eliminar	* Dejar el campo vacío	* No permite completar el formulario	OK

Tabla 6: Pruebas unitarias de la plataforma web (Administrador/Profesor)

Entidad	Formulario	Campos	Pruebas	Salida	Resultado
Pregunta	Nueva Pregunta	* Enunciado * Dificultad * Número de respuestas * Deberes o Presencial * Respuestas * Respuesta correcta * (Opcional) Justificación de la respuesta	* Dejar uno de los campos vacíos * Introducir fechas de expiración y activación incoherentes * No seleccionar grupo y tema donde crear	* No permite completar el formulario * Error: La fecha de activación no puede ser posterior (o igual) a la fecha de expiración * Error: Debe seleccionar un grupo y un tema de destino en el panel de la izquierda	OK
	Exportar Pregunta	* Pregunta a exportar * Campos de la pregunta bloqueados	* No seleccionar ninguna pregunta * No seleccionar grupo y tema donde exportar	* Error: Debe seleccionar una pregunta para exportar * Error: Debe seleccionar un grupo y un tema de destino en el panel de la izquierda	OK

	Editar Pregunta Eliminar Pregunta Duplicar Pregunta	* Enunciado * Dificultad * Número de respuestas * Deberes o Presencial * Respuestas * Respuesta correcta * (Opcional) Justificación de la respuesta	* Dejar uno de los campos vacíos * Introducir fechas de expiración y activación incoherentes * No seleccionar grupo y tema donde crear	* No permite completar el formulario * Error: La fecha de activación no puede ser posterior (o igual) a la fecha de expiración * Error: Debe seleccionar un grupo y un tema de destino en el panel de la izquierda	OK
--	--	---	--	--	----

Tabla 7: Pruebas unitarias de la plataforma web (Pregunta)

## 5.4 Mejoras en el proyecto

A medida que se realizaban las pruebas anteriormente descritas en las aplicaciones surgían nuevas ideas y mejoras que aumentaban las prestaciones del proyecto y añadían características muy interesantes para alumnos y docentes. Todas las ideas y mejoras que se realizaron fueron previamente recogidas como *peticiones de cambio* que se describen en los siguientes apartados.

### 5.4.1 Recordar correo y contraseña para iniciar sesión

<b>PC-01</b>	<b>Recordar correo y contraseña para iniciar sesión</b>
<b>Proyecto/s y versión</b>	ClickEPS (Android) v1.0.2
<b>Fecha</b>	22/01/2018
<b>Descripción</b>	Cada vez que se accede a la aplicación móvil hay que introducir el correo y la contraseña, ya que la plataforma no recuerda nunca las credenciales.
<b>Impacto directo</b>	Únicamente afectaría a la actividad del proyecto Android que contiene el formulario de iniciar sesión, llamado <i>LoginActivity</i> .

Tabla 8: Recordar correo y contraseña para iniciar sesión

Esta característica ofrece la posibilidad de recordar el correo y contraseña del estudiante cada vez que éste accede a la aplicación móvil, y en Android se conoce como *Shared Preferences*. Su implementación resulta muy sencilla y reduce el tiempo de acceso del estudiante a ClickEPS. La actividad que se modificó fue *LoginActivity* (ver **Anexo D – Fragmentos de código 40 y 41**).

Se decidió aprobar debido a las valoraciones en la encuesta realizada en la versión de partida v1.0.1, donde manifestaban la inconveniencia de tener que introducir el correo y la contraseña cada vez que querían iniciar sesión en la aplicación móvil.

#### 5.4.2 Actualizar la contraseña desde la aplicación móvil

<b>PC-02</b>	<b>Actualizar la contraseña desde la aplicación móvil</b>
<b>Proyecto/s y versión</b>	ClickEPS (Android y web) v1.0.2
<b>Fecha</b>	29/01/2018
<b>Descripción</b>	Permitir actualizar la contraseña del estudiante desde la aplicación móvil, ya que actualmente no pueden y el administrador les registra por normativa con el NIA como su contraseña.
<b>Impacto directo</b>	Se necesita modificar ambas aplicaciones: la aplicación móvil para introducir el enlace a la sección, y la aplicación web para implementar el formulario que consultará a la base de datos, que será reutilizado del RF10

**Tabla 9: Actualizar la contraseña desde la aplicación móvil**

Esta mejora permitiría a los estudiantes elegir la contraseña que les parezca más cómoda y sencilla para ellos, ya que por defecto el administrador registra a los estudiantes con sus NIA como contraseña.

#### 5.4.3 No mostrar la respuesta hasta que no finalice la pregunta

<b>PC-03</b>	<b>No mostrar la respuesta hasta que no finalice la pregunta</b>
<b>Proyecto/s y versión</b>	ClickEPS (Android) v1.0.2
<b>Fecha</b>	30/01/2018
<b>Descripción</b>	Este cambio consiste en impedir visualizar la respuesta hasta que finalice la pregunta.
<b>Impacto directo</b>	Afecta directamente a las actividades del proyecto Android relacionadas con las preguntas, llamadas <i>QuestionActivity</i> , <i>QuestionItemAdapter</i> y <i>QuestionListActivity</i>

**Tabla 10: No mostrar la respuesta hasta que no finalice la pregunta**

Actualmente cuando un estudiante contesta a una pregunta le aparece al instante la respuesta correcta y si ha fallado o no. Esto a nivel académico supone un problema si lo que queremos es examinar mediante la aplicación, porque se pueden comentar las soluciones en clase o en casa mientras la pregunta sigue activa (el tiempo de expiración no ha finalizado). Con esta propuesta el problema se evitaría pues los estudiantes podrían visualizar el resultado únicamente cuando el tiempo para responder la pregunta hubiera terminado.

Para implementar esta funcionalidad, se utiliza el campo *selection* de *Question* que contiene la respuesta del estudiante. Si el campo está vacío o es nulo, la pregunta no ha

sido contestada. Esto se guarda en una variable booleana llamada *answered* que, junto con los tiempos de expiración y activación, forman las condiciones para mostrar o no las respuestas (ver Anexo D – Fragmento de código 43).

#### 5.4.4 Preguntas con un número variable de respuestas

<b>PC-04</b>	<b>Preguntas con un número variable de respuestas</b>
<b>Proyecto/s y versión</b>	ClicEPS (Android y web) v1.0.3
<b>Fecha</b>	15/02/2018
<b>Descripción</b>	Este cambio consiste en que el número de respuestas de una pregunta no sea siempre cuatro (A, B, C o D) permitiendo crear preguntas con dos respuestas (A o B) y tres respuestas (A, B, o C)
<b>Impacto directo</b>	Afecta directamente a las actividades del proyecto Android relacionadas con las preguntas, llamadas <i>QuestionActivity</i> , <i>QuestionItemAdapter</i> y <i>QuestionListActivity</i> . En la aplicación web habría que añadir un campo en la tabla <i>Question</i> de MySQL llamado <i>numAnswers</i> que contemplara el nuevo aspecto y adaptar los formularios y el fichero <i>controller.js</i>

**Tabla 11: Preguntas con un número variable de respuestas**

Como se menciona en la mejora, ésta consistía en variar algunas partes del *controller.js* de la interfaz de administración del proyecto web. Las respuestas que no se elijan reciben como contenido el valor “empty”. Este valor es utilizado junto con un nuevo campo de *Question* llamado *numAnswers* en la aplicación Android, como condición para añadir o no los botones de la respuesta. De esta forma, conseguimos diversidad en las preguntas con poco código (ver Anexo D – Fragmento de código 44).

La mejora surgió tras el estudio de las aplicaciones similares en el apartado Estado del Arte. Para extender la funcionalidad se decidió poder variar únicamente el número de respuestas. Tener siempre cuatro posibles respuestas resultaba algo pesado y poco dinámico.

#### 5.4.5 Añadir a las preguntas una explicación de la respuesta

<b>PC-05</b>	<b>Añadir a las preguntas una explicación de la respuesta</b>
<b>Proyecto/s y versión</b>	ClicEPS (Android y web) v1.0.3
<b>Fecha</b>	18/04/2018
<b>Descripción</b>	Consiste en añadir un campo más a las preguntas llamado “explicación” que ayude a los alumnos a comprender por qué han podido fallar la pregunta desde la aplicación móvil.

<b>Impacto directo</b>	Los elementos afectados serían los formularios de las preguntas debido a que habría que añadir un nuevo campo, y también afectaría a la interfaz Android, añadiendo el botón “Explicación” en la sección de cada pregunta y el campo en MySQL
------------------------	---

**Tabla 12: Añadir a las preguntas una explicación de la respuesta**

Con esta mejora conseguiríamos aumentar el conocimiento del estudiante sin necesidad de una interacción directa con el docente, porque permitiríamos a los estudiantes saber por qué han acertado o fallado una pregunta.

Este cambio conllevó añadir un campo más en la entidad *Question* de MySQL, y añadir un botón en *QuestionActivity* con un evento *OnClick* que muestre bajo unas condiciones, el contenido de la explicación (ver **Anexo D – Fragmento de código 45**).

## 5.5 Resultados del proyecto

El resultado del trabajo es un conjunto de utilidades que permiten al profesor desarrollar las clases de una forma más dinámica y realizar un seguimiento del rendimiento de los estudiantes mediante su sistema estadístico integrado.

Todo el proyecto está disponible en GitHub pero actualmente con el código en privado por cuestiones de seguridad. El repositorio que contiene la plataforma web y la aplicación móvil de administración se encuentra en <https://github.com/Oscararroyon/tfg17846> y el repositorio que contiene la aplicación móvil para los estudiantes se encuentra en [https://github.com/Oscararroyon/tfg\\_android](https://github.com/Oscararroyon/tfg_android).

Como se ha explicado en apartados anteriores, también se realizaron varias encuestas de evaluación de las aplicaciones. Parte de los resultados de los que hablaremos aquí pueden consultarse en el **Anexo B**. En la **primera encuesta** realizada a los estudiantes en la versión v1.0.2 de la aplicación móvil llegamos a la conclusión de que los alumnos tenían problemas a la hora de navegar por la aplicación debido a que en el encabezado no aparecía en qué sección se encontraba el estudiante, lo que se conoce como contexto de la aplicación. También aconsejaron que en el detalle de las preguntas renombrásemos la etiqueta de “tiempo” como “tiempo restante”.

Con respecto a la **segunda encuesta** realizada a los estudiantes sobre la versión final de la aplicación móvil, destacaron consejos como que la filtración de los grupos fuese más visual. Por ello se crearon etiquetas más orientativas y visuales. También pedían una versión para IOS, ya que hay alumnos que no pueden disfrutar de la aplicación sin utilizar un simulador de Android en el ordenador. Por otra parte, en la sección de *login* destacaron la característica de actualizar la contraseña. Por último, en el listado de preguntas no añadirían nada nuevo, pero sí en el detalle de pregunta, el poder adjuntar una imagen durante la misma.

Todos los dispositivos móviles que fueron utilizados durante las pruebas de los estudiantes fueron recogidos en una tabla que puede consultarse en el **Anexo B**, junto con los resultados que describieron en la encuesta.

Para finalizar se realizó una **tercera encuesta** destinada a los profesores que probasen la plataforma web de profesorado y administración en su última versión. Destaca que los profesores prefieran juntar la elección de grupo y tema en los formularios en vez de en el panel lateral izquierdo a pesar de que el proceso sea amigable. En las estadísticas por pregunta algunos de los encuestados prefieren que no se incluyan los alumnos ausentes como parte del gráfico circular. Si observamos los resultados de las estadísticas por estudiante, un encuestado prefiere cambiar el texto de “no sabe” por “no sabe/no contesta” como en la versión original, y aconseja para un futuro crear un sistema estadístico que muestre la evolución del estudiante con el tiempo.

Por último, en relación a la compatibilidad, algunos de los encuestados tuvieron problemas con el navegador Firefox, mientras que con el Chrome no experimentaron ninguno.

## 6 Conclusiones y trabajo futuro

---

### 6.1 Conclusiones

ClicEPS ha finalizado en este trabajo en la versión v1.0.3 totalmente funcional con características muy importantes, como una interfaz de administración, un sistema de preguntas presenciales y múltiples mejoras en ambas aplicaciones con el propósito de crear una experiencia sencilla y a la vez muy útil para todos los usuarios.

Se han implementado todos los requisitos recogidos durante la fase de análisis y diseño, siendo finalmente probados por los alumnos y profesores en un entorno real mediante las pruebas unitarias y funcionales descritas anteriormente.

No creemos que se trate de un proyecto complejo, pero sí muy extenso al haber añadido un número elevado de características nuevas y haber modificado los estándares de conexión y seguridad que se implementaron en la versión original. Como el proyecto partía de un código ya desarrollado, se ha tenido que realizar el trabajo de entender y comprender su funcionamiento e informarse del *framework* en el que se ejecuta, AngularJs.

Pensando en versiones futuras por temas relacionados con el mantenimiento de las aplicaciones, se ha documentado todo el código para simplificar la labor del desarrollador que retome el proyecto.

### 6.2 Trabajo futuro

Con este proyecto se da por finalizada la ampliación de ambas aplicaciones con un sistema de administración capaz de manejar todos los datos de la aplicación junto con múltiples funcionalidades que mejoran la experiencia en ClicEPS. Sin embargo y, como hemos ido mencionando en algunos apartados, ClicEPS puede ir más allá. Una buena propuesta extraída del resultado de las encuestas consiste en crear una aplicación análoga a la aplicación Android para los estudiantes que operen bajo dispositivos IOS.

Por otra parte, se podrían añadir mejoras en el sistema estadístico para que pudiese mostrar un seguimiento evolutivo de cada uno de los estudiantes a lo largo del tiempo.

La característica que más valoramos de la nueva versión del proyecto es la creación de preguntas presenciales, pero creemos que se puede mejorar. Para ello sería necesario conseguir los planos de la facultad y darlos de alta en Google Maps Indoor.

Una vez dados de alta, la idea sería diseñar el sistema presencial con Situm Indoor Positioning [\[20\]](#), una tecnología muy avanzada de posicionamiento en interiores que da soporte en Android e IOS con una extensa documentación.

Por último, le daríamos un toque de red social, donde los estudiantes y profesores pudiesen interactuar en una especie de chat. Son ideas que mejorarían todo lo que hemos creado en ClicEPS y la harían destacar aún más en el ámbito educativo, no sólo creando distribuciones para la Escuela Politécnica Superior, sino cualquier otra facultad de la Universidad.

# Referencias

---

- [1] “What are Clickers?”, Northern Illinois University, Diciembre 2007, <https://www.niu.edu/facdev/resources/quicktips/transcripts/clickers.shtml>
- [2] Android, <https://www.android.com/>
- [3] Hitesh Ramoliya, “Why AngularJs is Best Javascript Framework in 2017”, LinkedIn, Marzo 2017, <https://www.linkedin.com/pulse/why-angularjs-best-javascript-framework-2017-ramoliya-hitesh>
- [4] Jose M<sup>a</sup> Baquero García, “¿Qué tiene de bueno NodeJS?”, Arsys blog Octubre 2015, <https://www.arsys.es/blog/programacion/ventajas-nodejs/>
- [5] “Why MySQL?”, <https://www.mysql.com/why-mysql/>
- [6] Laurence Moroney, “The Definitive Guide to Firebase – Build Android Apps on Google’s Mobile Platform”, Noviembre 2017, Apress
- [7] “Create and monitor geofences”, <https://developer.android.com/training/location/geofencing>
- [8] Virginia Gaitán, “Gamificación: el aprendizaje divertido”, educativa blog, Octubre 2013, <https://www.educativa.com/blog-articulos/gamificacion-el-aprendizaje-divertido/>
- [9] Sarah Moody, “Quizalize Helps Tamaini Win Most Valuable Educator Award”, Quizalize blog, Abril 2018, <https://www.quizalize.com/blog/2018/04/04/teacher-story-quizalize-helps-tamaini-win-valuable-educator-award/>
- [10] José Luis Alejandro Marco, “Buenas prácticas en la docencia universitaria con apoyo de TIC. Experiencias en 2016 – Herramientas gratuitas para la gamificación”, Mayo 2017, Prensas de la Universidad de Zaragoza
- [11] Iphone OS, <https://es.wikipedia.org/wiki/IOS>
- [12] Google Chrome, <https://www.google.es/chrome/index.html>
- [13] Alfredo Weitzenfeld Ridet, Silvia Guardati Buemo “Ingeniería de software: el proceso para el desarrollo de Software – Capítulo 12.2.1 Modelo de cascada”, 2008
- [14] HMAC, <https://es.wikipedia.org/wiki/HMAC>
- [15] Mike Coutermarsh, “Heroku Cookbook”, Noviembre 2014, Packt Publishing
- [16] Marc Stevens, Elie Bursztein, Pierre Karpman, Ange Albertini, Yarik Markov, Alex Petit Blanco, Clement Baisse, “Announcing the first SHA1 collision”, Google Security Blog, Febrero 2017, <https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html>
- [17] Cecilio Álvarez Caules, “Frameworks CSS (Less y Sass)”, Cantabria TIC, Junio 2014, <http://www.cantabriatic.com/frameworks-css-less-y-sass/>
- [18] WebView Android, <https://developer.android.com/reference/android/webkit/WebView>
- [19] ClearDB, <http://w2.cleardb.net/>
- [20] Situm Indoor Positioning, <https://situm.es/es/tecnologia>
- [21] Twitter, <https://twitter.com/>



## Glosario

---

API	Application Programming Interface
CSV	Comma Separated Values
IOS	Iphone OS
OS	Operating System
IP	Internet Protocol
HMAC	Keyed-Hash Message Authentication Code
SHA	Secure Hash Algorithm
IDE	Integrated Development Environment
HTML	HyperText Markup Language
FAQ	Frequently Asked Questions
CSS	Cascading Style Sheets
APK	Android Application Package
Callback	Es una mecánica a partir de la cual se permite al cliente devolverle la llamada en un tiempo determinado



## Anexos

---

### *A Manual de Usuario*



**Figura 8: Logo de ClickEPS v.1.0.3**

ClickEPS puede parecer una aplicación muy sencilla, pero al disponer de tantas herramientas en la plataforma web, a veces llega a resultar complejo para el usuario. Por eso, con el objetivo de ayudar tanto a los administradores como a los profesores, se han creado dos vídeos a modo de tutorial que muestran cómo manejarse por la plataforma web.

