



ugr | Universidad
de **Granada**

TRABAJO FIN DE GRADO
GRADO DE INGENIERÍA EN INFORMÁTICA

Aplicación multiplataforma para el aprendizaje del lenguaje musical

Subítulo del Proyecto

Autor

Sergio Hervás Cobo

Directores

Luis López Escudero
Germán Arroyo Moreno



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE
TELECOMUNICACIÓN

—
Granada, mes de 201



Aplicación multiplataforma para el aprendizaje del lenguaje musical

Subtítulo del proyecto.

Autor

Sergio Hervás Cobo

Directores

Luis López Escudero
Germán Arroyo Moreno

Aplicación multiplataforma para el aprendizaje del lenguaje musical: Subtítulo del proyecto

Sergio Hervás Cobo

Palabras clave: palabra_clave1, palabra_clave2, palabra_clave3,

Resumen

Poner aquí el resumen.

Project Title: Project Subtitle

Sergio Hervás Cobo

Keywords: Keyword1, Keyword2, Keyword3,

Abstract

Write here the abstract in English.

Yo, **Nombre Apellido1 Apellido2**, alumno de la titulación **TITULACIÓN de la Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI XXXXXXXXX, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Nombre Apellido1 Apellido2

Granada a X de mes de 201 .

D. **Nombre Apellido1 Apellido2 (tutor1)**, Profesor del Área de XXXX del Departamento YYYY de la Universidad de Granada.

D. **Nombre Apellido1 Apellido2 (tutor2)**, Profesor del Área de XXXX del Departamento YYYY de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado ***Título del proyecto, Subtítulo del proyecto***, ha sido realizado bajo su supervisión por **Nombre Apellido1 Apellido2 (alumno)**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a X de mes de 201 .

Los directores:

Nombre Apellido1 Apellido2 (tutor1) **Nombre Apellido1 Apellido2 (tutor2)**

Agradecimientos

Poner aquí agradecimientos...

Índice general

1. Introducción	1
1.1. Motivacion	2
1.2. Objetivos	3
2. Estado del Arte	5
2.1. Software a desarrollar	5
2.1.1. Aplicaciones de aprendizaje	5
2.1.2. Aplicaciones de aprendizaje musical	10
2.2. Desarrollo de Software	12
2.2.1. Flutter	12
2.2.2. React Native	12
2.2.3. Dart	13
2.2.4. Kotlin	13
2.2.5. NodeJS	14
2.2.6. React	14
2.2.7. Angular	15
2.2.8. MongoDB	15
2.2.9. Pilas MERN & MEAN	16
2.2.10. MariaDB	16
3. Especificación de requisitos	19
3.1. Introducción	19
3.1.1. Propósito	19
3.1.2. Ámbito del Sistema	19
3.1.3. Definiciones, Acrónimos y Abreviaturas	20
3.1.4. Referencias	20
3.1.5. Visión General del Documento	20
3.2. Descripción General	21
3.2.1. Perspectiva del producto	21
3.2.2. Funciones del producto	22
3.2.3. Características de los usuarios	23
3.2.4. Restricciones	23
3.2.5. Suposiciones y dependencias	23

3.3. Requisitos Específicos	23
3.3.1. Interfaz de usuario	24
3.3.2. Requisitos funcionales	25
3.3.3. Requisitos no funcionales	26
4. Planificación	29
4.1. Introducción	29
4.2. Velocidad	31
4.3. Product Backlog	32
4.4. Sprints	34
4.4.1. Sprint #1 - Documentación inicial	34
4.4.2. Sprint #2 - Documentación de requisitos	34
4.4.3. Sprint #3 - Documentación de HU	35
4.4.4. Sprint #4 - Diseño y comienzo del desarrollo	35
4.4.5. Sprint #5 - Registro y login de usuarios	35
4.4.6. Sprint #6 - Tests y progreso	36
4.4.7. Sprint #7 - Entrada de micrófono y detección de tono	36
4.4.8. Sprint #8 - Funcionalidades del profesor con lecciones	37
4.4.9. Sprint #9 - Funcionalidades del profesor con tests .	37
4.4.10. Sprint #10 - Funcionalidades del administrador y ranking	38
4.4.11. Sprint #11 - Funcionalidades poco prioritarias y valor añadido	38
4.4.12. Sprint #12 - Mejoras de código	38
4.4.13. Sprint #13 - Pruebas	39
4.4.14. Sprint #14 - Conclusión y finalización del proyecto .	39
4.5. Diagrama de Gantt	39
5. Análisis del problema	43
5.1. Introducción	43
5.2. Historias de Usuario	43
5.3. Diagrama de Clases	48
6. Diseño	51
6.1. Introducción	51
6.2. Arquitectura del sistema	51
6.3. Diagrama de base de datos	52
6.4. Diagrama de clases	54
6.5. Diagramas de secuencia	54
6.6. Diseño de interfaces de usuario	55

7. Implementación	63
7.1. Estructura del proyecto	63
7.2. Funcionalidad genérica	64
7.3. Funcionalidad de usuario	64
8. Pruebas	65
9. Conclusiones	67

Índice de figuras

2.1.	Logo de Duolingo	5
2.2.	Ejercicio de Duolingo donde el usuario debe adivinar qué palabra corresponde al audio	6
2.3.	Menú de selección de las lecciones de Artly clasificadas en movimientos artísticos	7
2.4.	Ciudad de BMath donde se seleccionan los niveles y se construyen los edificios para mostrar el progreso del usuario	8
2.5.	Ejercicio de BMath sobre cálculo matemático (sumas y restas) con calculadora, con papel y mentalmente	9
2.6.	Lista de lecciones que ofrece la aplicación Curso de Lenguaje Musical	10
2.7.	Logo de Flutter	12
2.8.	Logo de React Native	12
2.9.	Logo de Dart	13
2.10.	Logo de MongoDB	15
2.11.	Pilas MEAN y MERN	16
2.12.	Logo de MariaDB	16
3.1.	Diagrama general de la perspectiva del producto a desarrollar.	21
3.2.	Primer boceto de la interfaz de usuario de la página de inicio de la aplicación para los usuarios, donde se muestra la lista de las lecciones	24
4.1.	Diagrama de la arquitectura del sistema, donde se muestra la comunicación entre el servidor y la aplicación móvil.	30
4.2.	Diagrama de Gantt de la planificación del proyecto, divivido por Sprints.	40
4.3.	Diagrama de Gantt de los cuatro primeros Sprints.	41
4.4.	Diagrama de Gantt de los Sprints 5, 6, 7 y 8.	41
4.5.	Diagrama de Gantt de los Sprints 9, 10 y 11.	42
4.6.	Diagrama de Gantt de los Sprints 12, 13 y 14.	42
5.1.	Diagrama de clases de nuestro proyecto donde se muestran las relaciones entre las diferentes clases que componen el sistema.	48

6.1.	Diagrama de la arquitectura del sistema, donde se muestra la comunicación entre el servidor y la aplicación móvil.	52
6.2.	Diagrama de base de datos del sistema, donde se muestran los documentos que se van a almacenar en nuestra base de datos de MongoDB.	53
6.3.	Diagrama de clases del sistema donde se detallan las propiedades y las relaciones de las distintas clases o entidades que tendrá el software.	54
6.4.	Diagrama de secuencia de un usuario registrándose, el cual interactuará con la vista para poder hacer la petición al controlador y guardar su usuario en el modelo.	55
6.5.	Boceto de la pantalla de inicio de sesión, donde se pedirá al usuario el correo electrónico y la contraseña.	56
6.6.	Boceto de la pantalla de registro, donde se pedirá al usuario sus datos personales necesarios para crear la cuenta como el nombre, los apellidos, el correo y la contraseña.	56
6.7.	Boceto de la pantalla principal de la aplicación donde se muestran las lecciones disponibles para el usuario.	57
6.8.	Boceto de la pantalla del perfil de usuario con la sesión iniciada donde se muestran sus datos.	57
6.9.	Boceto de la pantalla de logros conseguidos por el usuario. . .	58
6.10.	Boceto de la pantalla de una lección, con el texto, el contenido multimedia y el botón para comenzar el test.	58
6.11.	Boceto de la pantalla de una pregunta de test de tipo selección única.	59
6.12.	Boceto de la pantalla de una pregunta de test de tipo selección multiple.	59
6.13.	Boceto de la pantalla de una pregunta de test de tipo escritura de texto.	60
6.14.	Boceto de la pantalla de una pregunta de test de tipo entrada por micrófono.	60
6.15.	Boceto de la pantalla de una pregunta de test de tipo entrada por micrófono.	61

Capítulo 1

Introducción

Durante toda la historia de la humanidad, la música ha sido siempre una parte muy importante de la vida de la mayoría de personas y, a día de hoy, es una pieza clave y fundamental en nuestra sociedad y tradiciones. Escuchamos canciones a todas horas: mientras andamos por la calle, cuando cocinamos, mientras hacemos ejercicio, conduciendo en el coche, etc. La música está presente en gran parte de nuestros hábitos y de nuestra cultura y es por esto que, además de ser una de las siete bellas artes, mucha gente se dedica a estudiarla y a aprender sobre esta.

Sin embargo, por desgracia, a lo largo de la historia el aprendizaje del lenguaje musical ha sido elitista y la mayoría de personas con pocos recursos económicos no ha podido acceder a gran parte de este conocimiento, ya que antiguamente solo las familias nobles y adineradas podían darse el lujo de disfrutar de las melodías clásicas. Puede parecer que no pero, incluso en la actualidad, la formación musical a veces puede ser vista como un privilegio reservado para unos pocos, pues solo quienes pueden permitirse ir a una escuela de música o conservatorio pueden adquirir esos conocimientos.

Por otro lado, pese a ser un aspecto fundamental en nuestras vidas, el aprendizaje de esta disciplina es a menudo visto como algo aburrido debido a clases desactualizadas, poco motivadoras y enfocadas únicamente a un género musical en específico: la música clásica.

Todo esto plantea las siguientes cuestiones, ¿es posible democratizar el aprendizaje musical? ¿Ha habido un cambio en la forma en la que se estudia y/o practica música con la aparición de los dispositivos móviles, de internet y de las aplicaciones software? ¿Es posible aprender música de forma autodidacta, divertida y gratuita?

1.1. Motivacion

El surgimiento de nuevas aplicaciones que ofrecen formas de aprendizaje lúdicas y divertidas en distintos ámbitos como idiomas, matemáticas, arte, geografía, etc, me ha hecho pensar en lo útiles que pueden llegar a ser estas para personas con pocos recursos que buscan una forma rápida, barata y sencilla de aprender sobre algún tema en específico. Además, el acceso a la música es algo fundamental que todas las personas deberían tener al alcance y siempre lo he creído así.

En cuanto al aprendizaje musical, al estar relacionado con la música clásica y tradicional, suele ser visto como algo aburrido, por lo que muchos no se llegan a interesar por esto o suelen abandonar su formación en mitad. Sin infravalorar las clases de música tradicionales, el mundo está cambiando y existe una sensación de que la docencia musical se ha quedado atrás y no se ha adaptado a las necesidades de los jóvenes ni de la sociedad actual. Habiendo pocas opciones que ofrezcan un aprendizaje de la música de forma divertida y lúdica, muchas personas pierden una oportunidad valiosa de llegar a un mundo muy valioso.

Por otro lado, mi experiencia estudiando en una escuela de música durante 6 años me ha hecho entender que el aprendizaje musical, pese a ser un proceso complejo y exigente, es gratificante y satisfactorio; y que cualquier persona, tenga o no recursos económicos, debería tener la oportunidad de poder aprender música y disfrutar de ella sin impedimentos. La música y la formación en esta debería ser accesible para cualquier persona.

Al unir el surgimiento de distintas aplicaciones software educativas que ofrecen una visión novedosa y entretenida del aprendizaje con mi experiencia en el conservatorio de música, se me ocurrió la idea de desarrollar un sistema de aprendizaje musical que permita a cualquier persona, independientemente de su nivel original de conocimientos, aprender conceptos básicos e intermedios sobre lenguaje musical, así como la posibilidad de instruirse para empezar a tocar instrumentos musicales con un enfoque didáctico, divertido y progresivo.

Todo esto y la falta de aplicaciones similares y de calidad en el mercado respecto al conocimiento musical, han convertido en una necesidad el desarrollo de este proyecto.

1.2. Objetivos

Se pretende desarrollar una aplicación software que permita al usuario aprender conceptos de lenguaje musical de forma autodidacta y sencilla, proporcionando un enfoque que anime a las personas a involucrarse en la adquisición de conocimientos y a entender la mayoría de conceptos que la envuelven, desde el lenguaje musical hasta la práctica de un instrumento.

También se desea un sistema que proporcione una metodología de enseñanza mediante gamificación; es decir, con contenidos lúdicos y divertidos y con un progreso organizado en logros y niveles que permita a los usuarios sentirse satisfechos con su progreso y motivados a seguir aprendiendo. Los usuarios podrán ver su progreso mediante un sistema de cuentas de usuario que permitirá también la posibilidad de compartir sus logros con otros usuarios.

Por último, otro de los objetivos será incorporar distintos tipos de ejercicios y actividades que faciliten este aprendizaje, como por ejemplo la inclusión de un sistema que detecte las notas musicales tocadas por el usuario o la incorporación de un teclado virtual simple y sencillo que permita al usuario practicar y aprender a tocarlo.

Capítulo 2

Estado del Arte

2.1. Software a desarrollar

Para tener una perspectiva más amplia y clara de lo que se pretende desarrollar, se ha realizado una búsqueda de software similar al de este proyecto, tanto de aprendizaje musical como de otros temas que utilicen las técnicas de gamificación y de aprendizaje lúdico.

2.1.1. Aplicaciones de aprendizaje

En la actualidad existen numerosas aplicaciones en el mercado que ayudan al aprendizaje de conceptos y a la adquisición de conocimientos sobre distintos temas, como los idiomas, las matemáticas o el arte, entre otros. Es tan grande el auge de este tipo de herramientas que muchas de ellas están siendo recomendadas por el colectivo docente como apoyo a los contenidos que se dan en clase. A continuación se detallarán las aplicaciones más destacadas encontradas:

Duolingo

Duolingo es una de las aplicaciones más populares que existen para aprender idiomas. Esta aplicación se basa en el método de aprendizaje por inmersión y en la gamificación: el usuario estudia el idioma mediante una serie de actividades lúdicas. Aunque su principal motivación es la enseñanza de hablar y escribir en un idioma, también se incluyen actividades de comprensión auditiva, de lectura y de vocabulario.



Figura 2.1: Logo de Duolingo

La aplicación se divide en lecciones, las cuales contienen una serie de ejercicios (Figura 2.2) que el usuario debe completar para ir avanzando por las distintas secciones.

