



TELECOMUNICACIÓN

Campus Sur  
POLITÉCNICA

# ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA Y SISTEMAS DE TELECOMUNICACIÓN

## PROYECTO FIN DE GRADO

**TÍTULO:** ESTUDIO E IMPLEMENTACIÓN DE NUEVOS TIPOS DE  
EJERCICIOS EN LA PLATAFORMA MALL UP2B2

**AUTOR:** ROCIO MAGALÍ ROMERO PANERO

**TITULACIÓN:** GRADO EN INGENIERÍA TELEMÁTICA

**TUTOR:** ANTONIO DA SILVA FARIÑA

**DEPARTAMENTO:** Departamento de Ingeniería Telemática y Electrónica

VºBº

**Miembros del Tribunal Calificador:**

**PRESIDENTE:** IRINA ARGÜELLES ALVAREZ

**VOCAL:** ANTONIO DA SILVA FARIÑA

**SECRETARIO:** ANA BELÉN GARCÍA HERNANDO

**Fecha de lectura:**

**Calificación:**

El Secretario,



# Agradecimientos

*A mis padres, Adriana y Jorge*

*A mis hermanos, Gonzalo y Leandro*

*Y a Miguel*

*Que gracias a su apoyo han hecho este trabajo posible.*



## Resumen

Este proyecto fin de grado tiene como objetivo la implementación de nuevos tipos de ejercicios de Listening con imágenes y audios en el aplicativo UP2B2 para el aprendizaje del idioma inglés. Para ello hemos estudiado el marco tecnológico relativo a las aplicaciones MALL y hemos realizado una ampliación de la aplicación UP2B2 agregando nuevas funcionalidades para el desarrollo de ejercicios con audio

En la actualidad existen muchos tipos de herramientas que permiten el aprendizaje de un idioma de manera autónoma a través de dispositivos móviles, permitiendo al alumno decidir cuándo hacer ejercicios y de qué manera. En la ETSIST es necesario disponer de un nivel B2 en inglés para finalizar la titulación y esta competencia se puede obtener mediante las pruebas del Test of English for International Communication (TOEIC). Por tanto, es importante aportar al alumnado material y ejercicios para la preparación de esta prueba para que puedan afrontarla con éxito.

En esta prueba, se nos evaluará las competencias de Reading y Listening, y es por ello que surge la necesidad de ampliar la aplicación UP2B2 para adaptar los ejercicios al tipo de examen que los alumnos han de afrontar. En la aplicación de UP2B2 actualmente disponemos de ejercicios para la parte de Reading, es decir, para poner a prueba nuestras habilidades de vocabulario y gramática. Sin embargo, no se ofrecen ejercicios para la parte de Listening, tales como conversaciones o descripción de imágenes.

Así, en el presente proyecto fin de grado por un lado se estudia las posibilidades y usos que aportan las aplicaciones móviles y por otro lado se realiza el desarrollo de una aplicación MALL, para la realización de ejercicios con imágenes y audios con el fin de facilitar al alumnado la realización del dicho examen.



## Abstract

This final degree work aims to implement new types of listening exercises with images and audios in the UP2B2 application for English language learning. We have studied the technological framework related to MALL applications and we have extended the UP2B2 application framework adding new functionalities for the development of listening exercises.

Currently, there are many different types of tools to accomplish the learning of a foreign language, such as translators, applications or games. This allows the student to decide when and how they want to do exercises. A B2 level in English according to the Common European Framework of Reference for Languages (CEFR) is compulsory in ETSIST to finish the degree, and this competency can be demonstrated through the Test of English for International Communication (TOEIC) exams. Therefore, there is a clear need to offer material and exercises for the preparation of this test for the students.

In this exam, the abilities of reading and listening are evaluated, and therefore a need exists to extend the UP2B2 application to adapt the exercises to the ones found at the TOEIC exam. Currently UP2B2 offers exercises for the reading part, such as exercises of vocabulary and grammar, but there are none for the listening part, such as conversations or spoken image descriptions.

Thereby, in this project we will be studying the possibilities and uses that the mobile application provide at the moment on the one hand, and on the other hand, we will develop a web and mobile application, with the objective to grant exercises with images and audios in the UP2B2 platform for the students, to help them face the TOEIC exam.





# Índice de contenido

<b>Resumen .....</b>	<b>i</b>
<b>Abstract .....</b>	<b>iii</b>
<b>Índice de figuras .....</b>	<b>vii</b>
<b>Lista de acrónimos.....</b>	<b>ix</b>
<b>1      Introducción y objetivos.....</b>	<b>1</b>
1.1    Marco y motivación del proyecto .....	1
1.2    Objetivos del proyecto .....	2
1.3    Organización del resto de la memoria .....	2
<b>2      Marco tecnológico .....</b>	<b>3</b>
2.1    Introducción .....	3
2.1.1    MALL.....	3
2.1.2    Aplicaciones existentes.....	6
2.1.3    Tecnologías empleadas .....	9
2.2    Node.js .....	9
2.2.1    Concurrencia.....	9
2.2.2    V8.....	10
2.2.3    Módulos.....	10
2.2.4    Npm.....	10
2.3    Express.js .....	10
2.4    MongoDB .....	11
2.5    Ionic .....	12
2.6    AngularJS .....	12
2.6.1    Módulos.....	12
2.6.2    Directivas .....	12
2.6.3    Modelos.....	12
2.6.4    Controladores.....	12
2.6.5    Scope .....	13
2.7    Cordova.....	13
<b>3      Descripción del diseño y desarrollo realizados .....</b>	<b>15</b>
3.1    Estructura de la aplicación.....	15
3.2    Modelos de datos en base de datos .....	15
3.2.1    Modelo de usuario.....	15
3.2.2    Modelo de pregunta .....	16
3.3    Estructura de módulos en cliente .....	18
3.3.1    Servicios.....	20
3.3.2    Valores .....	20
3.4    Estructura del servidor .....	21
3.4.1    Archivo connectMongoose.js .....	22
3.4.2    questionModel y userModel .....	22
3.4.3    Archivo middleware.js .....	22
3.4.4    Archivo services.js .....	22

3.4.5	Archivos password.js, question.js, upload.js y users.js .....	22
3.4.6	Archivo app.js .....	22
3.4.7	Archivo config.js .....	22
3.4.8	Archivo package.json .....	23
3.4.9	Módulos utilizados .....	23
3.5	Acceso a los archivos de audio e imagen .....	23
3.6	Reproducción de los audios.....	24
3.7	Software utilizado para el desarrollo.....	24
3.7.1	GITHUB y SourceTree .....	24
3.7.2	Sublime Text 3.....	24
3.7.3	Robomongo .....	24
3.7.4	Postman .....	24
3.8	Medidas de seguridad.....	25
3.8.1	Archivo de configuración .....	26
3.8.2	Registro en la aplicación .....	26
3.8.3	Token de acceso .....	26
3.8.4	Autenticación .....	27
3.9	Funcionalidades de la aplicación.....	27
3.9.1	Autenticación .....	28
3.9.2	Administración .....	30
3.9.3	Test.....	37
3.9.4	Ranking .....	40
3.9.5	Perfil .....	41
3.9.6	Desconectar .....	42
<b>4</b>	<b>Conclusiones y trabajos futuros.....</b>	<b>45</b>
4.1	Conclusiones .....	45
4.2	Trabajos futuros.....	46
	<b>Bibliografía .....</b>	<b>49</b>
	<b>Anexo A. Manual de usuario .....</b>	<b>51</b>
A.1	Parte servidor.....	51
A.1.1.	Node.JS.....	51
A.1.2.	MongoDB.....	51
A.1.3.	Express.....	52
A.1.4.	Npm .....	53
A.2	Parte cliente .....	53
A.2.1.	Ionic.....	53
	<b>Anexo B. Recortar audio.....</b>	<b>55</b>
	<b>Anexo C. Descargar material de Listening.....</b>	<b>59</b>

## Índice de figuras

Figura 1. Resultados de encuesta en [2] .....	4
Figura 2. Google Traductor .....	6
Figura 3. TOEIC Test, Practice TOEIC .....	7
Figura 4. English Listening .....	8
Figura 5. Duolingo .....	9
Figura 6. ExpressJS .....	11
Figura 7. Ejemplo de Hello_world con Express .....	11
Figura 8. Scope de AngularJS .....	13
Figura 9. Estructura de la aplicación implementada .....	15
Figura 10. Esquema de usuario .....	16
Figura 11. Ejemplo de usuario en Robomongo .....	16
Figura 12. Modelo de pregunta .....	17
Figura 13. Ejemplo de pregunta en Robomongo .....	18
Figura 14. Estructura de módulo en cliente .....	18
Figura 15. Especificación del módulo del ranking .....	19
Figura 16. Configuración del módulo del ranking .....	19
Figura 17. Controlador del módulo del ranking .....	19
Figura 18. Archivo APIClient .....	20
Figura 19. Archivo sessionService .....	20
Figura 20. Archivo values .....	21
Figura 21. Estructura de carpetas del servidor .....	21
Figura 22. Archivo connectMongoose .....	22
Figura 23. Establecimiento de ruta para acceso a archivos desde public .....	23
Figura 24. Descarga de imagen de una pregunta .....	23
Figura 25. Creación de audio .....	24
Figura 26. Ejecución y pausa de audio .....	24
Figura 27. Ejemplo de llamada en Postman .....	25
Figura 28. Ejemplo de respuesta en Postman .....	25
Figura 29. Función de creación del token .....	27
Figura 30. Diferente menú según el tipo de usuario .....	28
Figura 31. Pantalla de autenticación .....	28
Figura 32. Popup de autenticación fallida .....	29
Figura 33. Local storage .....	29
Figura 34. Popups de recordar contraseña .....	29
Figura 35. Correo recibido al cambiar de contraseña .....	30
Figura 36. Pantalla de administración .....	30
Figura 37. Popup de creación de usuario .....	31
Figura 38. Email de creación de usuario .....	31
Figura 39. Pantalla de gestión de usuarios .....	32
Figura 40. Popup de eliminación de usuario .....	32

Figura 41. Popups de modificación de usuario .....	33
Figura 42. Pantalla de estadísticas .....	33
Figura 43. Pantalla de estadísticas por usuarios .....	33
Figura 44. Pantalla de estadísticas por preguntas .....	34
Figura 45. Pantalla de creación de pregunta sin rellenar .....	35
Figura 46. Pantalla de creación de pregunta con los campos rellenados .....	35
Figura 47. Popup de creación correcta de pregunta .....	36
Figura 48. Popup de eliminación de pregunta .....	36
Figura 49. Pantalla de gestión de preguntas .....	37
Figura 50. Pantalla principal de test .....	37
Figura 51. Pantalla de test 1 .....	39
Figura 52. Pantalla de test 2 .....	39
Figura 53. Pantalla de test 3 .....	40
Figura 54. Pantalla de test al finalizarlo .....	40
Figura 55. Pantalla de Ranking .....	41
Figura 56. Pantalla de perfil de usuario .....	41
Figura 57. Popups de cambio de contraseña de usuario .....	42
Figura 58. Popup de desconexión .....	42
Figura 59. Descarga de Node.JS .....	51
Figura 60. Descarga de MongoDB .....	52
Figura 61. Instalación Express .....	52
Figura 62. Instalación Express-generator .....	52
Figura 63. Ejecución node .....	52
Figura 64. Archivo package.json .....	53
Figura 65. Instalación Cordova e Ionic .....	53
Figura 66. Comienzo nuevo proyecto Ionic .....	53
Figura 67. Iniciar proyecto en Ionic .....	54
Figura 68. AudioTrimmer: Paso 1 .....	55
Figura 69. AudioTrimmer: Paso 2 .....	56
Figura 70. AudioTrimmer: Paso 3 .....	56
Figura 71. AudioTrimmer: Paso 4 .....	57
Figura 72. AudioTrimmer: Paso 5 .....	57
Figura 73. Ejemplo de audios en ELLLO .....	59
Figura 74. Descarga de audio en ELLLO (1) .....	60
Figura 75. Descarga de audio en ELLLO (2) .....	60
Figura 76. Descarga de audio en ELLLO (3) .....	61

## Lista de acrónimos

MALL (Mobile-Assisted Learning Language)  
HTML (HyperText Markup Language)  
CSS (Cascading Style Sheets)  
DNI (Documento Nacional de Identidad)  
SQL (Structured Query Language)  
DB (Data Base)  
JS (JavaScript)  
JSON (JavaScript Object Notation)  
APK (Android Application Package)  
TOEIC (Test of English for International Communication)  
URL (Uniform Resource Locator)  
ELLLO (English Language Listening Lab Online)



# 1 Introducción y objetivos

## 1.1 Marco y motivación del proyecto

Para poder finalizar el grado en la ETSIST actualmente es necesario obtener el título B2 de inglés. Para ello, la escuela ofrece convocatorias para la realización de exámenes de la prueba TOEIC, que incluye ejercicios de Reading, gramática, vocabulario y Listening. De esta manera, se evalúan las habilidades de comprensión tanto lectora como auditiva del alumno en el inglés.

La prueba se compone de dos partes:

1. Parte de Listening, con 4 secciones diferentes:
  - a. Fotografías: 10 preguntas en las que tendremos 1 fotografía y un audio, en el que se darán 4 posibles descripciones a la imagen y que tendremos que elegir la opción que mejor describa la situación que observamos en la fotografía. El contenido del audio no estará transcrito en texto.
  - b. Preguntas-Respuestas: 30 ejercicios en las que oiremos una pregunta y 4 posibles respuestas a la pregunta, que no estarán transcritas en el papel.
  - c. Conversaciones cortas: 30 ejercicios en que se oirán conversaciones cortas entre 2 personas. Para cada audio se responderán 3 preguntas asociadas a la conversación. En este caso tendremos las preguntas y las respuestas escritas en texto.
  - d. Monólogos cortos: 30 preguntas en las que oiremos un audio con 3 preguntas asociadas. En cada audio se oirá a una persona hablando sobre un tema, y dispondremos de las preguntas y las respuestas transcritas en el papel.
2. Parte de Reading, con 3 secciones:
  - a. Oraciones incompletas: 40 ejercicios en que tendremos diferentes oraciones con huecos que tendremos que rellenar dadas 4 posibles respuestas.
  - b. Completar texto: 12 ejercicios en los que tendremos textos con partes incompletas, teniendo 4 posibles opciones a dichas partes incompletas que tendremos que seleccionar.
  - c. Comprensión lectora: Tendremos por un lado 28 ejercicios con un texto simple, y por otro lado 20 ejercicios con 2 textos relacionados. Para cada texto o conjunto de textos tendremos preguntas y 4 posibles respuestas para cada pregunta.

Previamente, en un PFG anterior, se desarrolló la aplicación UP2B2, que incluía ejercicios de gramática, vocabulario y verbos. Debido a que el examen ofrecido por la escuela también evalúa ejercicios de tipo Listening, nos encontramos con una clara necesidad de realizar una ampliación de la plataforma para así ofrecer al alumnado más herramientas para practicar y desarrollar estas habilidades y así poder afrontar positivamente el examen. De esta manera,

surge el presente PFG, con la intención de estudiar las posibilidades ofrecidas por las tecnologías actuales para la implantación de ejercicios tanto con imágenes como con audios.

### **1.2 Objetivos del proyecto**

Los objetivos de este proyecto fin de carrera son, desde el punto de vista técnico:

- Estudio de las metodologías en el aprendizaje de un idioma asistido a través de un dispositivo móvil (MALL).
- Continuación del desarrollo de la aplicación UP2B2 añadiendo ejercicios tanto con imágenes como con audios.
- Aprendizaje de tecnologías basadas en Javascript, como son Ionic junto con AngularJS y Node.JS, HTML5, CSS3, además de bases de datos no SQL como MongoDB.
- Estudio de la implementación de una plataforma con imágenes y audios en Ionic y Node.JS.

### **1.3 Organización del resto de la memoria**

En los próximos capítulos de la memoria se ofrece información por un lado del marco tecnológico en que se engloba el desarrollo del proyecto, específicamente de las metodologías MALL, Node.js, Express.js, MongoDB, Ionic, AngularJS y Cordova. Por otro lado, respecto al diseño y desarrollo realizado, destacamos la estructura de la aplicación, los modelos de datos implementados en la base de datos, el software utilizado para la implementación, las medidas de seguridad desarrolladas y las funcionalidades de la aplicación, como son la autenticación, el área de administración, la realización de tests, el ranking, el perfil y la desconexión. Además, nos encontramos con un capítulo de resultados y discusión, en que analizamos los pasos seguidos para el desarrollo de la aplicación y los resultados obtenidos. Para finalizar, tenemos un capítulo de conclusión y trabajos futuros en el que se ofrecen líneas de continuación del trabajo realizado y conclusiones del desarrollo realizado.



## 2 Marco tecnológico

### 2.1 Introducción

#### 2.1.1 MALL

MALL, se define como “tecnologías móviles aplicadas al aprendizaje de un lenguaje, especialmente en situaciones donde el dispositivo ofrece especificaciones de portabilidad específicas” [1], e incluye todo tipos de dispositivos, desde reproductores MP3 o MP4 hasta ordenadores portátiles y tabletas.

En [2] encontramos un estudio de las razones que llevan al uso de dispositivos móviles para el aprendizaje de un lenguaje, así como los mecanismos de implementación de estos. Para que el estudio de un lenguaje sea exitoso, es necesario que el alumno tenga flexibilidad y poder de elección para así dirigir su aprendizaje según sus necesidades. Normalmente los alumnos consideran estas actividades más motivacionales que el tipo de aprendizaje clásico en un aula, pues pueden realizar las actividades que más les convenga en cada momento.

Respecto a cómo se experimenta el aprendizaje a través de dispositivos fuera de las clases, en un estudio en [3], nos encontramos que el uso de técnicas de memorización de vocabulario de inglés teniendo en cuenta el entorno y contenido de cada situación fue beneficioso para el correcto aprendizaje. Por tanto, la monitorización y evaluación del aprendizaje es importante, incrementando la motivación y ayudando a los alumnos.

Además, también es importante conocer las motivaciones de los alumnos, añadiendo nuevos tipos de contenidos e interacciones con la aplicación, permitiendo diferentes maneras de acceso a la aplicación, teniendo en cuenta las motivaciones de los alumnos, y ayudándolos a localizar sus necesidades específicas.