Para acceder al tutorial sobre la plataforma de profesorado se debe visitar el siguiente enlace: <https://www.youtube.com/watch?v=nLtbMXKpHH0&feature=youtu.be>, mientras que para acceder al tutorial sobre la plataforma de administración hay que visitar <https://www.youtube.com/watch?v=riPnmP8kCN4&feature=youtu.be>. Ambos tutoriales están narrados y muestran visualmente cómo hacer cada una de las funciones disponibles en ClickEPS.

Esta aplicación también cuenta con un servicio de soporte pensando más para el futuro, y es que desde la aplicación web o móvil podemos acceder a un perfil que nos redirige directamente a la red social Twitter [21], que es el siguiente: <https://twitter.com/ClickEps>. En este perfil, cualquier usuario podrá pedir ayuda en caso de no saber cómo utilizar la aplicación y también podrá reportar errores que encuentren durante su uso.



Figura 9: Perfil de Twitter ClickEPS

Con la existencia de esta red social, se decidió que no era necesario crear un tutorial para la aplicación móvil debido a la facilidad de uso, y por ello se hizo hincapié en mostrar a los estudiantes que disponen desde la APP un enlace directo a soporte para consultar cualquier tipo de duda.

Por último, hay que destacar también el pequeño F.A.Q que se muestra en la plataforma web cada vez que el profesor o administrador inicia sesión para recordarle cómo manejarse por la web. La Figura 10 muestra el F.A.Q que se encuentra un administrador.

**F.A.Q**

**¿Cómo se distribuye el menú de la plataforma de administración?**  
 Se distribuye de dos formas: horizontal y vertical. Vamos a describir los 3 paneles existentes:  
 \*Si observas en el **panel de la izquierda**, verás que pone 'Edición'. A partir de esos enlaces directos, podrás editar tanto un Usuario (Profesor o estudiante) como un Administrador, Asignatura, Tema y Grupo.  
 \*En el **panel de la derecha** tendrá diversas funcionalidades, como por ejemplo volver a esta página, editar su perfil de administrador, importar archivos .CSV, resetear la base de datos o cerrar sesión.  
 \*Por último, tiene el **panel superior** que se distribuye de forma horizontal. Ahí tendrá todas las opciones posibles con los elementos que se describen (por ejemplo, creación, edición, eliminación, matriculación, etc)

**¿Existen guías sobre las distintas funcionalidades de la plataforma?**  
 Así es, aquí puedes encontrar enlaces a videotutoriales sobre toda la funcionalidad posible de ClickEPS:  
 \* [Funcionalidad Administrativa](#)

Figura 10: F.A.Q de administrador

## B Resultados de la encuesta

Esta pequeña sección recoge algunos de los resultados más importantes de las encuestas realizadas durante el proyecto, ya que la dimensión de las encuestas es demasiado grande. Finalmente, se recogerán en una lista todos los dispositivos Android que se utilizaron durante las pruebas con las conclusiones de los alumnos sobre el rendimiento.

5 [Filtrado por Grupos] ¿Añadiría algo a esta sección?

No (4x)

Pondría arriba un cartel grande que especificar el menú en el que estamos, en este caso Grupos

No.

Mis notas:

En la próxima versión v1.0.3 de la aplicación se incluirá el indicativo que especifique el menú en el que estamos

Figura 11: Encuesta sobre el filtrado por grupos

16 [Detalle de Preguntas] ¿Añadirías algo a esta sección?

En el tiempo pondría mejor "tiempo restante"

No (3x)

Pondría arriba un cartel grande que especificar el menú en el que estamos, en este caso Preguntas

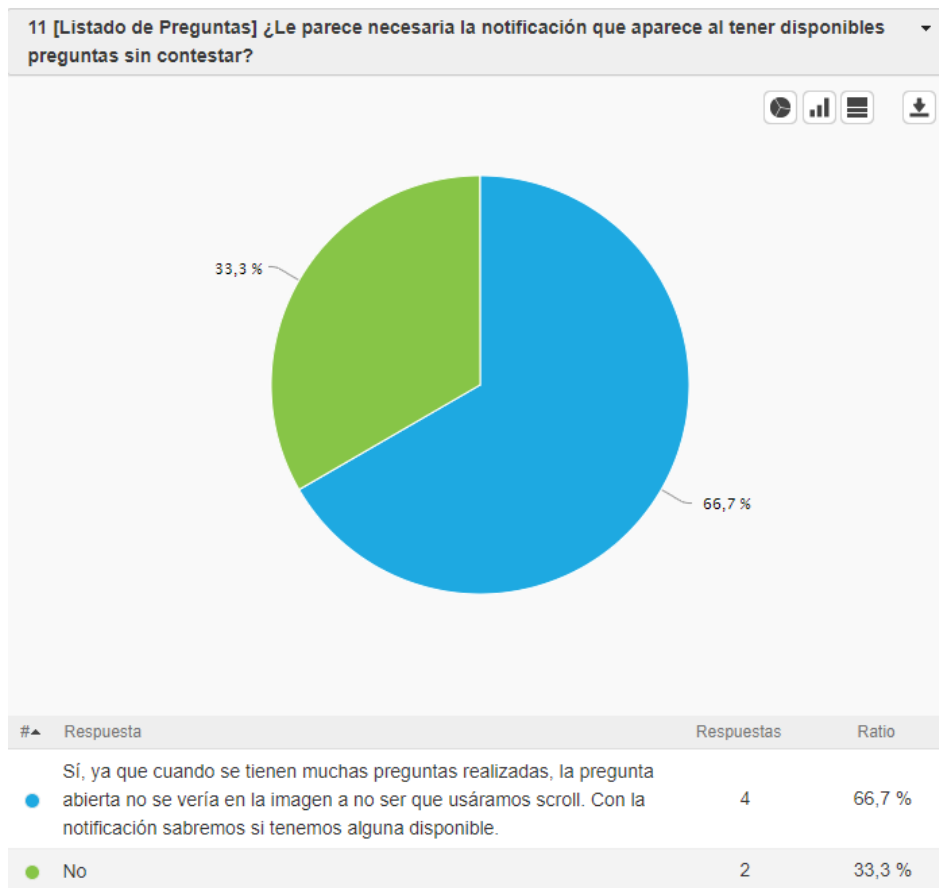
Realmente, con el ejemplo que habéis utilizado no parece que el enunciado sea precisamente un enunciado

Mis notas:

\* En la próxima versión v1.0.3 de la aplicación se incluirá el indicativo que especifique el menú en el que estamos

\* En la próxima versión v1.0.3 de la aplicación se incluirá la palabra "Enunciado" al principio de cada pregunta que se cree para recalcar el hecho de que se trata de un enunciado. Además, se añadirá alguna división física en la aplicación para separar el mismo de las posibles respuestas

Figura 12: Encuesta sobre el detalle de preguntas



**Figura 13: Encuesta sobre el listado de preguntas**

9 [Filtrado por Temas] ¿Añadirías algo a esta sección?

Podría ser de una manera más visual, no tan simple

No existe esta aplicación para usuarios de iOS, en ninguna asignatura se ofreció opción para esos alumnos, que han quedado incapacitados de utilizar dichas aplicaciones.

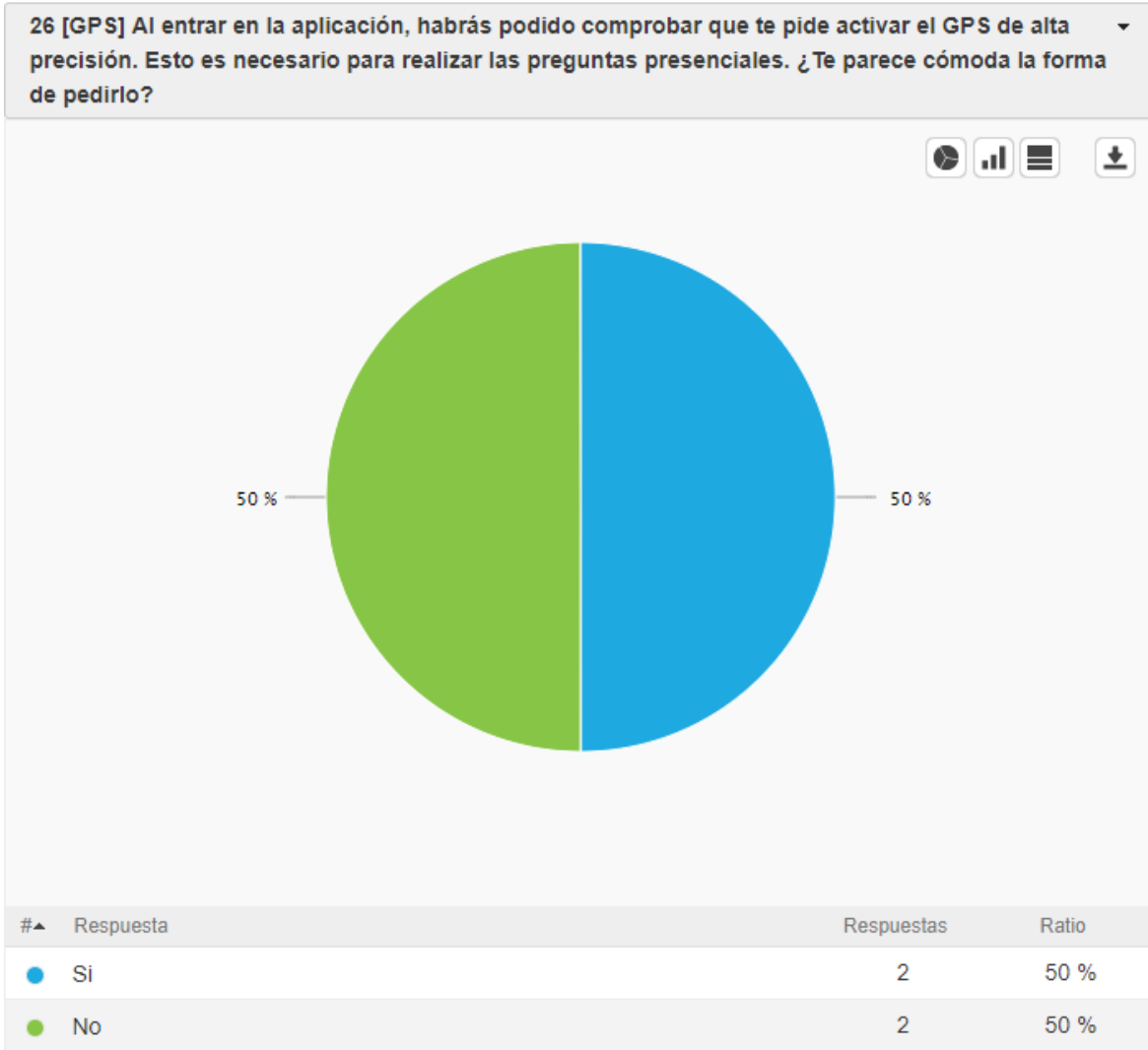
**Figura 14: Encuesta sobre el filtrado por temas**

23 [Login] ¿Añadirías algo a esta sección?

No, está bien que se pueda cambiar la contraseña

No existe esta aplicación para usuarios de iOS, en ninguna asignatura se ofreció opción para esos alumnos, que han quedado incapacitados de utilizar dichas aplicaciones.

**Figura 15: Encuesta sobre el login**

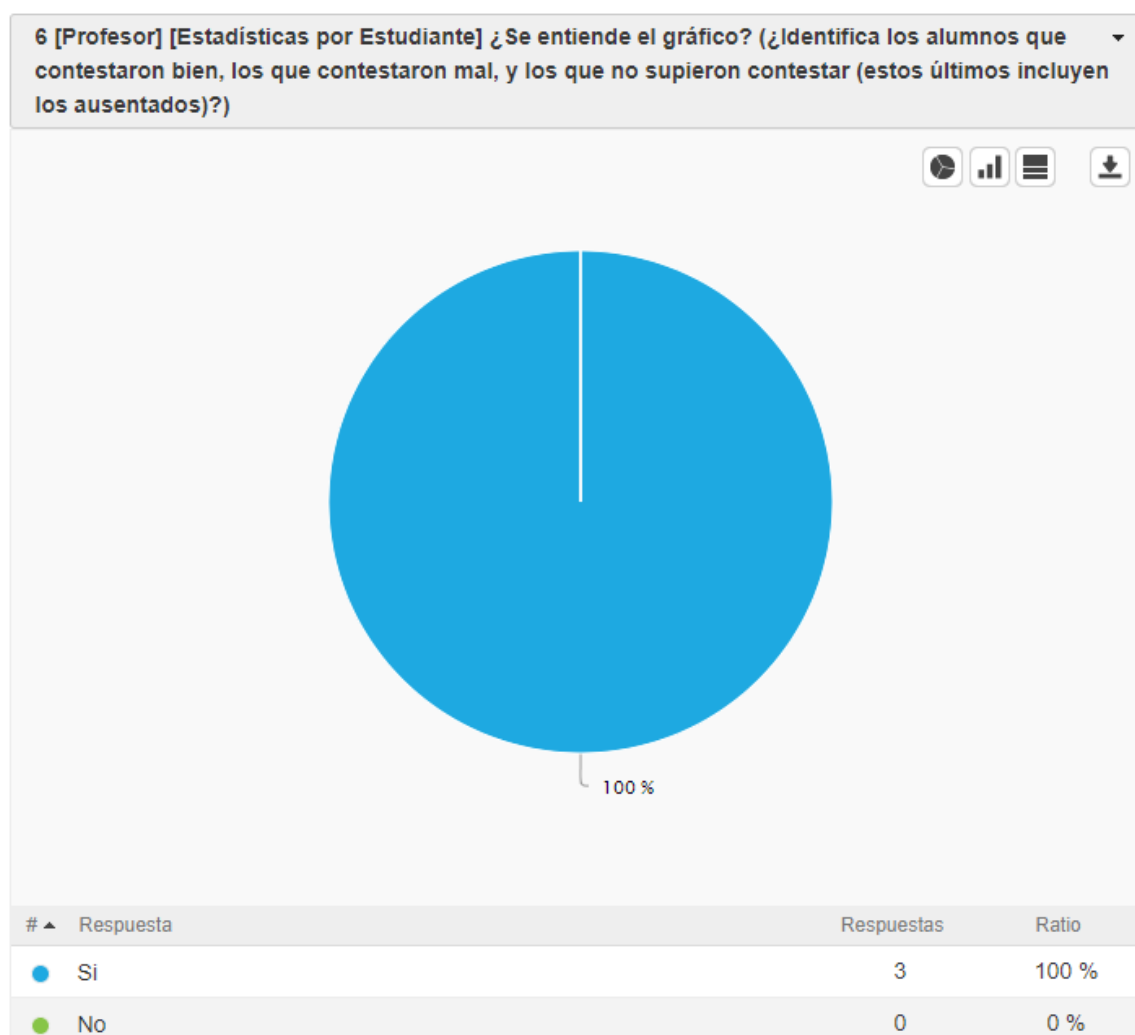


**Figura 16: Encuesta sobre el GPS**

2 [Filtrado por Grupos] ¿Ayuda el nuevo mensaje indicativo en la elección de grupo?	▼
3 [Filtrado por Grupos] ¿Te parece necesaria la notificación que aparece al confirmar un grupo?	▼
4 [Filtrado por Grupos] Cuando tu cuenta no dispone de ningún grupo, aparece un mensaje nuevo indicando de la situación. ¿Lo ves necesario?	▼
5 [Filtrado por Grupos] ¿Añadirías algo a esta sección?	▼
6 [Filtrado por Temas] ¿Se entiende qué contiene el listado?	▼
7 [Filtrado por Temas] ¿Ayuda el nuevo mensaje indicativo en la elección de tema?	▼
8 [Filtrado por Temas] Cuando tu cuenta no dispone de ningún tema, aparece un mensaje nuevo indicando de la situación. ¿Lo ves necesario?	▼
9 [Filtrado por Temas] ¿Añadirías algo a esta sección?	▼
10 [Listado de Preguntas] ¿Se entiende qué contiene el listado?	▼
11 [Listado de Preguntas] ¿Te parece necesaria la notificación que aparece al tener disponibles preguntas sin contestar?	▼
12 [Listado de preguntas] ¿Te parecen intuitivos los iconos?	▼
13 [Listado de preguntas] ¿Añadirías algo a esta sección?	▼
14 [Detalle de Pregunta] ¿Se lee bien la pregunta y sus opciones?	▼
15 [Detalle de Pregunta] ¿Se entiende el contador de tiempo o el mensaje de información sobre la respuesta dada?	▼
16 [Detalle de Pregunta] ¿Se entienden los botones de enviar respuesta?	▼
17 [Detalle de Preguntas] ¿Añadirías algo a esta sección?	▼
18 [Diseño] Con la versión anterior, no se podía girar la pantalla y continuar con la aplicación en ese ángulo. Sin embargo, ahora que puede, ¿Te resulta útil tener esa posibilidad?	▼
19 [Diseño] Observando las siguientes imágenes relacionadas con las versiones anteriores (v1.0.1 del año pasado y v1.0.2 de Enero), ¿crees que ha mejorado la interfaz o no?	▼

**Figura 17: Parte de la encuesta realizada a los estudiantes**





**Figura 18: Encuesta sobre el gráfico circular en las estadísticas por estudiante**

7 [Profesor] [Estadísticas por Estudiante] ¿Echa algo en falta? Explíquenos también su experiencia en esta sección

Cambiaría el texto por de no sabe por no sabe/no contesta para que quede más claro. Haría gráficas que muestren cómo evoluciona un estudiante con el tiempo