Algunos de los ejercicios que presenta la aplicación son:

- Traducción de palabras o frases con el teclado
- Traducción seleccionando bloques de palabras
- Pronunciación de palabras o frases mediante el micrófono del dispositivo
- Selección de imagen para el vocabulario

Además, Duolingo ofrece en cada lección, antes de los ejercicios, una breve explicación de la gramática o vocabulario con imágenes y definiciones sencillas, que ayudarán al usuario a entender el temario antes de comenzar la evaluación de la lección. En cuanto al diseño, la herramienta ofrece una interfaz sencilla y fácil de usar, con botones intuitivos y coloridos que mejoran la experiencia de usuario, algo primordial en este tipo de aplicaciones. Por último, cabe destacar que esta aplicación es gratuita (aunque ofrece funcionalidad adicional de pago dentro de esta) y está disponible para dispositivos móviles (Android, iOS y Windows Phone).

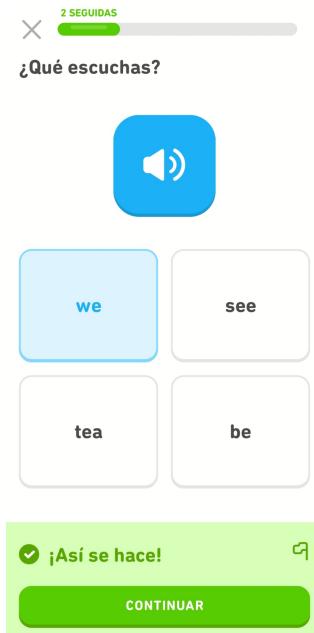


Figura 2.2: Ejercicio de Duolingo donde el usuario debe adivinar qué palabra corresponde al audio

Artly

Artly es una aplicación móvil que permite aprender cientos de obras de arte de forma interactiva. De forma parecida a Duolingo, cuando el usuario abre la aplicación, se encuentra una lista de secciones (Figura 3.2) correspondientes a los distintos movimientos artísticos. Estos apartados se irán desbloqueando uno a uno a medida que el usuario vaya completando las lecciones y ejercicios.

En cada uno de los apartados el usuario podrá ver una lista de cuadros y esculturas de los distintos artistas que pertenecen a ese movimiento. Al seleccionar uno de ellos, se abrirá una pantalla con la imagen del cuadro o escultura, en la que el usuario podrá ver la obra más de cerca, junto con el título, el autor y una descripción. Artly incorpora además una serie de preguntas sobre las obras vistas en dicho apartado (seleccionar el autor, seleccionar el título de la obra...). Al finalizar las preguntas, el usuario sumará un progreso en el tema que permitirá desbloquear otros apartados y conseguir ciertos logros en su perfil. La aplicación es gratuita pero tiene mejoras de pago que te permitirán avanzar con más facilidad y sin anuncios.



Figura 2.3: Menú de selección de las lecciones de Artly clasificadas en movimientos artísticos

BMath

BMath es un software para móviles para el aprendizaje de conceptos matemáticos mediante gamificación. Esta aplicación crea un programa personalizado para el usuario en función del curso académico en el que esté, adaptándose a su conocimiento y a su nivel. Pese a ser gratuita, el tiempo diario que te permite el modo básico es de 5 minutos, mientras que si pagas te permitirá ampliarlo en 10 minutos cada día.

Es un ejemplo de aprendizaje por gamificación ya que la aplicación es prácticamente un videojuego (de hecho, se consideran así), pues muestra una ciudad (Figura 2.4) por la que puedes mover la cámara y donde construyes distintos edificios. Para construirlos, deberás superar una serie de pruebas sobre álgebra, geometría, etc. También obtendrás puntos de experiencia al superar los ejercicios (Figura 2.5) y, al subir de nivel, recibirás nuevos edificios y decoraciones para tu ciudad. Con esto, se pretende que el usuario tenga la motivación de realizar ejercicios y superarlos correctamente para lograr que su ciudad prospere. Como alumno también puedes elegir tu propio personaje animado que te identificará en el juego.

Los diseños del software están muy cuidados y posee colores muy llamativos para que el usuario se sienta motivado y atraido. Además, la aplicación proporciona una experiencia de usuario agradable y motivadora, ayudando al alumno en todo momento a solucionar los ejercicios y enseñándole técnicas para resolverlos.

La aplicación también posee otras funciones como un apartado donde añadir recordatorios para acceder a la aplicación, preguntas para conocer cómo se siente el usuario tras cada sesión de estudio, sección parental, etc.



Figura 2.4: Ciudad de BMath donde se seleccionan los niveles y se construyen los edificios para mostrar el progreso del usuario

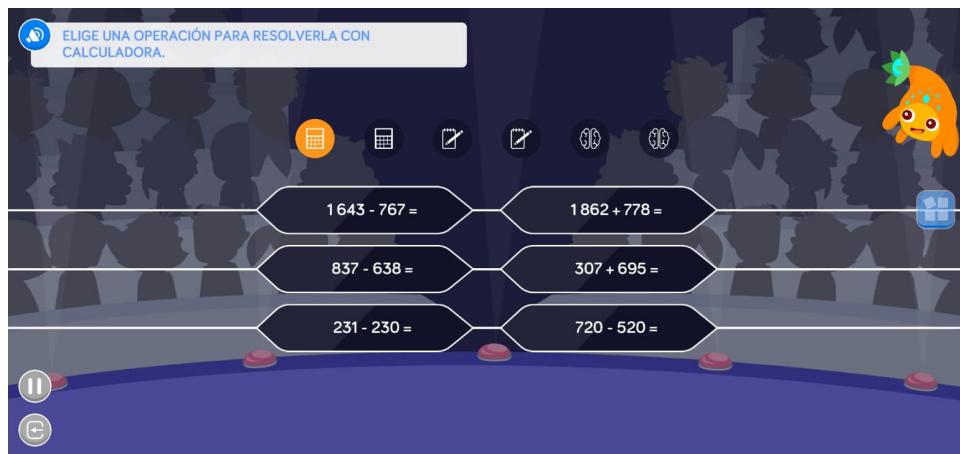


Figura 2.5: Ejercicio de BMATH sobre cálculo matemático (sumas y restas) con calculadora, con papel y mentalmente

2.1.2. Aplicaciones de aprendizaje musical

Centrándonos ya en el tema que nos ocupa (el aprendizaje musical) existen numerosas aplicaciones que ayudan a la adquisición de conocimientos musicales o a la ayuda a aprender a tocar un instrumento musical. A continuación se muestran algunas de ellas.

Curso de Lenguaje Musical

Esta herramienta para móvil tiene el objetivo de enseñar los conceptos básicos de lenguaje musical. La aplicación proporciona una forma de aprendizaje muy tradicional y que no está basada en la gamificación, ya que no dispone de ningún tipo de seguimiento del progreso ni de ejercicios para realizar sobre las lecciones aprendidas. La app contiene una lista de lecciones (Figura 2.6) y cada una de ellas muestra un vídeo de un docente explicando el contenido del temario.

Por último, en cuanto a la interfaz, la aplicación es muy sencilla y tiene un diseño poco atractivo, dando una simple lista con enlaces a los vídeos de las lecciones sin ningún tipo de decoración o imagen.



Figura 2.6: Lista de lecciones que ofrece la aplicación Curso de Lenguaje Musical

Sonid

Sonid consiste en una aplicación para el aprendizaje de piano a través de la gamificación. La aplicación se divide en distintas lecciones (Figura 2.7b) que el usuario irá desbloqueando al pasarlas correctamente. Cada lección explica unos conceptos básicos de música y piano y, al finalizar, el usuario deberá realizar una serie de ejercicios para comprobar que ha entendido y aprendido el contenido del temario.

Algunos de los ejercicios que posee esta aplicación son:

- Seleccionar la tecla en el piano que corresponde a una nota musical. (Figura 2.7a)
- Responder si la tecla que se ha pulsado es una nota musical o no.
- Decidir qué nota musical corresponde a una tecla del piano que se ha pulsado.

Además de las lecciones, el usuario también puede practicar con escalas y acordes del piano, así como aprenderlas en la wiki y en el diccionario que posee la aplicación. También dispone de un foro para que toda la gente pueda compartir sus dudas.

Por último, en cuanto al seguimiento, el usuario puede ver su progreso y sus estadísticas en su perfil, donde aparece el número de lecciones completadas, el número de errores que ha tenido, la experiencia que tiene. Además, el usuario consigue logros al completar las lecciones y comparte una clasificación global con otros usuarios que también usan la aplicación.

(a) Ejercicio de Sonid para conocer las notas de un piano

(b) Menú de selección de lecciones de Sonid dividido por escalas

2.2. Desarrollo de Software

En cuanto a las tecnologías y herramientas que existen para el desarrollo de aplicaciones, existen numerosas opciones que se pueden utilizar. En este apartado se muestran algunas de ellas.

2.2.1. Flutter

Flutter es un kit de desarrollo software (framework) de código abierto destinado al desarrollo de aplicaciones móviles multiplataforma. Está programado en Dart, fue creado por Google y su primera versión estable se lanzó en 2018. Se emplea para desarrollar aplicaciones para móvil, web y escritorio desde una sola base de código, lo cual permite mucha más agilidad y consistencia.

Flutter ofrece tres ventajas respecto a otros frameworks utilizados para el desarrollo de aplicaciones multiplataformas:

- **Compilación en nativo**
- **Flexibilidad** para crear interfaces gráficas
- **Desarrollo veloz**, permitiendo ver el resultado del código al instante.

Flutter está basado en el concepto de widgets, que son los elementos que componen la interfaz y la interacción del usuario. Estos widgets pueden ser modificados, añadidos o eliminados dinámicamente, permitiendo una gran flexibilidad a la hora de crear interfaces gráficas.

2.2.2. React Native

Framework de código abierto creado por Meta Platforms (Facebook) y destinado al desarrollo de aplicaciones multiplataforma para Android, iOS, Web, Windows, etc. Está basado en Javascript como lenguaje de programación y en React como herramienta para la interfaz de usuario. La diferencia con React es que, en lugar de trabajar con el navegador, trabaja con las plataformas móviles al transformar los componentes en nativos en función de la plataforma en la que se esté ejecutando la aplicación.

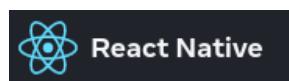


Figura 2.7: Logo de Flutter

Esta tecnología permite construir aplicaciones móviles usando solamente Javascript, con el mismo diseño que utiliza React permitiéndote realizar una interfaz de usuario completa mediante componentes.

Podemos encontrar dos ventajas en este framework:

- **Código compartido:** tu código puede ser compartido para diferentes plataformas.
- **Comunidad:** Existe una amplia comunidad sobre React y React Native, lo cual permite encontrar soluciones a los problemas que se puedan presentar en el desarrollo.

Como desventaja, encontramos una limitación en los componentes nativos, pues seguramente se necesite escribir algo de código específico para un plataforma en concreto

2.2.3. Dart

Dart es un lenguaje de programación de código abierto desarrollado por Google y utilizado por Flutter. Es orientado a objetos y de tipado estático. Surgió en 2011 como alternativa a Javascript.



Figura 2.9: Logo de Dart

Dart posee las siguientes ventajas:

- **Lenguaje fácil y sencillo** de aprender
- **Acceso gratuito**
- **Funciona en todas las plataformas**
- **Programación estructurada y flexible**
- Al ser **orientado a objetos**, facilita la encapsulación y reutilización de código gracias al uso de clases.

2.2.4. Kotlin

Kotlin es un lenguaje de programación construido sobre Java, desarrollado por JetBrains y lanzado en 2016. Es orientado a objetos y tiene tipado estático, pero también soporta la programación por procedimientos y funciones.

Algunas de sus ventajas son:

- **Compatibilidad con Java:** permite utilizar código Java en Kotlin y viceversa
- **Lenguaje fácil y sencillo** de aprender y usar por su sintaxis similar a Java
- **Robusto:** Es un lenguaje seguro con los valores nulos
- **Exactitud y claridad:** reduce el código repetido de forma sustancial y disminuye la probabilidad de error.

2.2.5. NodeJS

NodeJS es un entorno en tiempo de ejecución de código abierto, del lado del servidor y programado en Javascript. Es asíncrono y dirigido por eventos. Node es utilizado para el desarrollo de aplicaciones de red escalables y rápidas, ofreciendo beneficios en rendimiento, velocidad de desarrollo, etc.

Se utiliza en el desarrollo de aplicaciones web porque permite ejecutar Javascript del lado del servidor en lugar de en el navegador.

Los principales motivos para utilizar NodeJS son:

- **Simultaneidad de peticiones:** NodeJS es capaz de manejar múltiples peticiones al mismo tiempo gracias al modelo dirigido por eventos, lo que permite una mayor escalabilidad.
- **Lenguaje sencillo** basado en Javascript.
- **Gestión de paquetes de calidad** gracias a NPM.

2.2.6. React

React es una biblioteca Javascript de código abierto que se utiliza para la creación de interfaces de usuario de forma fácil y sencilla. Es mantenida por una gran comunidad de desarrolladores de software libre y fue lanzada en 2013.

En React se trabaja con componentes, los cuales son elementos o partes de la interfaz de usuario que facilitarán la reutilización de código y la modularidad.

Entre las ventajas de React encontramos:

- **Componentes reutilizables:** React permite crear componentes que pueden ser reutilizados en otras partes de la aplicación.
- **Fácil de aprender:** React es fácil de aprender y de utilizar, además de tener una documentación sólida y una gran cantidad de recursos online gratuitos.
- **Rendimiento:** React es rápido y eficiente gracias al DOM virtual.

2.2.7. Angular

Angular es un framework de código abierto para el desarrollo de aplicaciones web y móviles. Está basado en TypeScript y está desarrollado por Google. Esta tecnología también está basada en componentes, lo cual permite la creación de aplicaciones web escalables

2.2.8. MongoDB

MongoDB es una base de datos no relacional de código abierto y orientada a documentos. Es de tipo NoSQL, es decir, que no utiliza el modelo relacional de bases de datos tradicionales. Cada registro de la base de datos es un documento que consta de pares clave - valor (muy similar a los objetos JSON). Está escrito en C++ y las consultas se hacen pasando objetos JSON como parámetros.

MongoDB proporciona las siguientes ventajas:

- **Rendimiento:** MongoDB es más rápido porque almacena los datos directamente en un mismo sitio (colección).
- **Simplicidad:** No hay que seguir un esquema ni unir tablas como en SQL.
- **Flexibilidad** para diseñar el esquema y crear las relaciones.



Figura 2.10: Logo de MongoDB

2.2.9. Pilas MERN & MEAN

MERN es un conjunto de tecnologías que se utilizan para el desarrollo de aplicaciones web. Está compuesto por las siglas de MongoDB, Express (ayuda para crear y gestionar el backend en Node), React y NodeJS. MEAN es igual, pero sustituye React por Angular.