En este mismo estudio, respondiendo a las preguntas de cuáles son las motivaciones de los alumnos para usar los dispositivos móviles para aprendizaje de un idioma, y de cuáles son las prácticas emergentes en este tipo de actividades, se realizó una encuesta obteniendo los siguientes resultados:

Number of students invited	1525
Number of responses	269
Age of respondents	20 – 65 +
Gender of respondents	65.4% female; 34.6% male
Complete responses	243
Response rate based on complete responses	15.9%
Numbers by language based on complete responses	French (69), Spanish (90), German (33), Italian (42), Chinese (9)
Numbers by level (4) based on complete responses	Beginner (79), Intermediate (94), Upper Intermediate (38), Advanced (32)
Of complete responses, number and percentage of people using mobile devices for language learning	143 (53%)

**Figura 1. Resultados de encuesta en [2]**

Respecto a los dispositivos utilizados, el primer dispositivo más utilizado es el teléfono móvil (45.9% para los alumnos principiantes, 51.65% para los alumnos intermedios y 60.25% para los alumnos avanzados), seguido de iPads y tabletas (32.79% para los alumnos principiantes, 26.65% para alumnos intermedios y 23.75% para alumnos avanzados), y por último reproductores MP3 e iPods.

Respecto a las actividades preferidas, la principal actividad desarrollada fue para practicar la habilidad de escucha del idioma elegido. Esta es una actividad clave e importante para el desarrollo del aprendizaje de un lenguaje, y es ideal para los dispositivos móviles, pues permite hacerlo mientras se está en movimiento, en espacios de tiempo reducidos o realizando otras actividades. Así, una de las actividades más populares entre los alumnos es la de escuchar audios o ver vídeos.

Entonces, si estudiamos el por qué los alumnos están comenzando a utilizar los dispositivos móviles para el aprendizaje de un idioma, nos encontramos en que una de las razones es que ya tienen un dispositivo de este tipo y saben que tiene un gran potencial para enfocar su uso a este tipo de actividades. Otra de las razones, es la posibilidad de completar los espacios de tiempo vacío entre actividades, como puede ser viajes entre el hogar y el trabajo, o los descansos en el trabajo, realizando ejercicios con sus dispositivos móviles para ejercitar el idioma elegido.

Así, las actividades de escucha son percibidas como una preparación para el mundo real, ya sea escribir mientras una persona habla rápido, o practicar diferentes acentos y voces, con la intención de obtener un acento auténtico. Además, de esta manera algunos alumnos consiguen la confianza necesaria para sobrellevar el miedo a hablar el lenguaje. Por otro lado, los dispositivos móviles también brindan diversión y se diferencian de las actividades típicas de las clases, haciendo que los alumnos lo vean como algo divertido y diferente.

Por tanto, los alumnos que utilizan dispositivos móviles para el aprendizaje de un idioma se sienten con más control de su aprendizaje, entienden sus propias necesidades y frecuencias de uso necesarias de las aplicaciones para poder conseguir sus objetivos.

En [4] se especifican tres claves en MALL: los problemas físicos, pedagógicos y psicosociales:

- En primer lugar, respecto a los problemas físicos, los dispositivos son portátiles y relativamente pequeños, además de disponer de problemas de capacidades de almacenamiento, velocidad de procesador, batería y compatibilidad de los distintos dispositivos.
- En segundo lugar, respecto a los problemas pedagógicos, nos encontramos con que uno de los grandes desafíos es asegurarse que las tareas son viables en el dispositivo a usar. Normalmente, es común encontrar ejercicios diseñados para resolverlos con papel y lápiz, y será necesario cambiar y adaptar los ejercicios a los dispositivos. También, se destaca el problema de que no todo usuario tiene competencias plenas para utilizar correctamente el dispositivo.
- Por último, los problemas psicosociales. La población relaciona los dispositivos móviles como elementos de ocio y no de trabajo o de estudio. Si observamos el tipo de aplicaciones que la mayoría de la población tiene instalada en sus dispositivos, vemos que la mayoría son para comunicaciones entre personas o además de videojuegos. Este hecho nos hace evaluar si la sociedad percibe a los dispositivos móviles como herramientas aptas para el aprendizaje.

Por otro lado, en este mismo estudio, se definen diez principios a seguir en el desarrollo e implementación del aprendizaje de un lenguaje en plataformas móviles.

1. Los ejercicios y aplicaciones han de tener presente las limitaciones tanto del dispositivo móvil como del entorno donde la aplicación será utilizada.
2. Necesidad de limitar las distracciones del entorno y la multitarea, de cara a evitar el aumento del estrés, errores y disminución de la productividad.
3. La utilización de notificaciones de tipo “push” han de estar limitadas y hay que ofrecer la posibilidad de configurar tanto la frecuencia de uso como la desactivación total de las éstas.
4. Esforzarse por mantener la equidad en la clase, esto es, tener conocimientos de las limitaciones de los dispositivos que disponen los alumnos y de la posible existencia de alumnos sin dispositivos móviles.
5. Planificar y tener conocimiento de las diferencias de aprendizaje de alumnos, teniendo en cuenta los distintos estilos de aprendizaje.
6. Conocer las culturas de uso de los dispositivos por parte de los alumnos.
7. Mantener las sesiones y ejercicios con duración limitada y corta. Es mejor dividir las actividades o tareas largas en otras más pequeñas.
8. Permitir que el aprendizaje del lenguaje se adapte a las tecnologías y entornos del momento. Hay que tener en cuenta que si los alumnos tienen intención de utilizar el dispositivo en momentos donde es difícil incorporar audios o vídeos, las actividades deberían de adaptarse al entorno.

9. Algunos, probablemente la mayoría de los alumnos, necesitarán un pequeño entrenamiento en el uso de los dispositivos móviles y aplicaciones.
10. En las clases es necesario proveer y dar cabida a múltiples actores, realizando una adecuada preparación y motivar tanto a los alumnos como al profesorado.

### 2.1.2 Aplicaciones existentes

Si analizamos las aplicaciones existentes actualmente que nos ayudan a estudiar o mejorar nuestro nivel de inglés, las distinguimos según la funcionalidad y tipos de ejercicios que nos ofrecen.

#### 2.1.2.1 Traductores

En la sección de traductores destacamos a Google Traductor. Disponemos de más de 90 idiomas, traducción de texto, imágenes, audios, dibujos y realidad aumentada. Permite traducciones inmediatas tanto de texto escrito con el teclado como texto de aplicaciones, contenido en imágenes, voz, texto escrito a mano, o texto apuntando con nuestra cámara y obteniendo la traducción en la misma pantalla.

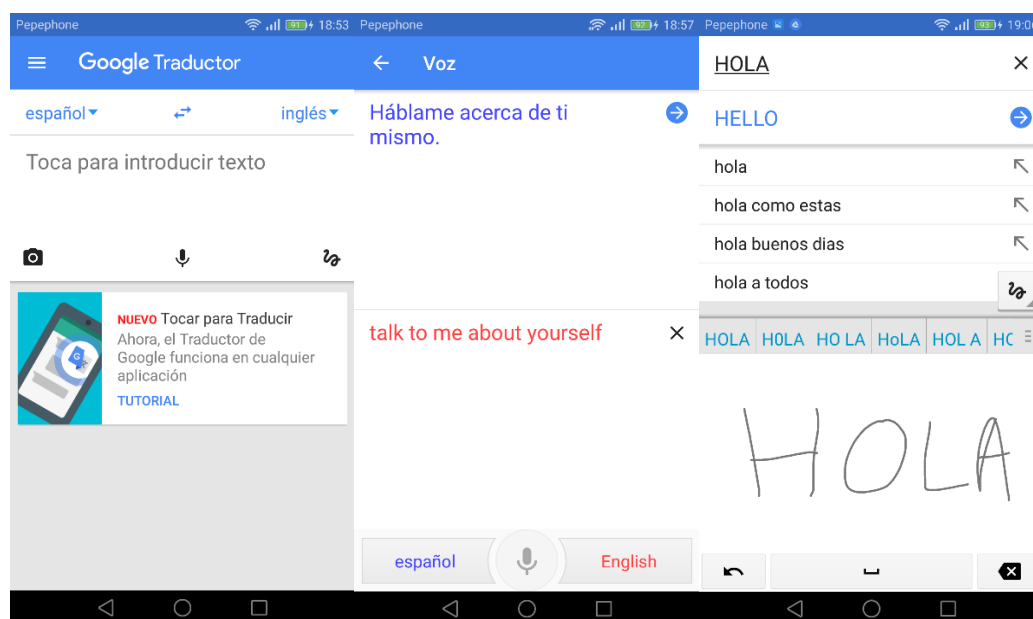


Figura 2. Google Traductor

#### 2.1.2.2 Aplicaciones específicas para la preparación del examen TOEIC

Actualmente disponemos para la preparación específica del examen TOEIC de varias aplicaciones con ejercicios que imitan dicho examen. Si buscamos en la Play Store por “TOEIC test Listening reading” obtenemos muchos resultados, en el que destacamos la aplicación “TOEIC Test, Practice TOEIC” con una valoración de 4.9. En esta aplicación disponemos de ejercicios separados en secciones del examen, tanto de la parte de Listening como de Reading.

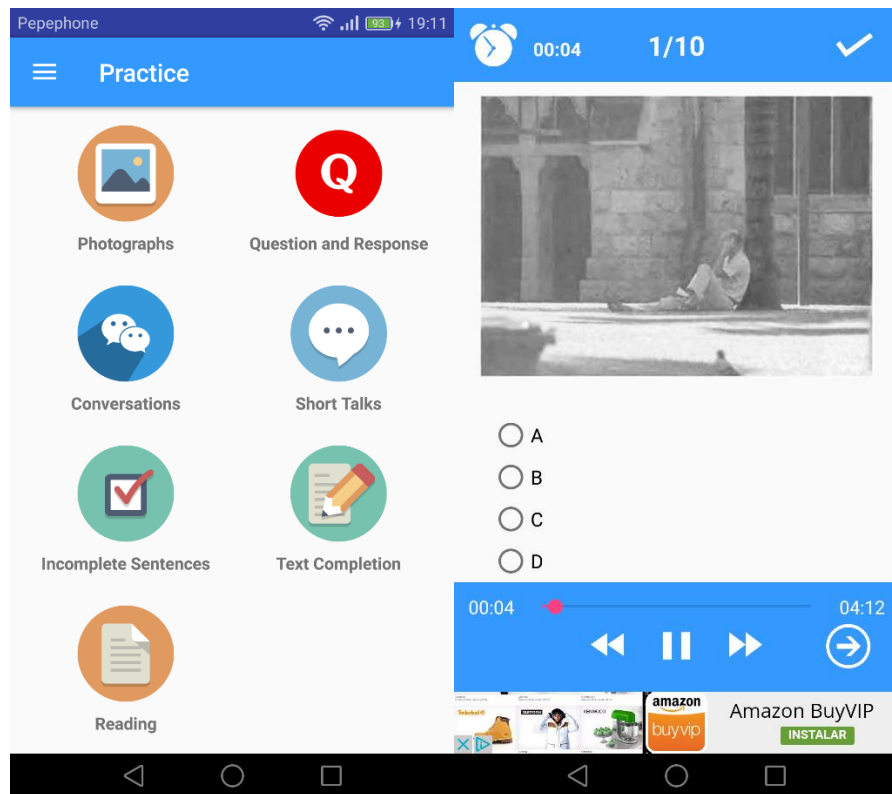


Figura 3. TOEIC Test, Practice TOEIC

Además, si nos centramos en aplicaciones específicas para el aprendizaje de la parte de Listening y buscamos por “Listening”, también obtenemos varios resultados.

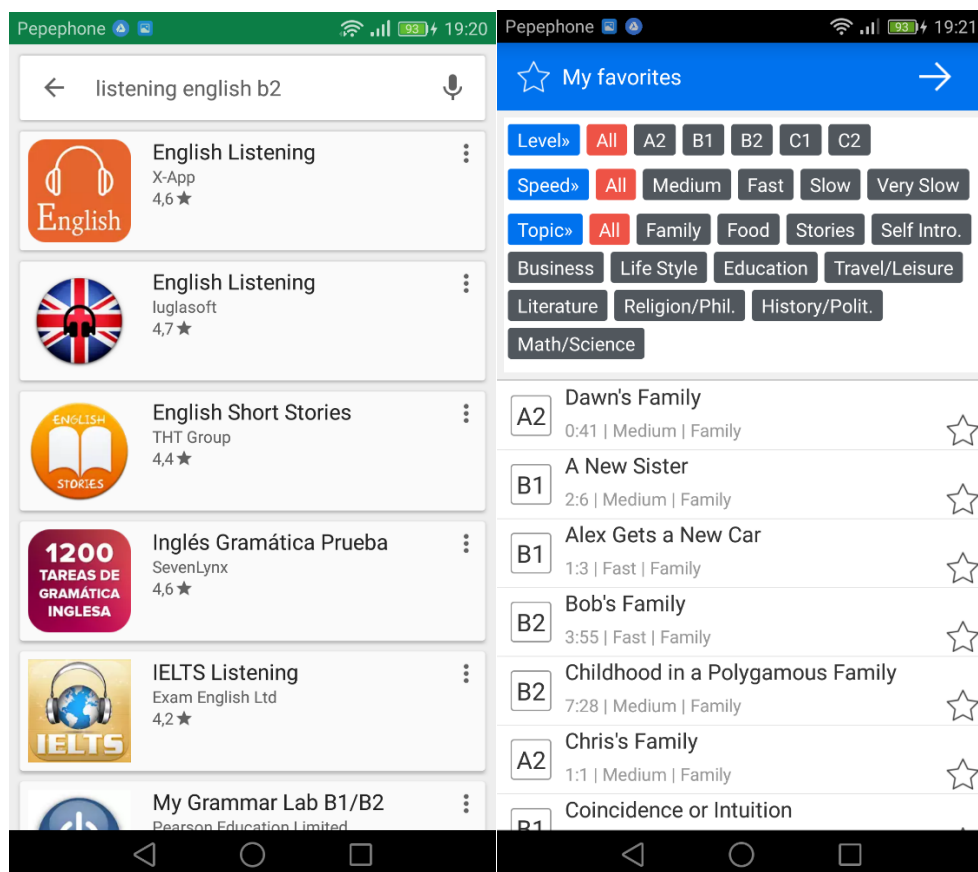


Figura 4. English Listening

Si seleccionamos la aplicación “English Listening”, vemos que podemos practicar Listenings con varios niveles, velocidades y categorías de audios.

### 2.1.2.3 Duolingo

Esta aplicación está concebida para el aprendizaje de idiomas en general, sin tener en cuenta un examen específico. La metodología de aprendizaje es amena, incluyendo distintos idiomas, juegos, metas diarias, etc. Además, incluye diferentes tipos de ejercicios, combinando audios, comprensión lectora, traducciones, entre otros. De esta manera, consigue que el aprendizaje sea divertido y motiva al estudiante a seguir utilizando la aplicación.

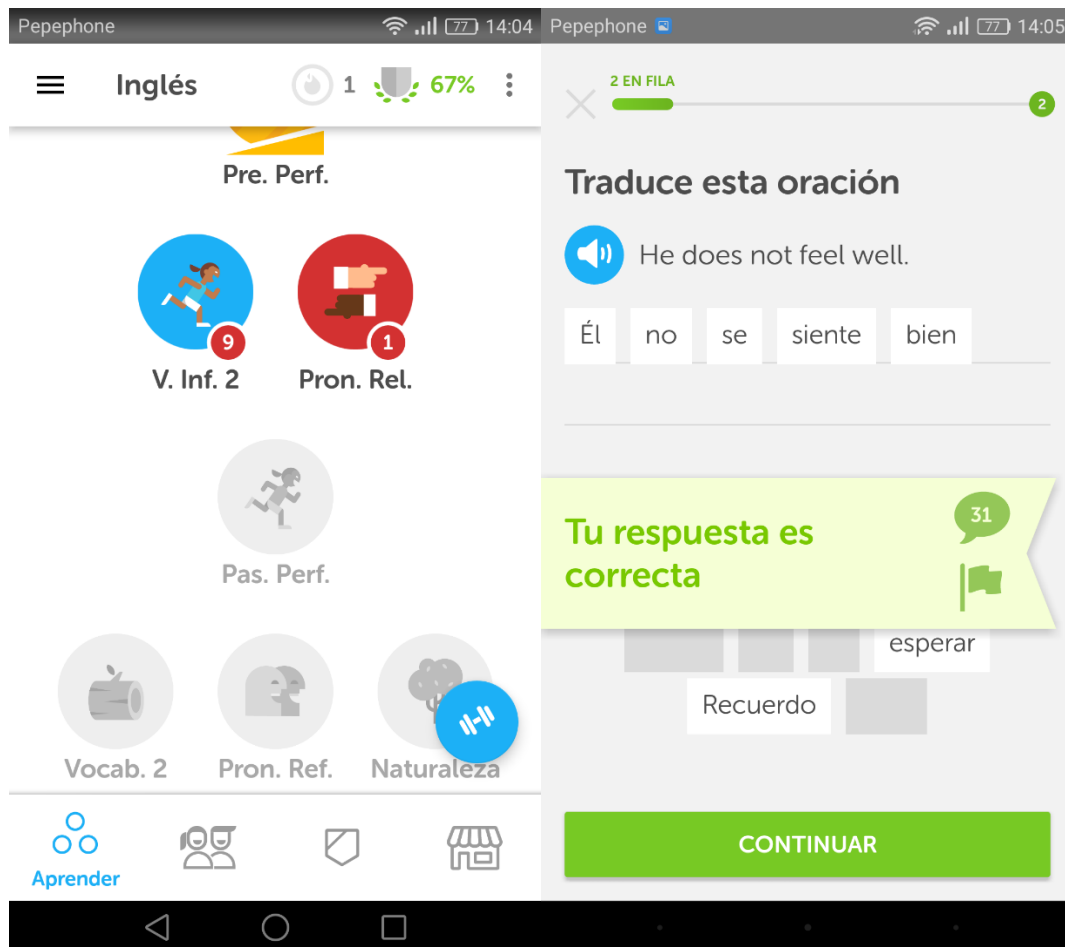


Figura 5. Duolingo

### 2.1.3 Tecnologías empleadas

Las tecnologías empleadas para el desarrollo de la aplicación las podemos separar en 2 tipos: las utilizadas para la parte del cliente, tales como Ionic, AngularJS y Cordova; y las utilizadas para la parte del servidor: NodeJS, ExpressJS y MongoDB.