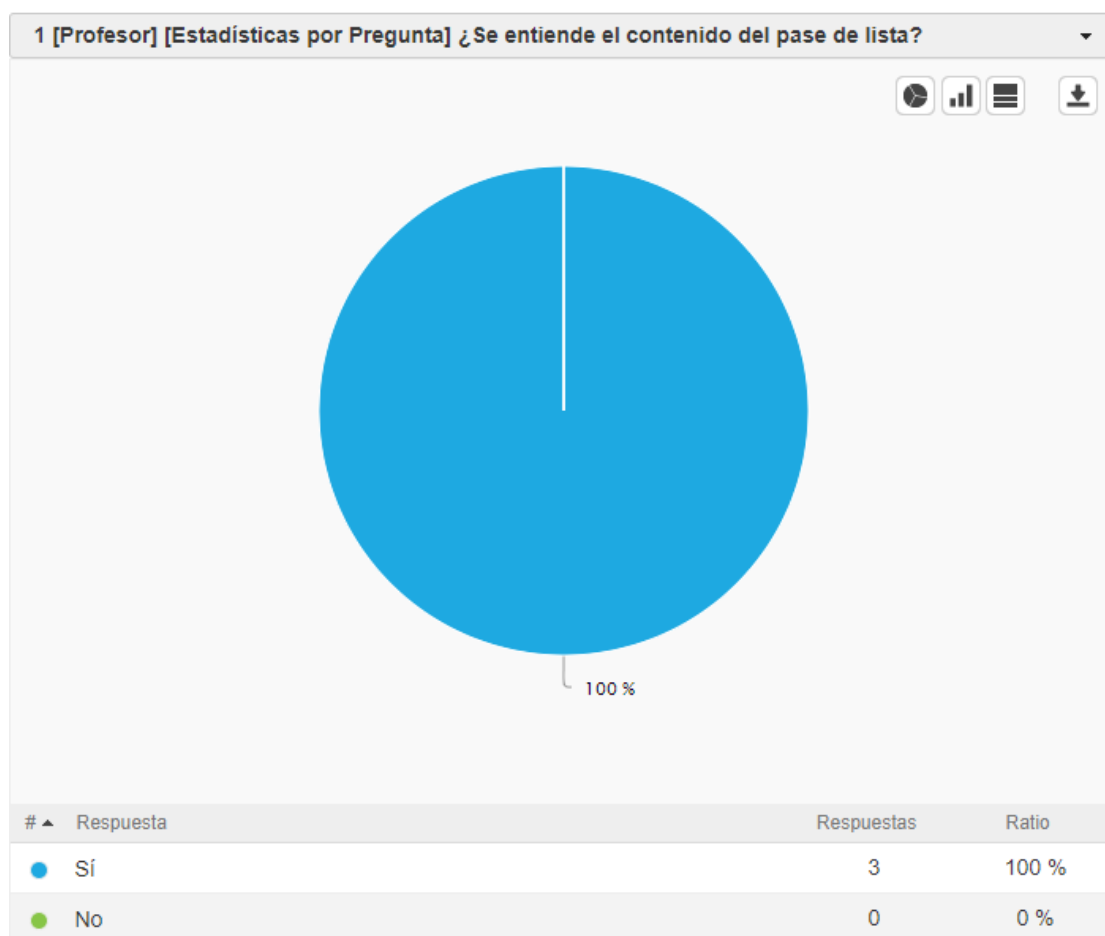
**Figura 19: Encuesta sobre las estadísticas por estudiante**

4 [Profesor] [Estadísticas por Pregunta] ¿Echa algo en falta? Explíquenos también su experiencia en esta sección ▾

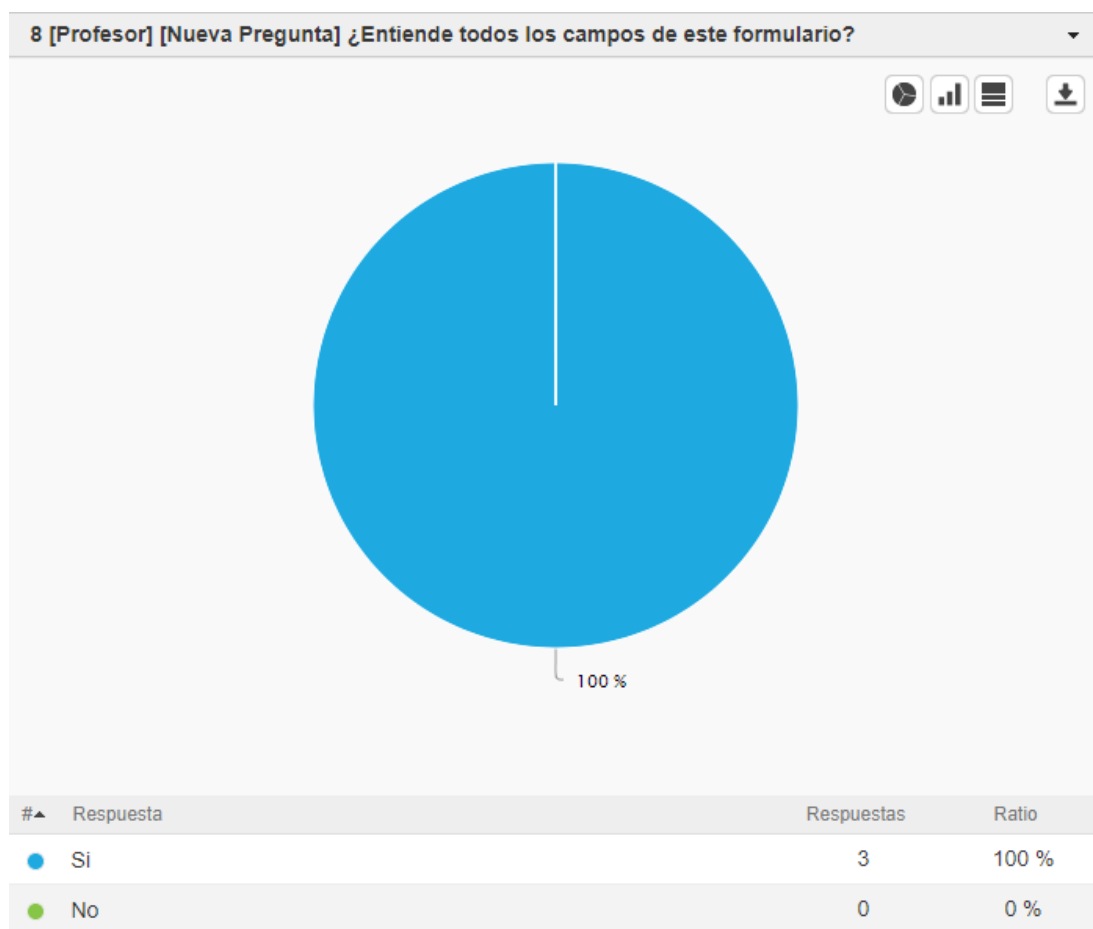
En general es bastante amigable el proceso salvo a la hora de ligar una pregunta a un tema. Creo que ayudaría que el tema al que se une una pregunta también forme parte de la plantilla, en lugar de estar fuera.

Echo de menos que se puedan ver las estadísticas eliminando a los que no han contestando

**Figura 20: Encuesta sobre las estadísticas por pregunta**



**Figura 21: Encuesta sobre el pase de lista**



**Figura 22: Encuesta sobre el formulario de nueva pregunta**

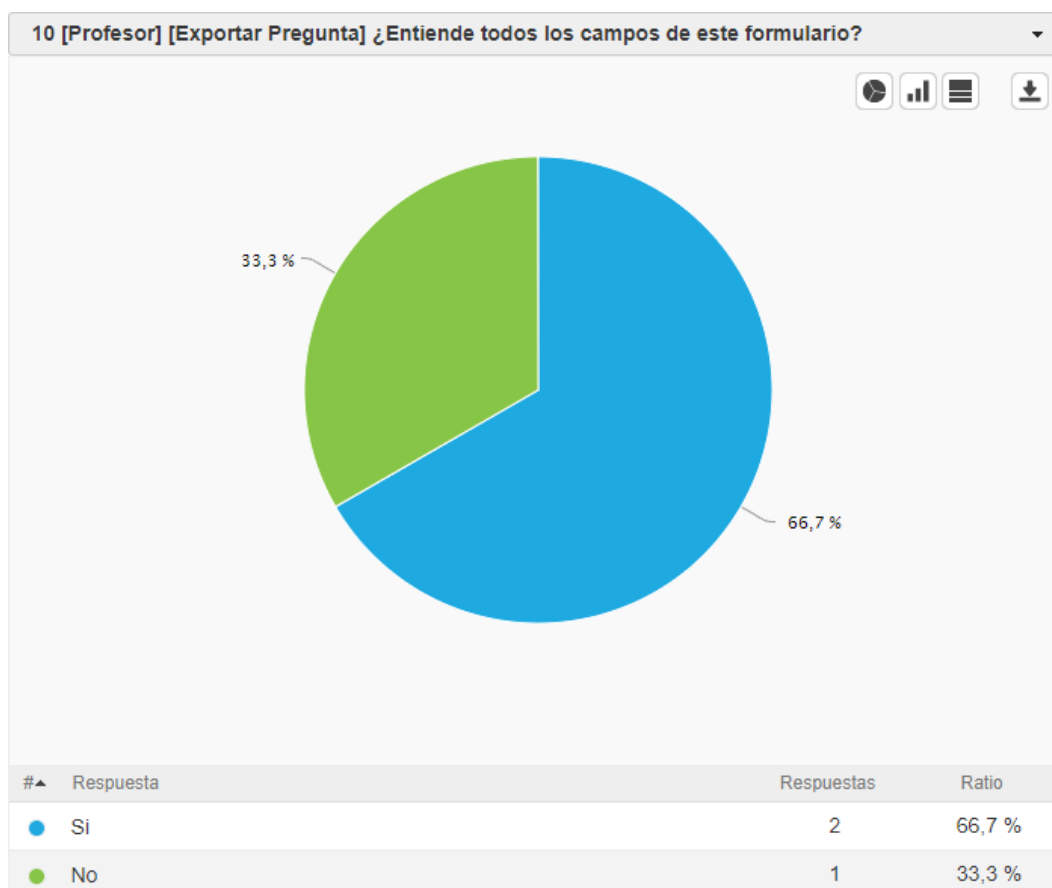
9 [Profesor] [Nueva Pregunta] ¿Echa algo en falta? Expliquenos también su experiencia en esta sección ▾

El tema al que pertenece la pregunta, que no forma parte del cuestionario, sino que se encuentra fuera del mismo.

En firefox, este formulario se ve mal. Los campos se ponen por debajo de del panel izquierdo y algunos campos no se ven. Creo que la selección del tema de la pregunta debe estar en el formulario y no en el panel lateral.

En mi ordenador la barra lateral azul derecha tapaba parte del texto y formularios, que continuaban hacia la derecha.

**Figura 23: Encuesta sobre nueva pregunta**



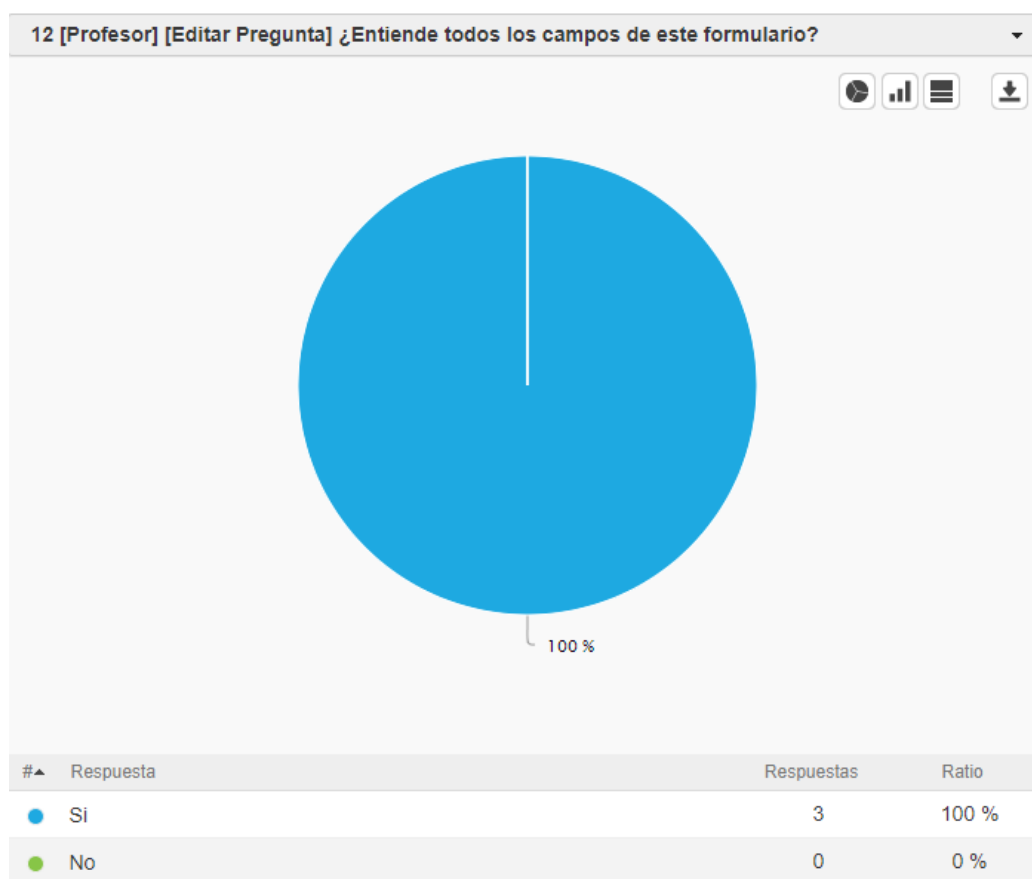
**Figura 24: Encuesta sobre el formulario de exportar pregunta**

11 [Profesor] [Exportar Pregunta] ¿Echa algo en falta? Explíquenos también su experiencia en esta sección

El tema creo que es mejor mostrarlo en la pregunta y no en el panel lateral

Es algo confuso que exportar pregunta esté justo debajo de nueva pregunta. Debería estar en dos páginas diferentes, por ejemplo separadas por fichas o por un menú previo.

**Figura 25: Encuesta sobre exportar pregunta**



**Figura 26: Encuesta sobre el formulario de editar pregunta**

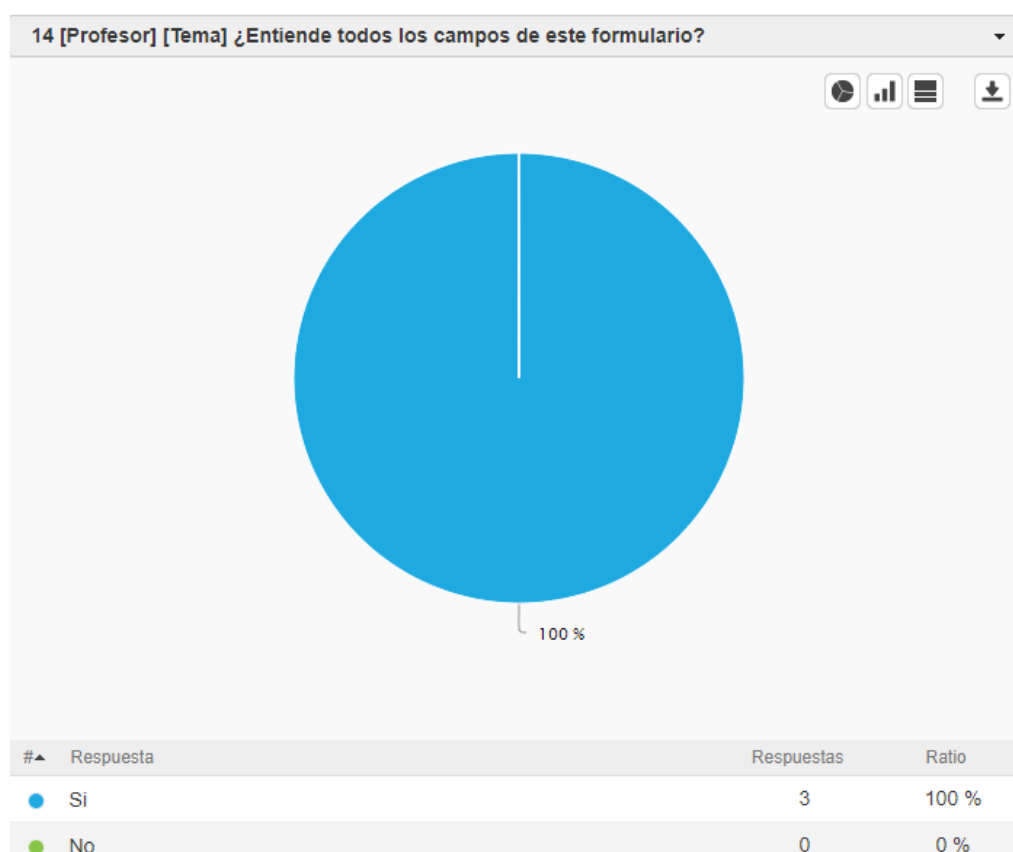
13 [Profesor] [Editar Pregunta] ¿Echa algo en falta? Explíquenos también su experiencia en esta sección

No he podido cambiar una pregunta de tema. He creado primero la pregunta y la he situado en un tema que creo que se llamaba PROG2. Luego he añadido un nuevo tema. He editado la pregunta para cambiarla de tema y, al menos yo, no he encontrado cómo.

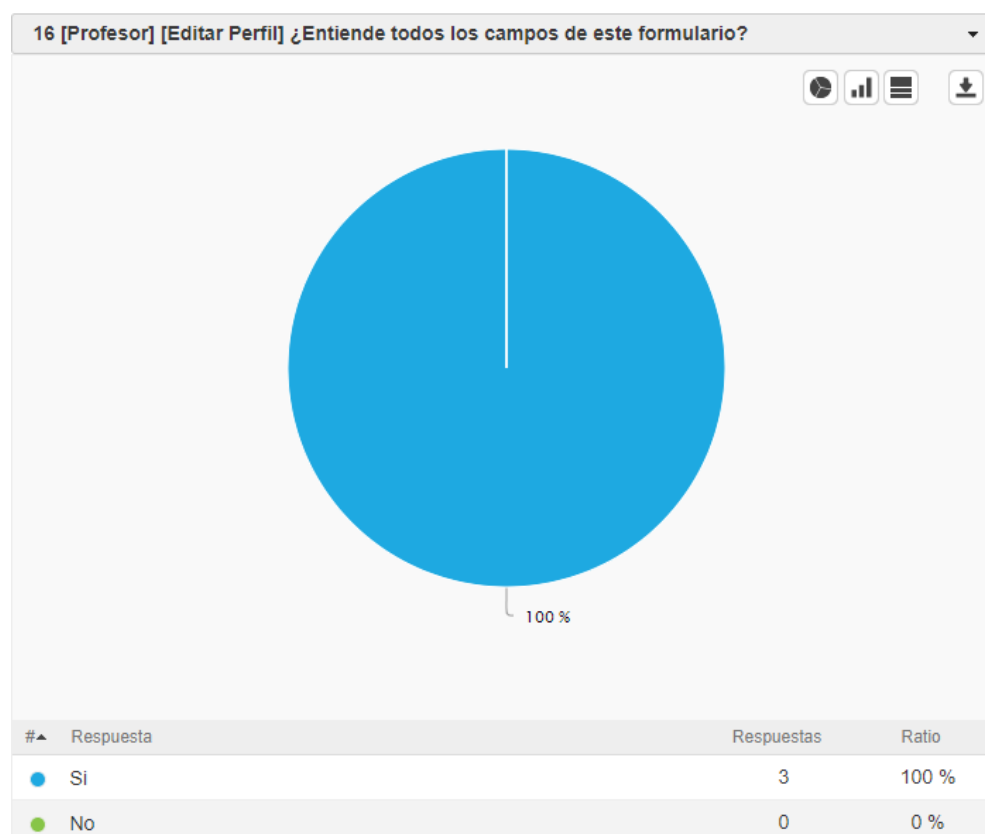
Tiene el mismo problema que crear pregunta: se ve mal en firefox

Al seleccionar editar pregunta confunde algo que aparezca directamente seleccionada una pregunta. Sería más claro si primero hubiera que elegir pregunta para editar y luego apareciera.