Hay una gran cantidad de beneficios al utilizar una de estas pilas para desarrollar una aplicación web como por ejemplo:

- **Cubre todo el ciclo de desarrollo:** Desde el backend hasta el frontend.
- **Facilita el trabajo** con la arquitectura modelo-vista-controlador.
- **Rápido desarrollo:** El desarrollo de una aplicación web con estas tecnologías es rápido y eficiente.
- **Código abierto:** Todas las tecnologías que componen estas pilas son de código abierto y gratuitas y están respaldadas por la comunidad.
- **Fácil mantenimiento:** El mantenimiento de una aplicación web con estas tecnologías es fácil y sencillo.



Figura 2.11: Pilas MEAN y MERN

2.2.10. MariaDB

MariaDB es un sistema de gestión de bases de datos relacional de código abierto. Es un fork de MySQL y está desarrollado por su comunidad de usuarios. Surgió en 2009 como consecuencia de la compra de MySQL por parte de Sun Microsystems. Sigue un modelo relacional, lo cual significa que el contenido de los datos se almacena en tablas con un esquema más rígido que en MongoDB u otra base de datos noSQL.



Figura 2.12: Logo de MariaDB

Pese a haber mencionado solamente MariaDB como ejemplo de sistema de gestión de bases de datos relacional, existen muchos otros como por ejemplo PostgreSQL, MySQL, etc.

Capítulo 3

Especificación de requisitos

3.1. Introducción

En esta sección se describirá qué sistema vamos a construir y cómo lo vamos a hacer, con sus restricciones específicas. Esto se realizará especificando y describiendo los requisitos del sistema que vamos a desarrollar.

3.1.1. Propósito

En este capítulo se pretende describir de forma clara y precisa las funciones, características y restricciones del sistema que se va a desarrollar. Estas definiciones servirán al equipo de desarrollo para conocer las necesidades del sistema y a los usuarios finales. Además, este capítulo será consultado como base para el desarrollo de las funcionalidades descritas en la sección de implementación.

3.1.2. Ámbito del Sistema

El sistema que vamos a desarrollar es una aplicación multiplataforma llamada "Meloudy" que permitirá a los usuarios aprender conceptos musicales de una forma amena, fácil y divertida. Esto se logrará mediante el método de gamificación, utilizando un sistema de logros y de recompensas con la superación de los distintos ejercicios y actividades de distintos tipos que el usuario deberá completar. Además, la aplicación llevará el progreso de los usuarios que les permitirá saber cómo van en su aprendizaje.

3.1.3. Definiciones, Acrónimos y Abreviaturas

A continuación se detallará el significado de algunos conceptos importantes para la comprensión del capítulo y de nuestro sistema.

- **Requisito:** Es una condición o característica que debe cumplir el sistema para satisfacer una necesidad o cumplir una función.
- **Funcionalidad:** Descripción de lo que debe hacer el producto software.
- **Restricción:** Condición que limita la funcionalidad del sistema.
- **Interfaz de usuario:** Requisitos que describen los diseños de pantallas que utilizará el usuario para utilizar el software.
- **Usuario:** Persona que utilizará la aplicación para la intención final de esta.
- **Administrador:** Persona encargada del sistema software y del mantenimiento de este para su buen funcionamiento.
- **RF / RNF:** Requisito funcional / Requisito No Funcional.

3.1.4. Referencias

Este capítulo de Especificación de requisitos ha sido redactada consultando los documentos del estándar IEEE Recommended Practice for Software Requirements Specification ANSI/IEEE 830, 1998.

3.1.5. Visión General del Documento

La Especificación de Requisitos consta de tres partes bien diferenciadas:

- **Introducción:** Proporciona una visión general sobre el apartado de Especificación de Requisitos sin profundizar en los requisitos como tal.
- **Descripción General:** Se describirá el sistema a construir para saber las funciones principales, los datos necesarios, las restricciones y otros aspectos que puedan afectar al desarrollo de la aplicación.
- **Requisitos Específicos:** Se profundiza en las necesidades del usuario definiendo los requisitos que debe tener nuestro sistema tras el desarrollo y la implementación de este.

3.2. Descripción General

A continuación, se procederá a describir la visión del sistema y los factores que afectan al producto de una forma general.

3.2.1. Perspectiva del producto

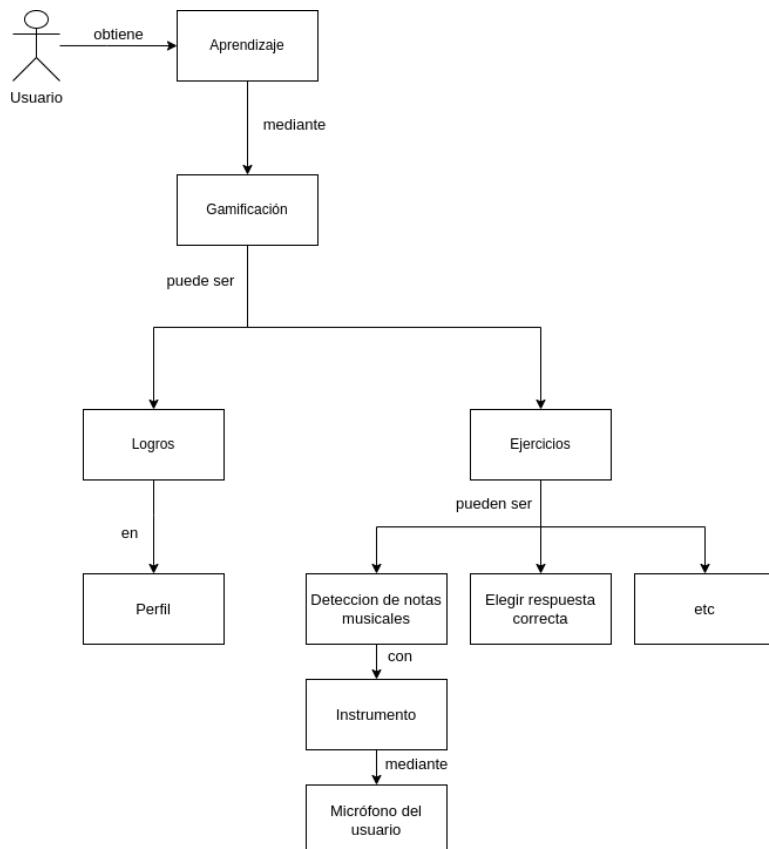


Figura 3.1: Diagrama general de la perspectiva del producto a desarrollar.

Se pretende implementar un sistema que permita el aprendizaje del usuario mediante técnicas de gamificación y ejercicios a resolver. Algunas de estas actividades utilizarán librerías para la detección de notas musicales a partir de la frecuencia captada por el micrófono del usuario y, por tanto, la aplicación dependerá de dichas librerías que se incluyan. Por otro lado, el sistema de administración y el progreso de los usuarios se llevará a cabo utilizando una base de datos en la que se almacenará la información necesaria de los usuarios y su progreso (con un sistema de logros asociados al perfil).

La interacción de los usuarios con la aplicación será mediante una interfaz

gráfica que podrá ser utilizada con la pantalla táctil o el ratón del dispositivo usado.

3.2.2. Funciones del producto

Las principales funciones que el usuario podrá realizar dentro de la aplicación son:

- Selección de lecciones a aprender.
- Respuesta (mediante selección o escritura) de las preguntas y actividades que ofrezca la aplicación.
- Consulta del progreso individual, de los logros obtenidos y de la clasificación global.
- Modificación de los datos del perfil.

Además, el encargado/profesor se encargará de parte de la administración de las lecciones mediante:

- Modificación y creación del texto de cada lección
- Gestión de contenido multimedia de cada lección (modificación, creación y eliminación)
- Gestión de las preguntas y actividades (modificación, creación y eliminación)
- Creación de nuevas lecciones
- Eliminación de lecciones

Por último, el administrador, además de las funcionalidades del encargado/profesor, podrá realizar las siguientes funcionalidades

- Creación de nuevos usuarios
- Modificación de los datos de los usuarios
- Borrado de usuarios
- Creación de nuevas lecciones
- Borrado de lecciones

3.2.3. Características de los usuarios

Los usuarios que usarán la aplicación tendrán distintos perfiles y abarcarán edades muy distintas. Aún así, el perfil objetivo para el uso del sistema será las personas jóvenes, pues suelen utilizar mucho más las nuevas tecnologías y estarán más familiarizados con este tipo de aplicaciones. No se necesita conocimiento musical previo para utilizar el software y, de hecho, las lecciones pueden ser bastante básicas y sencillas para las personas que ya conozcan conceptos y aspectos avanzados del lenguaje musical.

3.2.4. Restricciones

Las restricciones del sistema a desarrollar son:

- La aplicación del cliente estará desarrollada en Flutter y/o Dart y para la del servidor se utilizará NodeJS con Express.
- Se utilizarán las consultas, inserciones, modificaciones y borrados de MongoDB para la base de datos, usando un modelo no relacional.
- Al hacer uso de librerías externas para desarrollar algunas de las funciones, el rendimiento y las limitaciones hardware pueden depender de estas.

Además, es posible que el sistema tenga que adaptarse a los cambios del entorno y necesite agregar con el paso del tiempo funcionalidades que añadan valor al producto.

3.2.5. Suposiciones y dependencias

Los requisitos especificados en este capítulo pueden estar sujetos a cambios por motivos técnicos, de alcance o de refinamiento por la necesidad de la adaptación al entorno y a la evolución continua del proyecto.

No habrá ninguna suposición respecto al sistema operativo a utilizar puesto que pretendemos desarrollar una aplicación multiplataforma, aunque se priorizará el buen funcionamiento en Android y en navegadores web.

Asumiremos también que las bibliotecas a utilizar para el desarrollo de ciertos requisitos funcionan correctamente y son estables.

3.3. Requisitos Específicos

A continuación se listarán todos los requisitos que debe cumplir el sistema a desarrollar. Se dividirán en distintos tipos en función de sus objetivos.

3.3.1. Interfaz de usuario

Se tratará que la interfaz de usuario siga un diseño minimalista y sencillo para facilitar el uso de la aplicación a todo tipo de personas y garantizar cierto grado de accesibilidad. En cuanto a los colores, se usarán tonalidades de azul junto con grises. En móviles la pantalla estará dividida en dos partes principales:

- La cabecera con el título de la aplicación, un menú estilo hamburguesa que se abrirá lateralmente y un ícono para el acceso al perfil del usuario.
- El cuerpo, donde se encuentra el contenido principal de la sección correspondiente.

A continuación se presenta un primer boceto de lo que podría ser la interfaz de usuario para móviles.

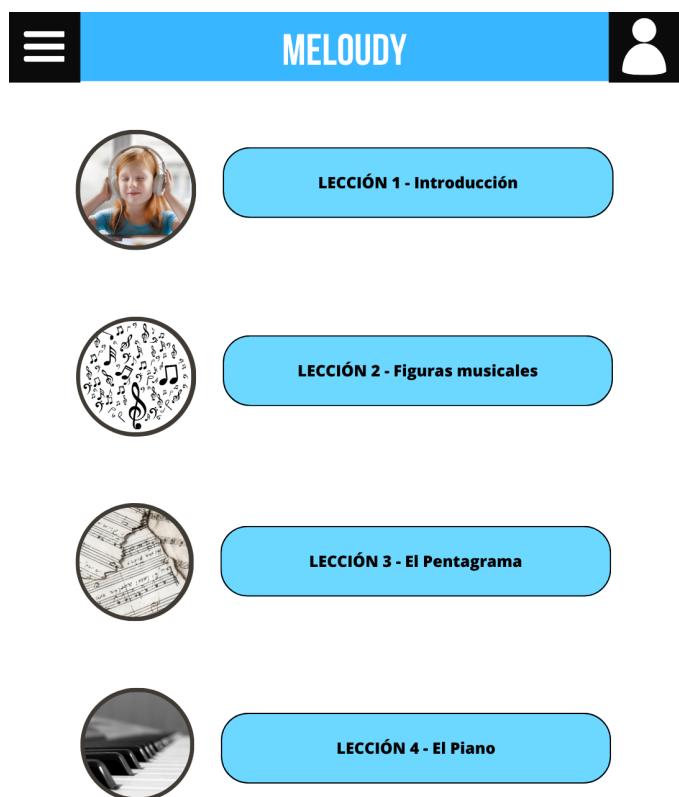


Figura 3.2: Primer boceto de la interfaz de usuario de la página de inicio de la aplicación para los usuarios, donde se muestra la lista de las lecciones

3.3.2. Requisitos funcionales

En esta sección se presentan los requisitos relacionados con las funcionalidades del sistema y su comportamiento, sus servicios y las tareas que este debe realizar.

▪ General

- **RF 1 - Registro de usuarios:** Los usuarios deberán registrarse en el sistema para utilizar la funcionalidad de la aplicación.
- **RF 2 - Identificar usuarios:** Los usuarios deberán identificarse en el sistema con su cuenta para poder utilizar la funcionalidad de la aplicación.

▪ Usuarios

- **RF 3 - Consultar datos de la cuenta:** Los usuarios podrán consultar los datos de su perfil.
- **RF 4 - Modificar datos de la cuenta:** Los usuarios podrán modificar los datos de su perfil (imagen de perfil, nombre, contraseña...).
- **RF 5 - Selección de lección:** Los usuarios podrán seleccionar la lección a estudiar de la lista de lecciones de la pantalla principal.
- **RF 6 - Responder preguntas:** Los usuarios podrán responder las preguntas de cada tipo en la lección que estén.
 - **RF 6.1 - Responder con texto:** Los usuarios podrán responder preguntas que necesiten insertar texto como respuesta.
 - **RF 6.2 - Responder seleccionando varias opciones:** Los usuarios podrán responder preguntas que necesiten seleccionar varias opciones de un conjunto.
 - **RF 6.3 - Responder seleccionando una opción:** Los usuarios podrán responder preguntas que necesiten seleccionar una opción de un conjunto.
 - **RF 6.4 - Responder tocando una nota:** Los usuarios podrán responder preguntas que necesiten tocar una nota de un instrumento y captarla por el micrófono.
- **RF7 - Consultar ranking:** Los usuarios podrán consultar la clasificación global de los usuarios.
- **RF8 - Consultar progreso de perfil:** Los usuarios podrán consultar su progreso y sus logros obtenidos en su perfil.
- **RF9 - Consultar perfiles ajenos:** Los usuarios podrán visualizar el perfil de otros usuarios y ver sus logros.

- **RF22 - Consultar progreso de lección:** Los usuarios podrán consultar el progreso de una lección en concreto.