## 2.2 Node.js

Esta tecnología [5] la hemos utilizado en la parte del servidor. Nos ofrece un entorno de ejecución de Javascript orientado a eventos asíncronos. A continuación analizamos las características más relevantes.

### 2.2.1 Concurrencia

Node.js funciona con un único hilo de ejecución con entradas y salidas asíncronas que se ejecutan concurrentemente. Así, todas las entradas y salidas disponen de “callbacks”, que son funciones que se ejecutan una vez la funcionalidad de la petición se ha completado, evitando así bloqueos.

### 2.2.2 V8

Motor de código abierto para Javascript desarrollado por Google, es utilizado por Node.js como entorno de ejecución. Compila el código Javascript en código máquina en vez de realizar la interpretación en tiempo real.

### 2.2.3 Módulos

Incluye módulos compilados en el propio binario con funcionalidades básicas. Además, permite utilizar módulos de terceros, que pueden extender a Node.js o añadir otros niveles de abstracción, como es Express, que explicaremos seguidamente.

### 2.2.4 Npm

Npm [6] (Node Package Manager) es el manejador de paquetes de Node.js y es instalado automáticamente al instalar Node.js. Éste se ejecuta con línea de comandos y maneja las dependencias de la aplicación, permitiendo la instalación de módulos que se encuentran en su repositorio.

## 2.3 Express.js

Además de Node.js, en la parte servidora hemos utilizado un framework llamado Express.js [7] que nos ayuda a organizar la parte servidora con una estructura MVC, a la vez que nos facilita el manejo de rutas. Este framework permite utilizar muchas extensiones muy útiles y que nos ayuda a realizar un desarrollo más rápido.

Comenzar a crear una API con Express es muy sencillo, basta con ejecutar el comando “express [nombre de la aplicación deseada]”. A continuación vemos un ejemplo para una aplicación llamada “Hello\_world”:



```
C:\Users\ROCIO\Desktop\Hello world>express Hello_world

create : Hello_world
create : Hello_world/package.json
create : Hello_world/app.js
create : Hello_world/routes
create : Hello_world/routes/index.js
create : Hello_world/routes/users.js
create : Hello_world/views
create : Hello_world/views/index.jade
create : Hello_world/views/layout.jade
create : Hello_world/views/error.jade
create : Hello_world/public/images
create : Hello_world/public/stylesheets
create : Hello_world/public/stylesheets/style.css
create : Hello_world/public
create : Hello_world/bin
create : Hello_world/bin/www

install dependencies:
> cd Hello_world && npm install

run the app:
> SET DEBUG=Hello_world:* & npm start

create : Hello_world/public/javascripts

C:\Users\ROCIO\Desktop\Hello world>
```

Figura 6. ExpressJS

Como podemos observar, una vez ejecutado el comando se generan carpetas y archivos con todo lo necesario para comenzar a desarrollar nuestra aplicación, no teniendo que preocuparnos por la redirección ni la estructura de carpetas. Seguidamente instalaremos las dependencias con “npm install”.

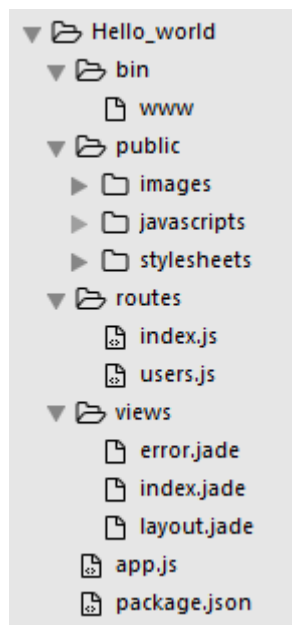


Figura 7. Ejemplo de Hello\_world con Express

## 2.4 MongoDB

MongoDB [8] es una base de datos NoSQL ágil en la que la estructura de los datos se basa en documentos.

El software utilizado para la gestión de la base de datos es Robomongo [9], pues nos ofrece una interfaz sencilla en la que poder visualizar fácilmente los documentos existentes en la base de datos y poder rápidamente modificarlos, eliminarlos o incluso añadir nuevos.

### 2.5 Ionic

Es un framework [10] para construir aplicaciones móviles. Sigue el patrón MVC (Modelo-Vista-Controlador) en el que se separa los datos de la lógica y las interfaces de usuario. Permite el desarrollo de aplicaciones híbridas basadas en Javascript, HTML5 y CSS, optimizado con AngularJS y Sass.

Ionic nos ofrece un alto rendimiento, desarrollando una sola aplicación y pudiendo compilar tanto para Android como para IOS y web. Además disponemos de plantillas para no tener que comenzar el desarrollo desde cero, plugins de Cordova [11] y componentes desarrollados como directivas, tales como botones, formularios o tarjetas, todos con sus CSS asociados.

### 2.6 AngularJS

Es un framework javascript [12] que también sigue el patrón MVC. Permite “Data binding”, es decir, la actualización automática del contenido de la vista cuando el modelo cambia, o al revés, modificar el modelo cuando la vista cambia. Por otro lado, los controladores son los encargados de controlar el comportamiento de los elementos del DOM, como puede ser la acción a realizar al pulsar un botón. AngularJS también nos ofrece validación de formularios, comunicaciones asíncronas con el servidor, directivas o componentes reusables, entre otros.

#### 2.6.1 Módulos

Cada parte de la aplicación dispone de un módulo, además de un controlador para dicho módulo. En nuestra aplicación disponemos de módulos para la autenticación, para la administración, para la creación de tests, etc.

#### 2.6.2 Directivas

Con las directivas se permite extender HTML con nuevos atributos, ofreciendo nuevas funcionalidades a la aplicación. Las propias de AngularJS empiezan siempre por el prefijo ng-. Unos ejemplos de directivas son: ng-app, ng-init, ng-model. Además, también podemos crear las nuestras propias o descargar de otros proyectos.

#### 2.6.3 Modelos

Los modelos conectan los controles y valores de HTML, como son los input, selects o textareas, a los datos de la aplicación.

#### 2.6.4 Controladores

Las aplicaciones de AngularJS son manejadas por los controladores. Con la directiva ng-controller definimos el área de control de cada controlador. En éste componente tendremos toda la funcionalidad del módulo específico que controle, como puede ser la función a realizar

cuando se pulsa un botón, o el comportamiento a seguir cuando cambia un valor en la vista que ejerce control.

### 2.6.5 Scope

El scope es la conexión entre la vista (HTML) y el controlador (Javascript). Es un objeto con propiedades y métodos accesibles, y está disponible en ambas partes, tanto en la vista como en el controlador. Un ejemplo de uso del scope puede ser el siguiente:

```
<div ng-app="myApp" ng-controller="myCtrl">

  <h1>{{carname}}</h1>

</div>

<script>
var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope) {
    $scope.carname = "Volvo";
});
</script>
```

Figura 8. Scope de AngularJS

Como vemos, en el controlador llamado “myCtrl” tenemos en el scope una propiedad llamada “carname”, que en la vista se conecta como “{{carname}}”. Así, en la vista dentro del elemento h1 tendremos escrito “Volvo” pues en el controlador tiene ese valor.

## 2.7 Cordova

Es un framework [11] para el desarrollo de aplicaciones móviles que permite usar HTML5, CSS3 y Javascript para poder desarrollar aplicaciones en distintas plataformas. Permite utilizar una misma aplicación en distintas plataformas sin tener que implementarla para cada una. Las aplicaciones web no pueden utilizar las funcionalidades nativas de los dispositivos móviles, y es por eso que necesitamos Cordova. Nos ofrece una conexión entre las aplicaciones web y los dispositivos móviles, adaptando las funcionalidades para ambos lados. Así, podemos desarrollar aplicaciones híbridas pudiendo utilizar la cámara, la geolocalización, el sistema de archivos y otras funcionalidades nativas de los teléfonos móviles.



### 3 Descripción del diseño y desarrollo realizados

#### 3.1 Estructura de la aplicación

El esquema que sigue la aplicación la podemos representar con la siguiente estructura:

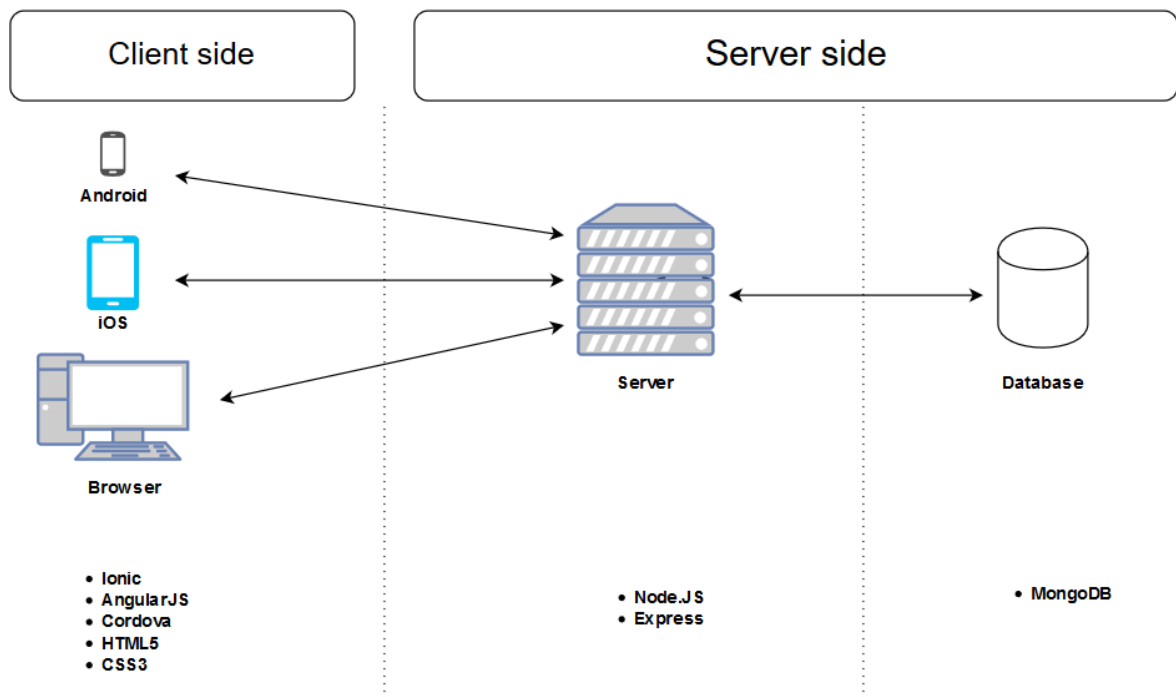


Figura 9. Estructura de la aplicación implementada

La parte cliente se compone por los dispositivos, que acceden al sistema gracias a la generación de la aplicación con Ionic, tanto en Android como en iOS, como también a través de navegadores web. Por otro lado, el servidor accede a la base de datos desarrollada en MongoDB para obtener la información solicitada por la parte cliente, proveyendo de toda la información que dispone el sistema y que se muestra en la aplicación.

#### 3.2 Modelos de datos en base de datos

Al haber utilizado el módulo de Mongoose en Express para el manejo de la base de datos, hemos podido establecer de antemano el modelo que seguirán las estructuras de datos de los documentos en MongoDB. Esto nos permite que en el caso de que se intente guardar contenido que no corresponde con el modelo, éste no se guarde.

##### 3.2.1 Modelo de usuario

El modelo del objeto de cada usuario se compone por: nombre completo, contraseña, email, puntuación actual, titulación, si es o no administrador, DNI, fecha de creación del usuario, y una lista con información de cada test hecho, conteniendo el tiempo utilizado para hacer el test,

## ESTUDIO E IMPLEMENTACIÓN DE NUEVOS TIPOS DE EJERCICIOS EN LA PLATAFORMA MALL UP2B2

la puntuación obtenida, si es de entrenamiento o no, el número de respuestas correctas, el número de respuestas incorrectas, si ha obtenido el bonus de tiempo, y la fecha en la que realizó el test. A continuación podemos ver cómo queda este esquema en código:

```
let userSchema = mongoose.Schema({
  fullName: String,
  pass: String,
  email: String,
  score: Number,
  degree: String,
  admin: Boolean,
  dni: String,
  createTime: Date,
  testDone: [{
    timeSpent: Number,
    score: Number,
    training: Boolean,
    numberCorrect: Number,
    numberWrong: Number,
    timeBonus: Boolean,
    date: Date
  }]
});
```

Figura 10. Esquema de usuario

Si accedemos con Robomongo a la base de datos, para un usuario creado con anterioridad que ha realizado 1 test, su estructura sería la siguiente:

The screenshot displays a document in Robomongo with the following structure:

- Document ID:** ObjectId("5882076b89f67716c0b9a7ba")
- Fields:**
  - `_id`: ObjectId("5882076b89f67716c0b9a7ba")
  - `pass`: String
  - `email`: String
  - `dni`: String
  - `fullName`: String
  - `score`: Int32
  - `degree`: String
  - `admin`: Boolean
  - `createTime`: Date
  - `testDone`: Array
    - Element 0:** Object
      - `date`: Date
      - `training`: Boolean
      - `timeBonus`: Boolean
      - `numberWrong`: Int32
      - `numberCorrect`: Int32
      - `score`: Int32
      - `timeSpent`: Int32
      - `_id`: ObjectId("5882219889f67716c0b9a7cb")

Figura 11. Ejemplo de usuario en Robomongo

### 3.2.2 Modelo de pregunta

El modelo de cada pregunta se compone de: fecha de creación, lista con la ruta a los archivos asociados, el enunciado de la pregunta, las 4 posibles respuestas, la respuesta correcta, si es de entrenamiento o no, si es de tipo test o no, el tiempo para contestar la pregunta, y 2 listas con

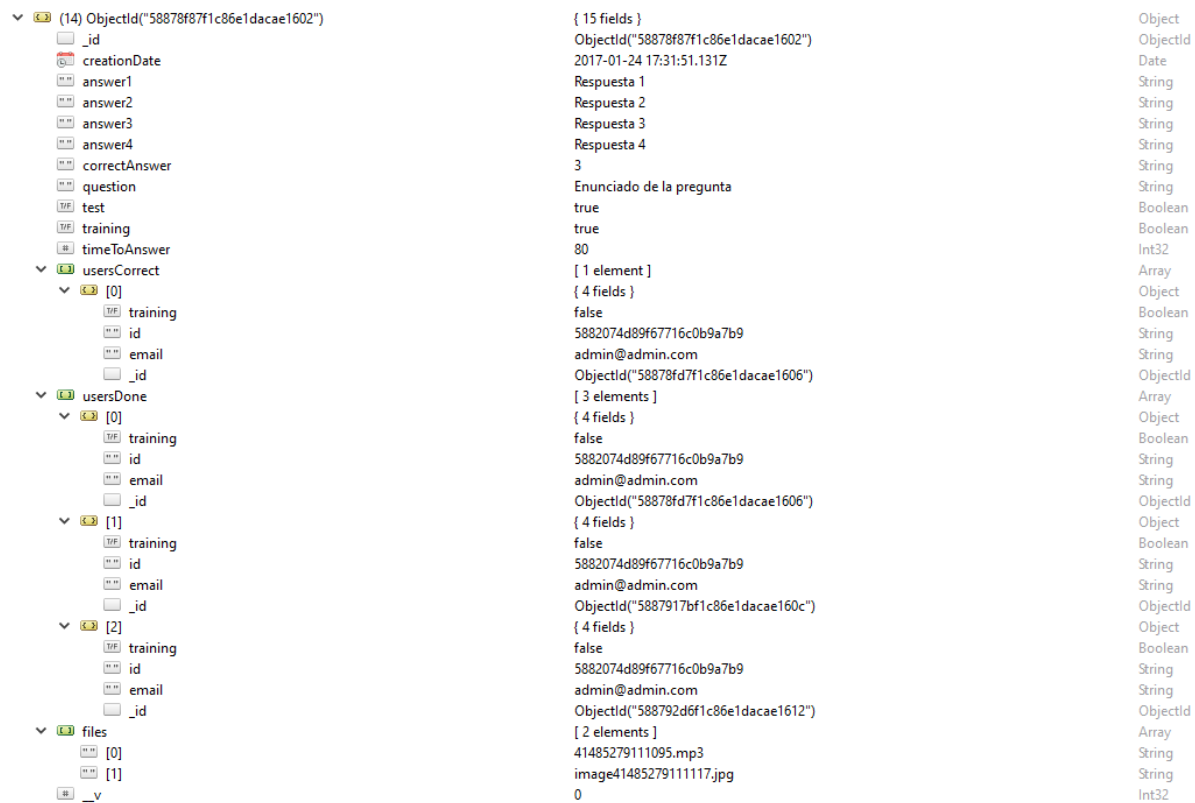
los usuarios que han respondido a la pregunta y los que la han respondido correctamente. En código ésta estructura es la siguiente:

```
let QuestionSchema = mongoose.Schema({  
  createDate: Date,  
  files: [String],  
  question: String,  
  answer1: String,  
  answer2: String,  
  answer3: String,  
  answer4: String,  
  correctAnswer: String,  
  usersDone: [{  
    id: String,  
    email: String,  
    training: Boolean  
  }],  
  usersCorrect: [{  
    id: String,  
    email: String,  
    training: Boolean  
  }],  
  training: Boolean,  
  test: Boolean,  
  timeToAnswer: Number  
});
```

Figura 12. Modelo de pregunta

Así, en la base de datos accediendo desde Robomongo, podemos ver un ejemplo de una pregunta con 2 archivos asociados, 3 usuarios que han respondido a la pregunta, y 1 usuario que ha respondido a la pregunta correctamente:

## ESTUDIO E IMPLEMENTACIÓN DE NUEVOS TIPOS DE EJERCICIOS EN LA PLATAFORMA MALL UP2B2



▼ (14) ObjectId("58878f87f1c86e1dcae1602")	{ 15 fields }	Object
_id	ObjectId("58878f87f1c86e1dcae1602")	ObjectId
creationDate	2017-01-24 17:31:51.131Z	Date
answer1	Respuesta 1	String
answer2	Respuesta 2	String
answer3	Respuesta 3	String
answer4	Respuesta 4	String
correctAnswer	3	String
question	Enunciado de la pregunta	String
test	true	Boolean
training	true	Boolean
timeToAnswer	80	Int32
usersCorrect	[ 1 element ]	Array
▼ [0]	{ 4 fields }	Object
training	false	Boolean
id	5882074d89f67716c0b9a7b9	String
email	admin@admin.com	String
_id	ObjectId("58878fd7f1c86e1dcae1606")	ObjectId
usersDone	[ 3 elements ]	Array
▼ [0]	{ 4 fields }	Object
training	false	Boolean
id	5882074d89f67716c0b9a7b9	String
email	admin@admin.com	String
_id	ObjectId("58878fd7f1c86e1dcae1606")	ObjectId
▼ [1]	{ 4 fields }	Object
training	false	Boolean
id	5882074d89f67716c0b9a7b9	String
email	admin@admin.com	String
_id	ObjectId("5887917bf1c86e1dcae160c")	ObjectId
▼ [2]	{ 4 fields }	Object
training	false	Boolean
id	5882074d89f67716c0b9a7b9	String
email	admin@admin.com	String
_id	ObjectId("588792d6f1c86e1dcae1612")	ObjectId
files	[ 2 elements ]	Array
[0]	41485279111095.mp3	String
[1]	image41485279111117.jpg	String
_v	0	Int32

Figura 13. Ejemplo de pregunta en Robomongo

### 3.3 Estructura de módulos en cliente

Para cada funcionalidad de la aplicación, ésta se ha separado en módulos. Cada módulo tendrá su controlador, su HTML asociado, su archivo de configuración y su archivo de especificación de módulo. Así, tenemos los siguientes módulos: login, administración, añadir pregunta, hacer test, ver perfil, ver ranking, administrar preguntas y administrar usuarios.