**Figura 27: Encuesta sobre editar pregunta**



**Figura 28: Encuesta sobre tema**



**Figura 29: Encuesta sobre editar perfil**

11 [Profesor] [Exportar Pregunta] ¿Echa algo en falta? Explíquenos también su experiencia en esta sección

12 [Profesor] [Editar Pregunta] ¿Entiende todos los campos de este formulario? ▾

13 [Profesor] [Editar Pregunta] ¿Echa algo en falta? Explíquenos también su experiencia en esta sección ▾

14 [Profesor] [Tema] ¿Entiende todos los campos de este formulario? ▾

15 [Profesor] [Tema] ¿Echa algo en falta? Explíquenos también su experiencia en esta sección

16 [Profesor] [Editar Perfil] ¿Entiende todos los campos de este formulario? ▾

17 [Profesor] [Editar Perfil] ¿Echa algo en falta? Explíquenos también su experiencia en esta sección

18 [Profesor] [Home y Soporte] ¿Entiende ambas secciones dedicadas a la información actual de la aplicación y al soporte vía redes sociales (Twitter)? ▾

19 [Profesor] En caso de haber utilizado la versión v1.0.1 de ClickEPS (la del año pasado) por favor, explique si le parecen útiles las nuevas funcionalidades, su experiencia en comparación, etc.

20 [Administrador] [Login] ¿Le parece buena la posibilidad de que la plataforma ofrezca la elección entre la plataforma de profesorado y administración en caso de que el administrador también tenga una cuenta de profesor con los mismos datos? ▾

21 [Administrador] [Login] ¿Echa algo en falta? Explíquenos también su experiencia en esta sección ▾

22 [Administrador] [Grupo] ¿Entiende los campos de los formularios? ▾

23 [Administrador] [Grupo] ¿Echa algo en falta? Explíquenos también su experiencia en esta sección ▾

24 [Administrador] [Estudiante] ¿Entiende los campos de los formularios? ▾

25 [Administrador] [Estudiante] ¿Echa algo en falta? Explíquenos también su experiencia en esta sección ▾

**Figura 30: Parte de la encuesta realizada a los profesores y administradores**

Por último vamos a hablar sobre los dispositivos utilizados por los estudiantes. Observamos a continuación en la Tabla 13 que 7 de un total de 8 dispositivos obtuvieron un rendimiento positivo de la aplicación móvil, mientras que sólo un dispositivo tuvo problemas de ejecución durante las pruebas.

<b>Fabricante</b>	<b>Modelo y versión</b>	<b>Android</b>	<b>Resultado</b>
<b>Lenovo</b>	YT3-X50F	6.0.1	“Totalmente funcional”
<b>Samsung</b>	Galaxy J7 SM-J710FN	7.0	“Perfecto”
<b>Samsung</b>	Galaxy S7 SM-G930F	7.0	“Va perfectamente”
<b>Huawei</b>	Yumo	4.5	“Funciona”
<b>Huawei</b>	VNS-AL00	6.0	“No he tenido inconvenientes”
<b>BQ</b>	Aquaris M5.5	7.1.2	“La aplicación va lenta en este dispositivo, a veces no arranca y queda bloqueada”
<b>Samsung</b>	Galaxy Tab SM-T230	4.4.2	“En este dispositivo funciona con normalidad”
<b>Huawei</b>	Y6 Pro SLA-L22	7.0	“Funciona correctamente”

**Tabla 13: Dispositivos Android utilizados durante las pruebas**



## C Historial de versiones e incidencias resueltas

En esta sección podemos ver qué versiones ha tenido la aplicación desde su inicio hasta el final del proyecto, así como la fecha orientativa de cuándo se empezó. Las versiones fueron cambiando a la vez tanto en la web como en la aplicación móvil. Por último, añadiremos las incidencias resueltas al final de cada una de las versiones, a modo de mantenimiento del software.

Versión	Fecha inicio	Fecha fin
V1.0.1	-	15/02/2017
V1.0.2	16/12/2017	31/01/2018
V1.0.3	01/02/2018	-

Tabla 14: Historial de versiones

Las versiones cuya fecha es un guión indica que o está actualmente en esa versión, o la fecha inicial del proyecto.

ID	Descripción	Prioridad	Versión
INC0001	Al crear una pregunta, la hora de expiración se almacenaba en la base de datos mal	9	V1.0.1 web
INC0002	Las preguntas que no están contestadas pero han expirado, se muestran como disponibles	6	V1.0.2 Android
INC0003	Cuando confirmo un grupo o tema sin existir ninguno, la aplicación se cierra	10	V1.0.2 Android
INC0004	Borrar pregunta no funciona si hay respuestas asociadas	10	V1.0.2 web
INC0005	El tiempo en todos los campos se muestra en <i>timezone</i> americano	6	V1.0.2 web
INC0006	Al editar la pregunta no se guarda el campo <i>homework</i>	9	V1.0.2 web
INC0007	Al duplicar la pregunta no se guarda el campo <i>homework</i>	9	V1.0.2 web
INC0008	Error de las cookies al hacer <i>login</i>	10	V1.0.3 web
INC0009	" <i>method ID not in [0, 0xffff]: 65536</i> "	10	V1.0.3 Android
INC0010	Cuando matriculo a un usuario que coincide en nombre con otro, selecciona el último	10	V1.0.3 web
INC0011	Las estadísticas en directo salen con números negativos	7	V1.0.3 web
INC0012	Al editar una pregunta con un enunciado largo, se descuadran las cajas de texto	10	V1.0.3 web
INC0013	Cuando hay preguntas preparadas pero no activadas, las muestra como disponibles	9	V1.0.3 Android

Tabla 15: Historial de incidencias



## ***D Fragmentos de código***

En este anexo se adjuntan todas las capturas de pantalla utilizadas durante el trabajo que contienen fragmentos de código.

```
var mysql = require('mysql');
var pool = mysql.createPool({
  connectionLimit : 500,
  port      : '3306',
  host      : 'us-cdbr-iron-east-05.cleardb.net', //localhost
  user      : 'b8c4da0fa7a6a4',
  password  : '63544fcc',
  database  : 'heroku_502e45638ebd447'
});
```

### **Fragmento de código 1: Conexión MySQL mediante un pool**

```
app.post('/resetIDUsers/', function(req, res) {
  var query = "ALTER TABLE heroku_502e45638ebd447.User AUTO_INCREMENT = 1";
  pool.getConnection(function(err, connection) {
    connection.query(query, function(err, result) {
      connection.release();
      if(err) {
        console.error("Problem with MySQL" + err);
      } else {
        res.send(result);
      }
    });
  });
});
```

### **Fragmento de código 2: Ejemplo de consulta MySQL**

```
var isCorrectPassword = function(password, userEncryptPassword) {
  vm.passwordDecrypted = password;
  var expired = new Date();
  expired.setMinutes(expired.getMinutes() + 60);
  /*Insertamos Cookie que después utilizaremos para editar el password*/
  $cookies.putObject('passwordDecrypted', vm.passwordDecrypted, { expires : expired });
  var encryptPassword = cryptoJS.HmacSHA1(password,"epsuam").toString();
  return encryptPassword === userEncryptPassword;
};
```

### **Fragmento de código 3: Implementación del cifrado HMAC-SHA1 (web)**

```

try {
    JSONObject jsonObject = new JSONObject(s);

    String emailResponse = jsonObject.getString("email");
    String passwordResponse = jsonObject.getString("password");
    String userId = jsonObject.getString("idUser");
    /*Aumentamos seguridad con HmacSHA1 y una clave de cifrado*/
    String hexString;
    hexString = hmacDigest(passwordEditText.getText().toString(), "epsuam", "HmacSHA1"); /*Realizamos la encriptación con el nuevo algoritmo*/

    if (hexString.toString().equals(passwordResponse)) {
        if (emailResponse.contains("@estudiante.uam")) {
            Intent intent = new Intent("es.uam.eps.tfg17846.mariopolo2805.clickeps.SUBJECTACTIVITY");
            intent.putExtra(Constants.USERID_KEY, userId);
            startActivity(intent);
        } else {
            Toast.makeText(LoginActivity.this, "Acceso solo autorizado a estudiantes", Toast.LENGTH_SHORT).show();
        }
    } else {
        AlertDialog.Builder dialogBuilder = new AlertDialog.Builder(LoginActivity.this);
        dialogBuilder.setTitle("Login");
        dialogBuilder.setMessage("Usuario o contraseña incorrectos");
        dialogBuilder.setNeutralButton("Atrás", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
            }
        });
        dialogBuilder.show();
    }
} catch (Exception e) {
    AlertDialog.Builder dialogBuilder = new AlertDialog.Builder(LoginActivity.this);
    dialogBuilder.setTitle("Oops!");
    dialogBuilder.setMessage("Algo fue mal");
    dialogBuilder.setNeutralButton("Atrás", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
        }
    });
    dialogBuilder.show();
}
}

```

#### Fragmento de código 4: Implementación del cifrado HMAC-SHA1 (Android)

```

public static String hmacDigest(String msg, String keyString, String algo) {
    String digest = null;
    try {
        SecretKeySpec key = new SecretKeySpec((keyString).getBytes("UTF-8"), algo);
        Mac mac = Mac.getInstance(algo);
        mac.init(key);
        byte[] bytes = mac.doFinal(msg.getBytes("ASCII"));
        StringBuffer hash = new StringBuffer();
        for (int i = 0; i < bytes.length; i++) {
            String hex = Integer.toHexString(0xFF & bytes[i]);
            if (hex.length() == 1) {
                hash.append('0');
            }
            hash.append(hex);
        }
        digest = hash.toString();
    } catch (UnsupportedEncodingException e) {
    } catch (InvalidKeyException e) {
    } catch (NoSuchAlgorithmException e) {
    }
    return digest;
}

```

#### Fragmento de código 5: Función *hmacDigest* para cifrado HMAC-SHA1 (Android)

```

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
<form class="new-question-layout">
<fieldset>
<legend><b>Panel de Administración ClickEPS v1.0.3</b></legend>

    <div class="new-question-text-content">
        
    </div>

    <div class="new-question-answers-content">

<p>Bienvenid@ a la plataforma web de administración de ClickEPS. Si necesita alguna ayuda, puede consultar en esta misma página el F.A.

    <p>Por esto mismo, si es su primera vez aquí, le recomendamos que lea el F.A.Q para familiarizarse con la plataforma.</p>

    <p>No olvide realizar la encuesta una vez haberse familiarizado con la plataforma web.</p>

    <p><b>Enlace de la encuesta: <li><a href="https://www.surveio.com/survey/d/U9N2A6C5Q9Q6D7U1Y">Encuesta</a></li></b></p>

    </div>

    </fieldset>
</form>

<form class="new-question-layout">
<fieldset>
<legend><b>Noticias:</b></legend>

    <div class="new-question-answers-content">

    <ul>
    <li><b>*04/04/2018 -</b> Se ha corregido una incidencia en las preguntas presenciales. En caso de seguir funcionando mal, se comunicará

    </ul>
    </div>
</fieldset>
</form>

```

## Fragmento de código 6: Parte del fichero *welcome*

```

* Editar Admin */
vm.editarAdmin = function() {
    if(vm.AdminEditSelected.name == null || vm.AdminEditSelected.surname == null || vm.AdminEditSelected.email == null){
        alert("Rellene todos los campos necesarios para editar el perfil");
    }else{
        /*Primero comprobamos si el correo existe*/
        AdminDataSer.comprobarSiExisteCorreoID(vm.AdminEditSelected.id,vm.AdminEditSelected.email).then(function(result){
            if(result.length != 0){ //No es correcto, el correo existe
                alert("La dirección de correo ya está en uso");
                $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
            }else{
                AdminDataSer.editarAdmin(vm.AdminEditSelected.id,vm.AdminEditSelected.name,vm.AdminEditSelected.surname,vm.AdminEditSelected.email);
                alert("Admin editado con éxito");
                $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
            }
        });
    }
};
}

```

## Fragmento de código 7: Función utilizada en *editAdmin*

```

* Editar Perfil Admin */
vm.editarPerfilAdmin = function() {
  if(vm.userProfile.name == null || vm.userProfile.surname == null || vm.userProfile.email == null || vm.userProfile.passwordDecrypted == null ){
    alert("Rellene todos los campos necesarios para editar el perfil");
  }else{
    /*Primero comprobamos si el correo existe*/
    AdminDataSer.comprobarSiExisteCorreoID(vm.userProfile.id,vm.userProfile.email).then(function(result){
      if(result.length != 0){ //No es correcto, el correo existe
        alert("La dirección de correo ya está en uso");
        $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
      }else{
        var Auxpassword;
        Auxpassword = cryptoJS.HmacSHA1(vm.userProfile.passwordDecrypted,"epsuam").toString();
        AdminDataSer.editarPerfil(vm.userProfile.id,vm.userProfile.name,vm.userProfile.surname,vm.userProfile.email,Auxpassword);
        alert("Perfil editado con éxito. Tendrá que iniciar sesión de nuevo con sus nuevas credenciales");
        vm.closeSesion();
        $state.go('wrapper.login');
      }
    });
  }
}

```

### Fragmento de código 8: Función utilizada en *editProfileAdmin*

```

m.editarGroup = function() {
  if(vm.groupEditSelected.group == null){
    alert("Rellene todos los campos necesarios para editar el grupo");
  }else{
    AdminDataSer.comprobarSiExisteGrupo(vm.groupEditSelected.group,vm.subjectSelected.id,vm.teacherSelected.id).then(function(result){
      if(result.length != 0){ //No es correcto, el correo existe
        alert("El grupo que intenta editar ya existe con los nuevos datos introducidos");
        $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
      }else{
        AdminDataSer.editarGroup(vm.groupEditSelected.id,vm.groupEditSelected.group,vm.subjectSelected.id,vm.teacherSelected.id);
        alert("Grupo editado con éxito");
        $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
      }
    });
  }
}

```

### Fragmento de código 9: Función utilizada en *editGroup*

```

* Editar Section */
m.editarSection = function() {
  if(vm.sectionEditSelected.id == null){
    alert("Rellene todos los campos necesarios para editar el tema");
  }else{
    AdminDataSer.editarSection(vm.sectionEditSelected.id,vm.sectionEditSelected.name);
    alert("Tema editado con éxito");
    $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
  }
}

```

### Fragmento de código 10: Función utilizada en *editSection*

```

/* Editar Subject */
n.editarSubject = function() {
  if(vm.subjectEditSelected.id == null || vm.subjectEditSelected.name == null || vm.subjectEditSelected.code == null || vm.subjectEditSelected.grad
    alert("Rellene todos los campos necesarios para editar la asignatura");
  }else if(vm.subjectEditSelected.grade <= 0 || vm.subjectEditSelected.grade > 4){
    alert("El curso de la asignatura debe estar contenido entre 1 y 4, ambos incluidos");
  }else if(vm.subjectEditSelected.period <= 0 || vm.subjectEditSelected.period > 2){
    alert("El periodo de la asignatura debe estar contenido entre 1 y 2, ambos incluidos");
  }
  }else{
    AdminDataSer.editarSubject(vm.subjectEditSelected.id,vm.subjectEditSelected.name,vm.subjectEditSelected.code,vm.subjectEditSelected.grade,vm.
    alert("Asignatura editada con éxito");
    $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
  }
}

```

### Fragmento de código 11: Función utilizada en *editSubject*

```

/* Editar User */
n.editarUser = function() {
  if(vm.userEditSelected.name == null || vm.userEditSelected.surname == null || vm.userEditSelected.email == null){
    alert("Rellene todos los campos necesarios para editar el perfil");
  }else{
    /*Primero comprobamos si el correo existe*/
    UserDataSer.comprobarSiExisteCorreoID(vm.userEditSelected.id,vm.userEditSelected.email).then(function(result){
      if(result.length != 0){ //No es correcto, el correo existe
        alert("La dirección de correo ya está en uso");
        $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
      }else{
        if(vm.tabActive == 2 && !vm.userEditSelected.email.includes('@estudiante.uam.es')){
          alert("Para editar un estudiante, el correo debe seguir siendo institucional (@estudiante.uam.es)");
          $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
        }else if(!vm.userEditSelected.email.includes('@estudiante.uam.es') && !vm.userEditSelected.email.includes('@uam.es')){
          alert("Para editar un usuario (Ya sea estudiante o profesor), el correo debe seguir siendo institucional (@estudian
          $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
        }else{
          AdminDataSer.editarUser(vm.userEditSelected.id,vm.userEditSelected.name,vm.userEditSelected.surname,vm.userEditSelected.email);
          alert("Usuario editado con éxito");
          $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
        }
      }
    });
  }
}

```

### Fragmento de código 12: Función utilizada en *editUser*

```

app.post('/upload/data/', upload.single('csvdata'), function(req, res, next) {

    var fullpath = __dirname + "/uploads/" + req.file.originalname;
    var relativepath_aux = fullpath.split("\\tfg17846\\tfg17846");
    var relativepath = relativepath_aux[0];
    relativepath = relativepath.replace(/\\/g, "/");
    var filename = req.file.originalname;

    var query = "";
    switch(filename){

        case "group.csv":
            query = "LOAD DATA LOCAL INFILE '" + relativepath + "/tfg17846/tfg17846/uploads/group.csv' INTO TABLE heroku_502e45638ebd44";
            break;

        case "group_new.csv":
            var query_1 = "DELETE FROM `GROUP`";
            pool.getConnection(function(err, connection) {
                connection.query(query_1, function(err, rows){
                    connection.release();
                    var json = JSON.stringify(rows);
                });
            });
            query_1 = "ALTER TABLE `GROUP` AUTO_INCREMENT = 1";
            pool.getConnection(function(err, connection) {
                connection.query(query_1, function(err, rows){
                    connection.release();
                    var json = JSON.stringify(rows);
                });
            });
            query = "LOAD DATA LOCAL INFILE '" + relativepath + "/tfg17846/tfg17846/uploads/group.csv' INTO TABLE heroku_502e45638ebd44";
            break;

        case "user.csv":
    
```