- **Profesores/Encargados**

- **RF 10 - Obtener lista de lecciones:** Los profesores podrán obtener una lista de todas las lecciones que existen en el sistema.
- **RF 11 - Crear lección:** Los profesores podrán crear nuevas lecciones.
- **RF 12 - Eliminar lección:** Los profesores podrán eliminar lecciones ya existentes.
- **RF 13 - Modificar lecciones:** Los profesores podrán modificar el texto y el contenido multimedia de las distintas lecciones.
- **RF 14 - Crear preguntas:** Los profesores podrán crear nuevas preguntas sobre las distintas lecciones.
- **RF 15 - Obtener lista de preguntas:** Los profesores podrán obtener una lista de todas las preguntas que existen en el sistema.
- **RF 16 - Modificar preguntas:** Los profesores podrán modificar preguntas ya creadas.
- **RF 17 - Eliminar preguntas:** Los profesores podrán eliminar preguntas del sistema.

- **Administradores**

- **RF 18 - Modificar datos de usuarios:** Los administradores podrán modificar los datos de un usuario.
- **RF 19 - Crear usuario:** Los administradores podrán crear usuarios en el sistema.
- **RF 20 - Obtener lista de usuarios:** Los administradores podrán obtener una lista con todos los usuarios registrados en el sistema.
- **RF 21 - Eliminar usuario:** Los administradores podrán eliminar usuarios registrados en el sistema.

3.3.3. Requisitos no funcionales

Estos requisitos se refieren a aquellas condiciones que debe cumplir el sistema en cuanto a rendimiento, accesibilidad, seguridad, robustez...

- **RNF 1 -** La aplicación cifrará las claves en todo momento para garantizar la seguridad de los usuarios.

- **RNF 2** - El número de usuarios simultáneos solo estará limitado por la capacidad del hardware del servidor.
- **RNF 3** - La cantidad máxima de datos a almacenar solo estará limitada por la capacidad del hardware del servidor de la base de datos.
- **RNF 4** - La aplicación será multiplataforma, pudiéndose ejecutar en dispositivos (móvil, ordenador, tablet...) diferentes con distintos sistemas operativos (Windows, Android...)
- **RNF 5** - La aplicación será lo más accesible posible para personas con distintas discapacidades.
- **RNF 6** - La aplicación será tolerante a fallos (será fiable).
- **RNF 7** - Se garantizará el cumplimiento de la Ley de Protección de Datos de Carácter Personal.

Capítulo 4

Planificación

4.1. Introducción

En este capítulo se describirá la planificación para el desarrollo del proyecto, la cual va a servir para poder realizar un control de los avances del producto y asegurar que se cumplan los objetivos marcados en cada sprint/iteración. Para ello, se utilizarán ciertos elementos de la metodología SCRUM, el cual definiremos brevemente a continuación:

SCRUM

SCRUM es un marco de trabajo iterativo e incremental que se utiliza para desarrollar proyectos de software. Este marco de trabajo se basa en la idea de que el desarrollo de software es un proceso complejo que se adapta continuamente a las circunstancias. Por ello, se divide en pequeñas iteraciones que se van realizando de forma incremental y utiliza una serie de elementos para poder controlar el desarrollo del proyecto:

- **Artefactos:** Son documentos que se utilizan para controlar el desarrollo del proyecto.
- **Reuniones:** Se realizan cuatro tipos de reuniones de forma periódica para poder controlar y seguir el progreso del proyecto.
- **Roles:** Representan la responsabilidad de cada persona en el proceso.

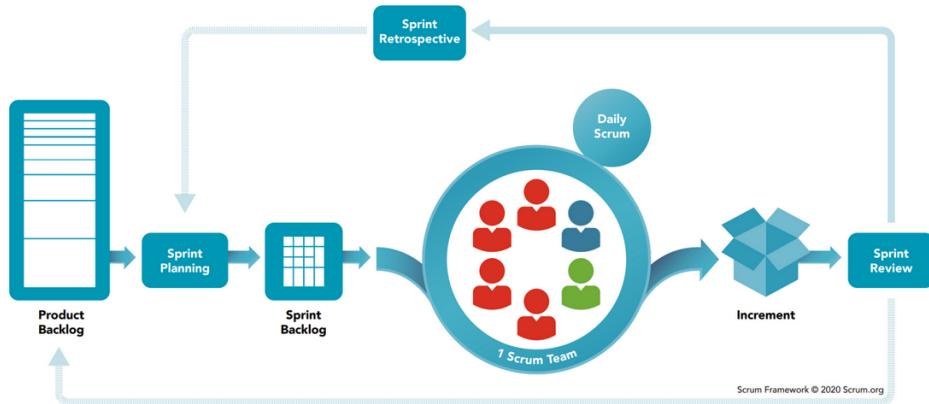


Figura 4.1: Diagrama de la arquitectura del sistema, donde se muestra la comunicación entre el servidor y la aplicación móvil.

Es importante recalcar que no voy a seguir la metodología SCRUM, sino que solo usaré algunos principios y conceptos de metodologías ágiles debido a las condiciones en las que se va a desarrollar el proyecto (no hay un cliente real, no hay un equipo de desarrollo real, etc.). Por tanto, se utilizarán los siguientes aspectos de SCRUM:

- **Sprints:** Se definirán los Sprints (iteraciones) que se realizarán para el desarrollo del proyecto, explicando el objetivo de cada uno de ellos y detallando la duración y las fechas de inicio y de fin.
- **Historias de Usuario:** Se listarán y describirán las historias de usuario que se van a implementar en cada sprint a partir de los requisitos funcionales del proyecto especificados en el capítulo de Especificación de Requisitos.
- **Product Backlog:** Se definirá el product backlog, que es una lista de todas las funcionalidades (Historias de usuario) que se pretenden implementar en el proyecto. Esta lista se ordenará por prioridad de acuerdo a la importancia que tienen para el usuario final.
- **Velocidad:** Se calculará la velocidad de trabajo que se tendrá en el desarrollo del proyecto a partir de la cantidad de horas que se pretenden dedicar al proyecto cada día y la duración de los Sprints.
- **Puntos de historia:** Se asignarán los puntos de historia a cada historia de usuario para poder estimar la cantidad de trabajo que se debería realizar en cada sprint. Los valores de estos puntos de historia se calcularán a partir de la estimación de la complejidad de cada historia de usuario.

- **Reuniones:** Pese a no encajar exactamente en ninguna de las reuniones que hay en SCRUM, se realizarán reuniones cada dos semanas para poder controlar el progreso del proyecto y realizar un seguimiento de los avances. En estas reuniones se tratará:
 - **Qué se ha hecho:** Se comentará el trabajo realizado desde la anterior reunión para comentar las posibles mejoras.
 - **Qué falta por hacer:** Se comparará la planificación realizada para el Sprint con el trabajo realizado, comprobando qué cosas han podido faltar.
 - **Qué se piensa hacer:** Se volverá a planificar si es necesario y se determinará qué trabajo se planea realizar durante el nuevo Sprint.

4.2. Velocidad

A continuación vamos a estimar la velocidad inicial de trabajo que se tendrá en el desarrollo. Esta será una aproximación que nos ayudará a estimar la cantidad de trabajo que se debería realizar en cada iteración. Sin embargo, esta podrá variar y ajustarse a lo largo del proyecto en función de la cantidad de trabajo que se logre completar en cada sprint.

Para calcular la velocidad, vamos a considerar las horas de trabajo que se pretenden dedicar al proyecto cada día y tratar de estimar cuanta cantidad de trabajo se podría realizar.

- Partimos de un “equipo de desarrollo” de 1 miembro.
- Se pretende dedicar 5 horas al día de trabajo aproximadamente.
- Cada Sprint dura 2 semanas (14 días). Si cada día trabajamos 5 horas de forma aproximada, obtenemos un total de 70 horas en cada Sprint.
- En mi entorno, se estima que 1 PH son dos días de trabajo ideal (es decir, 10 horas). Esto significa que cada dos jornadas de trabajo real, se debería completar 1 PH. Sin embargo, en la realidad, esto no siempre es así ya que estamos realizando una aproximación.
- Si multiplicamos un programador por las 70 horas de trabajo y lo dividimos por el número de horas de trabajo por cada punto de historia (10 horas), obtenemos que deberíamos completar **7 PH** por sprint aproximadamente.

Nota: La duración de los Sprints puede variar en función de factores externos al proyecto. Pero se tratará de que duren 2 semanas para seguir un ritmo de trabajo constante.

Nota 2: Puesto que estamos realizando estimaciones y suposiciones para la planificación, esta está sujeta a cambios a lo largo del proyecto.

4.3. Product Backlog

A continuación, se listarán todas las historias de usuario ordenadas por prioridad de acuerdo a la importancia que tienen para el usuario final del producto.

- **HU1** - Como usuario quiero registrarme en la aplicación para poder comenzar a utilizar sus funcionalidades. (Prioridad: Alta — Puntos de Historia: 2)
- **HU2** - Como usuario quiero iniciar sesión en la aplicación para poder acceder a sus funcionalidades. (Prioridad: Alta — Puntos de Historia: 2)
- **HU6** - Como usuario quiero seleccionar una lección para leer el temario. (Prioridad: Alta — Puntos de Historia: 1)
- **HU7** - Como usuario quiero empezar un test de una lección. (Prioridad: Alta — Puntos de Historia: 1)
- **HU26** - Como usuario quiero responder una pregunta de un test del tipo escritura de texto. (Prioridad: Alta — Puntos de Historia: 1)
- **HU8** - Como usuario quiero responder una pregunta de un test del tipo selección múltiple. (Prioridad: Alta — Puntos de Historia: 1)
- **HU9** - Como usuario quiero responder una pregunta de un test del tipo selección única. (Prioridad: Alta — Puntos de Historia: 1)
- **HU10** - Como usuario quiero responder una pregunta de un test del tipo respuesta por micrófono. (Prioridad: Alta — Puntos de Historia: 8)
- **HU29** - Como profesor quiero ver una lista de todas las preguntas. (Prioridad: Alta — Puntos de Historia: 0.5)
- **HU11** - Como usuario quiero ver el resultado de un test. (Prioridad: Media — Puntos de Historia: 2)

- **HU13** - Como usuario quiero ver mi progreso en las lecciones. (Prioridad: Media — Puntos de Historia: 2)
- **HU28** - Como profesor quiero ver una lista de todas las lecciones. (Prioridad: Media — Puntos de Historia: 0.5)
- **HU14** - Como profesor quiero crear una lección. (Prioridad: Media — Puntos de Historia: 1)
- **HU15** - Como profesor quiero modificar el texto de una lección. (Prioridad: Media — Puntos de Historia: 2)
- **HU16** - Como profesor quiero añadir contenido multimedia a una lección. (Prioridad: Media — Puntos de Historia: 2)
- **HU17** - Como profesor quiero eliminar contenido multimedia de una lección. (Prioridad: Media — Puntos de Historia: 1)
- **HU27** - Como profesor quiero añadir preguntas a un test del tipo entrada de texto. (Prioridad: Media — Puntos de Historia: 1)
- **HU18** - Como profesor quiero añadir preguntas a un test del tipo selección múltiple. (Prioridad: Media — Puntos de Historia: 1)
- **HU19** - Como profesor quiero añadir preguntas a un test del tipo selección única. (Prioridad: Media — Puntos de Historia: 1)
- **HU20** - Como profesor quiero añadir preguntas a un test del tipo respuesta por micrófono. (Prioridad: Media — Puntos de Historia: 2)
- **HU21** - Como profesor quiero eliminar preguntas de un test. (Prioridad: Media — Puntos de Historia: 0.5)
- **HU22** - Como profesor quiero modificar preguntas de un test. (Prioridad: Media — Puntos de Historia: 1)
- **HU30** - Como administrador quiero ver una lista de todos los usuarios. (Prioridad: Media — Puntos de Historia: 0.5)
- **HU24** - Como administrador quiero modificar los datos de un usuario. (Prioridad: Media — Puntos de Historia: 2)
- **HU25** - Como administrador quiero eliminar un usuario. (Prioridad: Media — Puntos de Historia: 0.5)
- **HU3** - Como usuario quiero ver los datos personales y los logros de mi perfil. (Prioridad: Media — Puntos de Historia: 1)
- **HU4** - Como usuario quiero editar los datos de mi perfil. (Prioridad: Media — Puntos de Historia: 2)

- **HU23** - Como administrador quiero crear un usuario. (Prioridad: Baja — Puntos de Historia: 1)
- **HU12** - Como usuario quiero ver el ranking de usuarios. (Prioridad: Baja — Puntos de Historia: 1)
- **HU5** - Como usuario quiero ver el perfil de otros usuarios. (Prioridad: Baja — Puntos de Historia: 1)

4.4. Sprints

Nuestro proyecto se pretende desarrollar en 14 Sprints. Cada Sprint tendrá una duración de 2 semanas, y se pretenden realizar una media de 7 puntos de historia por Sprint. A continuación se especificará el contenido de cada Sprint con las fechas de inicio y fin de cada uno de ellos.

4.4.1. Sprint #1 - Documentación inicial

24/11/2022 - 08/12/2022

En este Sprint se realizará:

- Definición del proyecto y del alcance.
- Redactar Capítulo 1 - Introducción.
- Redactar Capítulo 2 - Estado del Arte.

4.4.2. Sprint #2 - Documentación de requisitos

08/12/2022 - 12/01/2023

En este Sprint se planea realizar:

- Revisión de la documentación inicial y corrección de errores.
- Redactar Capítulo 3 - Especificación de Requisitos.

4.4.3. Sprint #3 - Documentación de HU

12/01/2023 - 03/02/2023

En este Sprint se realizará:

- Revisión de la documentación de requisitos y corrección de errores.
- Redactar Capítulo 4 - Planificación.
- Redactar Capítulo 5 - Análisis del problema.

4.4.4. Sprint #4 - Diseño y comienzo del desarrollo

03/02/2023 - 17/02/2023

En este Sprint se terminará la documentación y comenzará el desarrollo con:

- Revisión de la planificación y el análisis y corrección de errores.
- Redactar Capítulo 6 - Diseño.
- Preparación del backend para el desarrollo.
 - Creación de la base de datos.
 - Creación de los modelos, rutas y controladores del servidor con NodeJS.
- Preparación del frontend para el desarrollo (creación de la aplicación de Flutter).

4.4.5. Sprint #5 - Registro y login de usuarios

17/02/2023 - 03/03/2023

En este Sprint se comenzará el desarrollo de la aplicación realizando las funcionalidades iniciales del usuario que utilizará la aplicación, tales como el registro, el login, la selección de lección, etc.

- HU1 - Como usuario quiero registrarme en la aplicación.
- HU2 - Como usuario quiero iniciar sesión en la aplicación.
- HU6 - Como usuario quiero seleccionar una lección para leer el temario.

- HU7 - Como usuario quiero empezar un test de una lección.
- HU29 - Como profesor quiero ver una lista de todas las preguntas.