Como ejemplo, vemos para el caso de ver el ranking, que disponemos de su controlador (rankingController.js), su archivo de configuración (config.js) y el archivo de especificación del módulo (ranking.module.js). Los demás módulos siguen el mismo patrón de diseño. Respecto al HTML asociado (rankingTemplate.html), se encuentra junto a todos los archivos de tipo template (plantillas), en una carpeta con el mismo nombre.

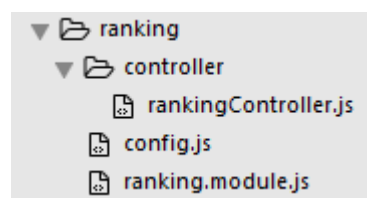


Figura 14. Estructura de módulo en cliente

En ranking.module.js tenemos el siguiente contenido, en que se especifica el nombre del módulo y se enlaza con el router de AngularJS. Por otro lado, en config.js se especifica el estado con el que se enlaza el módulo, la url, la vista asociada (rankingTemplate.html) y el controlador



asociado (rankingController.js). Por último, en el controlador del módulo rankingController.js, vemos la función de inicialización que hace la petición al servidor para obtener los datos a mostrar en el ranking.

```
'use strict';

angular.module('ranking.module', [
  'ui.router'
]);
```

Figura 15. Especificación del módulo del ranking

```
angular
  .module('ranking.module')
  .config(function config($stateProvider) {
    $stateProvider
      .state('app.ranking', {
        url: '/ranking',
        cache: false,
        views: {
          'menuContent': {
            templateUrl: 'templates/rankingTemplate.html',
            controller: 'rankingController'
          }
        }
      })
  });
```

Figura 16. Configuración del módulo del ranking

```
angular.module('ranking.module').controller('rankingController', function($scope, APIClient, utils) {
  $scope.$root.showMenuIcon = true;
  var initiate = function() {
    utils.showLoading();
    APIClient.getRanking().then(
      function(data) {
        utils.stopLoading();
        if (data.status !== 200) {
          utils.errorPopUp();
        } else {
          $scope.rankingReady = true;
          $scope.users = data.data.rows.reverse();
        }
      }
    )
  }
  initiate();
});
```

Figura 17. Controlador del módulo del ranking

### 3.3.1 Servicios

Además de los módulos, también se ha creado archivos como servicios para englobar en éste funcionalidades específicas.

1. APIClient: En este archivo tenemos todas las funciones para hacer peticiones al servidor. Como ejemplo, mostramos la primera función del archivo, que es la llamada para realizar la función de login.

```
angular.module('starter.services', []).factory('APIClient', function($http, APIPaths, sessionService) {
  return {
    login: function(user) {
      var url = APIPaths.server + APIPaths.users + APIPaths.login;
      return $http.post(url, user)
        .then(function(response) {
          return response;
        }, function(error) {
          return error;
        });
    },
  };
});
```

Figura 18. Archivo APIClient

2. SessionService: Archivo que ofrece funciones relacionadas con el control de las sesiones, pudiendo recoger, guardar y limpiar el sistema de almacenamiento local (localStorage) de manera centralizada.

```
angular.module('starter.session', []).factory('sessionService', function($http, APIPaths, $location) {
  return {
    get: function(name){
      return localStorage.getItem(name);
    },
    store: function(name, details){
      return localStorage.setItem(name, details);
    },
    clear: function () {
      localStorage.clear();
    }
  }
});
```

Figura 19. Archivo sessionService

3. Utils: Archivo en que se engloban las funciones que se utilizan en varias partes de la aplicación, y que permiten que el código no esté duplicado en varios puntos de la plataforma, como son las funciones de mostrar popup de error, o mostrar mensaje de cargando.

### 3.3.2 Valores

Por último, en el archivo APIPaths disponemos de los valores de las rutas del servidor.

```
angular.module("starter.values", []).value("APIPaths", {  
  server: "http://localhost:3000/api/v1",  
  users: "/users",  
  questions: "/questions",  
  login: "/login",  
  newUser: "/newUser",  
  password: "/password",  
  modifyUser: "/userData/",  
  newQuestion: "/newQuestion",  
  ranking: "/ranking",  
  test: "/test",  
  resolve : "/resolve",  
  upload: "/upload",  
  getFile: "/getFile"  
});
```

Figura 20. Archivo values

### 3.4 Estructura del servidor

La estructura de archivos que tiene el servidor es la siguiente:

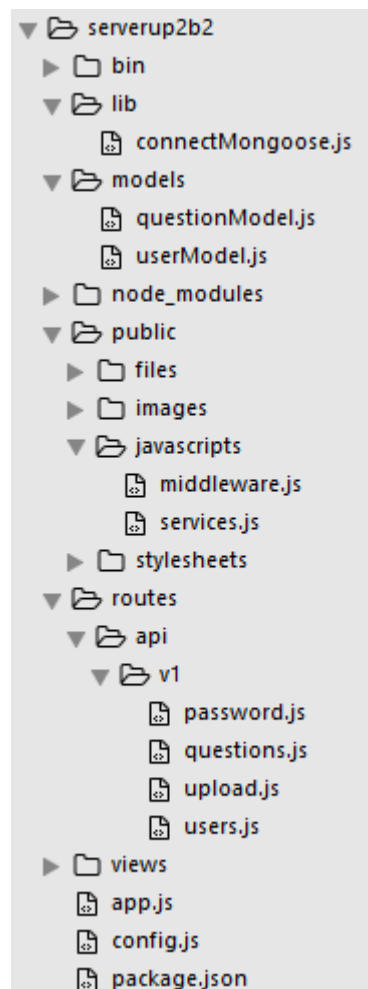


Figura 21. Estructura de carpetas del servidor

### 3.4.1 Archivo *connectMongoose.js*

Archivo que gestiona la conexión de Mongoose con MongoDB.

```
var config = require('../config.js');

var mongoose = require('mongoose');
var conn = mongoose.connection;
conn.on('error', console.error.bind(console, 'mongodb connection error:'));
conn.once('open', function() {
  console.info('Connected to mongodb.');
```

Figura 22. Archivo connectMongoose

### 3.4.2 *questionModel* y *userModel*

Archivos donde se especifican los esquemas de los modelos que siguen las preguntas y los usuarios en los documentos de MongoDB, ya explicados anteriormente en el apartado 3.2.

### 3.4.3 Archivo *middleware.js*

Es el middleware que se ejecuta antes de cada ruta que necesita un token de autenticación para poder acceder a ella. En este archivo tenemos la función de comprobación de la validez del token. Esta funcionalidad se explicará en el apartado 3.8.3.

### 3.4.4 Archivo *services.js*

Servicio que proporciona la función para la creación del token de acceso. Se llamará a la función cada vez que un usuario se autentique en la plataforma.

### 3.4.5 Archivos *password.js*, *question.js*, *upload.js* y *users.js*

Archivos específicos para cada ruta del servidor, donde se encuentran los métodos GET, POST, SET, DELETE de las rutas, junto con sus funcionalidades.

### 3.4.6 Archivo *app.js*

Es el control principal de Express.js. Aquí se especifican las rutas que tiene el servidor junto con los archivos asociados a dichas rutas, el puerto y dirección del servidor, etc.

### 3.4.7 Archivo *config.js*

Archivo donde se encuentran las variables susceptibles de cambio en la aplicación. De esta manera, se abstraen todas las variables a un único archivo para la fácil configuración del servidor y abstracción del código quitando la necesidad de tener conocimientos de la distribución del código.

### 3.4.8 Archivo package.json

Archivo que se crea y se modifica cada vez que añadimos un módulo a la aplicación en la parte servidor. Sirve para el control de las dependencias, y en caso de descarga de un proyecto nos facilita la instalación de dichos módulos con el simple uso del comando “npm install”, que lee este archivo y descarga e instala todos los módulos y dependencias en el proyecto.

### 3.4.9 Módulos utilizados

En la aplicación hemos utilizado los siguientes módulos de NodeJS:

- Crypto: para cifrar las contraseñas. En nuestro caso hemos utilizado sha256
- Jwt-simple: utilizado en el middleware para la gestión de los tokens, aporta funciones de cifrado y descifrado para la comprobación de la validez de los token, así como para su creación.
- Mongoose: Mongoose se conecta a MongoDB y nos permite especificar esquemas de datos, pudiendo añadirles métodos estáticos.
- Moment: Librería para parsear, validar, manipular y formatear fechas.
- Random-password-generator: Generador de cadenas de caracteres aleatorias. En nuestro caso lo hemos utilizado para generar las contraseñas de los nuevos usuarios, así como la funcionalidad de reiniciar la contraseña por olvido de ésta.
- Nodemailer: Permite el envío de emails. Usado para el envío de los emails de registro en la plataforma y para el reinicio de la contraseña.
- fs: Node file system, usado para almacenar los archivos de imagen y audio en el sistema de archivos del servidor.
- formidable: permite parsear <sup>1</sup>información de formularios. Usado para parsear la subida y descarga de los archivos de imagen y audio

## 3.5 Acceso a los archivos de audio e imagen

Los archivos de audio e imagen se encuentran almacenados en la ruta del servidor /public/files. Para permitir el acceso a dicha ruta, en el servidor en el archivo app.js hemos especificado que a la ruta /public/files se puede acceder a través de: url\_servidor/files

```
app.set('files', path.join(__dirname, 'public/files'));
```

Figura 23. Establecimiento de ruta para acceso a archivos desde public

Así podemos ver un ejemplo de cómo se descarga la imagen asociada a una de las preguntas desde la parte del cliente:

```
$scope.questions[0].image = 'http://localhost:3000/files/' + $scope.questions[0].files[m];
```

Figura 24. Descarga de imagen de una pregunta

<sup>1</sup> Del inglés “to parse”, equivalente en castellano a “analizar”, “tratar”, “procesar el texto”.

### 3.6 Reproducción de los audios

Para la reproducción de los audios se ha utilizado el elemento Audio que nos ofrece HTML5. Su utilización es sencilla y nos proporciona funciones como play o stop para controlar la reproducción y parada de los audios. Por un lado, para la vinculación del audio al elemento, basta con crear un objeto audio de la siguiente manera:

```
var audio = new Audio('url_audio')
```

Figura 25. Creación de audio

Por otro lado, para reproducir el audio o pararlo, basta con utilizar las funciones como sigue:

```
audio.play();  
audio.stop();
```

Figura 26. Ejecución y pausa de audio

### 3.7 Software utilizado para el desarrollo

#### 3.7.1 GITHUB y SourceTree

Para el control de versiones se ha utilizado GitHub, a través del programa SourceTree, que nos ofrece una interfaz gráfica. Con GitHub podemos crear ramas de trabajo para las distintas funcionalidades a desarrollar, y nos permite guardar cambios y documentar el desarrollo de la aplicación. Así, podemos ver las modificaciones, eliminaciones e inserciones de archivos en el proyecto, el porqué de los cambios, quién lo ha realizado y en qué momento.

#### 3.7.2 Sublime Text 3

Para el desarrollo del código se ha utilizado el software Sublime Text 3, que incluye paquetes que se pueden instalar que facilitan el desarrollo y ahorran tiempo. Por ejemplo, incluye paquetes de AngularJS con snippets de creación rápida de controladores o servicios, o paquetes para CSS en que al escribir el color en hexadecimal, el color se pinta en la pantalla, o autocompletados de funciones y corrección de cierres de funciones con paréntesis.

#### 3.7.3 Robomongo

Software para la gestión de la base de datos de MongoDB. Incluye un aspecto visual intuitivo y fácil, facilitando la gestión de la base de datos y permitiendo visualizar, modificar, crear y eliminar los documentos de manera sencilla.

#### 3.7.4 Postman

Software útil al principio del desarrollo de la parte servidor. Permite realizar llamadas al servidor y probar las rutas para comprobar el correcto funcionamiento del servidor sin

necesidad de tener operativo un cliente que llame a dichas rutas. A continuación se muestra una captura de pantalla del programa con un ejemplo de llamada para crear un nuevo usuario:

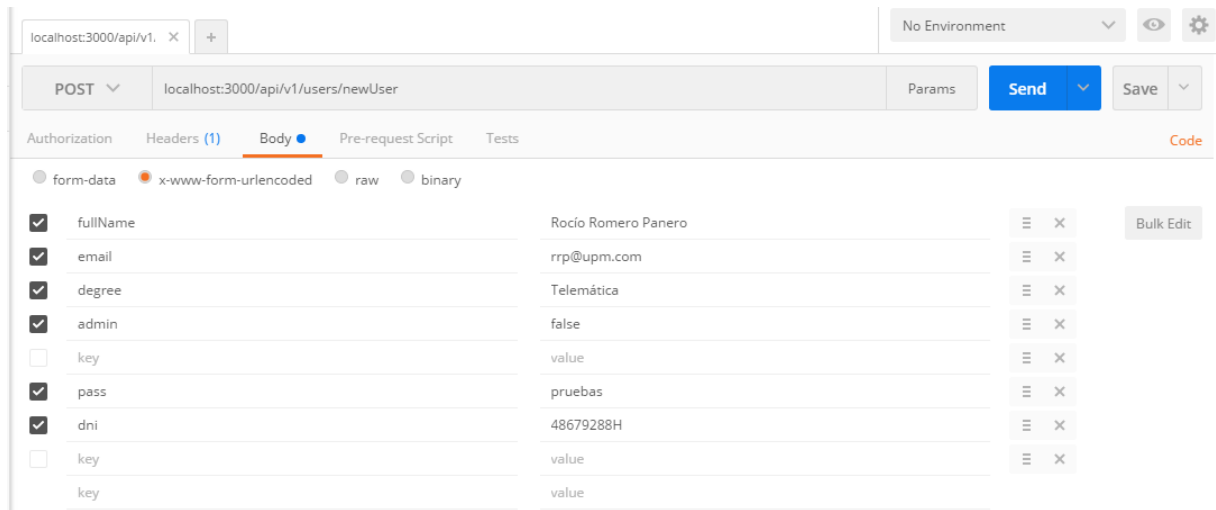


Figura 27. Ejemplo de llamada en Postman

Como vemos, configuramos que la llamada sea de tipo “POST”, a la ruta api/v1/users/newUser, e introducimos en Body los campos que tendría la petición si la hiciéramos desde un cliente real. Además, podemos visualizar la respuesta recibida del servidor, que es la información que recibirá la parte cliente:

```
1 {
2   "result": "user created",
3   "row": {
4     "__v": 0,
5     "pass": "qKY8K+nNY1CHjcsepLsBiYgy09um/xRFb4yrbJJZlpq8=",
6     "email": "rrp@upm.com",
7     "dni": "48679288H",
8     "fullName": "Rocío Romero Panero",
9     "score": 0,
10    "degree": "Telemática",
11    "admin": false,
12    "creationTime": "2017-01-30T12:00:46.247Z",
13    "_id": "588f2aee17e3f11a6c57e07c",
14    "testDone": []
15  }
16 }
```

Figura 28. Ejemplo de respuesta en Postman

## 3.8 Medidas de seguridad

Como medidas de seguridad en la aplicación, se ha optado por un lado del registro en la plataforma como control de acceso, como también la necesidad de autenticarse para poder acceder a la aplicación. Por otro lado, para el acceso a las rutas de la parte servidora es necesario tener un token de acceso que se genera una vez el usuario se ha autenticado en el sistema; así

como disponer de las contraseñas encriptadas en la base de datos. Otra medida tomada es la existencia de un archivo de configuración en la que extraemos las variables del código.

### **3.8.1 *Archivo de configuración***

Para almacenar las variables de configuración del servidor, se ha creado un archivo excluido del sistema de control de versiones. De esta manera, nos aseguramos que el contenido del archivo, que contiene información sensible, no sea accesible para el exterior del servidor, además de permitir una configuración de la aplicación centralizada en un solo documento, sin necesidad de modificar el código. En este archivo tenemos las variables de configuración siguientes:

- Usuario y contraseña de la cuenta para la gestión del envío de emails para la plataforma.
- Palabra secreta con la cual se codificará y decodificará el token de autenticación
- Dirección de la base de datos
- Dirección del servidor
- Puerto del servidor
- Asunto de los emails de registro y cambio de contraseña.

### **3.8.2 *Registro en la aplicación***

La responsabilidad del registro en la aplicación de los usuarios será por parte de un usuario de tipo administrador. Por tanto, el apartado de administración donde se encuentra la opción de registro sólo se puede acceder si el usuario dispone de esta modalidad.