### Fragmento de código 13: Parte de la consulta de *manageCSV*



```

        /* Matricular o desmatricular estudiante */
vm.matriculardesmatricularEstudiante = function() {
    if(auxiliarGroupSelectedID == null){
        alert("Seleccione un grupo válido");
    }
    var i = 0;
    if(auxiliarOptionsMatriculacion == 'Anular matrícula'){
        for(i = 0; i < vm.allTeachers.length; i++){
            if(vm.allTeachers[i].id==auxiliarUserSelectedID)
                auxiliarisTeacher = true;
        }
        if(auxiliarisTeacher==true){
            alert("No puede anular la matrícula de un profesor");
        }else{
            UserDataSer.desmatricularEstudiante(auxiliarUserSelectedID,auxiliarGroupSelectedID);
            alert("Matrícula anulada con éxito");
            $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
        }
    }else{
        i = 0;
        auxiliarisTeacher = false;
        for(i = 0; i < vm.allTeachers.length; i++){
            if(vm.allTeachers[i].id==auxiliarUserSelectedID)
                auxiliarisTeacher = true;
        }
        if(auxiliarisTeacher==true){
            alert("No puede matricular a un profesor");
        }else{
            UserDataSer.matricularEstudiante(auxiliarUserSelectedID,auxiliarGroupSelectedID);
            alert("Estudiante matriculado con éxito");
            $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
        }
    }
}
}
}

```

**Fragmento de código 14:** Parte de la funcionalidad de *manageEnrollment*

```

/*Creación y eliminación de grupos*/

vm.crearGrupo = function() {
  if(auxiliarGroupName == null || auxiliarSubjectSelected == null || auxiliarTeacherSelected == null){
    alert("Rellene todos los campos necesarios para crear el grupo");
  }else{

    AdminDataSer.comprobarSiExisteGrupo(auxiliarGroupName,auxiliarSubjectSelected,auxiliarTeacherSelected).then(function(result)
      if(result.length != 0){ //No es correcto, el correo existe
        alert("El grupo que intenta crear ya existe con los datos introducidos");
        $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
      }else{
        AdminDataSer.crearGrupo(auxiliarGroupName,auxiliarSubjectSelected,auxiliarTeacherSelected);
        alert("Grupo creado con éxito");
        $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
      }
    });

  }

}

vm.borrarGrupo = function() {
  if(vm.groupEditSelected == null){
    alert("Rellene todos los campos necesarios para borrar el grupo");
  }else{
    AdminDataSer.borrarAnswerGrupo(vm.groupEditSelected.id);
    AdminDataSer.borrarQuestionGrupo(vm.groupEditSelected.id);
    AdminDataSer.borrarTuitionGrupo(vm.groupEditSelected.id);
    AdminDataSer.borrarSectionGrupo(vm.groupEditSelected.id);
    AdminDataSer.borrarGrupo(vm.groupEditSelected.id);
    alert("Grupo borrado con éxito");
    $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
  }

}

}

```

**Fragmento de código 15: Parte de la funcionalidad de *manageGroup***

```

/* Section */
function getSections() {
    vm.sections = [];
    SectionDataSer.getSectionsData().then(function(sections) {
        vm.sections = _.map(sections, function(section) {
            var s = new SectionDataModel.SectionData(section);
            return s;
        });
    });
}

/*Creación y eliminación de temas*/
vm.crearTema = function() {
    if(vm.newSection.name == null || vm.newSection.group == null){
        alert("Rellene todos los campos necesarios para crear el tema");
    }else{
        AdminDataSer.crearTema(vm.newSection.name,vm.newSection.group.id);
        alert("Tema creado con éxito");
        $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
    }
}
}

```

**Fragmento de código 16:** Parte de la funcionalidad de *manageSubjectSection*

```

vm.crearAdmin = function() {
    if(vm.newAdmin.name == null || vm.newAdmin.surname == null || vm.newAdmin.email == null || vm.newAdmin.password == null ){
        alert("Rellene todos los campos necesarios para crear el administrador");
    }else{
        AdminDataSer.comprobarSiExisteCorreo(vm.newAdmin.email).then(function(result){
            if(result.length != 0){ //No es correcto, el correo existe
                alert("La dirección de correo ya está en uso");
                $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
            }else{ //El correo no existe, podemos crear el admin
                vm.newAdmin.password = cryptoJS.HmacSHA1(vm.newAdmin.password,"epsuam").toString();
                AdminDataSer.crearAdmin(vm.newAdmin.name,vm.newAdmin.surname,vm.newAdmin.email,vm.newAdmin.password);
                alert("Administrador creado con éxito");
                $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
            }
        });
    }
}

}

vm.borrarAdmin = function() {
    if(vm.AdminEditSelected == null){
        alert("Rellene todos los campos necesarios para eliminar el administrador");
    }else{

        if(vm.allAdmins.length == 1){
            alert("Error en la operación. No se puede quedar la base de datos sin administradores");
        }else{
            AdminDataSer.borrarAdmin(vm.AdminEditSelected.id);
            alert("Administrador eliminado con éxito");
            $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
        }
    }
}

}

```

**Fragmento de código 17: Parte de la funcionalidad de *manageUserAdmin***

```

/*Reset DB*/
vm.resetDB = function(){

    if(confirm("Ha seleccionado Resetear la base de datos. Con esta opción, usted eliminará toda la información de la misma.
    AdminDataSer.resetDBAnswers();
    AdminDataSer.resetDBQuestions();
    AdminDataSer.resetDBSections();
    AdminDataSer.resetDBTutions();
    AdminDataSer.resetDBGroups();
    AdminDataSer.resetDBSubjects();
    AdminDataSer.resetDBUsers();
    AdminDataSer.resetIDAnswers();
    AdminDataSer.resetIDQuestions();
    AdminDataSer.resetIDSections();
    AdminDataSer.resetIDTutions();
    AdminDataSer.resetIDGroups();
    AdminDataSer.resetIDSubjects();
    AdminDataSer.resetIDUsers();
    AdminDataSer.resetLoadSubjects();
    alert("Base de datos reseteada con éxito");
    $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
}else{
    alert("Reseteo cancelado");
    $state.go('wrapper.mainMenuAdmin.welcome', {}, {reload: true});
}
}
}

```

### Fragmento de código 18: Parte de la funcionalidad de Resetear DB

```

this.updatePassword = function(idAdmin,newPass) {
    return $http({
        url: '/updatePasswordAdmin/' + idAdmin + "/" + newPass,
        method: 'POST'
    }).then(function(result) {
        return result.data;
    });
};

this.editarPerfil = function(idAdmin,name,surname,email,password) {
    return $http({
        url: '/editarPerfilAdmin/' + idAdmin + "/" + name + "/" + surname + "/" + email + "/" + password,
        method: 'POST'
    }).then(function(result) {
        return result.data;
    });
};

```

### Fragmento de código 19: Ejemplo función en *service adminDataSer.js*

```

<form class="new-question-layout" ng-submit="vm.fnexportQuestion()">
<div class="new-question-layout">
<fieldset>
<legend><b>Exportar Pregunta</b></legend>
<div class="new-question-text-content">
<span class="new-question-select-title"><b>Seleccione en el panel izquierdo el GRUPO y TEMA de destino para la pregunta que se vaya a exp
</div>
<div class="new-question-select">
<span class="new-question-select-title">Seleccione la Pregunta de entre todas las disponibles de la aplicación:</span>
<select ng-show="vm.allQuestions.length > 0" ng-options="item as item.title for item in vm.allQuestions track by item.id" ng-model=
<select ng-show="vm.allQuestions.length === 0"><option>Todavía no existen preguntas en la aplicación</option></select>
</div>

<div class="new-question-text-content">
<span class="new-question-text-title">Enunciado de la pregunta a exportar:</span>
<textarea ng-disabled="vm" class="new-question-text-area" ng-model="vm.exportQuestion.text" required></textarea>
</div>

<div class="new-question-answers-options">
<span class="new-question-answers-options-title">Número de respuestas de la pregunta: </span>
<div class="new-question-answers-content">
<select ng-disabled="vm" ng-show="vm.exportQuestion.numAnswersName != undefined" ng-options="item as item for item in vm.c
<select ng-disabled="vm" ng-show="vm.exportQuestion.numAnswersName === undefined"><option>Seleccione una pregunta</option>
</div>
</div>
<div class="new-question-answers-options">
<span class="new-question-answers-options-title">¿Tarea o presencial?: </span>
<div class="new-question-answers-content">
<select ng-disabled="vm" ng-show="vm.exportQuestion.homework != undefined" ng-options="item as item for item in vm.homewc
<select ng-disabled="vm" ng-show="vm.exportQuestion.homework === undefined"><option>Seleccione una pregunta</option></sel
</div>
</div>

<div class="new-question-answers-content">
<span class="new-question-answers-title">Respuestas:</span>
<div class="new-question-answers-answer" ng-show="vm.exportQuestion.numAnswers >= 1" >
<label class="new-question-answers-answer-label">A</label>
<input type="text" ng-disabled="vm" class="new-question-answers-answer-input" ng-model="vm.exportQuestion.answerA" required
</div>
<div class="new-question-answers-answer" ng-show="vm.exportQuestion.numAnswers >= 1" >
<label class="new-question-answers-answer-label">B</label>
<input type="text" ng-disabled="vm" class="new-question-answers-answer-input" ng-model="vm.exportQuestion.answerB" required

```

## Fragmento de código 20: Parte del formulario de exportar pregunta

```

/*Conseguir todas las preguntas para funcionalidad de exportar pregunta*/
function getAllQuestions() {
    vm.allQuestions = [];
    QuestionDataSer.getQuestionsData().then(function(questions) {
        vm.allQuestions = _.map(questions, function(question, index) {
            var q = new QuestionDataModel.QuestionData(question);
            q.title = question.code + ' - ' + question.group + ' (' + question.
            return q;
        });
    });
};

```

**Fragmento de código 21: Carga de todas las preguntas en el objeto *vm.allQuestions***

```

function getAfkStudentsOfQuestion(id) {
    vm.afkStudents = [];
    auxAfkStudents=0;
    UserDataSer.afkStudentsOfQuestion(id).then(function(students) {
        vm.afkStudents = _.map(students, function(item, index) {
            var s = new UserDataModel.UserData(item);
            s.title = (index + 1) + ' - ' + s.surname + ', ' + s.name;
            return s;
        });
        if(vm.afkStudents.length!=0){
            auxAfkStudents = vm.afkStudents.length;
            if(vm.tabActive==0){
                calculateQuestionRates();
            }else{
                calculateStudentRates();
            }

        }else{
            auxAfkStudents=0;
        }
    });
};

```

**Fragmento de código 22: Función para conseguir los alumnos ausentes**

```

function calculateQuestionRates() {
    if(vm.questionSelected) {
        vm.rates = [0, 0, 0, 0];
        vm.colors
        _.each(vm.answers, function(answer) {
            if(answer.selection === null) {
                vm.rates[2]++;
            } else if(answer.selection === vm.questionSelected.solution) {
                vm.rates[0]++;
            } else {
                vm.rates[1]++;
            }
        });
        vm.rates[3] = auxAfKStudents;
        vm.rates[2] = vm.students.length - vm.rates[0] - vm.rates[1] - vm.rates[3];
        vm.sum = vm.rates[0] + vm.rates[1] + vm.rates[2] + vm.rates[3];
        vm.percentRight = Math.round((vm.rates[0] * 100 / vm.sum) * 100) / 100;
        vm.percentWrong = Math.round((vm.rates[1] * 100 / vm.sum) * 100) / 100;
        vm.percentNoAnswer = Math.round((vm.rates[2] * 100 / vm.sum) * 100) / 100;
        vm.percentAfK = Math.round((vm.rates[3] * 100 / vm.sum) * 100) / 100;
        vm.labels = [vm.percentRight + '%', vm.percentWrong + '%', vm.percentNoAnswer + '%', vm.percentAfK + '%'];
    }
}
/* Questions stats */

```

### Fragmento de código 23: Función para conseguir el gráfico estadístico

```

<div class="question-stats-graph" ng-show="vm.questions.length > 0 && vm.answers.length > 0 && vm.students.length > 1 && vm.questionSel

<canvas id="pie" class="chart chart-pie"
    chart-data="vm.rates" chart-labels="vm.labels" chart-legend="true" chart-colours="vm.colors">
</canvas>

<div class="question-stats-graph-percents">
    <div class="question-stats-graph-percent">
        Respuestas Correctas {{vm.rates[0]}}
    </div>
    <div class="question-stats-graph-percent">
        Respuestas Incorrectas {{vm.rates[1]}}
    </div>
    <div class="question-stats-graph-percent">
        No sabe {{ vm.rates[2] }}
    </div>
    <div class="question-stats-graph-percent">
        No asistió {{vm.rates[3]}}
    </div>
    <div class="question-stats-graph-percent">
        <span>Total de Respuestas: </span>{{vm.students.length - vm.rates[3]}}/{{vm.students.length}}
    </div>
</div>
</div>

```

### Fragmento de código 24: div que genera el gráfico estadístico

```

<legend><b>Pase de Lista</b></legend>
<span>Aquí se mostrarán los estudiantes que <b>NO</b> contestaron</span> a la pregunta, porque o bien no asistieron, o porque no pulsaron nin
<p><span class="question-stats-select-title" ng-show="vm.afkStudents.length > 0"><b><u>Estudiantes que no contestaron a esta pregunta:</u></b></span>
<p><span class="question-stats-select-title" ng-show="vm.afkStudents.length === 0"><b><u>A esta pregunta contestaron todos los alumnos</u></b></span>
<p><ul ng-disabled="vm" ng-show="vm.afkStudents.length > 0" class="new-question-text-area" ng-repeat="student in vm.afkStudents track i
</fieldset>

```

### Fragmento de código 25: div que genera la lista de estudiantes ausentes



```

<fieldset>
<legend><b>Actualizar Contraseña</b></legend>

<form ng-submit="vm.updatePassword()">
<div id = "correo">
<span>Correo Institucional</span>
  <input type="text" name="email" size="35" ng-model="vm.email"><br>
</div>
  <div id = "passold">
<span>Contraseña actual</span>
  <input type="password" name="actualPassword" size="35" ng-model="vm.oldpass"><br>
  </div>
    <div id = "passnew">
<span>Contraseña nueva</span>
  <input type="password" name="newPassword" size="35" ng-model="vm.newpass"><br>
  </div>

  <div id = "updatePass">
    <button>
      <input type="submit" value="Actualizar Contraseña" class="text-center">
      
    </button>
  </div>
</form>

</fieldset>

```

**Fragmento de código 26: *updatePassword.HTML***

```

public void updatePassword(View view) { /*Esto nos sirve para actualizar el password desde la APK*/

    WebView mywebView;
    mywebView = (WebView) findViewById(R.id.webView);
    mywebView.setVisibility(View.VISIBLE);
    View button_1 = findViewById(R.id.accept_login_button);
    button_1.setVisibility(View.GONE);
    View button_2 = findViewById(R.id.updatePassword);
    button_2.setVisibility(View.GONE);
    WebSettings webSettings = mywebView.getSettings();
    webSettings.setJavaScriptEnabled(true);
    webSettings.setDomStorageEnabled(true);
    webSettings.setLoadWithOverviewMode(true);
    webSettings.setUseWideViewPort(true);
    mywebView.setScrollBarStyle(WebView.SCROLLBARS_OUTSIDE_OVERLAY);
    mywebView.setScrollbarFadingEnabled(false);
    mywebView.getSettings().setSupportZoom(true);
    mywebView.getSettings().setBuiltInZoomControls(true);

    mywebView.loadUrl("https://serene-caverns-11982.herokuapp.com/#/updatePassword");

    mywebView.setWebChromeClient(new WebChromeClient());

}

```

### Fragmento de código 27: *updatePassword en Android*

```

/*Creación y eliminación de temas*/
vm.crearTema = function() {
    if(auxiliarnewSectionName == null || auxiliarnewSectionGroup == null){
        alert("Rellene todos los campos necesarios para crear el tema");
    }else{
        AdminDataSer.crearTema(auxiliarnewSectionName,auxiliarnewSectionGroup);
        alert("Tema creado con éxito");
        $state.go('wrapper.mainMenu.welcome', {}, {reload: true});
    }
}
}

```

### Fragmento de código 28: *Función para crear tema*

```

vm.borrarTema = function() {
    if(vm.sectionEditSelected == null){
        alert("Rellene todos los campos necesarios para eliminar el tema");
    }else{
        /*Primero eliminamos las respuestas asociadas a las preguntas que tienen ese tema*/
        SectionDataSer.borrarRespuestasByTema(vm.sectionEditSelected.id);
        SectionDataSer.borrarPreguntasByTema(vm.sectionEditSelected.id);
        SectionDataSer.borrarTema(vm.sectionEditSelected.id);
        alert("Tema eliminado con éxito");
        $state.go('wrapper.mainMenu.welcome', {}, {reload: true});
    }
}
}

```

### Fragmento de código 29: *Función para borrar tema*

```

public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;

    //Creamos el geofence con el círculo de área (EPS)
    LatLng question_area = new LatLng(EPS.get(0), EPS.get(1)); //Latitud, Longitud
    mMap.addCircle(new CircleOptions()
        .center(question_area)
        .radius(80) // en metros
        .strokeColor(Color.BLUE)
        .fillColor(0x220000FF)
        .strokeWidth(0.015f)
        .visible(false) //Queremos que el área presencial no sea visible para el estudiante
    );

    //Aquí preguntamos mediante GeoQuery nuestra posición, si es correcta respecto del geofence
    GeoQuery geoQuery = geoFire.queryAtLocation(new GeoLocation(question_area.latitude, question_area.longitude), 0.015f);
    geoQuery.addGeoQueryEventListener(new GeoQueryEventListener() {
        @RequiresApi(api = Build.VERSION_CODES.JELLY_BEAN)
        @Override
        public void onKeyEntered(String key, GeoLocation location) {

            if (key == userId) {

                QUESTION_AREA_AVAILABLE = true;
                Intent returnIntent = new Intent();
                returnIntent.putExtra("result", QUESTION_AREA_AVAILABLE);
                setResult(Activity.RESULT_OK, returnIntent);
                finish();

            }
        }

        @RequiresApi(api = Build.VERSION_CODES.JELLY_BEAN)
        @Override
        public void onKeyExited(String key) {

            if (key == userId) {

                QUESTION_AREA_AVAILABLE = false;
                Intent returnIntent = new Intent();
                returnIntent.putExtra("result", QUESTION_AREA_AVAILABLE);
                setResult(Activity.RESULT_CANCELED, returnIntent);
                finish();
            }
        }
    });
}