4.4.6. Sprint #6 - Tests y progreso

03/03/2023 - 17/03/2023

Este Sprint tendrá como objetivo el desarrollo de los tests permitiendo al usuario responder a las preguntas y ver el resultado. Además, podrá comprobar su progreso en cada lección y en la pantalla principal que muestra todas las lecciones (desbloqueando las correspondientes en su caso).

- HU26 - Como usuario quiero responder una pregunta de un test del tipo escritura de texto.
- HU8 - Como usuario quiero responder una pregunta de un test del tipo selección múltiple.
- HU9 - Como usuario quiero responder una pregunta de un test del tipo selección única.
- HU11 - Como usuario quiero ver el resultado de un test.
- HU13 - Como usuario quiero ver mi progreso en las lecciones.

4.4.7. Sprint #7 - Entrada de micrófono y detección de tono

17/03/2023 - 31/03/2023

En este Sprint se realizará la funcionalidad de la entrada de micrófono y la detección de tono. Puesto que la historia de usuario es muy grande y compleja, se reservará un Sprint entero para su desarrollo. Para desarrollar esta funcionalidad, se utilizarán librerías de Flutter que permitan la entrada de audio y trabajar con el mismo.

- HU10 - Como usuario quiero responder una pregunta de un test del tipo respuesta por micrófono.

4.4.8. Sprint #8 - Funcionalidades del profesor con lecciones

31/03/2023 - 14/04/2023

En la octava iteración se comenzarán a realizar las funcionalidades relacionadas con el profesor y la gestión de las lecciones. Además, puede que se necesite terminar la funcionalidad de la entrada de micrófono de la iteración anterior debido a que su estimación supera los puntos de historia de un Sprint.

- HU28 - Como profesor quiero ver una lista de todas las lecciones.
- HU14 - Como profesor quiero crear una lección.
- HU15 - Como profesor quiero modificar el texto de una lección.
- HU16 - Como profesor quiero añadir contenido multimedia a una lección.
- HU17 - Como profesor quiero eliminar contenido multimedia de una lección.

4.4.9. Sprint #9 - Funcionalidades del profesor con tests

14/04/2023 - 28/04/2023

En esta iteración se realizarán las funcionalidades relacionadas con el profesor y la gestión de los tests, tales como la creación, modificación y eliminación de las preguntas de los distintos tipos. Además, se realizará la funcionalidad de ver todos los usuarios registrados en la aplicación.

- HU27 - Como profesor quiero añadir preguntas a un test del tipo escritura de texto.
- HU18 - Como profesor quiero añadir preguntas a un test del tipo selección múltiple.
- HU19 - Como profesor quiero añadir preguntas a un test del tipo selección única.
- HU20 - Como profesor quiero añadir preguntas a un test del tipo respuesta por micrófono.
- HU21 - Como profesor quiero eliminar preguntas de un test.
- HU22 - Como profesor quiero modificar preguntas de un test.
- HU30 - Como administrador quiero ver una lista de todos los usuarios.

4.4.10. Sprint #10 - Funcionalidades del administrador y ranking

28/04/2023 - 12/05/2023

En la decima iteración se pretenden implementar todas las funcionalidades que tienen que ver con la administración de los usuarios, además de dos historias relacionadas con el perfil propio del usuario.

- HU24 - Como administrador quiero modificar los datos de un usuario.
- HU25 - Como administrador quiero eliminar un usuario.
- HU3 - Como usuario quiero ver los datos personales y los logros de mi perfil.
- HU4 - Como usuario quiero editar los datos de mi perfil.
- HU23 - Como administrador quiero crear un usuario.

4.4.11. Sprint #11 - Funcionalidades poco prioritarias y valor añadido

12/05/2023 - 26/05/2023

El objetivo de este Sprint es acabar todas las historias de usuario que quedan del product backlog y cuya prioridad es baja como lo es la interacción entre los distintos usuarios de la aplicación. Además, se añadirá una funcionalidad de valor añadido en función del tiempo disponible y de las funcionalidades que se hayan podido implementar en los Sprints anteriores.

- HU5 - Como usuario quiero ver el perfil de otros usuarios.
- HU12 - Como usuario quiero ver el ranking de usuarios.
- Añadir funcionalidad de valor añadido (a definir)

4.4.12. Sprint #12 - Mejoras de código

26/05/2023 - 09/06/2023

En este Sprint se pretende realizar mejoras de código en cuanto a estilo, formato y calidad del mismo. Además, se pretende dejar terminado la redacción del séptimo capítulo del documento.

- Revisar Capítulo 7 - Implementación
- Realizar mejoras de código.

4.4.13. Sprint #13 - Pruebas

09/06/2023 - 23/06/2023

Este Sprint se dedicará a realizar todas las pruebas necesarias para comprobar que la aplicación funciona correctamente y que no hay errores en el código, tanto en el frontend como en el backend. También se redactará el octavo capítulo del documento con las pruebas realizadas.

- Realizar pruebas de integración.
- Realizar pruebas de unidad.
- Redactar Capítulo 8 - Pruebas.

4.4.14. Sprint #14 - Conclusión y finalización del proyecto

23/06/2023 - 15/07/2023

El último Sprint se dedicará por completo a la redacción del último capítulo del documento, en el que se recogerán las conclusiones del proyecto y se realizará una revisión de todos los capítulos para corregir errores y mejorar la redacción. Se preparará también la presentación del proyecto para la defensa ante el tribunal.

- Redactar Capítulo 9 - Conclusiones.
- Revisión de todo el documento y corrección de errores.
- Preparación de la presentación del proyecto.

4.5. Diagrama de Gantt

Por último, se muestran los diagramas de Gantt de la planificación del proyecto.

El primero es más general y representa la división del desarrollo del producto en Sprints. En él se puede ver la duración de cada Sprint y la proporción de estos en el tiempo total del proyecto.

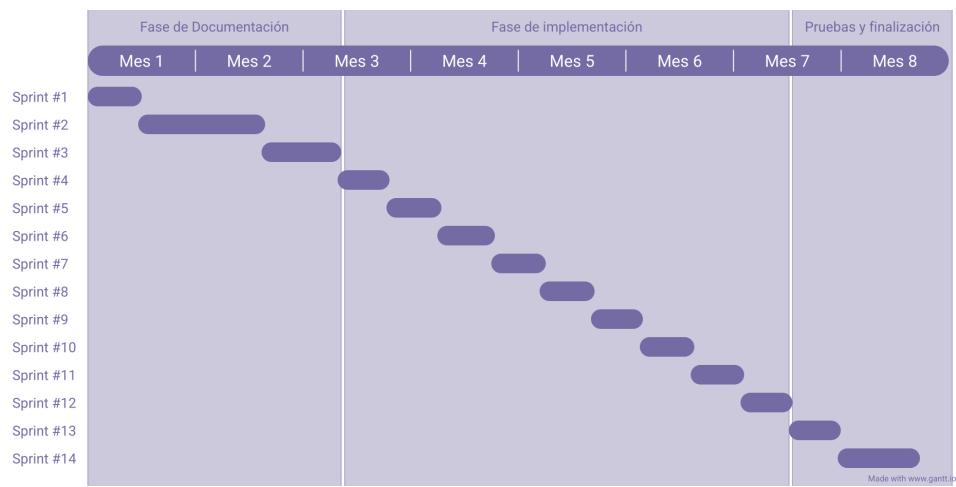


Figura 4.2: Diagrama de Gantt de la planificación del proyecto, divivido por Sprints.

Como se puede ver, el proyecto se ha dividido en 14 Sprints, cada uno de ellos con una duración de dos semanas aproximadamente. El desarrollo de cada Sprint se realizará normalmente de forma secuencial, es decir, no se empezará a desarrollar el siguiente Sprint hasta que no se haya terminado el anterior. Además, es importante recalcar que los Sprints del proyecto se dividen en tres fases:

- **Fase de Documentación:** en esta fase se redactará la documentación necesaria para el desarrollo del proyecto, como puede ser la especificación de requisitos, el estado del arte, la planificación, el diseño... A pesar de que esta sea la fase donde se redactará la mayor parte de la documentación, también se realizará esta en las fases posteriores.
- **Implementación:** en esta fase se desarrollará el producto, es decir, se implementarán las funcionalidades del proyecto.
- **Pruebas y finalización:** en esta fase se realizarán las pruebas necesarias para comprobar que el producto funciona correctamente y que no hay errores en el código. Además, se finalizará la documentación del proyecto y se preparará la presentación del mismo.

Por otro lado, a continuación se encuentra un diagrama de Gantt más detallado, en el que se puede ver la división de cada Sprint en historias de usuario o tareas y la duración de cada una de ellas. El diagrama se ha dividido en cuatro partes para facilitar su visualización y su lectura, debido a la gran cantidad de información que contiene.

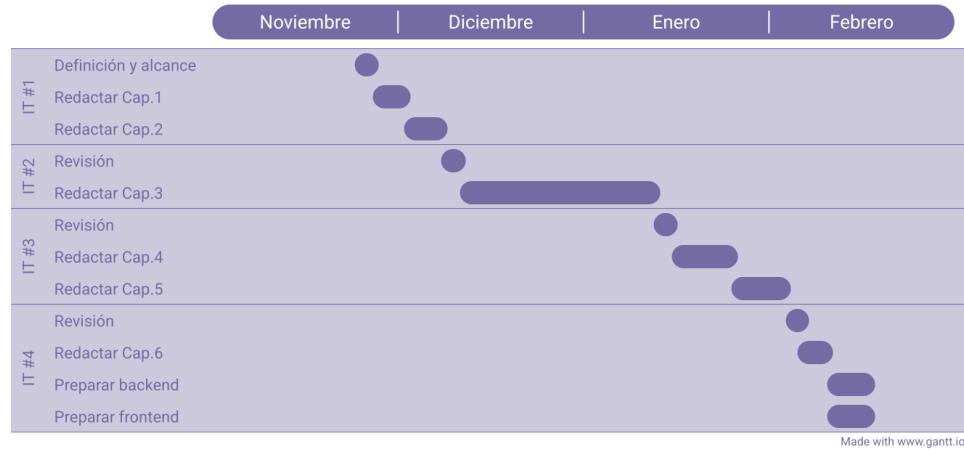


Figura 4.3: Diagrama de Gantt de los cuatro primeros Sprints.

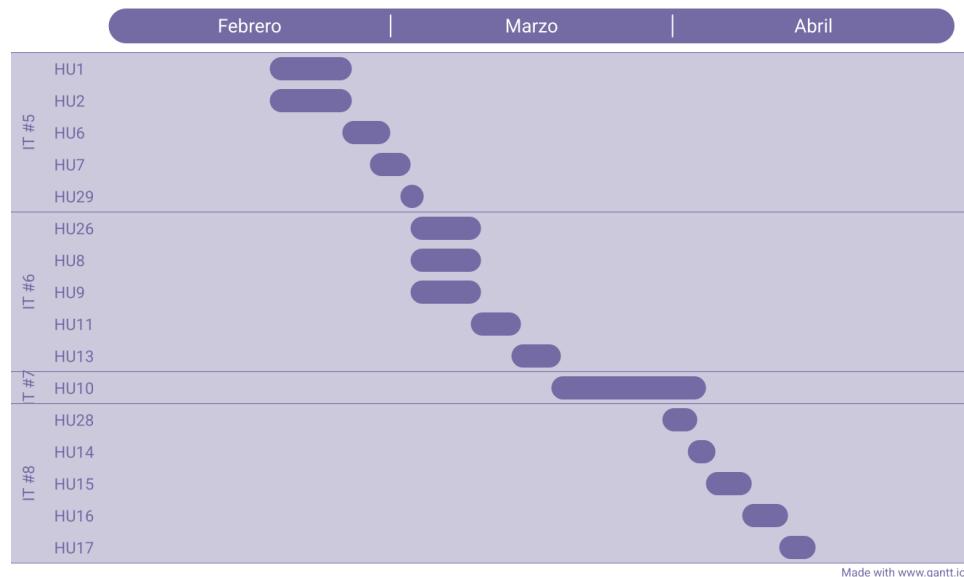


Figura 4.4: Diagrama de Gantt de los Sprints 5, 6, 7 y 8.

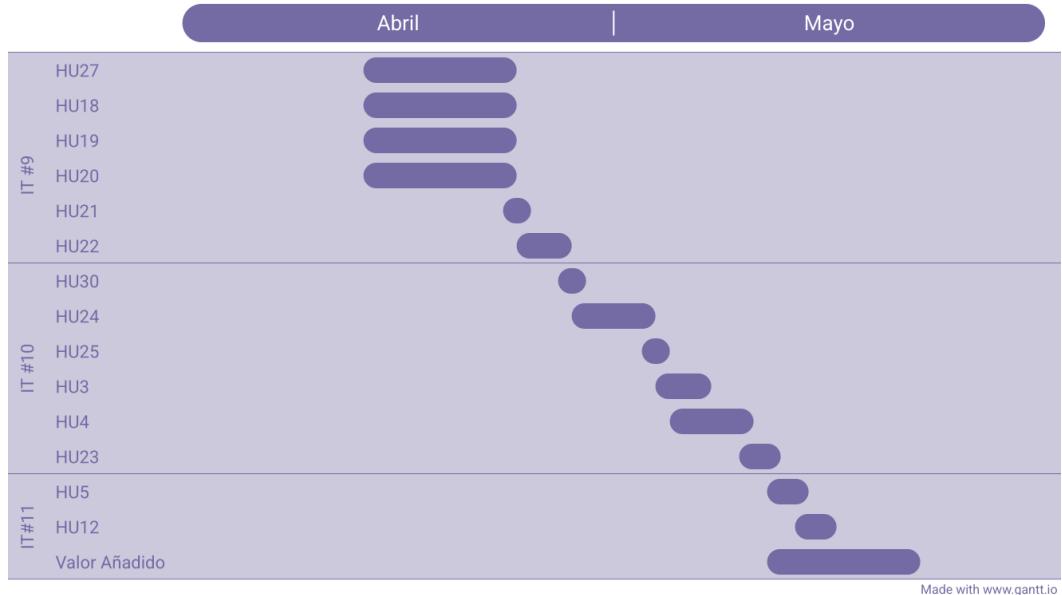


Figura 4.5: Diagrama de Gantt de los Sprints 9, 10 y 11.