### **3.8.3 *Token de acceso***

Una vez autenticado el usuario, se generará un token de acceso al servidor, y todas las peticiones realizadas al servidor desde el cliente tendrán que tener en la cabecera de la petición el token de acceso adjunto. De este modo, en el servidor antes de procesar cada petición, se recogerá dicho token y se comprobará si es válido.

Al crear un token de acceso, primero se establece su contenido, y después se codifica con una palabra secreta previamente elegida al configurar el servidor. En nuestro caso, el token se conforma por un payload con el id del usuario, el momento de la creación del token, y el momento de expiración del token.



```
exports.createToken = function(user) {  
  var payload = {  
    sub: user._id,  
    iat: moment().unix(),  
    exp: moment().add(14, "days").unix(),  
  };  
  return jwt.encode(payload, config.TOKEN_SECRET);  
};
```

Figura 29. Función de creación del token

Así, para comprobar la validez del token, se siguen los pasos siguientes:

1. Se decodifica el token con la clave secreta, y en caso de que se produzca una excepción al decodificar, significará que el token no ha sido codificado con la misma clave secreta del servidor y por tanto no es válido.
2. En caso de que se decodifique correctamente, se comprobará si el token ha expirado su periodo de validez, que en nuestro caso lo hemos configurado como 14 días.

En caso de que el token se haya codificado con la misma clave secreta y que no haya expirado, se permitirá el acceso a la ruta solicitada del servidor.

#### 3.8.4 Autenticación

Para poder pasar de la pantalla de Login de la aplicación, es necesario autenticarse con email y contraseña válidos presentes en la base de datos. Una vez el usuario cumplimenta el formulario de autenticación, se realiza una petición al servidor de tipo POST a la ruta /api/v1/users/login, en el que se siguen los siguientes pasos:

1. Se comprueba si el usuario con dicho email existe en la base de datos, y en caso de que no exista el servidor responderá con un código de estatus 404 de usuario no encontrado
2. En caso de que exista dicho usuario, se codificará la contraseña con sha256 y se comprobará si coincide con la contraseña almacenada en la base de datos. Si las contraseñas no coinciden, el servidor responderá con un código de estatus 401.
3. Si la contraseña coincide, se generará un token de acceso y se responderá con un código de estatus 200 de éxito y el contenido del id, email, nombre completo, DNI, titulación, si es o no administrador, y la puntuación del usuario autenticado.

### 3.9 Funcionalidades de la aplicación

La aplicación dispone por un lado de una sección de administración al que sólo puede acceder el usuario que es de tipo administrador, y por otro lado, todos los usuarios pueden acceder a los distintos apartados que se encuentran en el Menú. Según el tipo de usuario, éste tendrá acceso desde el menú al apartado de Administración o no.

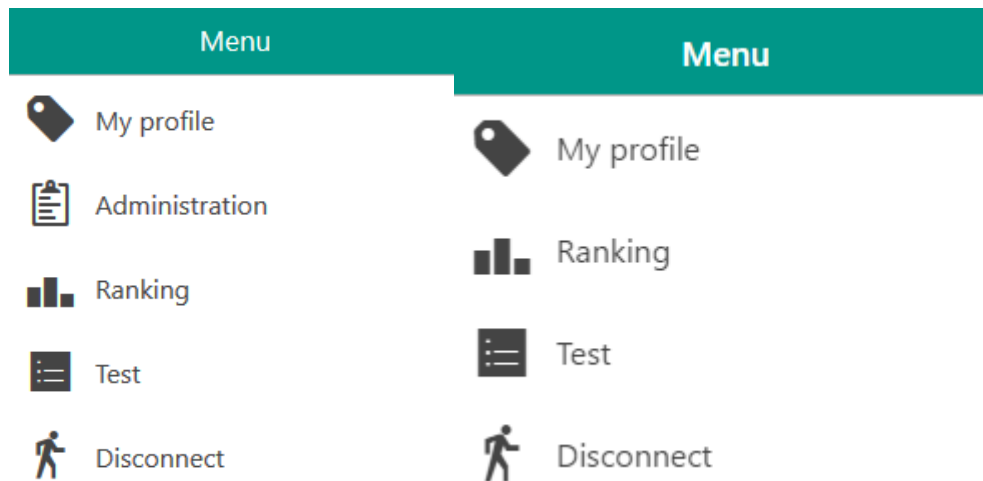


Figura 30. Diferente menú según el tipo de usuario

### 3.9.1 Autenticación

Al entrar a la aplicación, la primera pantalla que se visualiza es la de autenticación. En esta pantalla tendremos que completar el email y la contraseña del usuario. En caso de que surja un error en el proceso de autenticación como puede ser que el usuario con dicho email no exista, o que las contraseñas no coincidan, se abrirá un popup de información.

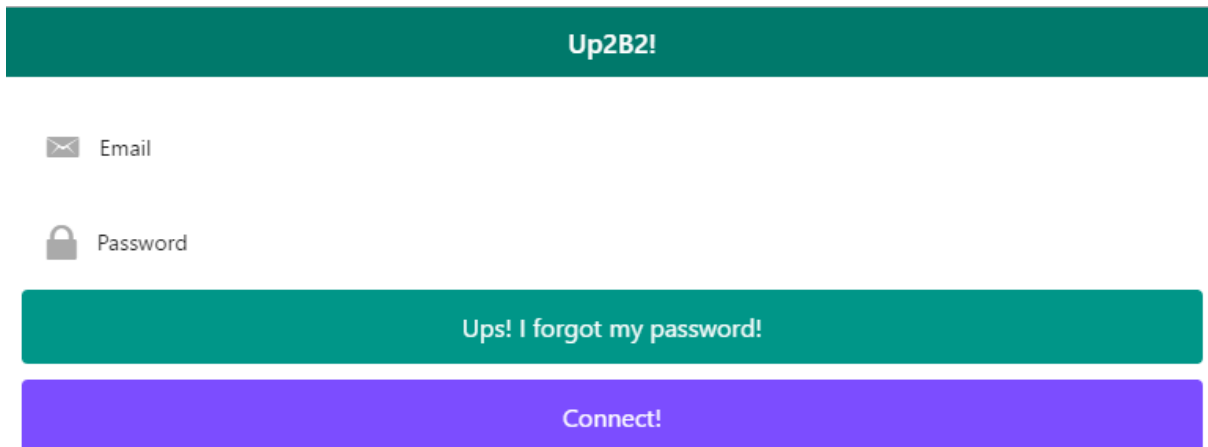


Figura 31. Pantalla de autenticación



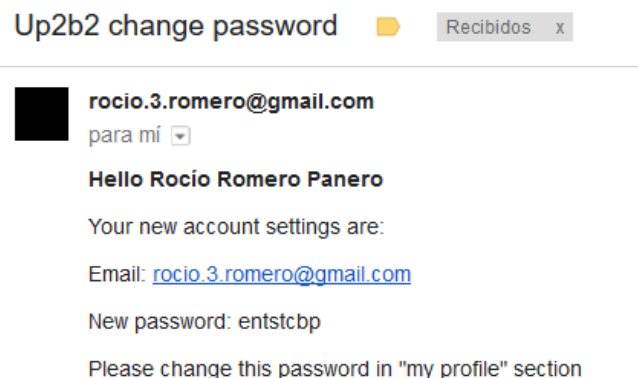


Figura 35. Correo recibido al cambiar de contraseña

### 3.9.2 Administración

En la pantalla de administración disponemos de distintos apartados tanto para administración de usuarios, como de preguntas y de visualización de estadísticas.

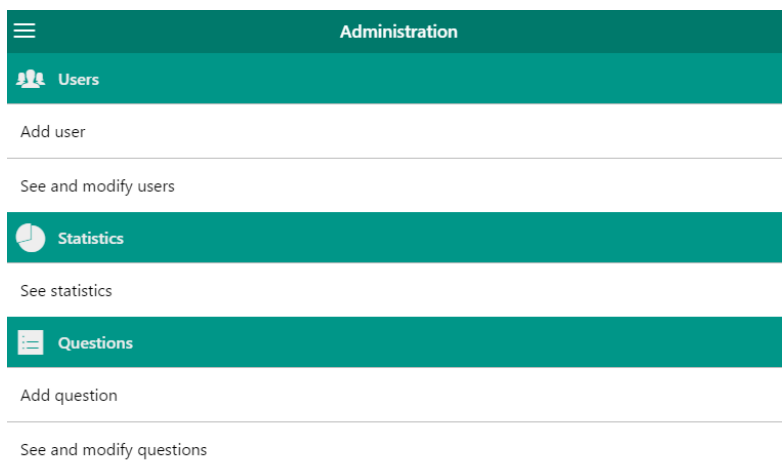


Figura 36. Pantalla de administración

#### 3.9.2.1 Registro de usuario

Una de las opciones para la gestión de los usuarios es la de registrar nuevos usuarios en la aplicación. Al pulsar en la opción, se abrirá una ventana popup en la que tendremos que rellenar los datos del nuevo usuario: email, nombre completo, titulación, DNI y si será de tipo administrador o no. Al pulsar en el botón “OK!” primero en la parte cliente se comprobará si todos los campos están completados, y en caso contrario se abrirá otro popup pidiendo que se rellenen todos los datos. Si todos los datos son correctos, se enviará una petición de tipo POST a la ruta `/api/v1/users/newUser` y se seguirán los siguientes pasos:

1. Se comprobará que el email del usuario no existe actualmente en la aplicación. En caso de que ya exista un usuario registrado con este email, se responderá con un código de estado 499.
2. En caso de que no exista un usuario con este email, se generará una contraseña aleatoria y se codificará con sha256 para guardarla en la base de datos. Seguidamente se salvará

el nuevo usuario en la base de datos y se enviará un email al usuario nuevo registrado y al usuario de administración con sus credenciales de acceso a la aplicación. Una vez se han enviado los emails, se responde al cliente con un código de estado 200.

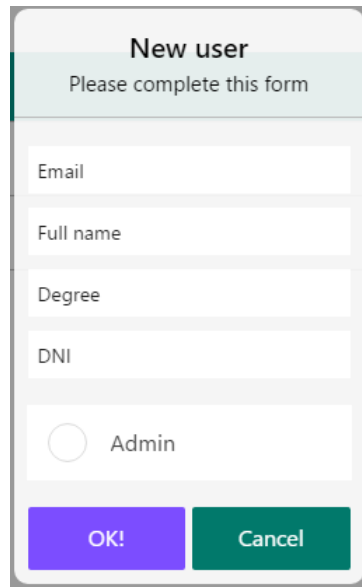


Figura 37. Popup de creación de usuario

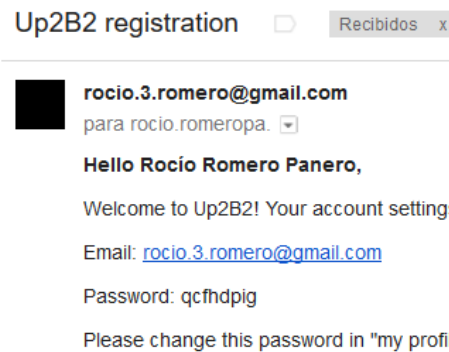

















Figura 38. Email de creación de usuario

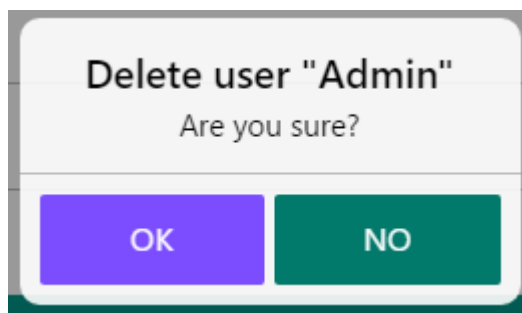
### 3.9.2.2 Gestión de usuarios

En este apartado podremos ver los usuarios registrados en la aplicación, modificar sus datos o eliminarlos. Al entrar en la pantalla primero se realizará una llamada al servidor de tipo GET a la ruta `/api/v1/users/` para obtener todos los usuarios registrados en la aplicación y así poder mostrarlos en la vista.

Users	
<b>Admin</b>	
Email: admin@admin.com	
Degree: Admin	
DNI: 123456789A	
Admin: Yes	
<b>Pruebas</b>	
Email: prueba@prueba.com	
Degree: Pruebas	
DNI: 123456789P	
Admin: No	
<b>Rocío Romero Panero</b>	
Email: rocio.3.romero@gmail.com	
Degree: Telemática	
DNI: 48679288H	
Admin: No	

**Figura 39. Pantalla de gestión de usuarios**

Al pulsar en el botón de arriba a la derecha de cada usuario, nos aparecerá un popup en que tendremos que confirmar la eliminación del usuario. En caso de que pulsemos en “OK”, se realizará una petición al servidor de tipo DELETE a la ruta `/api/v1/users/:id` en la que se eliminará al usuario de la base de datos y se responderá al cliente con el código de estado 200.



**Figura 40. Popup de eliminación de usuario**

Por otro lado, si pulsamos en los iconos de la derecha de email, degree, DNI o admin, nos aparecerá un popup donde podremos poner el nuevo valor a modificar. Una vez introduzcamos el nuevo valor y pulsemos en el botón “OK”, se hará una petición al servidor de tipo PUT a la ruta `/api/v1/users/userData/:id` en la que se modificará el valor del usuario en la base de datos y en caso de que se modifique correctamente, se responderá con un código de estado 200.

### Modify Admin's email

OK

Cancel

### Modify Admin's degree

OK

Cancel

### Modify Admin's dni

OK

Cancel

Figura 41. Popups de modificación de usuario

### 3.9.2.3 Estadísticas

En el apartado de estadísticas podremos obtener estadísticas de los usuarios y de las preguntas de la aplicación.



Figura 42. Pantalla de estadísticas

En el caso de estadísticas por usuarios, tendremos una lista con los usuarios de la aplicación, que al desplegarlo tendremos información detallada del usuario: número de tests realizados, el tiempo total haciendo tests, número de respuestas correctas totales, número de respuestas incorrectas totales, el número de tests de tipo entrenamiento y el número de tests de tipo real. Además, tendremos una lista de los tests realizados por el usuario, especificando para cada uno el tipo de test, la puntuación total obtenida, la fecha de realización del test, número de respuestas correctas, número de respuestas incorrectas, el tiempo para realizar la pregunta y si ha recibido el bonus de tiempo.

Para obtener esta información, una vez pulsado en “By User” se hace una llamada al servidor de tipo GET a la ruta `/api/v1/users` para obtener los usuarios de la aplicación y así poder mostrarlos en la vista.

<div> <div></div> <div>admin@admin.com</div> </div>	
Number of tests done by user: 19   Time spent doing tests: 3m 57s   Number of correct answers: 9   Number of wrong answers: 46   Number of training tests: 6   Number of real tests: 13	
<b>Training test   Total score: -3   20/01/2017  </b> N° correct: 0   N° wrong: 4   Time spent: 3 seconds   Time bonus: yes	
<b>Real test   Total score: 1   20/01/2017  </b> N° correct: 1   N° wrong: 3   Time spent: 4 seconds   Time bonus: yes	

Figura 43. Pantalla de estadísticas por usuarios

## ESTUDIO E IMPLEMENTACIÓN DE NUEVOS TIPOS DE EJERCICIOS EN LA PLATAFORMA MALL UP2B2

Por otro lado, en el caso de estadísticas por preguntas, tendremos una lista con las preguntas añadidas a la aplicación, con información del número de usuarios que han respondido la pregunta, el número de usuarios que la han respondido correctamente, y el email de los usuarios para ambos casos. Para obtener esta información, una vez pulsado en “By question” se hace una llamada al servidor de tipo GET a la ruta /api/v1/questions.

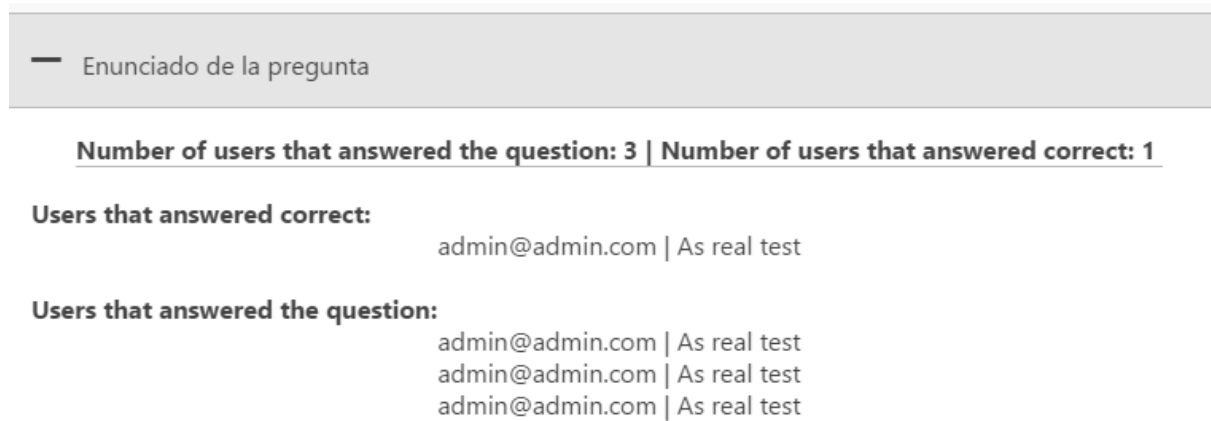


Figura 44. Pantalla de estadísticas por preguntas

### 3.9.2.4 Creación de preguntas

Al crear una pregunta tendremos que completar los campos de enunciado, respuestas 1, respuesta 2, respuesta 3, respuesta 4, respuesta correcta, segundos para contestar a la pregunta (sin tener en cuenta la longitud del audio adjunto), el tipo de test (entrenamiento y/o test real) y los archivos de imagen y/o audio asociados a la pregunta. El tiempo total para responder a la pregunta se calculará como el tiempo insertado en el formulario más la duración del audio en caso de existencia de éste.