```

### Fragmento de código 30: *Implementación de un geofence para la facultad*

```

/**
 * checkPlayServices
 * <p>
 * Comprueba si está conectado a PlayServices de Google
 *
 * @return Devuelve un booleano con el resultado
 */
private boolean checkPlayServices() {
    int resultCode = GooglePlayServicesUtil.isGooglePlayServicesAvailable(this);
    if (resultCode != ConnectionResult.SUCCESS) {

        if (GooglePlayServicesUtil.isUserRecoverableError(resultCode))
            GooglePlayServicesUtil.getErrorDialog(resultCode, this, PLAY_SERVICES_RESOLUTION_REQUEST).show();
        else {
            Toast.makeText(this, "This device is not supported", Toast.LENGTH_SHORT).show();
            finish();
        }
        return false;
    }
    return true;
}

```

### **Fragmento de código 31: Conexión al PlayServices de Google**

```

/**
 * buildGoogleApiClient
 * <p>
 * Este método se encarga de conectar a la API de Google
 */
private void buildGoogleApiClient() {
    mGoogleApiClient = new GoogleApiClient.Builder(this)
        .addConnectionCallbacks(this)
        .addOnConnectionFailedListener(this)
        .addApi(LocationServices.API)
        .build();
    mGoogleApiClient.connect();
}

```

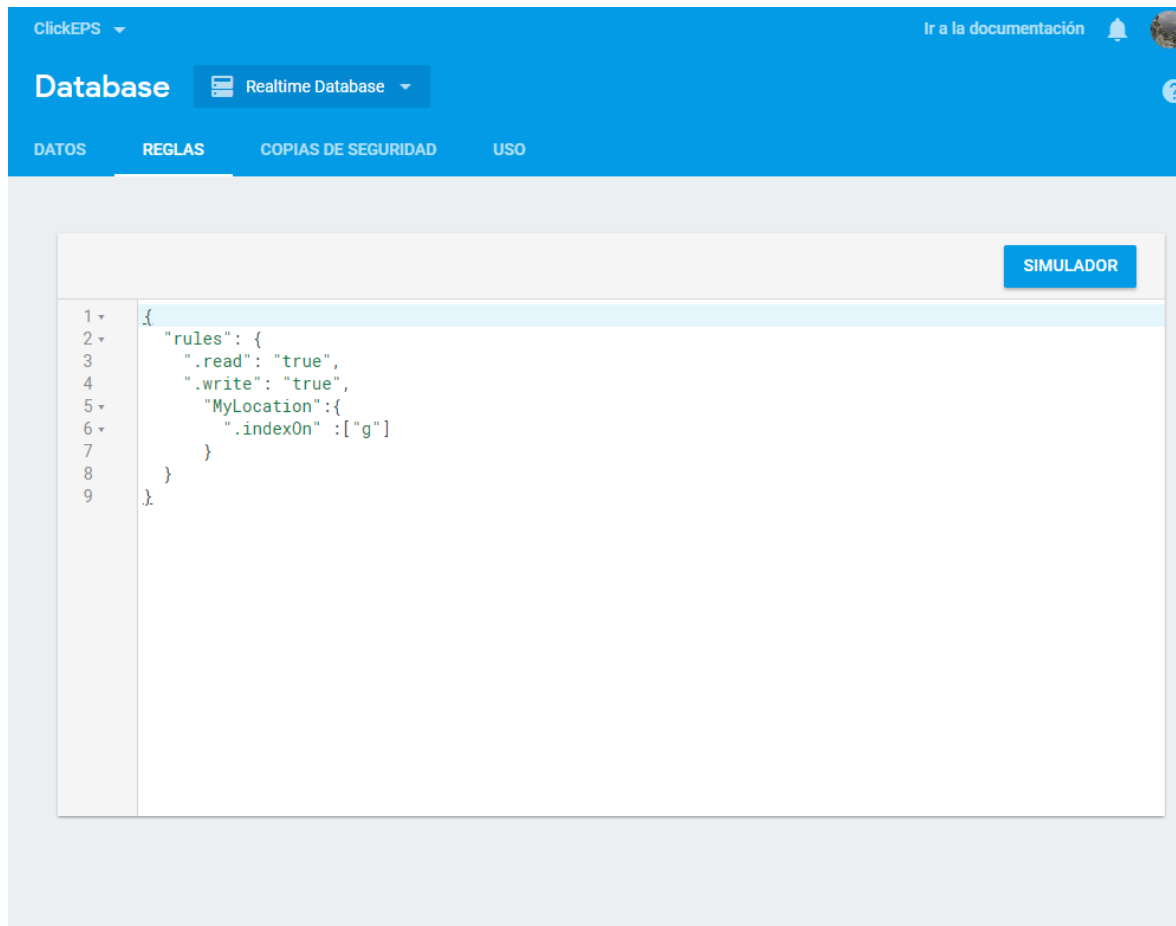
### **Fragmento de código 32: Conexión a la API de Google**

```

ref = FirebaseDatabase.getInstance().getReference("MyLocation");
geoFire = new GeoFire(ref);
ref.child(userId).child("userId").setValue(userId);

```

### **Fragmento de código 33: Objeto geoFire y referencia a Firebase**



**Fragmento de código 34: Reglas de la base de datos de Firebase**

```
//Actualizamos para Firebase
geoFire.setLocation(userId, new GeoLocation(latitude, longitude),
    new GeoFire.CompletionListener() {
        @Override
        public void onComplete(String key, DatabaseError error) {

            if(key == userId){
                //Añadimos marcador
                if (mCurrent != null)
                    mCurrent.remove(); // eliminamos el antiguo marcador
                mCurrent = mMap.addMarker(new MarkerOptions()
                    .position(new Latlng(latitude, longitude))
                    .title(userId));

                //Movemos la cámara a la nueva posición
                mMap.animateCamera(CameraUpdateFactory.newLatLngZoom(new Latlng(latitude, longitude), 12.0f));
            }
        }
    });
```

**Fragmento de código 35: Actualizaciones a base de datos de Firebase**

```

LocationCallback comprobarLocation = new LocationCallback() {
    @Override
    public void onLocationResult(String key, GeoLocation location) {

        if(key == userId){

            actualLocation.setLatitude(location.latitude);
            actualLocation.setLongitude(location.longitude);

            float distanceInMeters = targetLocation.distanceTo(actualLocation);

            if (distanceInMeters <= 80) { //80 es la distancia en metros que ocupa el radio del geofence que hemos generado
                // Estaremos dentro del geofence
                Toast.makeText(MapsActivity.this, "Estás en la zona presencial", Toast.LENGTH_LONG).show();
                QUESTION_AREA_AVAILABLE = true;
                Intent returnIntent = new Intent();
                returnIntent.putExtra("result", QUESTION_AREA_AVAILABLE);
                setResult(Activity.RESULT_OK, returnIntent);
                finish();
            }

        }

    }

    @Override
    public void onCancelled(DatabaseError databaseError) {

        // No hacemos nada
    }

};

geoFire.getLocation(userId.comprobarLocation):

```

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {

    if (requestCode == 1) {
        /*Comprobación del MockLocation*/
        Context context;
        context = getApplicationContext();
        boolean isMockSettingON = isMockSettingsON(context);
        boolean areThereMockPermissionApps = areThereMockPermissionApps(context);
        if (resultCode == MapsActivity.RESULT_OK) {
            if (areThereMockPermissionApps == true || isMockSettingON == true) {

                new android.app.AlertDialog.Builder(this).setMessage("Existen aplicaciones en su dispositivo que podrían generar una ubicación falsa.")
                    .setNeutralButton("Aceptar", new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                            finish();
                        }
                    }).create().show();

            }
        } else {
            new AlertDialog.Builder(this).setMessage("No puede responder a esta pregunta porque es una pregunta presencial. Si está en una ubicación falsa, ¿cómo puede responder a esta pregunta?")
                .setNeutralButton("Aceptar", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        finish();
                    }
                }).create().show();
        }
    }
}

//onActivityResult
```

```

/**
 * isMockSettingsON
 * <p>
 * Comprueba que mock Location está o no activado
 * Esto sirve principalmente para evitar que los estudiantes simulen el GPS mediante aplicaciones de terceros
 *
 * @param context
 * @return booleano con el resultado
 */
public static boolean isMockSettingsON(Context context) {
    // Devuelve true si mock location está activado, false si no.
    if (Settings.Secure.getString(context.getContentResolver(),
        Settings.Secure.ALLOW MOCK_LOCATION).equals("0"))
        return false;
    else
        return true;
}

```

### Fragmento de código 38: *isMockSettingsOn*

```

public static boolean areThereMockPermissionApps(Context context) {
    int count = 0;

    PackageManager pm = context.getPackageManager();
    List<ApplicationInfo> packages =
        pm.getInstalledApplications(PackageManager.GET_META_DATA);

    for (ApplicationInfo applicationInfo : packages) {
        try {
            PackageInfo packageInfo = pm.getPackageInfo(applicationInfo.packageName,
                PackageManager.GET_PERMISSIONS);

            // Conseguir permisos
            String[] requestedPermissions = packageInfo.requestedPermissions;

            if (requestedPermissions != null) {
                for (int i = 0; i < requestedPermissions.length; i++) {
                    if (requestedPermissions[i]
                        .equals("android.permission.ACCESS MOCK_LOCATION")
                        && !applicationInfo.packageName.equals(context.getPackageName())) {
                        count++;
                    }
                }
            }
        } catch (PackageManager.NameNotFoundException e) {
            Log.e("Got exception ", e.getMessage());
        }
    }

    if (count > 0)
        return true;
    return false;
}

```

### Fragmento de código 39: *areThereMockPermissionApps*

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    thread.start();

    super.onCreate(savedInstanceState);
    getSupportActionBar().setDisplayHomeAsUpEnabled(true);
    setContentView(R.layout.activity_login);

    sharedPreferences = getSharedPreferences(PREF_NAME, Context.MODE_PRIVATE);
    editor = sharedPreferences.edit();
    usernameEditText = (EditText) findViewById(R.id.username_login_field);
    passwordEditText = (EditText) findViewById(R.id.password_login_field);
    progressBar = (ProgressBar) findViewById(R.id.progress_bar);
    rem_userpass = (CheckBox) findViewById(R.id.checkBox);

    if (sharedPreferences.getBoolean(KEY_REMEMBER, false))
        rem_userpass.setChecked(true);
    else
        rem_userpass.setChecked(false);

    usernameEditText.setText(sharedPreferences.getString(KEY_USERNAME, ""));
    passwordEditText.setText(sharedPreferences.getString(KEY_PASS, ""));

    usernameEditText.addTextChangedListener(this);
    passwordEditText.addTextChangedListener(this);
    rem_userpass.setOnCheckedChangeListener(this);
}

```

#### Fragmento de código 40: Implementación de *Shared Preferences*



```

@Override
public void onTextChanged(CharSequence charSequence, int i, int i1, int i2) {
    managePrefs();
}

@Override
public void afterTextChanged(Editable editable) {

}

@Override
public void onCheckedChanged(CompoundButton compoundButton, boolean b) {
    managePrefs();
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case android.R.id.home:
            onBackPressed();
            return true;
    }
    return super.onOptionsItemSelected(item);
}

/**
 * managePrefs
 * <p>
 * Método que gestiona la opción de recordar el correo y contraseña siempre que entremos en la aplicación
 */
private void managePrefs() {
    if (rem_userpass.isChecked()) {
        editor.putString(KEY_USERNAME, usernameEditText.getText().toString().trim());
        editor.putString(KEY_PASS, passwordEditText.getText().toString().trim());
        editor.putBoolean(KEY_REMEMBER, true);
        editor.apply();
    } else {
        editor.putBoolean(KEY_REMEMBER, false);
        editor.remove(KEY_PASS);
        editor.remove(KEY_USERNAME);
        editor.apply();
    }
}
}

```

#### **Fragmento de código 41: Implementación de métodos de *Shared Preferences***

```

public class MainActivity extends AppCompatActivity {
    private WebView mywebView;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mywebView = (WebView) findViewById(R.id.webView);
        WebSettings webSettings = mywebView.getSettings();
        webSettings.setJavaScriptEnabled(true);
        webSettings.setDomStorageEnabled(true);
        webSettings.setLoadWithOverviewMode(true);
        webSettings.setUseWideViewPort(true);
        mywebView.setScrollBarStyle(WebView.SCROLLBARS_OUTSIDE_OVERLAY);
        mywebView.setScrollbarFadingEnabled(false);
        mywebView.getSettings().setSupportZoom(true);
        mywebView.getSettings().setBuiltInZoomControls(true);
        if(savedInstanceState==null){ /*De esta forma evitamos que el webView se reinicie cada vez que giramos la pantalla*/
            mywebView.loadUrl("https://serene-caverns-11982.herokuapp.com/");
        }
        mywebView.setWebChromeClient(new WebChromeClient());
    }

    @Override
    public void onBackPressed() {
        if(mywebView.canGoBack()){
            mywebView.goBack();
        }else {
            super.onBackPressed();
        }
    }

    @Override
    protected void onSaveInstanceState(Bundle outState )
    {
        super.onSaveInstanceState(outState);
        mywebView.saveState(outState);
    }
}

```

## Fragmento de código 42: Implementación sencilla de ClickEPSAdministración

```

if (question.getAnswered() == true && now.before(question.getExpiration())) { //Mediante esta comprobación evitamos ver resultados
    new android.app.AlertDialog.Builder(this).setMessage("Esta pregunta aún no ha finalizado. No puede ver los resultados. Vuelva a intentarlo").setNeutralButton("Aceptar", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            finish();
        }
    }).create().show();
} else {

    if (question.getAnswered() == false && now.before(question.getExpiration())) { //Pregunta abierta

        if (question.getHomework() == 0) { //Si la pregunta es presencial, pregunta por la ubicación

            //Comprobación de que estamos en la ubicación de la EPS
            Intent intent = new Intent("es.uam.eps.tfg17846.mariopolo2805.clickeps.MAPSACTIVITY");
            startActivityForResult(intent, 1);

        }
    }
}

```

## Fragmento de código 43: Parte de ocultar respuesta hasta expiración

```

        if (question.getNumAnswers().equals("1")) {
            answerCRadio.setVisibility(View.GONE);
            answerDRadio.setVisibility(View.GONE);
        } else if (question.getNumAnswers().equals("2")) {
            answerDRadio.setVisibility(View.GONE);
        }
    }
}

```

### Fragmento de código 44: Parte de la implementación de *numAnswers*

```

case R.id.explanationbutton:
    Date today = new Date();
    AlertDialog.Builder submitDialogBuilder_explanation = new AlertDialog.Builder(this);

    if(question.getExpiration().before(today)){
        String explanation = question.getExplanation();
        submitDialogBuilder_explanation.setTitle("Explicación de la respuesta");
        if(explanation.contentEquals("empty") || explanation.contentEquals("null") ){
            submitDialogBuilder_explanation.setMessage("El profesor no ha añadido una explicación a la respuesta");
        }else{
            submitDialogBuilder_explanation.setMessage(question.getExplanation());
        }
        submitDialogBuilder_explanation.setNeutralButton("Aceptar", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                dialog.dismiss();
            }
        });
        AlertDialog submitDialog = submitDialogBuilder_explanation.create();
        submitDialog.show();
    }else{
        submitDialogBuilder_explanation.setTitle("Explicación de la respuesta");
        submitDialogBuilder_explanation.setMessage("No podrá ver la explicación de la respuesta hasta que no finalice la pregun
        submitDialogBuilder_explanation.setNeutralButton("Aceptar", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                dialog.dismiss();
            }
        });
        AlertDialog submitDialog = submitDialogBuilder_explanation.create();
        submitDialog.show();
    }
}

break;

```

### Fragmento de código 45: Implementación de la explicación de las preguntas

```

/**
 * refresh
 * <p>
 * Este método actualiza la lista de las questiones, utilizado en el caso en que
 *
 * @param v
 */
public void refresh(View v) { /*Método para refrescar lista de cuestiones*/
    finish();
    overridePendingTransition(0, 0);
    startActivity(getIntent());
    overridePendingTransition(0, 0);
}

```

#### Fragmento de código 46: Implementación del evento *refresh*

```

@RequiresApi(api = Build.VERSION_CODES.JELLY_BEAN)
public void NotificarDisponibles() {
    Context context;
    context = getApplicationContext();
    group = (Group) getIntent().getExtras().getSerializable(Constants.GROUP_KEY);
    Intent intent = new Intent(context, Question.class);
    PendingIntent pi = PendingIntent.getActivity(context, 0, intent, 0);
    Notification.Builder builder = new Notification.Builder(this);
    builder.setContentTitle(";Preguntas Disponibles!")
        .setContentText("Aún tiene preguntas sin responder")
        .setSmallIcon(R.drawable.android_push_notification_icon)
        .setLargeIcon(BitmapFactory.decodeResource(getResources(), R.mipmap.launcher))
        .setContentIntent(pi)
        .setVibrate(new long[]{Notification.DEFAULT_VIBRATE})
        .setPriority(Notification.PRIORITY_MAX);
    NotificationManager notificationManager = (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
    notificationManager.notify(0, builder.build());
}

```

#### Fragmento de código 47: Notificar disponibles

## E Capturas de pantalla de ClickEPS

En esta sección aparecerán en primer lugar Figuras que corresponden a la aplicación web y por último a la aplicación móvil.