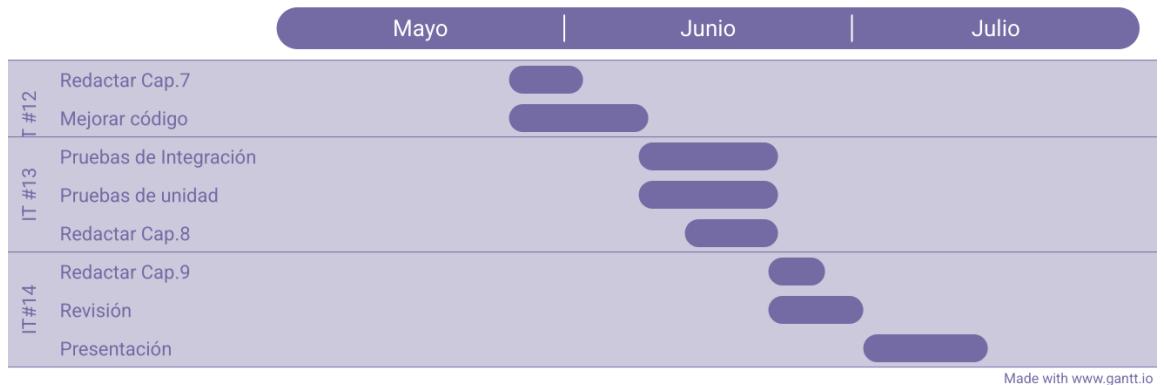


Figura 4.6: Diagrama de Gantt de los Sprints 12, 13 y 14.

Capítulo 5

Análisis del problema

5.1. Introducción

En este capítulo se va a describir de forma más detallada el problema que se va a resolver con la aplicación a desarrollar. De esta forma, vamos a especificar qué es lo que tiene que hacer el sistema para su futuro desarrollo. El objetivo del análisis es definir, estructurar y describir los requisitos o historias de usuario para conseguir una comprensión más precisa y detallada del problema a resolver.

5.2. Historias de Usuario

A continuación, se procederá a describir las historias de usuario que se han definido para el proyecto.

Funcionalidad del usuario

ID	HU1	Nombre	Como usuario quiero registrarme en la aplicación para poder comenzar a utilizar sus funcionalidades.
PH	2	Descripción	El usuario podrá registrarse en el sistema llenando un formulario con campos como el nombre de usuario, el correo electrónico, la contraseña...
Prioridad	Alta	Dependencias	— Requisitos Funcionales RF 1
Pruebas de aceptación		1. Todos los campos llenados por el usuario serán almacenados en la base de datos. 2. La contraseña se almacenará cifrada para no comprometer la seguridad del usuario. 3. Si el usuario introduce valores erróneos, se informará del campo en el que está el error. 4. Si el usuario no introduce datos en los campos obligatorios, se informará de los campos que faltan por llenar.	

ID	HU2	Nombre	Como usuario quiero iniciar sesión en la aplicación para poder acceder a sus funcionalidades.
PH	2	Descripcion	El usuario podrá iniciar sesión en el sistema rellenando un formulario con campos como el nombre de usuario o correo electrónico y la contraseña...
Prioridad	Alta	Dependencias	— Requisitos Funcionales RF 2
Pruebas de aceptacion			1. Se compararán los campos llenados por el usuario con los almacenados para comprobar la identidad del usuario. 2. La contraseña procesará de forma cifrada para no comprometer la seguridad del usuario. 3. Si el usuario introduce datos incorrectos, se informará del error al usuario.

ID	HU3	Nombre	Como usuario quiero ver los datos personales y los logros de mi perfil.
PH	1	Descripcion	El usuario podrá visualizar todos los datos de su propio perfil, así como los logros obtenidos en su aprendizaje.
Prioridad	Media	Dependencias	— Requisitos Funcionales RF 3
Pruebas de aceptacion			1. Los datos mostrados al usuario en su perfil se corresponderán con la información almacenada en la base de datos de dicho usuario.

ID	HU4	Nombre	Como usuario quiero editar los datos de mi perfil.
PH	2	Descripcion	El usuario podrá modificar los datos de su perfil como su nombre de usuario, correo electrónico, contraseña, etc...
Prioridad	Media	Dependencias	HU 3 Requisitos Funcionales RF 4
Pruebas de aceptacion			1. Los datos introducidos por el usuario en los campos del formulario se actualizarán en la base de datos 2. En caso de modificar la contraseña, esta se almacenará cifrada para no comprometer la seguridad del usuario. 3. Si el usuario introduce valores erróneos, se informará del campo en el que está el error.

ID	HU5	Nombre	Como usuario quiero ver el perfil de otros usuarios.
PH	1	Descripcion	El usuario podrá visitar los perfiles de otros usuarios registrados y ver sus logros
Prioridad	Baja	Dependencias	HU 3 Requisitos Funcionales RF 9
Pruebas de aceptacion			1. Los datos mostrados al usuario en su perfil se corresponderán con la información almacenada en la base de datos de dicho usuario. 2. Los datos privados del usuario no se mostrarán, solo los que no sean personales. 3. El usuario que no haya iniciado sesión no podrá ver ningún perfil.

ID	HU6	Nombre	Como usuario quiero seleccionar una lección para leer el temario.
PH	1	Descripcion	El usuario podrá hacer click en la lección que desea de la lista ubicada en la pantalla principal.
Prioridad	Alta	Dependencias	— Requisitos Funcionales RF 5
Pruebas de aceptacion			1. El contenido (textos, imágenes, videos...) almacenado en la base de datos de la lección seleccionada aparecerá en pantalla. 2. No se podrá elegir una lección que esté bloqueada.

ID	HU7	Nombre	Como usuario quiero empezar un test de una lección.
PH	1	Descripcion	El usuario pulsará el botón de empezar test ubicado dentro de la lección para poder responder a las preguntas.
Prioridad	Alta	Dependencias	— Requisitos Funcionales RF 6
Pruebas de aceptacion			1. Las preguntas almacenadas en la base de datos de dicha lección aparecerán en orden.

ID	HU8	Nombre	Como usuario quiero responder una pregunta de un test del tipo selección múltiple.
PH	1	Descripcion	El usuario podrá responder a una pregunta seleccionando más de una opción de la lista de posibles respuestas.
Prioridad	Alta	Dependencias	HU7 Requisitos Funcionales RF 6.2
Pruebas de aceptacion			1. La respuesta del usuario quedará registrada en el sistema para un futuro procesamiento. 2. El usuario deberá seleccionar una respuesta como mínimo.

ID	HU9	Nombre	Como usuario quiero responder una pregunta de un test del tipo selección única.
PH	1	Descripcion	El usuario podrá responder a una pregunta seleccionando una sola opción de la lista de posibles respuestas.
Prioridad	Alta	Dependencias	HU7 Requisitos Funcionales RF 6.3
Pruebas de aceptacion			1. La respuesta del usuario quedará registrada en el sistema para un futuro procesamiento. 2. El usuario deberá seleccionar una opción obligatoriamente.

ID	HU10	Nombre	Como usuario quiero responder una pregunta de un test del tipo respuesta por micrófono.
PH	8	Descripcion	El usuario podrá responder a una pregunta mediante la entrada de un sonido de un instrumento externo a través del micrófono.
Prioridad	Alta	Dependencias	HU7 Requisitos Funcionales RF 6.4
Pruebas de aceptacion			1. La respuesta del usuario quedará registrada en el sistema para un futuro procesamiento.

ID	HU26	Nombre	Como usuario quiero responder una pregunta de un test del tipo escritura de texto.
PH	1	Descripcion	El usuario podrá responder a una pregunta escribiendo la respuesta en un campo de texto.
Prioridad	Media	Dependencias	HU7 Requisitos Funcionales RF 6.1
Pruebas de aceptacion			1. La respuesta del usuario quedará registrada en el sistema para un futuro procesamiento.

ID	HU11	Nombre	Como usuario quiero ver el resultado de un test.
PH	2	Descripcion	El usuario podrá visualizar el resultado del test una vez finalizado.
Prioridad	Media	Dependencias	HU7, HU8, HU9, HU10, HU26 Requisitos Funcionales RF 6
Pruebas de aceptacion			1. El resultado del test será almacenado en la base de datos. 2. La puntuación mostrada en pantalla corresponderá a las preguntas acertadas en dicho test.

ID	HU12	Nombre	Como usuario quiero ver el ranking de usuarios.
PH	1	Descripcion	El usuario podrá visualizar la lista ordenada de usuarios con mayor progreso.
Prioridad	Baja	Dependencias	— Requisitos Funcionales RF 22
Pruebas de aceptacion			1. La lista mostrará a los usuarios con más puntuación en el sistema.

ID	HU13	Nombre	Como usuario quiero ver mi progreso en las lecciones.
PH	2	Descripcion	El usuario podrá ver su progreso en cada lección, los resultados de cada test y los puntos que le faltan para completar la lección.
Prioridad	Media	Dependencias	HU6 Requisitos Funcionales RF 1
Pruebas de aceptacion			1. Se mostrarán los datos relacionados con el progreso del usuario de la lección correspondiente.

Gestión de lecciones y tests

ID	HU14	Nombre	Como profesor quiero crear una lección.
PH	1	Descripcion	El profesor podrá crear una lección, añadiendo el texto y el contenido multimedia que desee.
Prioridad	Media	Dependencias	HU28 Requisitos Funcionales RF 11
Pruebas de aceptacion			1. Todos los campos rellenados por el profesor serán almacenados en la base de datos. 2. Si el usuario introduce valores erróneos, se informará del campo en el que está el error. 3. Si el usuario no introduce datos en los campos obligatorios, se informará de los campos que faltan por llenar.

ID	HU15	Nombre	Como profesor quiero modificar el texto de una lección.
PH	2	Descripcion	El profesor podrá modificar el texto de una lección en particular.
Prioridad	Media	Dependencias	HU14, HU28 Requisitos Funcionales RF 13
Pruebas de aceptacion			1. Todos los campos llenados por el profesor serán actualizados en la base de datos. 2. Si el usuario introduce valores erróneos, se informará del campo en el que está el error. 3. Si el usuario no introduce datos en los campos obligatorios, se informará de los campos que faltan por llenar.

ID	HU16	Nombre	Como profesor quiero añadir contenido multimedia a una lección.
PH	2	Descripcion	El usuario podrá añadir contenido multimedia al cuerpo de una lección.
Prioridad	Media	Dependencias	HU28, HU14 Requisitos Funcionales RF 13
Pruebas de aceptacion			1. Se agregará el contenido multimedia a la tabla de la lección en la base de datos. 2. Si el usuario introduce valores erróneos, se informará del campo en el que está el error. 3. Si el usuario no introduce datos en los campos obligatorios, se informará de los campos que faltan por llenar.

ID	HU17	Nombre	Como profesor quiero eliminar contenido multimedia de una lección.
PH	1	Descripcion	El usuario podrá eliminar contenido multimedia del cuerpo de una lección.
Prioridad	Media	Dependencias	HU28, HU14 Requisitos Funcionales RF 13
Pruebas de aceptacion			1. Se eliminará el contenido multimedia de la tabla de la lección en la base de datos.

ID	HU18	Nombre	Como profesor quiero añadir preguntas a un test del tipo selección múltiple.
PH	1	Descripcion	El profesor creará preguntas de tipo selección multiple y las añadirá a un test.
Prioridad	Media	Dependencias	— Requisitos Funcionales RF 14
Pruebas de aceptacion			1. Todos los campos llenados por el profesor serán almacenados en la base de datos. 2. La pregunta se mostrará en su test correspondiente.

ID	HU19	Nombre	Como profesor quiero añadir preguntas a un test del tipo selección única.
PH	1	Descripcion	El profesor creará preguntas de tipo selección única y las añadirá a un test.
Prioridad	Media	Dependencias	— Requisitos Funcionales RF 14
Pruebas de aceptacion			1. Todos los campos llenados por el profesor serán almacenados en la base de datos. 2. La pregunta se mostrará en su test correspondiente.

ID	HU20	Nombre	Como profesor quiero añadir preguntas a un test del tipo respuesta por micrófono.
PH	2	Descripcion	El profesor creará preguntas de tipo respuesta por micrófono y las añadirá a un test.
Prioridad	Media	Dependencias	— Requisitos Funcionales RF 14
Pruebas de aceptacion			1. Todos los campos llenados por el profesor serán almacenados en la base de datos. 2. La pregunta se mostrará en su test correspondiente.

ID	HU21	Nombre	Como profesor quiero eliminar preguntas de un test.
PH	0.5	Descripcion	El usuario podrá eliminar preguntas existentes de un test.
Prioridad	Media	Dependencias	HU29 Requisitos Funcionales RF 17
Pruebas de aceptacion			1. Se eliminará la información de dicha pregunta de la base de datos

ID	HU22	Nombre	Como profesor quiero modificar preguntas de un test.
PH	1	Descripcion	El profesor podrá modificar tanto el enunciado (la cuestión) como las posibles respuestas de cada pregunta de un test a través de un formulario.
Prioridad	Media	Dependencias	HU29 Requisitos Funcionales RF 16
Pruebas de aceptacion			1. Todos los campos llenados por el profesor serán actualizados en la base de datos 2. La contraseña se almacenará cifrada para no comprometer la seguridad del usuario. 3. Si el usuario introduce valores erróneos, se informará del campo en el que está el error. 4. Si el usuario no introduce datos en los campos obligatorios, se informará de los campos que faltan por llenar.

ID	HU27	Nombre	Como profesor quiero añadir preguntas a un test del tipo escritura de texto.
PH	1	Descripcion	El profesor creará preguntas de tipo escritura de texto y las añadirá a un test.
Prioridad	Media	Dependencias	— Requisitos Funcionales RF 14
Pruebas de aceptacion		1. Todos los campos rellenados por el profesor serán almacenados en la base de datos. 2. La pregunta se mostrará en su test correspondiente.	

ID	HU28	Nombre	Como profesor quiero ver una lista de todas las lecciones.
PH	0.5	Descripcion	El profesor podrá ver una lista de todas las lecciones almacenadas en el sistema.
Prioridad	Media	Dependencias	— Requisitos Funcionales RF 10
Pruebas de aceptacion		1. Todas las lecciones almacenados en la base de datos del sistema se mostrarán en pantalla	

ID	HU29	Nombre	Como profesor quiero ver una lista de todas las preguntas.
PH	0.5	Descripcion	El profesor podrá ver una lista de todas las preguntas almacenadas en el sistema.
Prioridad	Media	Dependencias	— Requisitos Funcionales RF 15
Pruebas de aceptacion		1. Todas las preguntas almacenadas en la base de datos del sistema se mostrarán en pantalla	

Gestión de usuarios

ID	HU23	Nombre	Como administrador quiero crear un usuario.
PH	1	Descripcion	El administrador podrá crear usuarios llenando un formulario con campos como el nombre de usuario, el correo electrónico, la contraseña...
Prioridad	Baja	Dependencias	— Requisitos Funcionales RF 19
Pruebas de aceptacion		1. Todos los campos rellenados por el administrador serán almacenados en la base de datos 2. La contraseña se almacenará cifrada para no comprometer la seguridad del usuario. 3. Si el usuario introduce valores erróneos, se informará del campo en el que está el error. 4. Si el usuario no introduce datos en los campos obligatorios, se informará de los campos que faltan por llenar.	