< Back Add question

**Question**  
Fill question

**Answer 1**  
Fill answer 1

**Answer 2**  
Fill answer 2

**Answer 3**  
Fill answer 3

**Answer 4**  
Fill answer 4

**Correct Answer**  
Select from 1 to 4

**Seconds to answer the question**

**Test type:** ☐ For training ☐ For real test

**Select image and/or audio for the question:**  
 Ningún archivo seleccionado

**Files uploaded:** None

Upload question without files

Figura 45. Pantalla de creación de pregunta sin rellenar

< Back Add question

**Question**  
Enunciado de la pregunta

**Answer 1**  
Respuesta 1

**Answer 2**  
Respuesta 2

**Answer 3**  
Respuesta 3

**Answer 4**  
Respuesta 4

**Correct Answer**  
3

**Seconds to answer the question**  
60

**Test type:** ☒ For training ☒ For real test

**Select image and/or audio for the question:**  
 Ningún archivo seleccionado

**Files uploaded:**

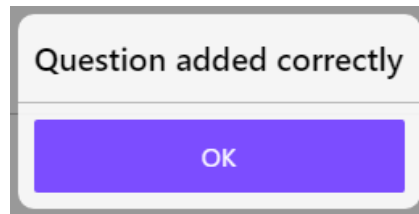
Name	Size
Allegro from Duet in C Major.mp3	31.36 MB
image1.jpg	0.01 MB

Upload question with files

Figura 46. Pantalla de creación de pregunta con los campos rellenados

Una vez pulsamos en “Upload question with files”, primero para los archivos adjuntos se hace una petición de tipo POST a la ruta /api/v1/upload en la que se parseará la información del formulario recibido con los archivos, se les añadirá al nombre del archivo la fecha de creación para evitar duplicados, y se guardarán en la ruta de carpetas public/files. Una vez recibida la

respuesta satisfactoria del servidor al cliente, se hace una petición de tipo POST a la ruta `/api/v1/questions/newQuestion` en la que se guarda en el campo `files` de la pregunta el nombre de los archivos asociados y seguidamente se guarda la pregunta completa en la base de datos.

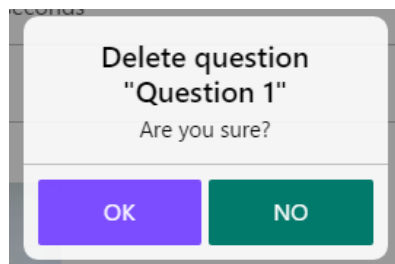


**Figura 47.** Popup de creación correcta de pregunta

### 3.9.2.5 Gestión de preguntas

En esta pantalla tendremos la información de todas las preguntas añadidas a la aplicación, pudiendo modificar sus campos o eliminarlas. Al entrar a la pantalla se hace una petición de tipo GET a la ruta `/api/v1/questions` que devuelve todas las preguntas creadas en el sistema.

Al pulsar en el botón de arriba a la derecha de cada pregunta, podremos eliminarla confirmando en el popup. Entonces, se hará una petición de tipo DELETE a la ruta `/api/v1/questions/:id` donde primero se eliminarán los archivos del servidor asociados a la pregunta y seguidamente se eliminará la pregunta de la base de datos.



**Figura 48.** Popup de eliminación de pregunta

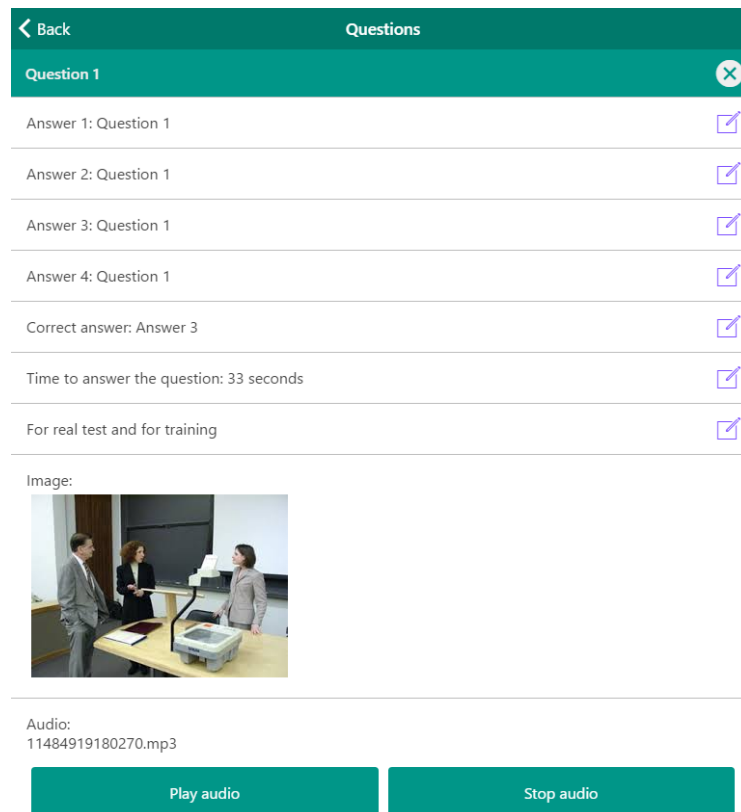


Figura 49. Pantalla de gestión de preguntas

### 3.9.3 Test

En la funcionalidad de test tenemos 2 modos:

- Entrenamiento: Tests sin límite de tiempo y la puntuación obtenida en el test no afecta a la puntuación de la cuenta del usuario. Además, los audios se pueden reproducir sin límite.
- Test real: Tests con límite de tiempo, y la puntuación obtenida sí que afectará a la puntuación final del usuario. Por otro lado, los audios sólo se pueden reproducir una vez.

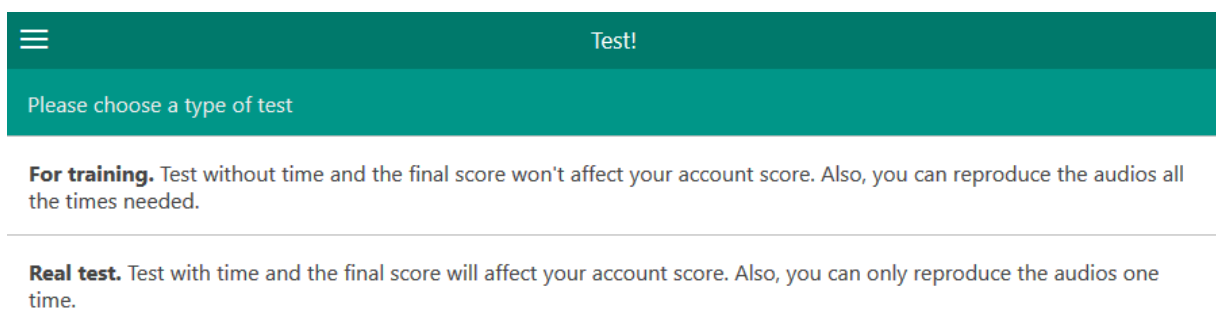


Figura 50. Pantalla principal de test

## ESTUDIO E IMPLEMENTACIÓN DE NUEVOS TIPOS DE EJERCICIOS EN LA PLATAFORMA MALL UP2B2

---

Una vez elegida la modalidad de test a realizar, se realiza una petición GET al servidor a la ruta `/api/v1/questions/test` en la que se seleccionará 10 preguntas del tipo seleccionado que el usuario no haya respondido correctamente con anterioridad. Al recibir las preguntas, en el cliente se mostrará la pantalla con una pregunta por pantalla, con su respectivo enunciado, respuestas, control del audio e imagen. Además se dispondrá de botones de adelante y atrás para poder navegar por las preguntas, y un botón de finalizar test en la última pregunta.

En el caso del tipo de test real, además dispondremos de un contador en la parte de arriba de la pregunta para poder visualizar el tiempo restante, además de que una vez pulsado el botón de iniciar el audio de cada pregunta, éste se bloqueará y no se permitirá volver a reproducirlo. En el caso de que el tiempo del contador llegue a 0 segundos, el test finalizará.

Cuando el test finalice, ya sea porque el tiempo disponible para responder el test se ha agotado o que el usuario ha pulsado en el botón de finalizar el test, en el cliente se recogerán las respuestas contestadas y se hará una petición de tipo POST a la ruta `/api/v1/questions/test/resolve` donde se seguirán los siguientes pasos:

1. Actualización de la puntuación total del usuario si el test era de tipo test real, teniendo en cuenta que el rango de valores a tomar es de -100 a 100 puntos. Además, se actualizará la lista de tests realizados del usuario, con el tiempo total empleado, si era de entrenamiento o test real, puntuación obtenida, número de respuestas correctas, número de respuestas incorrectas, si ha obtenido el bonus de tiempo y la fecha de realización.
2. Actualización de cada pregunta, añadiendo a la lista de usuarios que han realizado la pregunta y usuarios que han respondido a la pregunta, con la información de si era de entrenamiento o no, el id del usuario y el email del usuario.

Una vez recibido el estado de éxito 200 del servidor, en el cliente tendremos por un lado el resultado obtenido, el tiempo empleado, las respuestas correctas e incorrectas, y seguidamente el enunciado de cada pregunta junto con sus respuestas, indicando en rojo si la respuesta es incorrecta y en verde la correcta. En caso de no haber respondido a la pregunta se ha optado por no especificar la respuesta correcta.

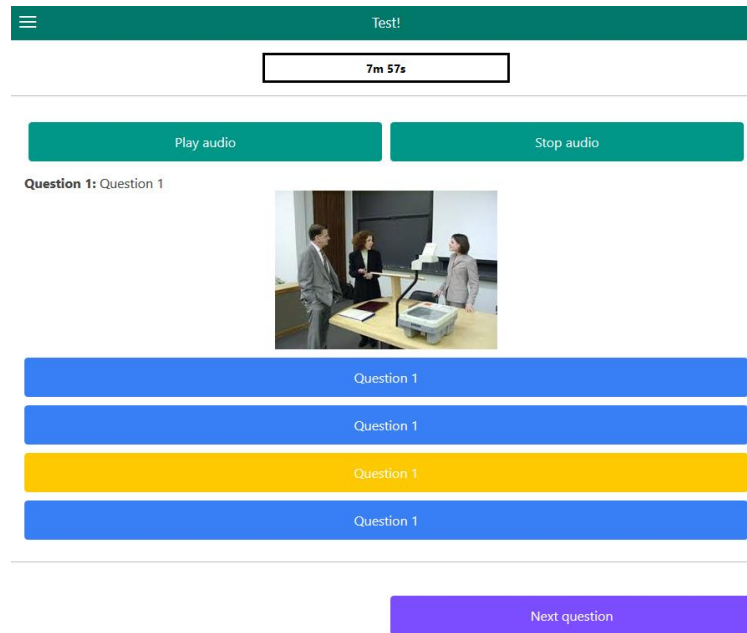


Figura 51. Pantalla de test 1

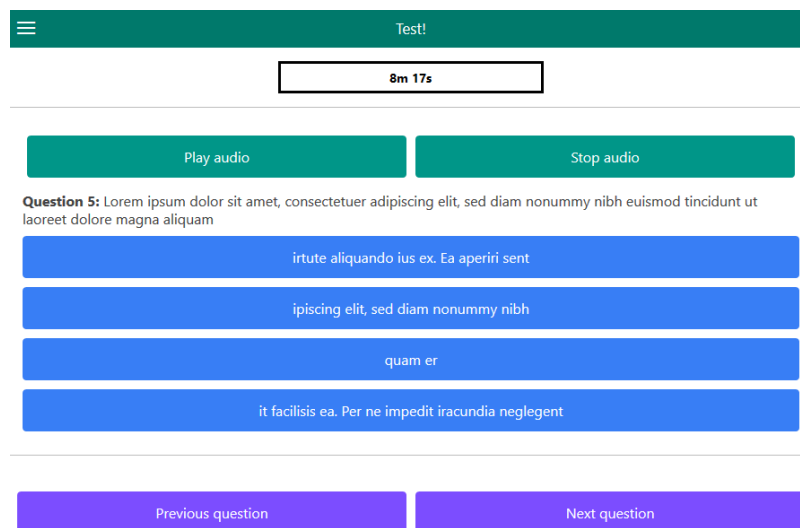


Figura 52. Pantalla de test 2



Figura 53. Pantalla de test 3

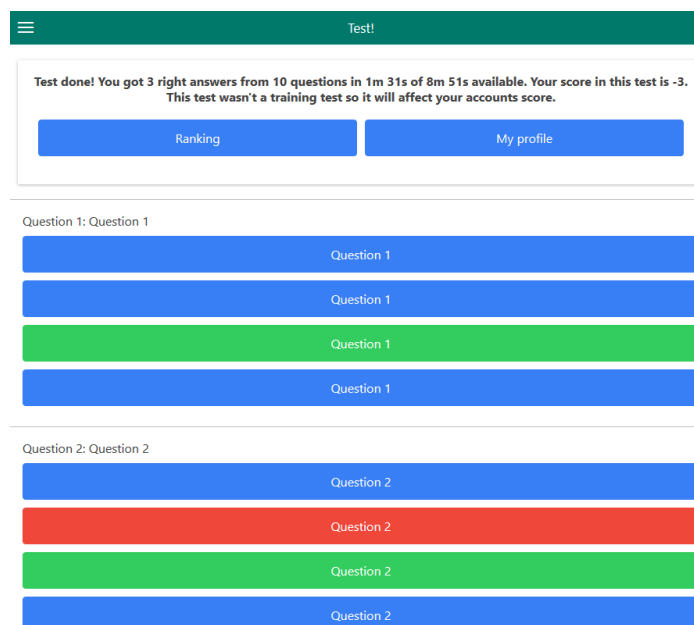


Figura 54. Pantalla de test al finalizarlo

### 3.9.4 Ranking

En la pantalla de ranking podremos ver la posición de cada usuario dependiendo de los puntos que tenga, ordenados de mayor a menor puntuación. Al entrar en la pantalla se hace una petición GET a la ruta `/api/v1/users/ranking` en la que se obtiene los usuarios ordenados por el campo score. El rango de la puntuación es de -100 a 100 puntos.

Ranking	
USER	SCORE
★   admin@admin.com	14
★   rocio.3.romero@gmail.com	0
☆   prueba@prueba.com	0
rocio.romeropaner@alumnos.upm.es	-3

Figura 55. Pantalla de Ranking

### 3.9.5 Perfil


En la pantalla de mi perfil el usuario podrá visualizar toda la información relativa a su cuenta, cambiar su contraseña y ver un registro de los tests que ha hecho. Al entrar en la pantalla se hará una petición de tipo GET a la ruta `/api/v1/users/:id` para obtener la información específica del usuario.

My Profile

Full name: Admin

DNI: 123456789A

Email: admin@admin.com

Password: \*\*\*\*\* 

Score: 14 points

Degree: Admin

All tests done

Training test | 20/01/2017 | Number questions correct: 0 | Number questions wrong: 4 | Time spent: 3 seconds | Total score: -3

Real test | 20/01/2017 | Number questions correct: 1 | Number questions wrong: 3 | Time spent: 4 seconds | Total score: 1

Training test | 21/01/2017 | Number questions correct: 0 | Number questions wrong: 4 | Time spent: 4 seconds | Total score: -3

Real test | 21/01/2017 | Number questions correct: 2 | Number questions wrong: 2 | Time spent: 6 seconds | Total score: 5

Figura 56. Pantalla de perfil de usuario

Si se selecciona el botón de cambiar de contraseña, aparecerá un popup en el que tendremos que rellenar el campo de contraseña actual y nueva contraseña. Una vez pulsado en el botón “Change”, se realizará una petición al servidor de tipo PUT a la ruta `/api/v1/password/:id` en la que se seguirán los siguientes pasos:

1. Comprobación de existencia del usuario en la base de datos, en caso contrario se responderá con un código de estado 404.

2. Si el usuario existe en la base de datos, se comprobará que la contraseña actual coincide con la existente, y en caso de que no coincidan se enviará un código de estado 401.
3. Por otro lado, si las contraseñas coinciden, se hashearán la nueva contraseña y se modificará en la base de datos, respondiendo con un código de estado 200.

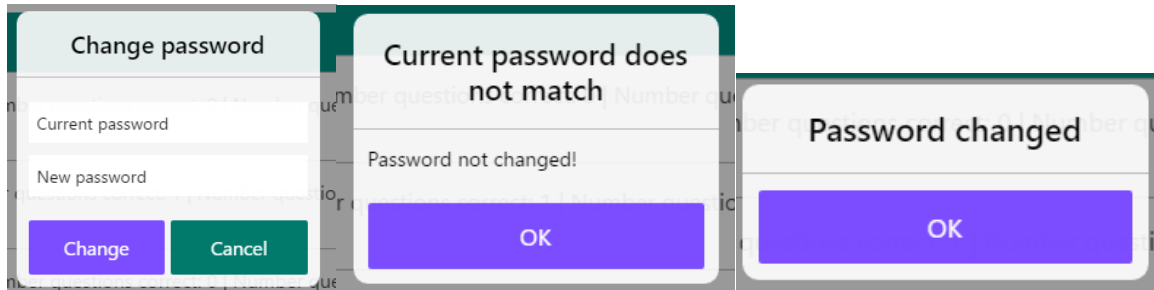


Figura 57. Popups de cambio de contraseña de usuario

### 3.9.6 Desconectar

Al pulsar es desconectar aparecerá un popup de confirmación que, en caso de confirmar la desconexión, se limpiará el contenido del local storage y se redirigirá a la pantalla de login.

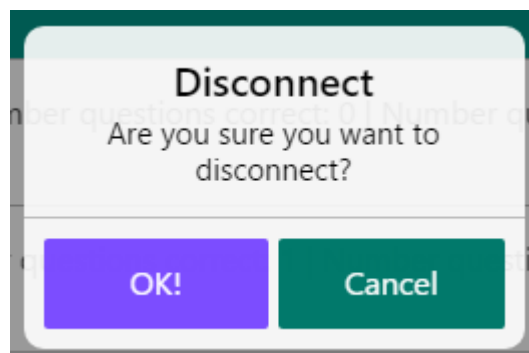


Figura 58. Popup de desconexión







## 4 Conclusiones y trabajos futuros

Como hemos ido analizando en los anteriores apartados de la memoria, aplicando las tecnologías de Ionic, AngularJS, Node.js y MongoDB hemos sido capaces de desarrollar satisfactoriamente una plataforma MALL que permita la reproducción de audios y visualización de imágenes para ejercicios de inglés.