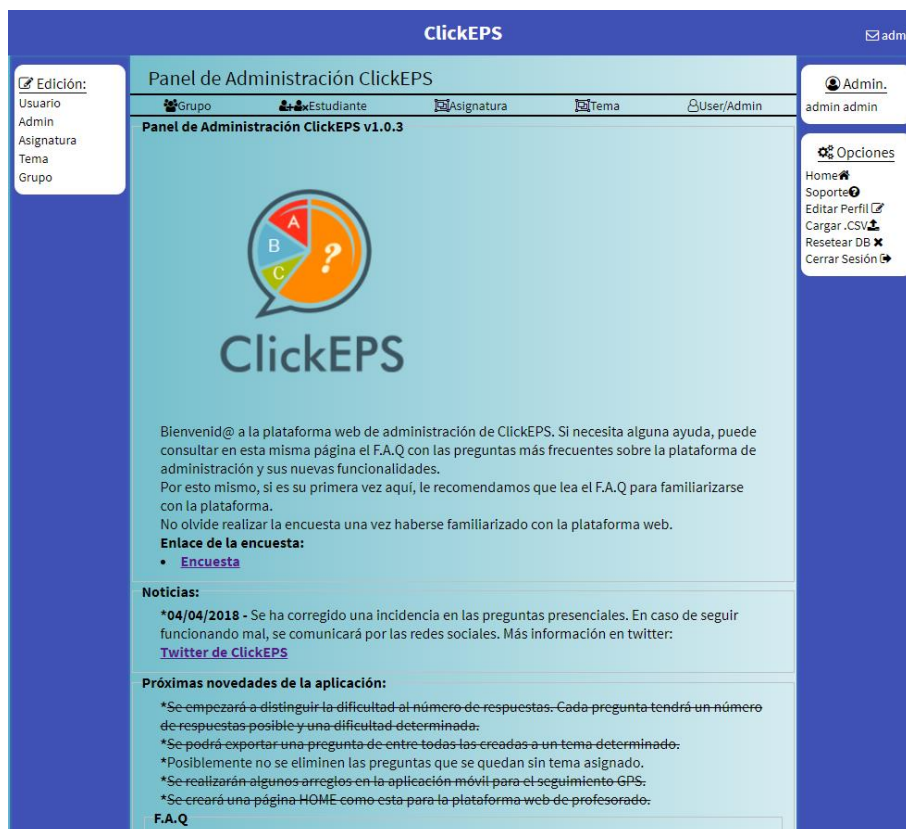


Figura 31: Página principal de la interfaz de administración con *welcome*



✉ admin

# ClickEPS

Edición:

- Usuario
- Admin
- Asignatura
- Tema
- Grupo

## Panel de Administración ClickEPS

Grupo
 xEstudiante
 Asignatura
 Tema
 User/Admin

### Editar Grupo

Seleccionar Grupo a editar:

Siglas del Grupo:

Seleccione una nueva asignatura:

Seleccione un nuevo profesor:

**Guardar Cambios**

Admin.

admin admin

---

Opciones

Home

Soporte

Editar Perfil

Cargar .CSV

Resetear DB

Cerrar Sesión

### Figura 34: Editar Grupo (*editGroup*)

Edición:

Usuario

Admin

Asignatura

Tema

Grupo

Panel de Administración ClickEPS

Grupo

Estudiante

Asignatura

Tema

User/Admin

Editar Tema

Seleccionar Tema a editar:

Nombre del tema:

Por ejemplo: Introducción a la asignatura

Guardar Cambios

\* Solamente es editable el nombre del tema debido a la facilidad de creación y eliminación

Admin.

admin admin

Opciones

Home

Soporte

Editar Perfil

Cargar .CSV

Resetear DB

Cerrar Sesión

**Figura 35: Editar Tema (*editSection*)**

Edición:

Usuario

Admin

Asignatura

Tema

Grupo

Panel de Administración ClickEPS

Grupo

Estudiante

Asignatura

Tema

User/Admin

Editar Asignatura

Seleccionar Asignatura a editar:

Nombre de la Asignatura:

Por ejemplo: Álgebra

Siglas de la asignatura:

Por ejemplo: ALG

Curso:

Debe elegir un número de 1 a 4

Cuatrimestre:

Debe elegir un número de 1 a 2

Guardar Cambios

\* Por cuestiones de diseño, no es editable el id de la asignatura

Admin.

admin admin

Opciones

Home

Soporte

Editar Perfil

Cargar .CSV

Resetear DB

Cerrar Sesión

Edición:

Usuario

Admin

Asignatura

Tema

Grupo

ClickePS

admin

Admin.

admin admin

Opciones

Home

Soporte

Editar Perfil

Cargar .CSV

Resetear DB

Cerrar Sesión

Panel de Administración ClickePS

Grupo

Estudiante

Asignatura

Tema

User/Admin

Editar Usuario (Estudiante o Profesor)

Seleccionar Usuario a editar:

Nombre:

Por ejemplo: Andrea

Apellidos:

Por ejemplo: Rodríguez Amado

Correo Institucional:

Si es profesor, no olvide utilizar el correo institucional @uam.es

Guardar Cambios

Actualizar Contraseña de Usuario (Estudiante o Profesor)

Seleccionar Usuario a editar:

Contraseña:

Inserte la nueva contraseña

Guardar Cambios





Figura 38: Cargar CSV (*manageCSV*)

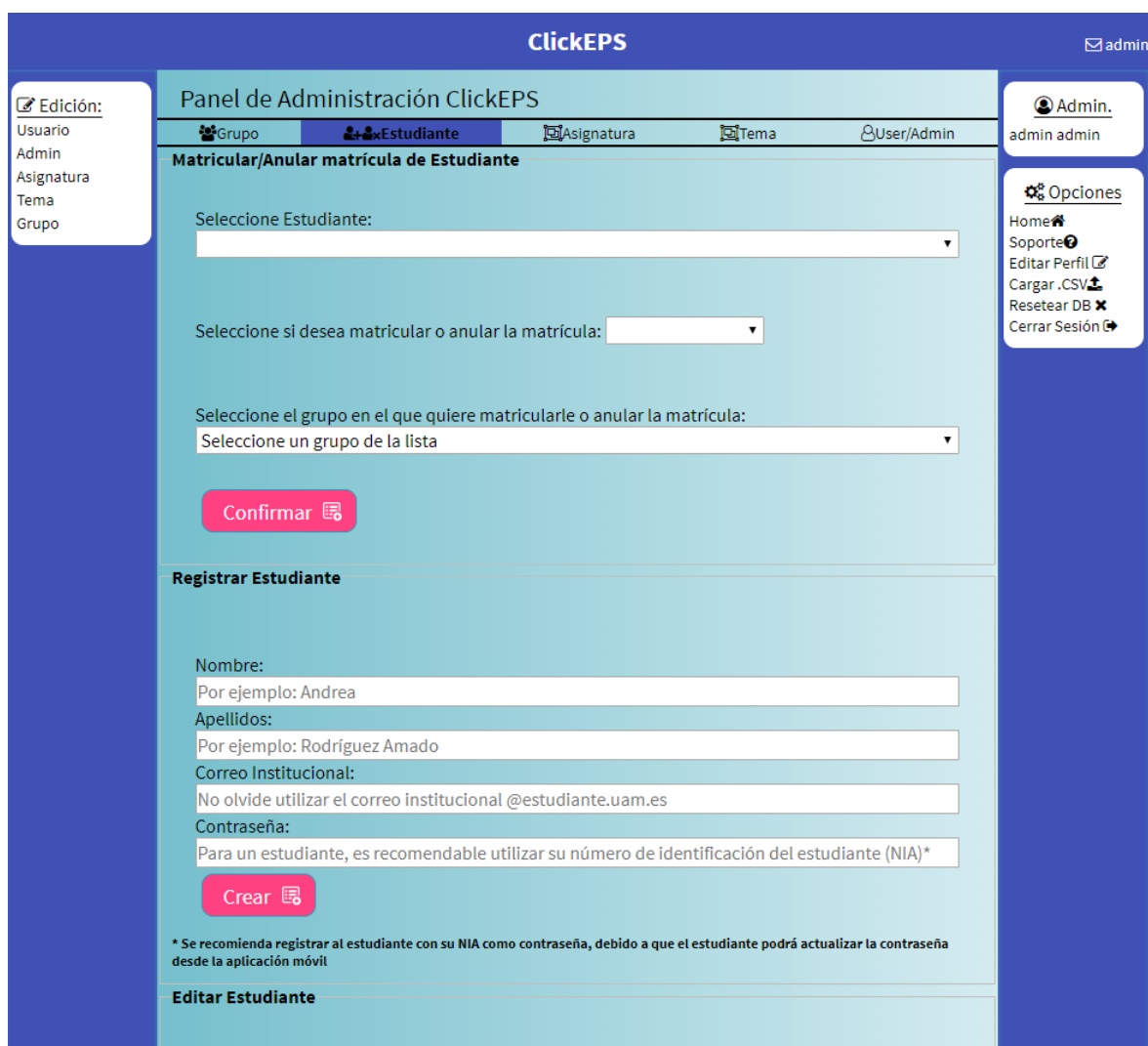


Figura 39: Parte 1 de *manageEnrollment*

### Editar Estudiante

Seleccionar Estudiante a editar:

Nombre:

Por ejemplo: Andrea

Apellidos:

Por ejemplo: Rodríguez Amado

Correo Institucional:

No olvide utilizar el correo institucional @estudiante.uam.es

Guardar Cambios

\* Por cuestiones de seguridad se ha decidido mostrar el campo password actual encriptado

### Actualizar Contraseña

Seleccionar Estudiante a editar:

Contraseña:

Inserte la nueva contraseña

Guardar Cambios

\* Por cuestiones de seguridad se ha decidido mostrar el campo password actual encriptado

### Eliminar datos de Estudiante

Estudiante a eliminar (Atención, no sólo eliminará el estudiante, sino todo lo relacionado con él)

Eliminar

Figura 40: Parte 2 de *manageEnrollment*

ClickEPS

admin

Edición:

Usuario

Admin

Asignatura

Tema

Grupo

Panel de Administración ClickEPS

Grupo

Estudiante

Asignatura

Tema

User/Admin

Crear Grupo

Introduzca las siglas del grupo:

Por ejemplo: G231

Asignatura del grupo:

Profesor del grupo:

Crear

Editar Grupo

Seleccionar Grupo a editar:

Siglas del Grupo:

Por ejemplo: G231

Seleccione una nueva asignatura:

Seleccione un nuevo profesor:

Guardar Cambios

Eliminar datos de Grupo

Grupo a eliminar (Atención, no sólo eliminará el grupo, sino todo lo relacionado con el)

Eliminar

Admin.

admin admin

Opciones

Home

Soporte

Editar Perfil

Cargar .CSV

Resetear DB

Cerrar Sesión

Figura 41: Grupo (*manageGroup*)

XLIX

Admin

Asignatura

Tema

Grupo

Crear Asignatura

Introduzca el código de la asignatura:

Es el código que viene en la guía docente de la misma. Ejemplo: 72635

Nombre de la asignatura:

Por ejemplo: Álgebra

Siglas de la asignatura:

Por ejemplo: ALG

Curso:

Cuatrimestre:

Crear

Editar Asignatura

Seleccionar Asignatura a editar:

Nombre de la Asignatura:

Por ejemplo: Álgebra

Siglas de la asignatura:

Por ejemplo: ALG

Curso:

Debe elegir un número de 1 a 4

Cuatrimestre:

Debe elegir un número de 1 a 2

Guardar Cambios

\* Por cuestiones de diseño, no es editable el id de la asignatura

Eliminar datos de Asignatura

Asignatura a eliminar (Atención, no sólo eliminará la asignatura, sino todo lo relacionado con ella)

Eliminar

Opciones

Home

Soporte

Editar Perfil

Cargar .CSV

Resetear DB

Cerrar Sesión

Figura 42: Asignatura (*manageSubjectSection*)

L

ClickEPS

admin

Edición:

Usuario

Admin

Asignatura

Tema

Grupo

Panel de Administración ClickEPS

Grupo

Estudiante

Asignatura

Tema

User/Admin

Crear Tema

Introduzca el nombre del tema:

Por ejemplo: Tema 1: Introducción a la asignatura

Grupo que se le asigna al tema:

Crear

Editar Tema

Seleccionar Tema a editar:

Nombre del tema:

Por ejemplo: Introducción a la asignatura

Guardar Cambios

\* Solamente es editable el nombre del tema debido a la facilidad de creación y eliminación

Eliminar datos de Tema

Tema a eliminar (Atención, no sólo eliminará el tema, sino todo lo relacionado con él)

Eliminar

Admin.

admin admin

Opciones

Home

Soporte

Editar Perfil

Cargar .CSV

Resetear DB

Cerrar Sesión

Figura 43: Tema (*manageSubjectSection*)





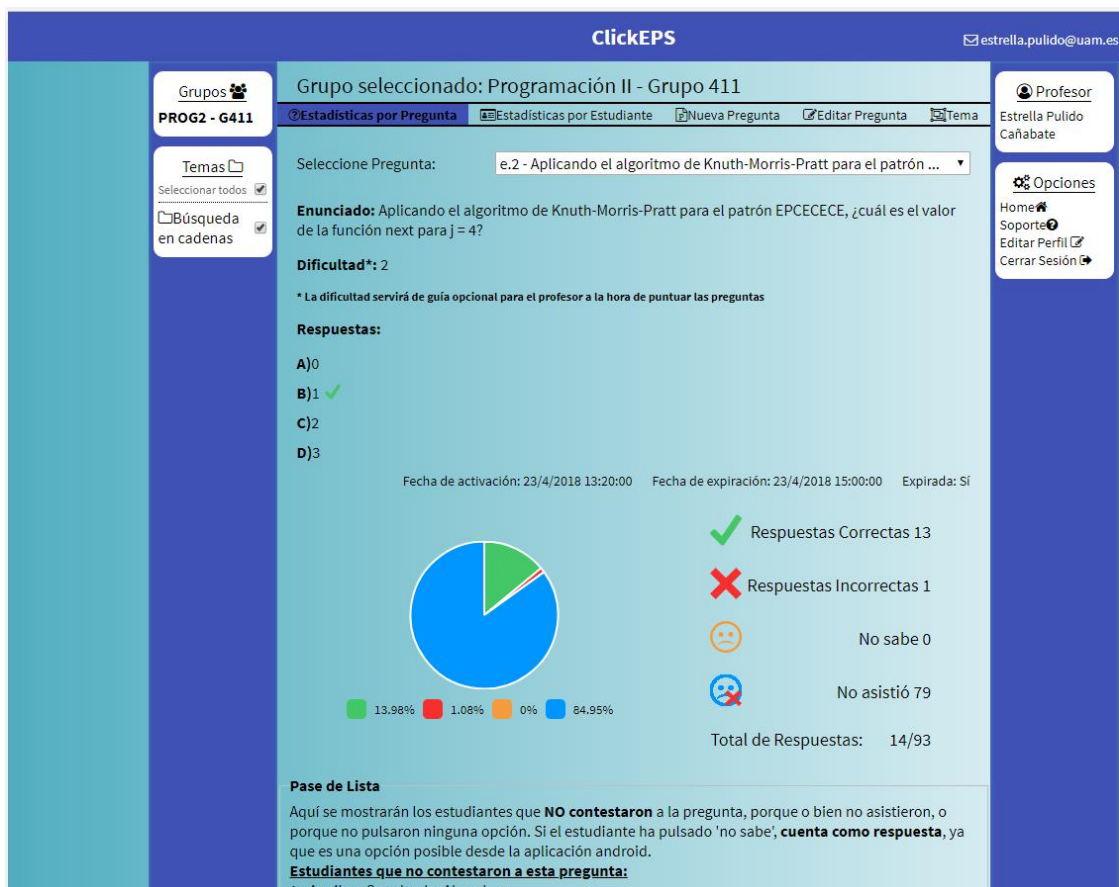


Figura 47: Captura de pantalla de Estadísticas por pregunta

**ClickEPS** estrella.pulido@uam.es

Grupo seleccionado: Programación II - Grupo 411

Estadísticas por Pregunta | Estadísticas por Estudiante | Nueva Pregunta | Editar Pregunta | Tema

**Crear Tema**

Introduzca el nombre del tema:

Por ejemplo: Tema 1: Introducción a la asignatura

Grupo que se le asigna al tema:

**Crear**

**Borrar Tema**

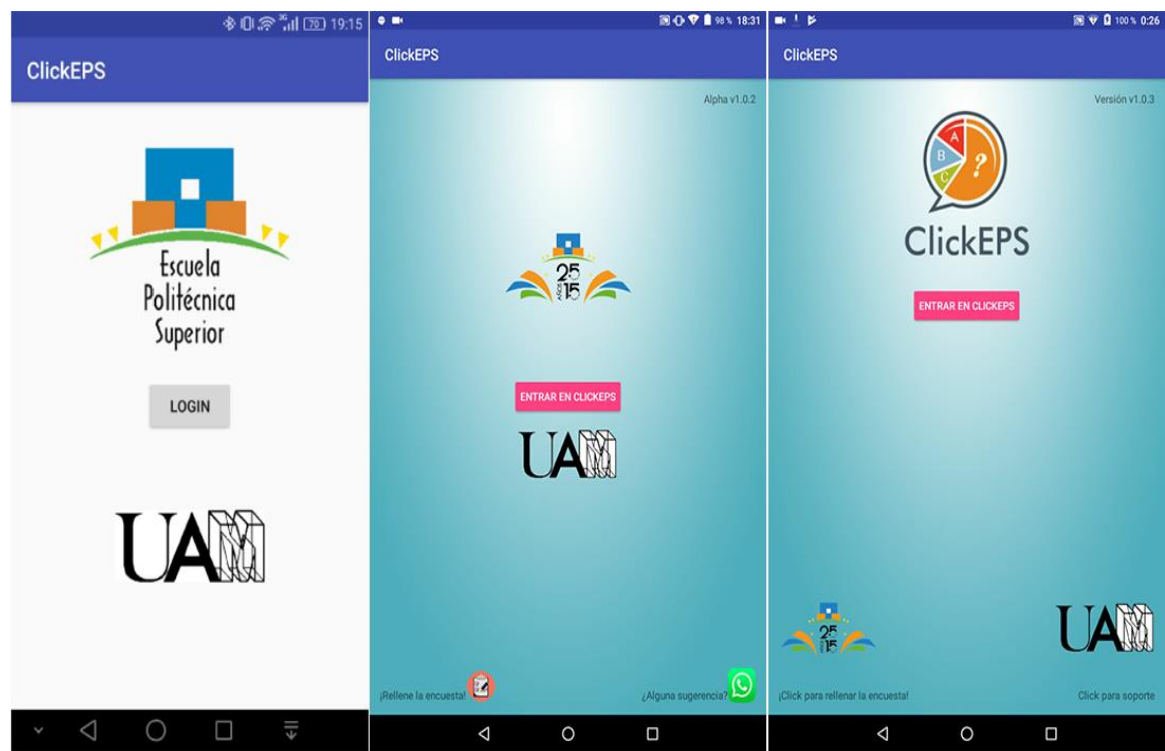
Tema a eliminar (Atención, no sólo eliminará el tema, sino todo lo relacionado con él)