ID	HU24	Nombre	Como administrador quiero modificar los datos de un usuario.
PH	2	Descripcion	El administrador podrá modificar los datos de un usuario registrado en el sistema rellenando un formulario con campos como el nombre de usuario, el correo electrónico, la contraseña...
Prioridad	Media	Dependencias	HU30 Requisitos Funcionales RF 18
Pruebas de aceptacion		1. Todos los campos rellenados por el usuario serán almacenados en la base de datos 2. La contraseña se almacenará cifrada para no comprometer la seguridad del usuario. 3. Si el usuario introduce valores erróneos, se informará del campo en el que está el error. 4. Si el usuario no introduce datos en los campos obligatorios, se informará de los campos que faltan por llenar.	

ID	HU25	Nombre	Como administrador quiero eliminar un usuario.
PH	0.5	Descripcion	El administrador podrá eliminar un usuario registrado en el sistema.
Prioridad	Media	Dependencias	HU30 Requisitos Funcionales RF 21
Pruebas de aceptacion		1. Los datos del usuario serán eliminados de la base de datos.	

ID	HU30	Nombre	Como administrador quiero ver una lista de todos los usuarios.
PH	0.5	Descripción	El profesor podrá ver una lista de todos los alumnos registrados en el sistema.
Prioridad	Media	Dependencias	— Requisitos Funcionales RF 20
Pruebas de aceptación			1. Todos los usuarios almacenados en la base de datos del sistema se mostrarán en pantalla

5.3. Diagrama de Clases

Para finalizar esta sección, se muestra un primer diagrama de clases de nuestro proyecto, que nos permite ver las relaciones entre las diferentes clases que compondrán el sistema y que nos ayudará a entender mejor el funcionamiento del mismo. Para ello, primero se han identificado las entidades en nuestro sistema a partir de las historias de usuario y de los requisitos funcionales. Tras esto, se han hallado las relaciones entre las diferentes entidades. Por último, se han definido los atributos (las propiedades) de dichas entidades.

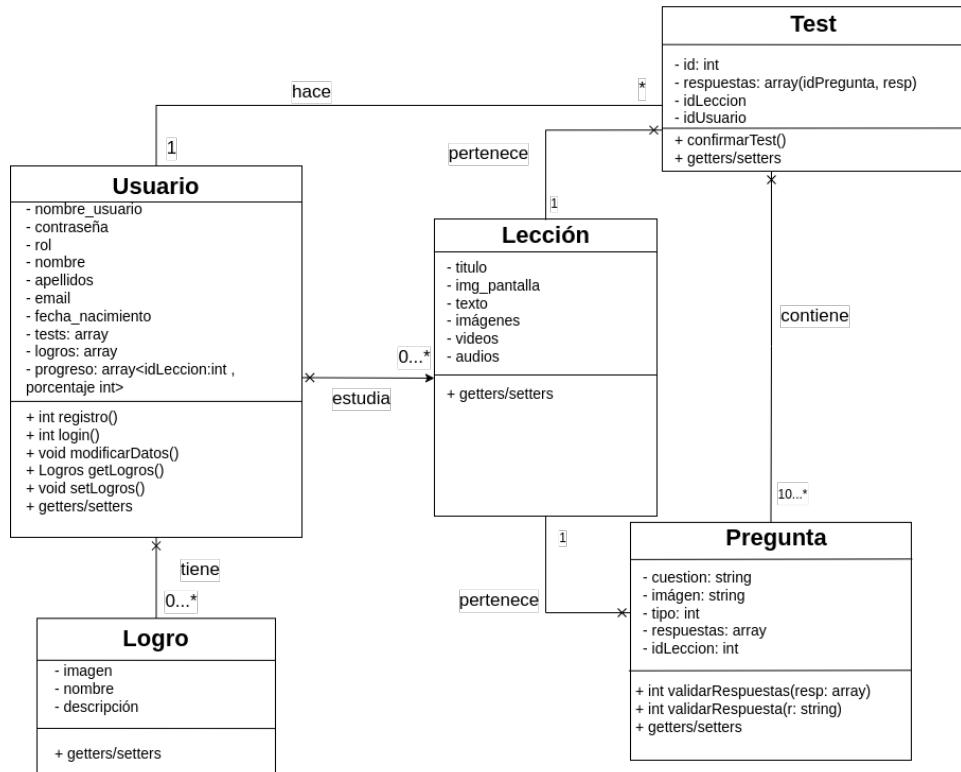


Figura 5.1: Diagrama de clases de nuestro proyecto donde se muestran las relaciones entre las diferentes clases que componen el sistema.

Como se puede observar, disponemos de cinco clases principales: Usuario, Lección, Test, Pregunta y Logro. Estas entidades se han relacionado

siguiendo la descripción del sistema que se ha realizado hasta ahora.

Capítulo 6

Diseño

6.1. Introducción

En este capítulo se realizará el diseño del software a desarrollar. Para ello, se detallará cómo se va a desarrollar la aplicación, qué arquitectura y tecnologías se van a utilizar y cómo se va a realizar el diseño de la interfaz de usuario. Esta fase tiene como objetivo definir la estructura del sistema y la función de cada una de sus partes, lo cual permitirá que el desarrollo sea más eficiente y que el resultado final tenga mayor calidad.

6.2. Arquitectura del sistema

Para el desarrollo de la aplicación se ha decidido utilizar una arquitectura **cliente-servidor**, donde el servidor será responsable de las operaciones relacionadas con la gestión (añadir, extraer, eliminar y/o modificar) de la información almacenada en la base de datos, mientras que el cliente será la aplicación móvil que se encargará de mostrar la información al usuario y de realizar las operaciones que el usuario solicite. Además, también será una arquitectura basada en el patrón de diseño **Modelo / Vista / Controlador (MVC)**, donde el modelo gestionará y mantendrá la estructura de la información de la base de datos, la vista se hará cargo de mostrar la información al usuario y facilitar la interacción de este con el sistema y el controlador se encargará de gestionar las operaciones que el usuario solicite. Todo esto mediante las tecnologías **Node.js** para el controlador, **Mongoose** para el modelo, **MongoDB** para la base de datos y **Flutter** para las distintas vistas que los usuarios tendrán.

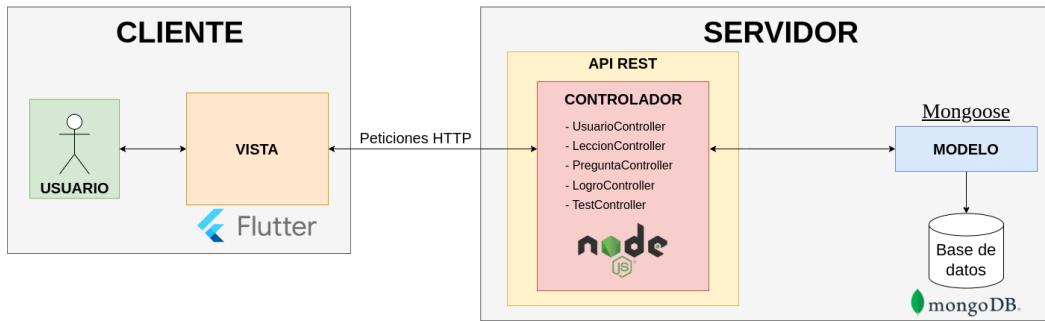


Figura 6.1: Diagrama de la arquitectura del sistema, donde se muestra la comunicación entre el servidor y la aplicación móvil.

Como podemos ver en el diagrama, el controlador tendrá una API REST que será la encargada de gestionar las peticiones que se realicen desde la aplicación móvil. La API REST es una interfaz que permite la comunicación entre dos sistemas de computación (en nuestro caso, entre el cliente y el servidor) para el intercambio de información de manera segura. REST definirá las restricciones a cumplir dentro de nuestra arquitectura, como por ejemplo la necesidad de utilizar los principales verbos HTTP (GET, POST, PUT y DELETE) para realizar las operaciones de lectura, creación, actualización y eliminación de la información.

6.3. Diagrama de base de datos

En cuanto al diagrama de base de datos, se ha realizado un diagrama de base de datos no relacional, que representa los distintos documentos que tendrá nuestra base de datos de MongoDB. En este caso, se han definido 5 documentos, uno para los usuarios, otro para las lecciones, otro para los tests, otro para las preguntas de los tests y otro para los logros. También se han definido las referencias que habrá entre los documentos, como por ejemplo que un usuario tiene varios logros.

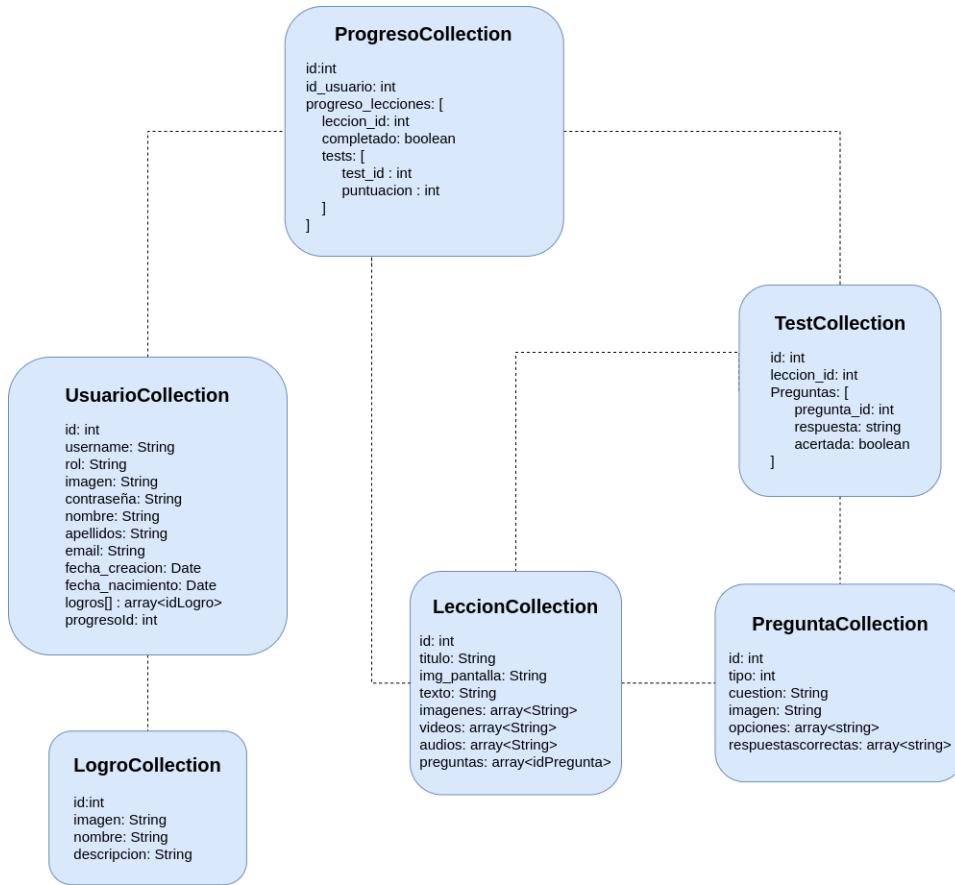


Figura 6.2: Diagrama de base de datos del sistema, donde se muestran los documentos que se van a almacenar en nuestra base de datos de MongoDB.

En el diagrama se pueden ver varias referencias entre las colecciones de datos que se explicarán a continuación:

- Un usuario tendrá los logros que consiga al progresar en la aplicación.
- Un usuario tiene un progreso, que contendrá las lecciones que realice junto con los tests que haya contestado.
- Un test tendrá varias preguntas (alrededor de 10) y estas se generarán de forma aleatoria cuando se cree el test.
- Una lección tendrá varias preguntas y de dicho conjunto de preguntas se seleccionarán algunas al azar para formar el test.
- Un test pertenecerá a una lección: un test solo podrá pertenecer a una lección (aquella donde se generó dicho test).

6.4. Diagrama de clases

A continuación se muestra el diagrama de clases de diseño, el cual se ha realizado a partir del diagrama de clases de análisis del capítulo anterior. En esta ocasión se han detallado más los atributos y los métodos de cada clase, facilitando así la comprensión de las relaciones entre las clases y de la estructura del sistema.

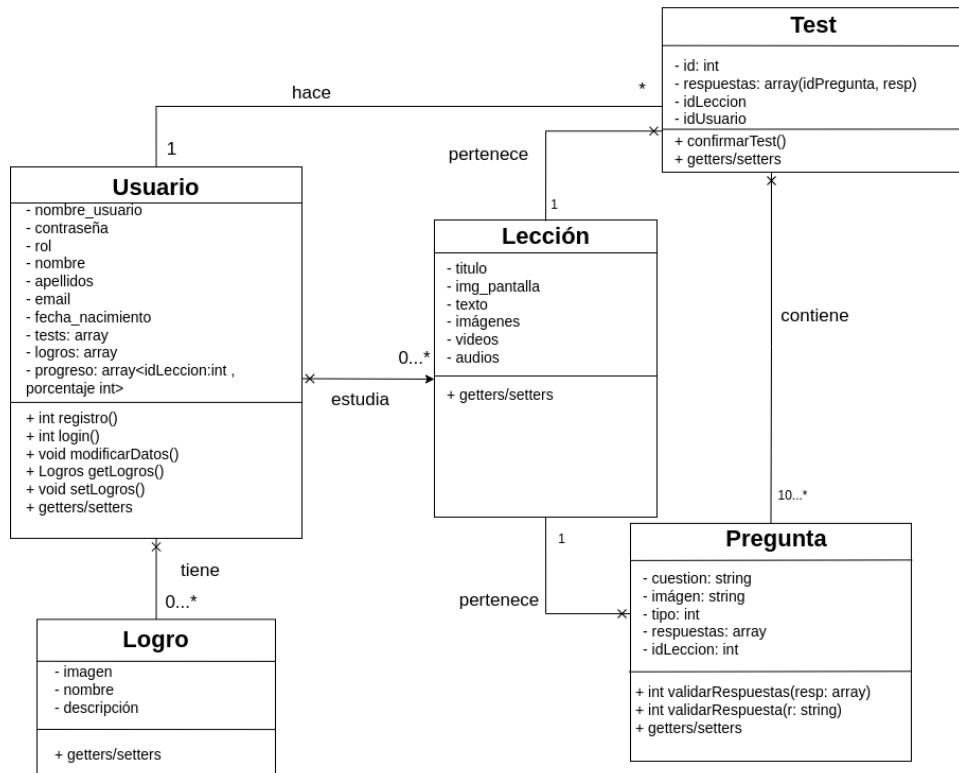


Figura 6.3: Diagrama de clases del sistema donde se detallan las propiedades y las relaciones de las distintas clases o entidades que tendrá el software.