Este desarrollo ha sido llevado a cabo debido a la necesidad de la escuela de disponer de una herramienta de aprendizaje para los alumnos, de cara a obtener el nivel B2 de inglés y afrontar satisfactoriamente la prueba TOEIC disponible en la ETSIST. De esta manera, conseguimos ampliar los tipos de ejercicios disponibles implementados en la plataforma UP2B2 y permitir al alumnado tener más control de su aprendizaje, pudiendo preparar las partes de Listening y Reading del examen.

Para el almacenamiento de los archivos de audio e imagen se ha tomado la decisión de guardarlos en el sistema de archivos que nos ofrece Node.js en la parte servidora, y para la visualización y reproducción en la parte cliente hemos utilizado las herramientas que nos ofrecen HTML5 y JavaScript con los elementos de Audio e Img.

De este modo, gracias a Ionic hemos podido generar una web y aplicación móvil que nos ofrece una plataforma MALL para el aprendizaje del inglés con ejercicios para practicar la parte de Listening del examen TOEIC. Así, a partir de este punto en un futuro tenemos la posibilidad de realizar más ejercicios con las herramientas y tecnologías aquí presentadas, y producir una integración de la versión previa de UP2B2 junto con esta.

### 4.1 Conclusiones

Tras la realización de este proyecto fin de carrera, podemos afirmar que se ha conseguido el objetivo, que era la realización de una plataforma MALL que disponga de ejercicios con audio e imágenes. De cara a la prueba TOEIC ofrecida en la ETSIST para obtener el título B2 en inglés, era necesario aportar herramientas para complementar el estudio específico de la parte de Listening. De esta manera, combinando la aplicación previa de UP2B2, podemos ofrecer al alumnado una herramienta completa para practicar ambas partes del examen, tanto Listening como Reading.

Para llevar a cabo este desarrollo fue necesario el estudio de las metodologías de implementación de las tecnologías MALL, así como el uso por parte de los usuarios y aplicaciones que tienen actualmente. Seguidamente, se realizó un estudio de las necesidades de la plataforma, tanto para la parte cliente como para la parte servidor, decidiendo las tecnologías y módulos a implementar en ambos lados. El proceso de analizar los requisitos de la plataforma y herramientas a utilizar es importante para el correcto desarrollo posterior, pues una mala elección puede implicar la lentitud en el desarrollo por necesidad de volver atrás y cambiar partes ya implementadas.

Además, se tuvo que tener en cuenta las medidas de seguridad a tomar, así como la mejor manera de almacenar los archivos tanto de audio como de imagen en el servidor. Una vez realizados los requisitos necesarios, se procedió al desarrollo de las rutas en el servidor, así como las vistas en la parte cliente, conectando finalmente ambas partes. Gracias a la gran comunidad de Node.js disponemos de muchos módulos en el repositorio de npm junto con explicaciones de cómo usarlos, por lo que es interesante antes de implementar una funcionalidad ver si no existe un módulo previamente desarrollado. Por otro lado, respecto a la comunidad de Ionic, ésta es más escasa y hemos necesitado apoyarnos en la comunidad de Angular.JS, que es mayor.

Una de las mayores dificultades encontradas fue la del almacenamiento de los archivos en la parte servidora, pues no existe un estándar de desarrollo en el servidor habiendo recibido los documentos desde un cliente de Ionic o Angular.JS. Finalmente, se tomó la decisión de almacenar los archivos en el sistema de archivos de Node.JS analizando la información con el módulo Formidable y enviándolo desde el cliente con el módulo de Angular FileUploader.

Acabado el desarrollo, fue necesario realizar pruebas de funcionamiento tanto en local como en remoto, poniendo la aplicación en servidores de la ETSIST y en los distintos dispositivos disponibles como son móviles, tablets y navegadores web.

Así, gracias al desarrollo de la aplicación he conseguido consolidar y ampliar mis conocimientos en tecnologías web obtenidos en la asignatura optativa “Aplicaciones Telemáticas Basadas en Web”, además de obtener otros nuevos como es el lenguaje Javascript, Node.js y bases de datos no SQL como MongoDB. Además, he obtenido conocimientos de control de versiones y de documentación.

### 4.2 Trabajos futuros

Como trabajos futuros, destacamos los siguientes:

1. Integración de ésta versión de la aplicación con la antigua, para así poder disponer de ejercicios gramaticales, de vocabulario, de Reading y de Listening en una misma plataforma.
2. Añadir funcionalidad de tests sin conexión a internet, ofreciendo la posibilidad de pre-descargarse tests cuando la aplicación detecte que se dispone de Wifi, para así posteriormente poder realizarlos sin necesidad de tener conexión a internet y evitar el consumo de datos.
3. Añadir ejercicios que permitan la grabación de voz dado un tema de conversación, como pueden ser preguntas de entrevistas, para su posterior corrección por personal docente.
4. Agregar metodologías de juego como pueden ser medallas por “X tests hechos” o “racha de respuestas correctas”.
5. Mejora visual de la aplicación.
6. Implementar más apartados en la sección de administración para obtener más información sobre el uso que hacen los alumnos de la aplicación.
7. Disponer de un apartado en la aplicación con contenido teórico de consulta.





## Bibliografía

- [1] A. Kukulska-Hulme, «Mobile-assisted language learning,» *C.Chapelle (Ed.) The encyclopedia of applied linguistics*, pp. 3701-3709, 2013.
- [2] A. J. Q. K. A. K.-H. a. A. E. Valérie Demouy, «Why and how do distance learners use mobile devices for language learning?,» *The EUROCALL Review*, vol. 23, nº 2, September 2015.
- [3] C.-M. a. L. Y.-L. Chen, «Personalised context-aware ubiquitous learning system for supporting effective English vocabulary learning,» *Interactive Learning Environments*, pp. 341-364, 2010.
- [4] G. S. a. P. Hubbard, «Some Emerging Principles for Mobile-assisted Language Learning,» *The International Research Foundation for English Language Education*, 2013.
- [5] Node.js, «Node.js,» [En línea]. Available: <https://nodejs.org/es/>. [Último acceso: 15 Octubre 2016].
- [6] NPM, «NPM,» [En línea]. Available: <https://www.npmjs.com/>. [Último acceso: 10 Octubre 2016].
- [7] Express, «Express,» [En línea]. Available: <http://expressjs.com/es/>. [Último acceso: 18 Septiembre 2016].
- [8] MongoDB, «MongoDB,» [En línea]. Available: <https://www.mongodb.com/es>. [Último acceso: Octubre 10 2016].
- [9] RoboMongo, «RoboMongo,» [En línea]. Available: <https://robomongo.org/>. [Último acceso: 10 Octubre 2016].
- [10] I. Framework, «Ionic Framework,» [En línea]. Available: <http://ionicframework.com/>. [Último acceso: 17 Octubre 2016].
- [11] Apache, «Apache Cordova,» [En línea]. Available: <https://cordova.apache.org/>. [Último acceso: 16 Octubre 2016].
- [12] AngularJS, «AngularJS,» [En línea]. Available: <https://angularjs.org/>. [Último acceso: 17 Octubre 2016].

- [13] AudioTrimmer, «AudioTrimmer,» [En línea]. Available: <https://audiotrimmer.com/>. [Último acceso: 20 Febrero 2017].
- [14] T. Beuckens, «ELLLO,» [En línea]. Available: <http://www.ello.org/>. [Último acceso: 20 Febrero 2017].
- [15] Mongoose, «Mongoose,» [En línea]. Available: <http://mongoosejs.com/>. [Último acceso: 8 Octubre 2016].
- [16] GitHub, «GitHub,» [En línea]. Available: <https://github.com/>. [Último acceso: 25 Septiembre 2016].
- [17] SourceTree, «SourceTree,» [En línea]. Available: <https://www.sourcetreeapp.com/>. [Último acceso: 15 Septiembre 2016].
- [18] PostMan, «PostMan,» [En línea]. Available: <https://www.getpostman.com/>. [Último acceso: 25 Octubre 2016].
- [19] SublimeText, «SublimeText,» [En línea]. Available: <https://www.sublimetext.com/>. [Último acceso: 28 Septiembre 2016].
- [20] JWT, «JSON Web Tokens,» [En línea]. Available: <https://jwt.io/>. [Último acceso: 15 Octubre 2016].



## Anexo A. Manual de usuario

### A.1 Parte servidor

#### A.1.1. Node.JS

Para instalar Node.JS en Windows basta con acceder a la web de Node.JS [5] y descargar e instalar el instalador.


#### Descargas


Versión actual: v6.9.5 (includes npm 3.10.10)


Descargue el código fuente de Node.js o un instalador pre-compilado para su plataforma, y comience a desarrollar hoy.

**LTS**  
Recomendado para la mayoría

**Actual**  
Últimas características

  
**Windows Installer**  
node-v6.9.5-x64.msi

  
**Macintosh Installer**  
node-v6.9.5.pkg

  
**Source Code**  
node-v6.9.5.tar.gz

**Windows Installer (.msi)**

**Windows Binary (.zip)**

**macOS Installer (.pkg)**

**macOS Binaries (.tar.gz)**

**Linux Binaries (x86/x64)**

**Linux Binaries (ARM)**

**Source Code**

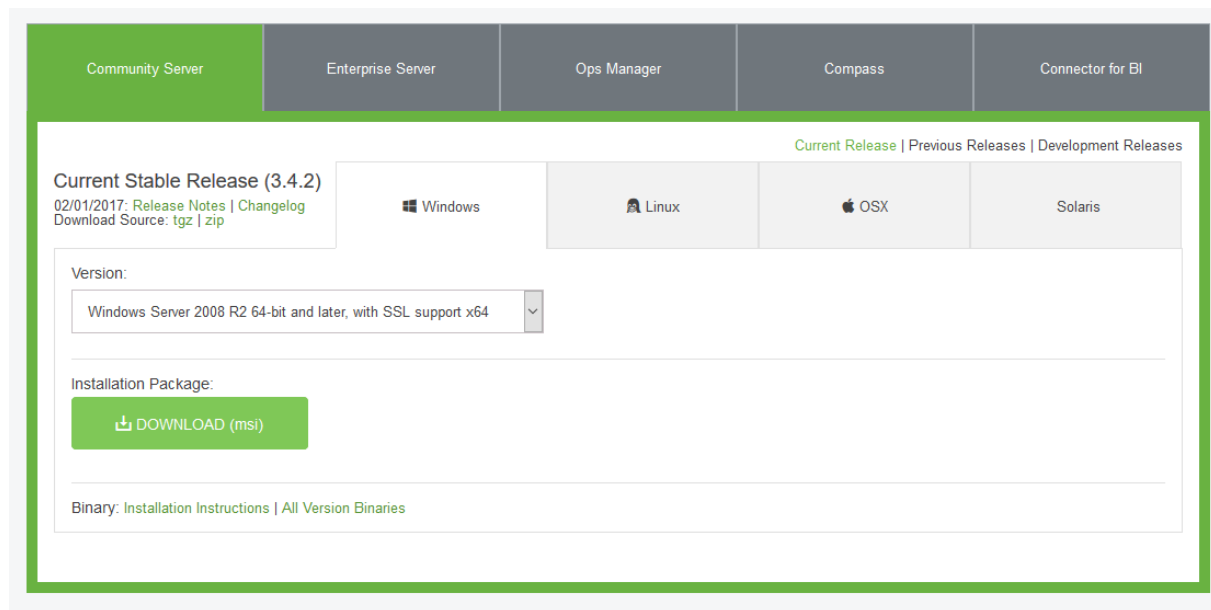
32-bit		64-bit	
32-bit		64-bit	
64-bit			
64-bit			
32-bit		64-bit	
ARMv6	ARMv7	ARMv8	
node-v6.9.5.tar.gz			

**Figura 59. Descarga de Node.JS**

Una vez instalado, podremos acceder a Node.js y npm desde cualquier directorio de nuestro sistema.

#### A.1.2. MongoDB

Para instalar MongoDB, también debemos acceder a la web de MongoDB [8] y descargar el instalador e instalarlo.



**Figura 60. Descarga de MongoDB**

Una vez instalado, para ejecutar MongoDB podemos ir al directorio donde se ha instalado (en nuestro caso es la ruta C:\Program Files\MongoDB\Server\3.2\bin) y hacer doble click en el archivo mongod.

### A.1.3. *Express*

Una vez instalado Node.JS, ya disponemos en nuestro ordenador del gestor de paquetes de Node.JS, npm. Para instalar Express en nuestro proyecto usaremos el comando siguiente [7]:

```
$ npm install express --save
```

**Figura 61. Instalación Express**

Si lo que queremos es disponer de un proyecto de Express con un esqueleto previamente creado, además necesitaremos de Express-generator, que se instala de la siguiente manera:

```
$ npm install express-generator -g
```

**Figura 62. Instalación Express-generator**

Una vez instalado, podremos crear un proyecto de Express con el comando “express [nombre\_proyecto]”.

Por último, para inicializar la parte servidora basta con ejecutar el comando como sigue:

```
$ node app.js
```

**Figura 63. Ejecución node**

#### A.1.4. Npm

Por otro lado, si lo que queremos es instalar los paquetes de un proyecto previamente desarrollado, ejecutaremos el comando “npm install”, que leerá el archivo package.json y descargará e instalará los paquetes especificados.

```
{
  "name": "HELLO",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node ./bin/www"
  },
  "dependencies": {
    "body-parser": "~1.15.1",
    "cookie-parser": "~1.4.3",
    "debug": "~2.2.0",
    "express": "~4.13.4",
    "jade": "~1.11.0",
    "morgan": "~1.7.0",
    "serve-favicon": "~2.3.0"
  }
}
```

Figura 64. Archivo package.json

## A.2 Parte cliente

### A.2.1. Ionic

Para crear un nuevo proyecto con Ionic primero debemos tener instalados Cordova e Ionic. Para ello, ejecutaremos el siguiente comando [10]:

```
npm install -g cordova ionic
```

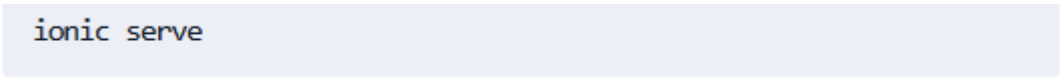
Figura 65. Instalación Cordova e Ionic

Una vez instalados, para crear un nuevo proyecto con plantilla llamado “myApp” ejecutaremos el comando siguiente:

```
ionic start --v2 myApp tabs
```

Figura 66. Comienzo nuevo proyecto Ionic

Finalmente, una vez creado el proyecto, para arrancar la aplicación tendremos que ejecutar el comando como sigue:



```
ionic serve
```

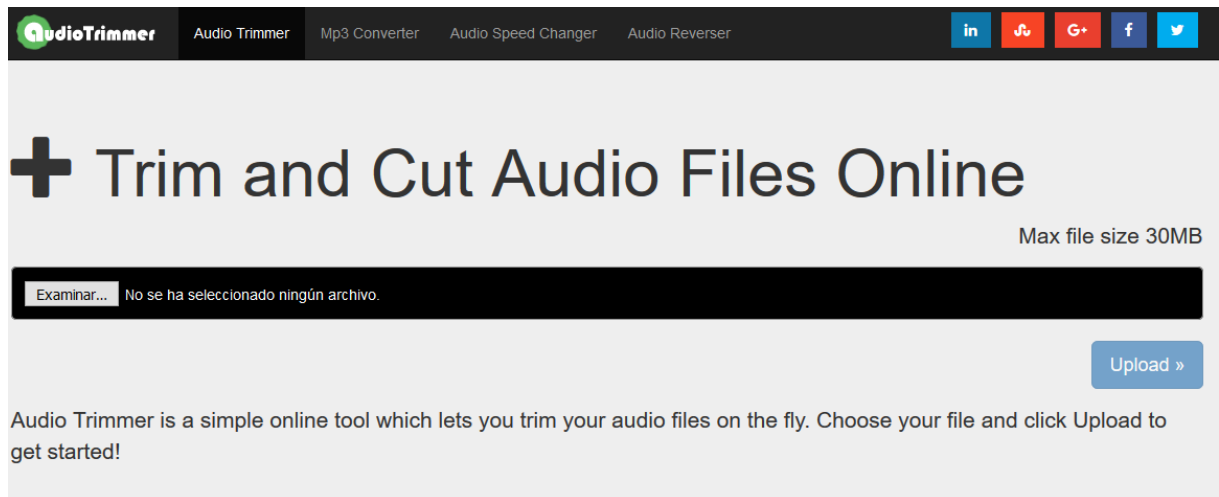
**Figura 67. Iniciar proyecto en Ionic**

Una vez iniciado el proyecto, los cambios en el código se detectarán automáticamente y se volverá a ejecutar la aplicación sin necesidad de volver a aplicar el comando.

## Anexo B. Recortar audio

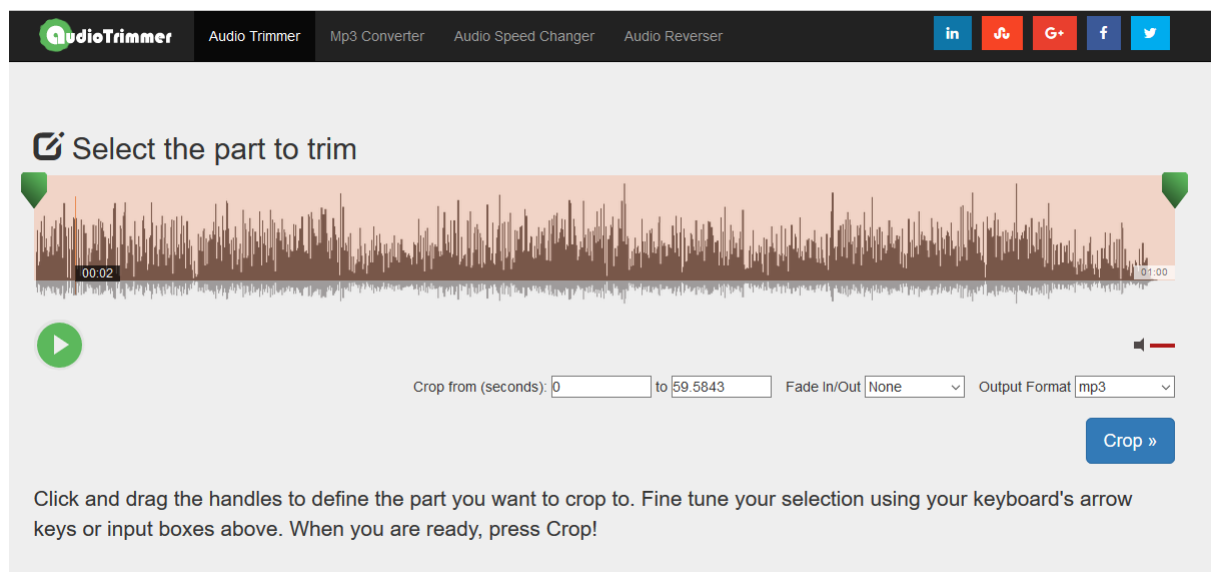
AudioTrimmer [13] es una herramienta online que permite recortar audios y que nos puede ser útil a la hora de obtener audios para la aplicación. A continuación se muestran los pasos a seguir para obtener un audio cortado.