**Eliminar**

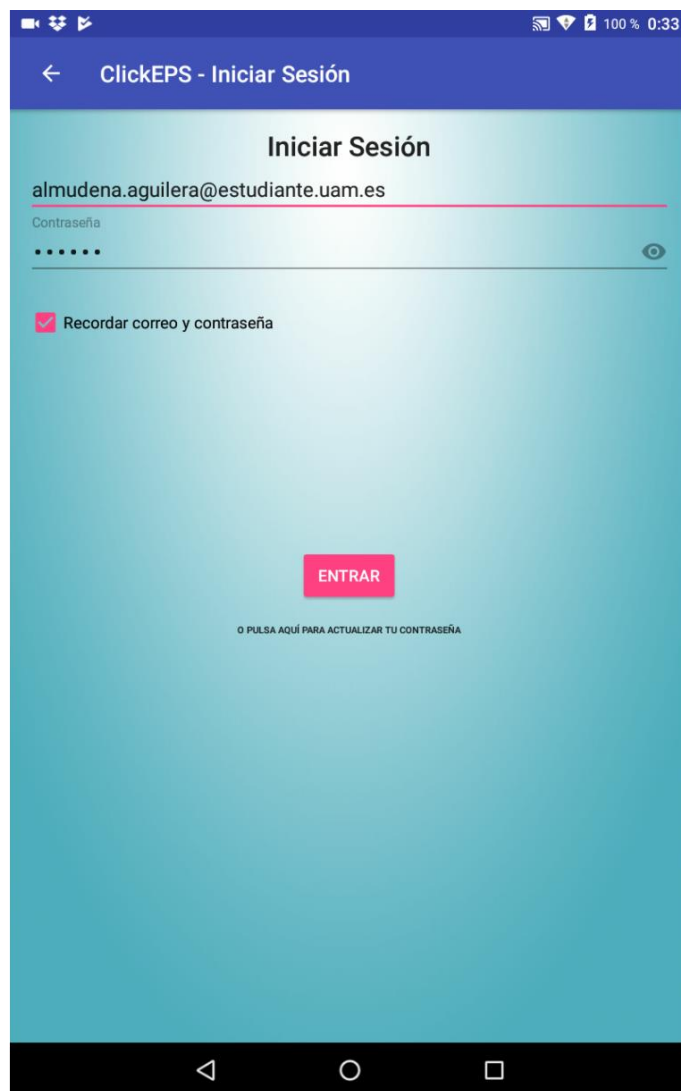
Figura 48: Captura de pantalla de crear tema (profesorado)



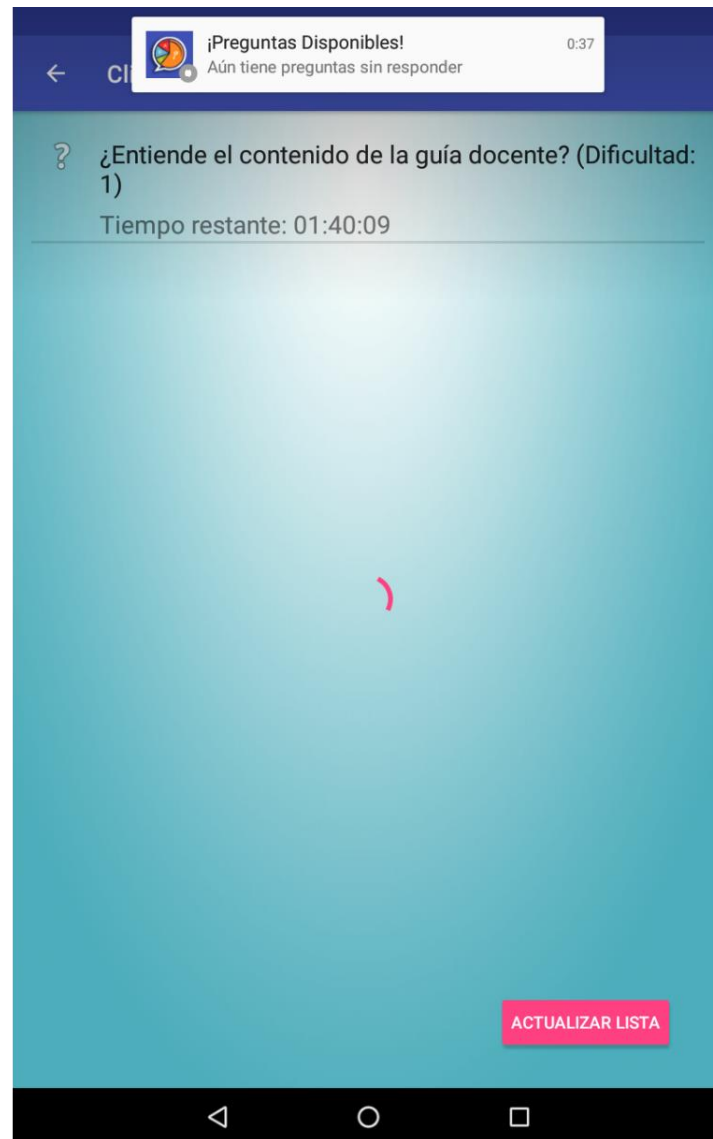
En la Figura 40 se observa de izquierda a derecha la evolución visual de la pantalla de inicio de la aplicación móvil de ClickEPS.



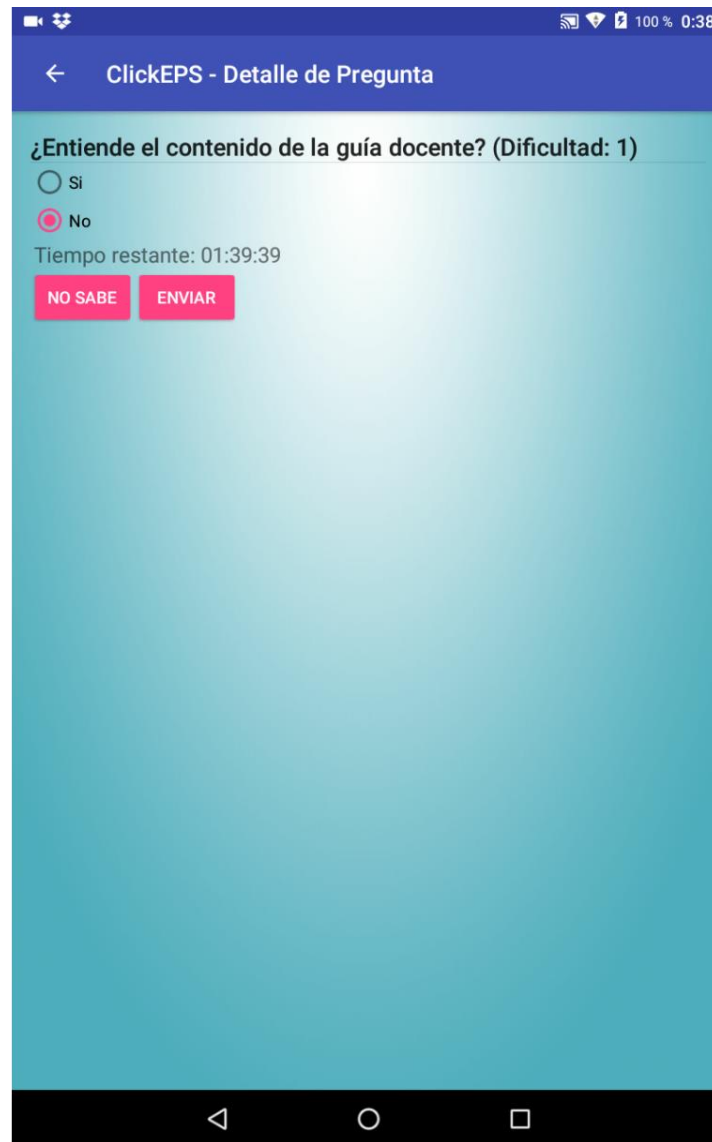
**Figura 49: Comparativa visual de versiones (Android)**



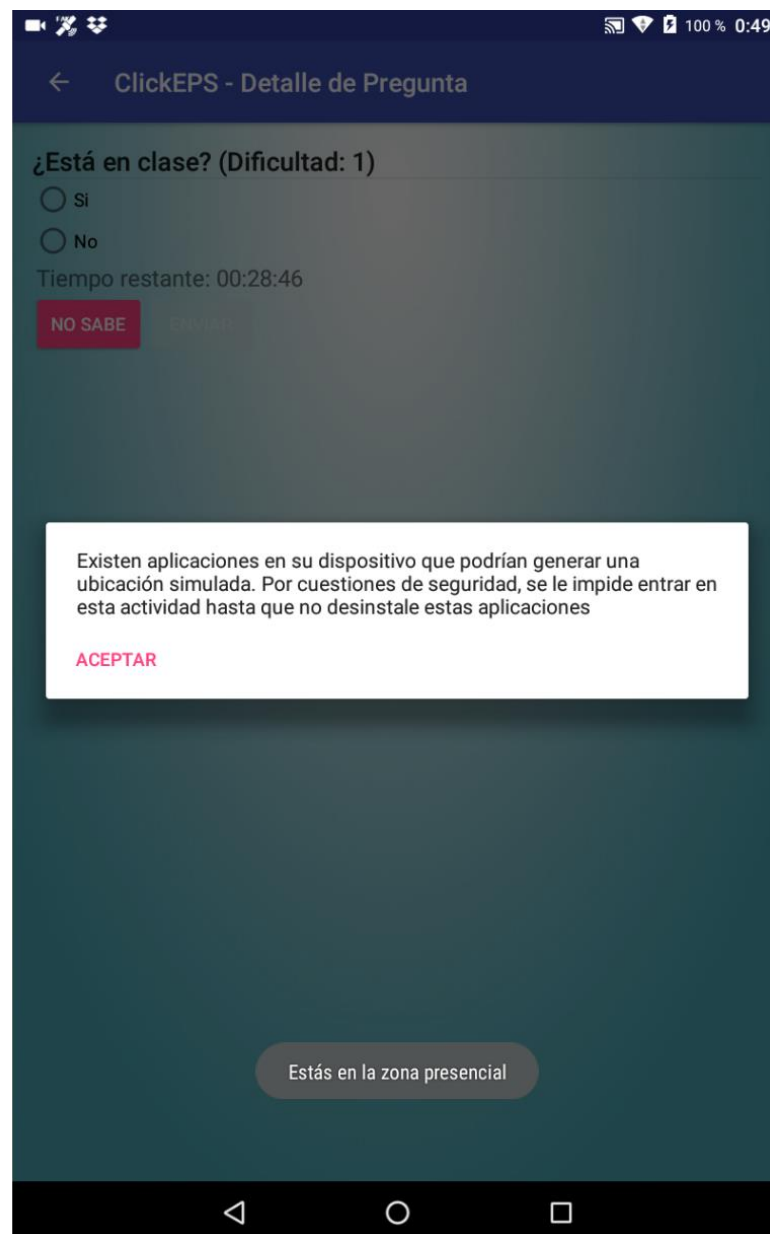
**Figura 50: Iniciar sesión (Android)**



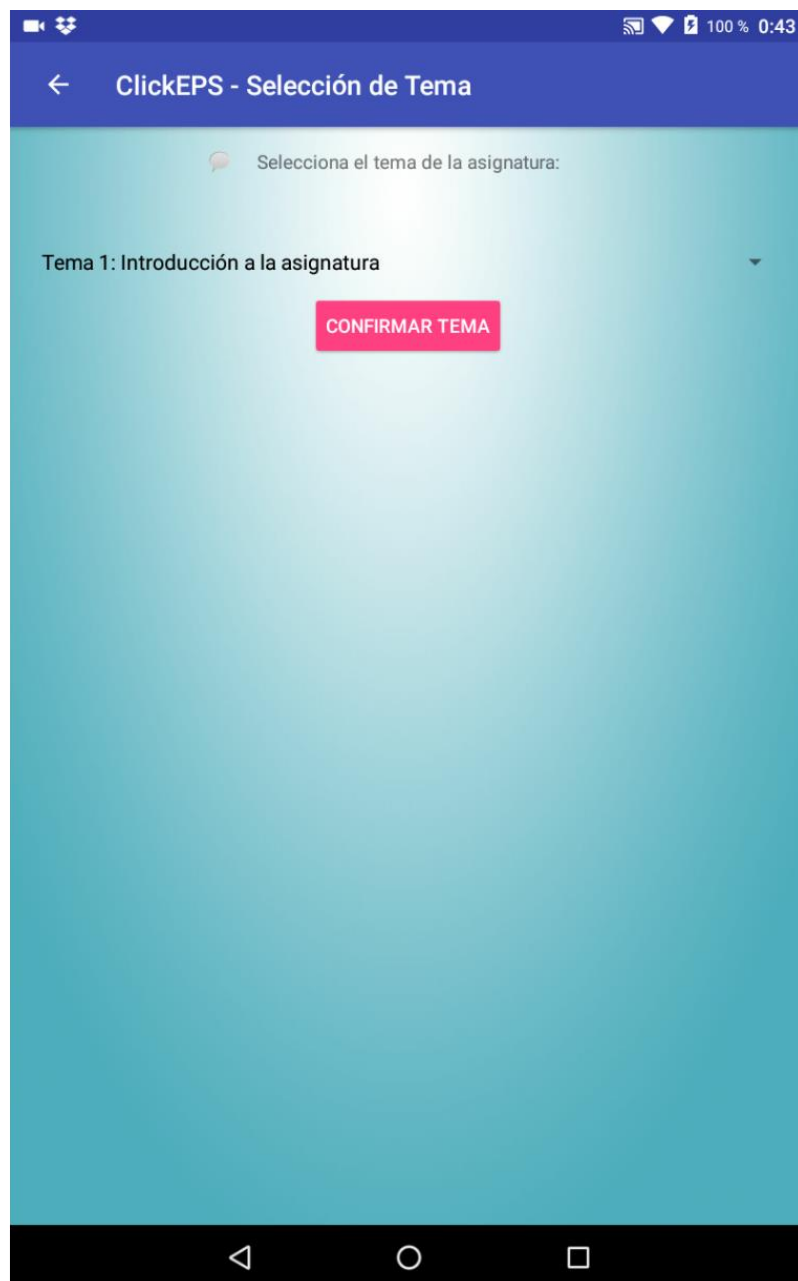
**Figura 51: Listado de preguntas (Android)**



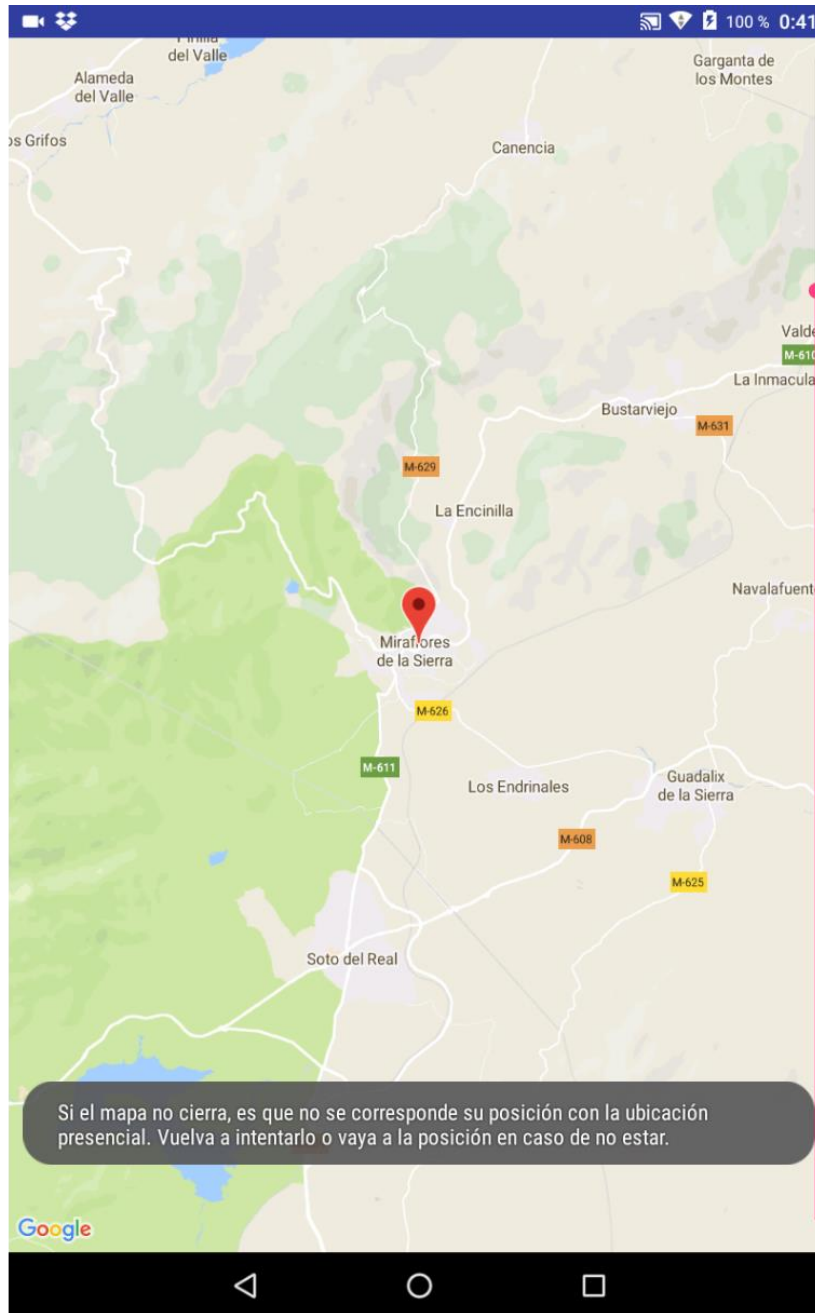
**Figura 52: Detalle de pregunta (Android)**



**Figura 53: MockLocation funcionando (Android)**



**Figura 54: Filtrado por temas (Android)**



**Figura 55: GPS (Android)**