6.5. Diagramas de secuencia

En lo que respecta a los diagramas de secuencia, se han realizado algunos ejemplos de cómo sería el flujo de una operación en el sistema. Como hemos visto anteriormente, se seguirá una arquitectura basada en el patrón de diseño MVC, por lo que las peticiones pasarán por la vista, luego por el controlador y finalmente por el modelo. En este caso, se ha realizado un diagrama de secuencia de cómo sería el flujo de una operación en la que un usuario se registra en la aplicación, otro para un usuario que contesta un test y otro para un profesor que modifica una lección.

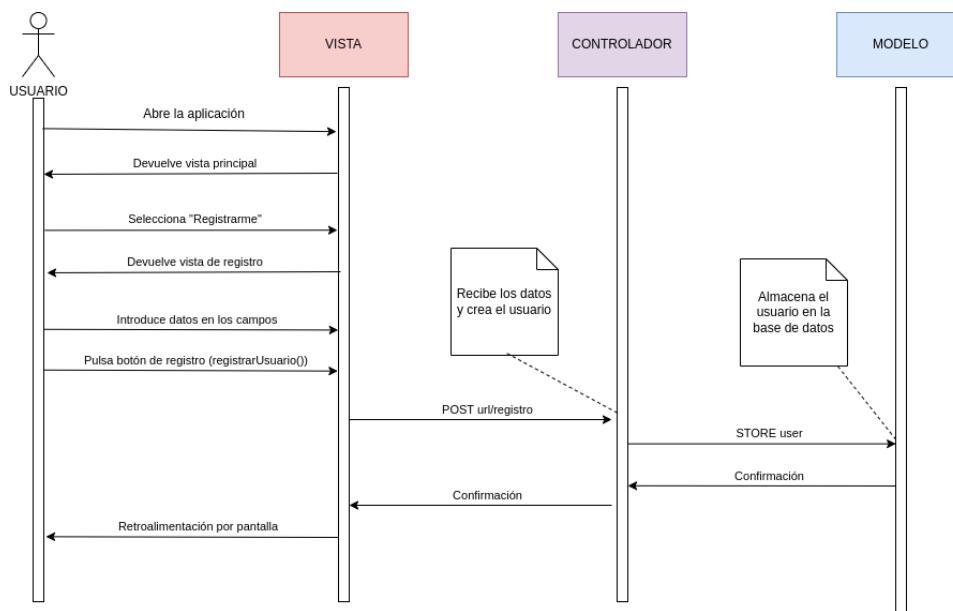


Figura 6.4: Diagrama de secuencia de un usuario registrándose, el cual interactuará con la vista para poder hacer la petición al controlador y guardar su usuario en el modelo.

6.6. Diseño de interfaces de usuario

Por último, en este apartado se presentan los bocetos de las interfaces de usuario que se han diseñado para la aplicación. Estos bocetos se han realizado con la herramienta Canva y serán de ayuda para la realización del diseño final de las interfaces de usuario, que, pese a que seguirán una estructura similar a la de los bocetos, podrán variar en algunos detalles y aspectos de diseño.



Figura 6.5: Boceto de la pantalla de inicio de sesión, donde se pedirá al usuario el correo electrónico y la contraseña.



Figura 6.6: Boceto de la pantalla de registro, donde se pedirá al usuario sus datos personales necesarios para crear la cuenta como el nombre, los apellidos, el correo y la contraseña.

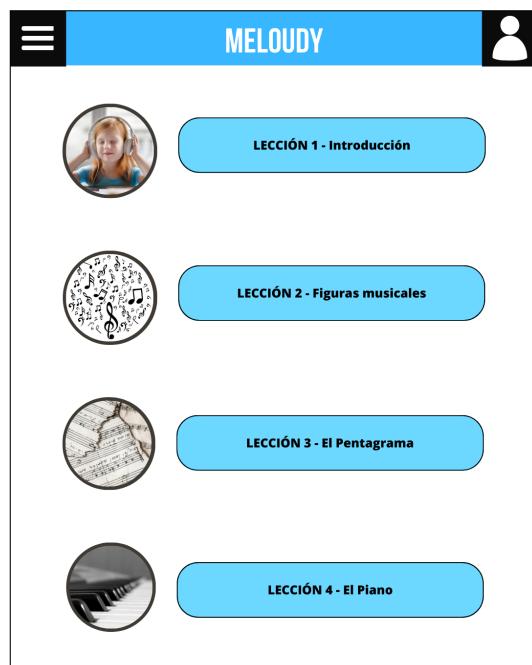


Figura 6.7: Boceto de la pantalla principal de la aplicación donde se muestran las lecciones disponibles para el usuario.

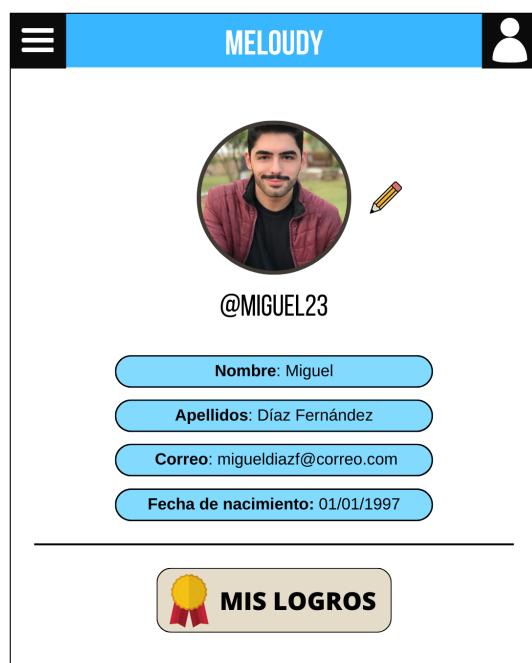


Figura 6.8: Boceto de la pantalla del perfil de usuario con la sesión iniciada donde se muestran sus datos.

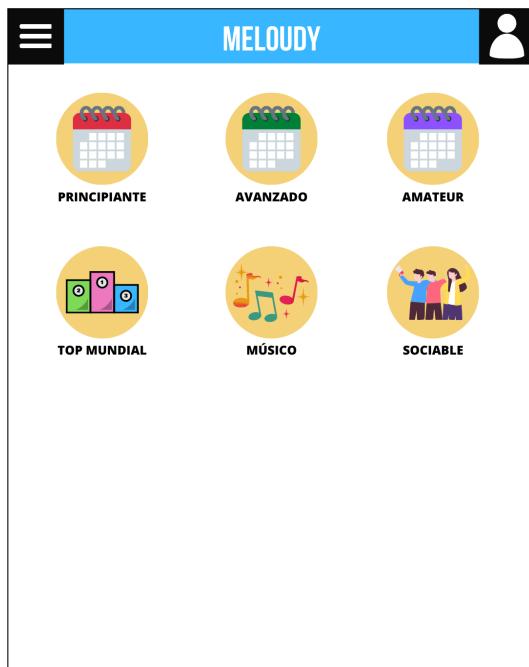


Figura 6.9: Boceto de la pantalla de logros conseguidos por el usuario.



Figura 6.10: Boceto de la pantalla de una lección, con el texto, el contenido multimedia y el botón para comenzar el test.

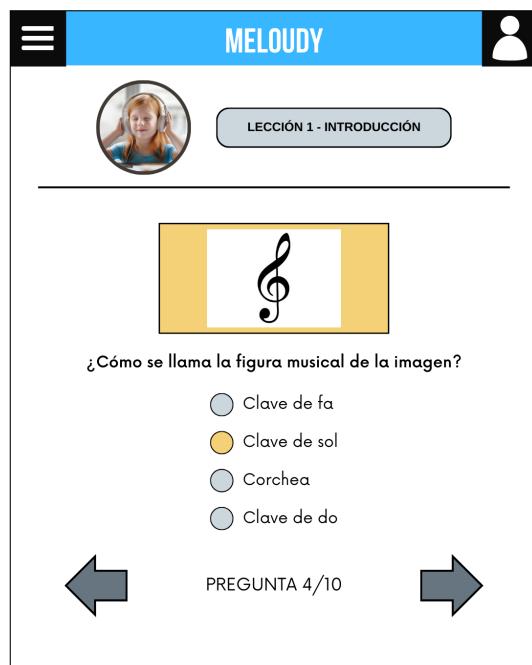


Figura 6.11: Boceto de la pantalla de una pregunta de test de tipo selección única.

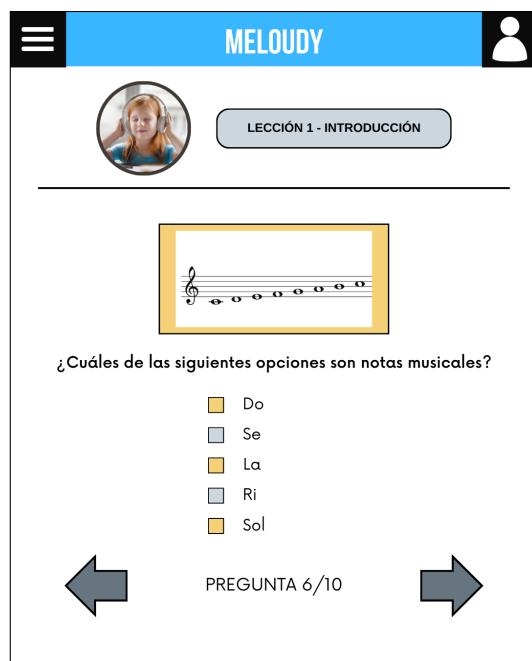


Figura 6.12: Boceto de la pantalla de una pregunta de test de tipo selección multiple.



Figura 6.13: Boceto de la pantalla de una pregunta de test de tipo escritura de texto.

X

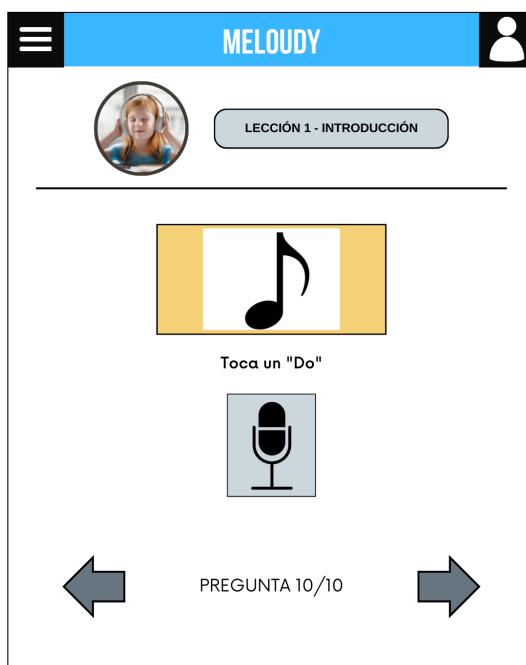


Figura 6.14: Boceto de la pantalla de una pregunta de test de tipo entrada por micrófono.

Para finalizar la sección se presenta un diagrama de navegación de la aplicación, el cual muestra las distintas pantallas que tendrá la aplicación siguiendo el flujo que un usuario tendría al utilizar la aplicación.

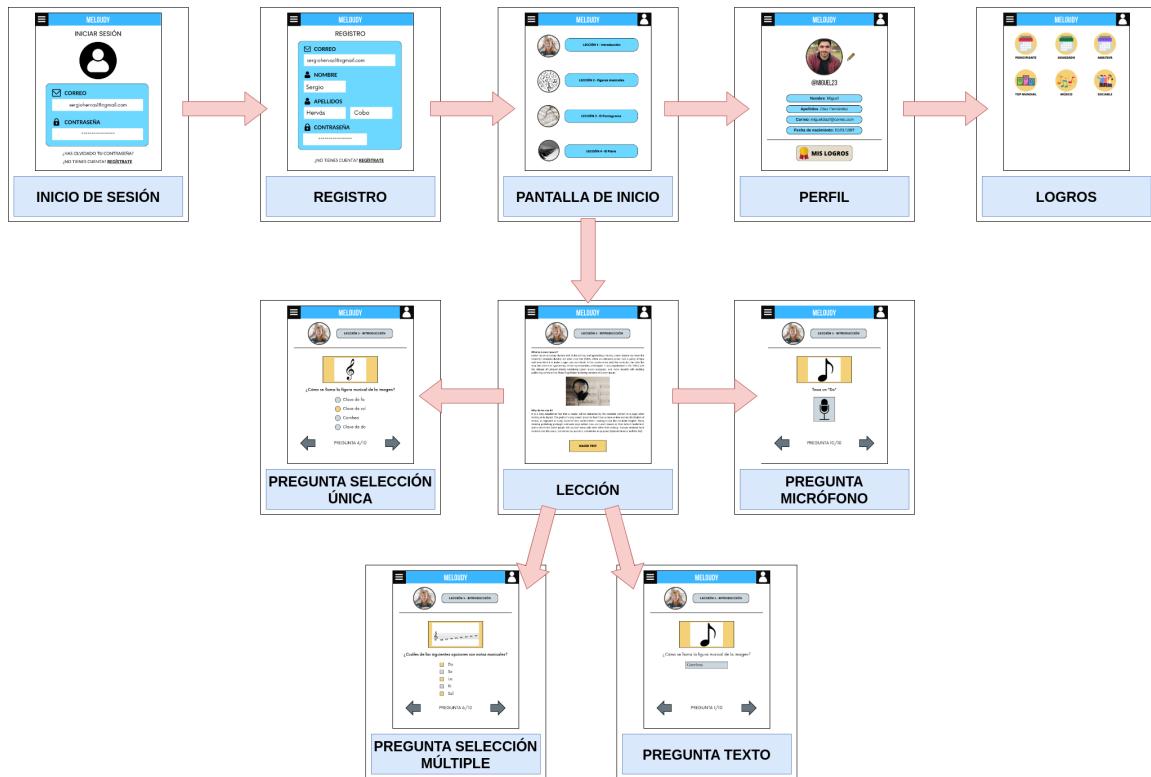


Figura 6.15: Boceto de la pantalla de una pregunta de test de tipo entrada por micrófono.

Capítulo 7

Implementación

En este séptimo capítulo se presentará la implementación del sistema. En primer lugar, se presentará la estructura del proyecto que se ha realizado en la preparación del frontend y del backend y a continuación se describirá la implementación de cada una de las funcionalidades del sistema, presentando la documentación de cada ruta de la API y describiendo el código escrito durante el desarrollo.

7.1. Estructura del proyecto

Como se mencionó en la arquitectura del sistema en el anterior capítulo, el proyecto se ha desarrollado en dos partes, una parte de frontend y otra de backend. La parte de frontend se ha desarrollado en Flutter y la parte de backend se ha desarrollado en Node.js.

La estructura del proyecto se puede ver en la siguiente imagen.

7.2. Funcionalidad genérica

Sesión

 Inicio de sesión

 Registro

 Cierre de sesión

7.3. Funcionalidad de usuario

Lista de lecciones

Lección

Capítulo 8

Pruebas

Capítulo 9

Conclusiones