En primer lugar, tendremos que pulsar en el botón “Examinar”, que nos abrirá una ventana para seleccionar el archivo de audio que queremos cortar.



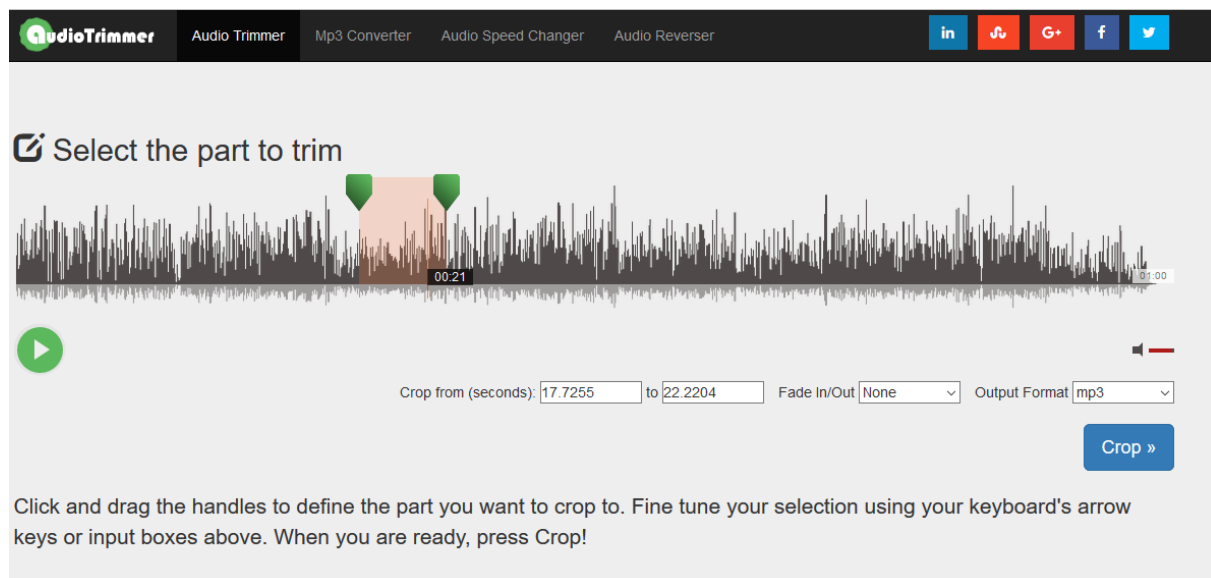
**Figura 68. AudioTrimmer: Paso 1**

Una vez seleccionado el archivo, éste se nos abrirá en la página web y podremos elegir el rango que queremos obtener tanto por las guías que tenemos arriba del audio como con los recuadros que tenemos en la parte de abajo donde pondremos el rango en segundos. Además, nos permitirá escuchar desde dónde a dónde lo hemos seleccionado, el formato de generación y si queremos que al finalizar el audio haya un efecto de apagado.



**Figura 69. AudioTrimmer: Paso 2**

Cuando tengamos el rango de la pista de audio que queremos, pulsaremos en el botón “Crop” para que se nos genere el archivo.



**Figura 70. AudioTrimmer: Paso 3**

Después, para poder descargar el archivo tendremos que pulsar en el botón “Download”.

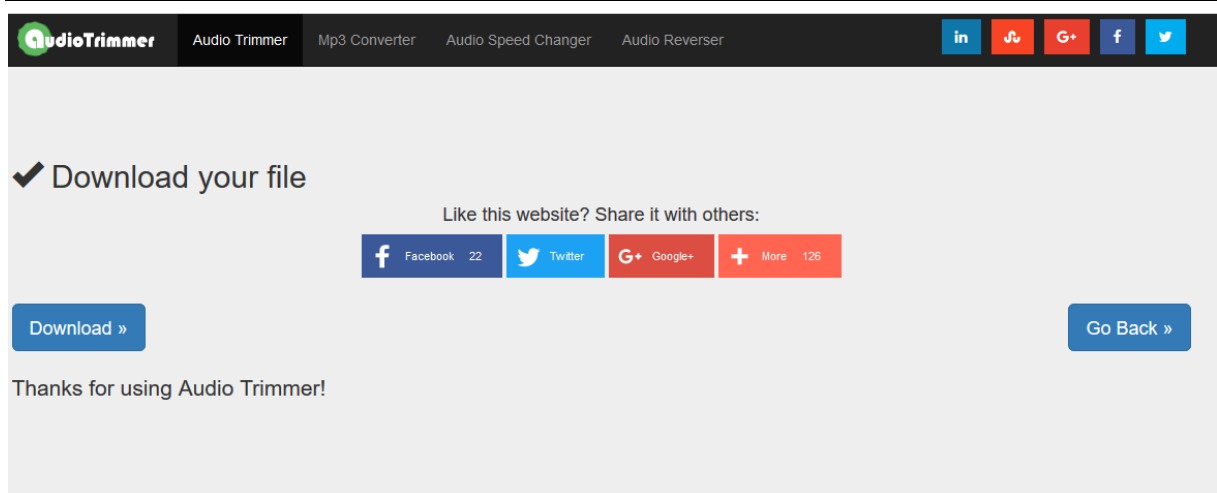


Figura 71. AudioTrimmer: Paso 4

Finalmente, una vez pulsado en “Download” elegiremos si queremos abrir o guardar el archivo en nuestro ordenador.

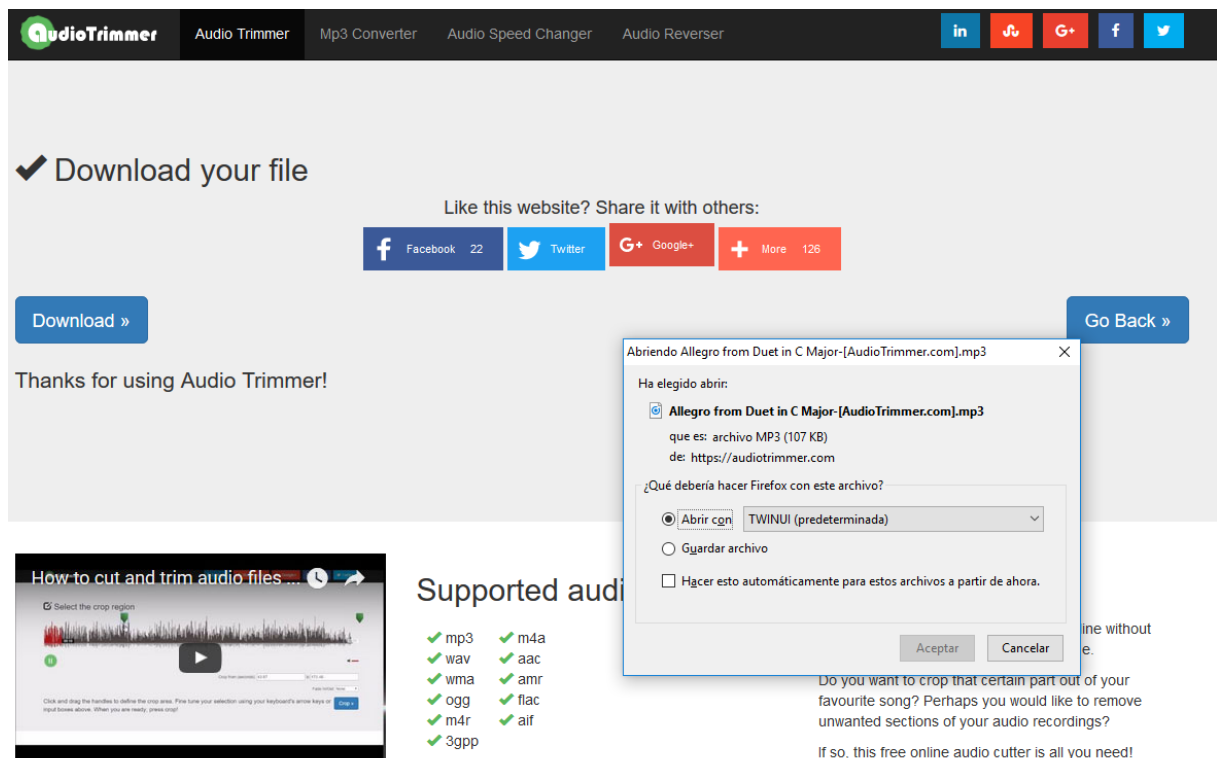



Figura 72. AudioTrimmer: Paso 5





## Anexo C. Descargar material de Listening

La página web ELLLO [14] dispone de una colección de más de 1000 ejercicios con Listenings de licencia Creative Commons separados en 5 niveles: principiante alto, intermedio bajo, intermedio medio, intermedio alto y avanzado.



**Level 6 : High-Intermediate**

These activities often cover more difficult topics and use some more advanced vocabulary.






















	<p>1151 <b><u>Buenos Aires</u></b></p> <p>Daniel gets some travel tips from Valeria about Buenos Aires.</p>	<p>Intermediate 6</p> <p>Time: 2:55</p> <p>Valeria Daniel</p> <div style="display: flex; align-items: center;">   </div>
	<p>1150 <b><u>Urban Plannning</u></b></p> <p>Shifani talks about making cities more efficient places to live.</p>	<p>Intermediate 6</p> <p>Time: 3:55</p> <p>Shifani Todd</p> <div style="display: flex; align-items: center;">   </div>
	<p>1149 <b><u>Cars and Traffic</u></b></p> <p>Shifani talks about how to look at traffic.</p>	<p>Intermediate 6</p> <p>Time: 3:53</p> <p>Shifani Todd</p> <div style="display: flex; align-items: center;">   </div>
	<p>1148 <b><u>Scary Food</u></b></p> <p>Paul and Todd look at the threat of disease to the food supply.</p>	<p>Intermediate 6</p> <p>Time: 3:25</p> <p>Paul / Todd</p> <div style="display: flex; align-items: center;">   </div>
	<p>1147 <b><u>Animal Workers</u></b></p> <p>Paul and Todd look at animals being part of the labor force.</p>	<p>Intermediate 6</p> <p>Time: 3:36</p> <p>Paul / Todd</p> <div style="display: flex; align-items: center;">   </div>
	<p>1146 <b><u>Captivity</u></b></p> <p>Todd and Paul discuss animals living in captivity.</p>	<p>Intermediate 6</p> <p>Time: 2:45</p> <p>Paul / Todd</p> <div style="display: flex; align-items: center;">   </div>
	<p>1145 <b><u>Tiger Farm</u></b></p> <p>Todd talks to Paul about visiting a tiger farm.</p>	<p>Intermediate 6</p> <p>Time: 4:05</p> <p>Paul / Todd</p> <div style="display: flex; align-items: center;">   </div>

Figura 73. Ejemplo de audios en ELLLO

## ESTUDIO E IMPLEMENTACIÓN DE NUEVOS TIPOS DE EJERCICIOS EN LA PLATAFORMA MALL UP2B2

Las actividades contienen un archivo MP3 para descargar, la transcripción del audio y un test interactivo. Los Listening incluyen diferentes temáticas tales como entrevistas, canciones, presentaciones, encuestas, juegos, noticias, etc. Además, en cada audio se nos informa del acento que tendrá cada interlocutor, así como la duración.

Así, para descargar los audios basta con pulsar en el enlace “Free MP3” que se encuentra debajo de la imagen de los protagonistas del audio y en la página que se abre pulsar botón derecho del ratón encima del reproductor de audio y seleccionar la opción de “Guardar audio como...”.

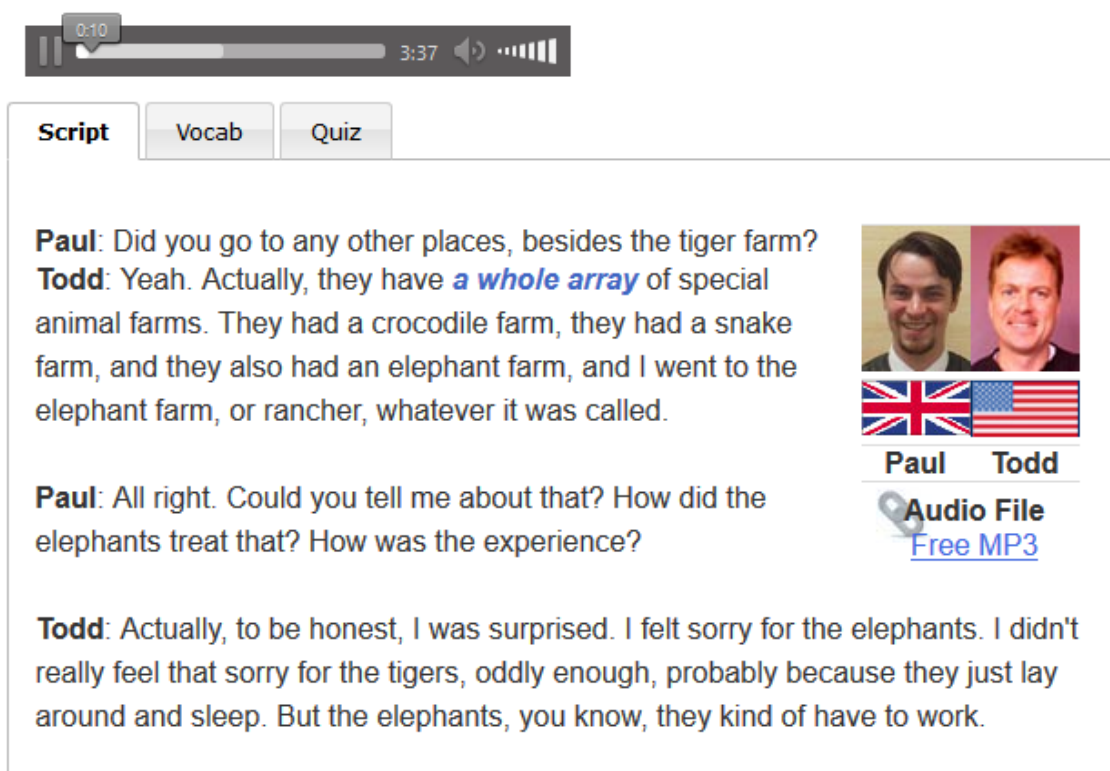


Figura 74. Descarga de audio en ELLLO (1)

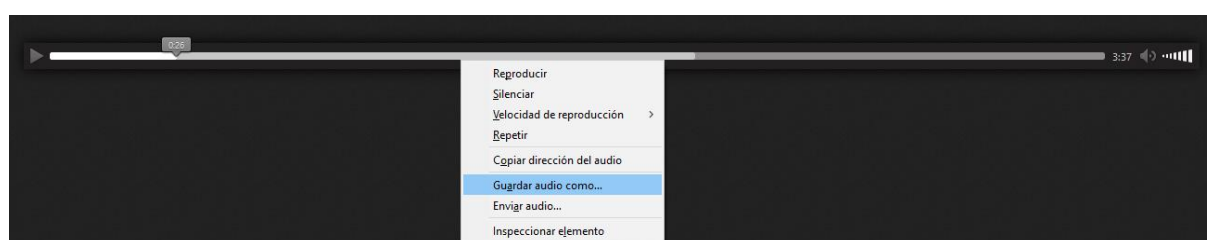


Figura 75. Descarga de audio en ELLLO (2)

Otra manera de descargarlo es directamente en la página del ejercicio, en el reproductor de audio pulsar con botón derecho y seleccionar “Guardar audio como...”

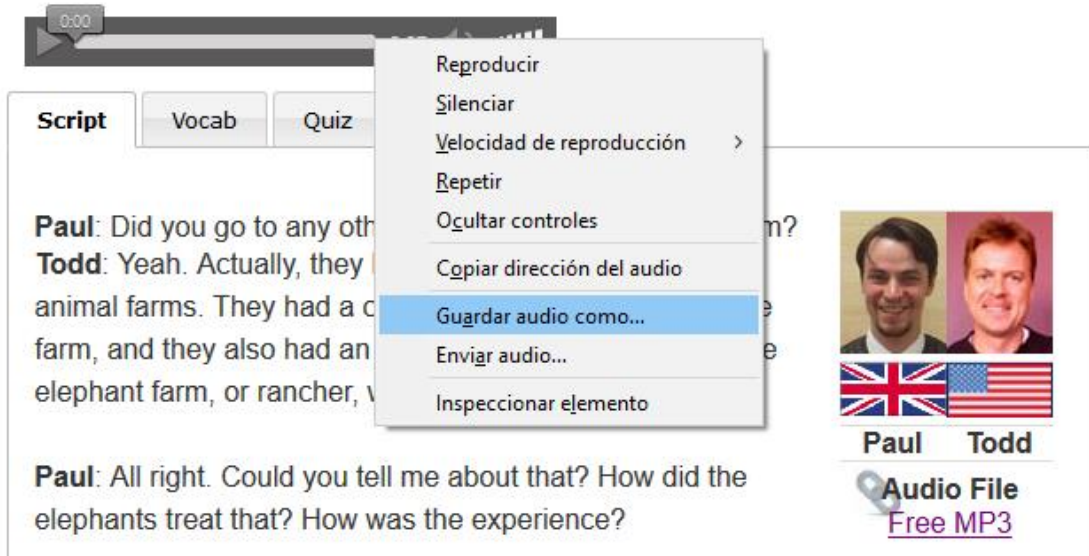


Figura 76. Descarga de audio en ELLLO (3)