MÁSTER EN PROGRAMACIÓN Y DESARROLLO DE APLICACIONES

**II EDICIÓN**



**Proyecto Final de Máster**

***Dirigido por D. Jose Alberto Rosales Navarro***

Julio de 2025

Autores:

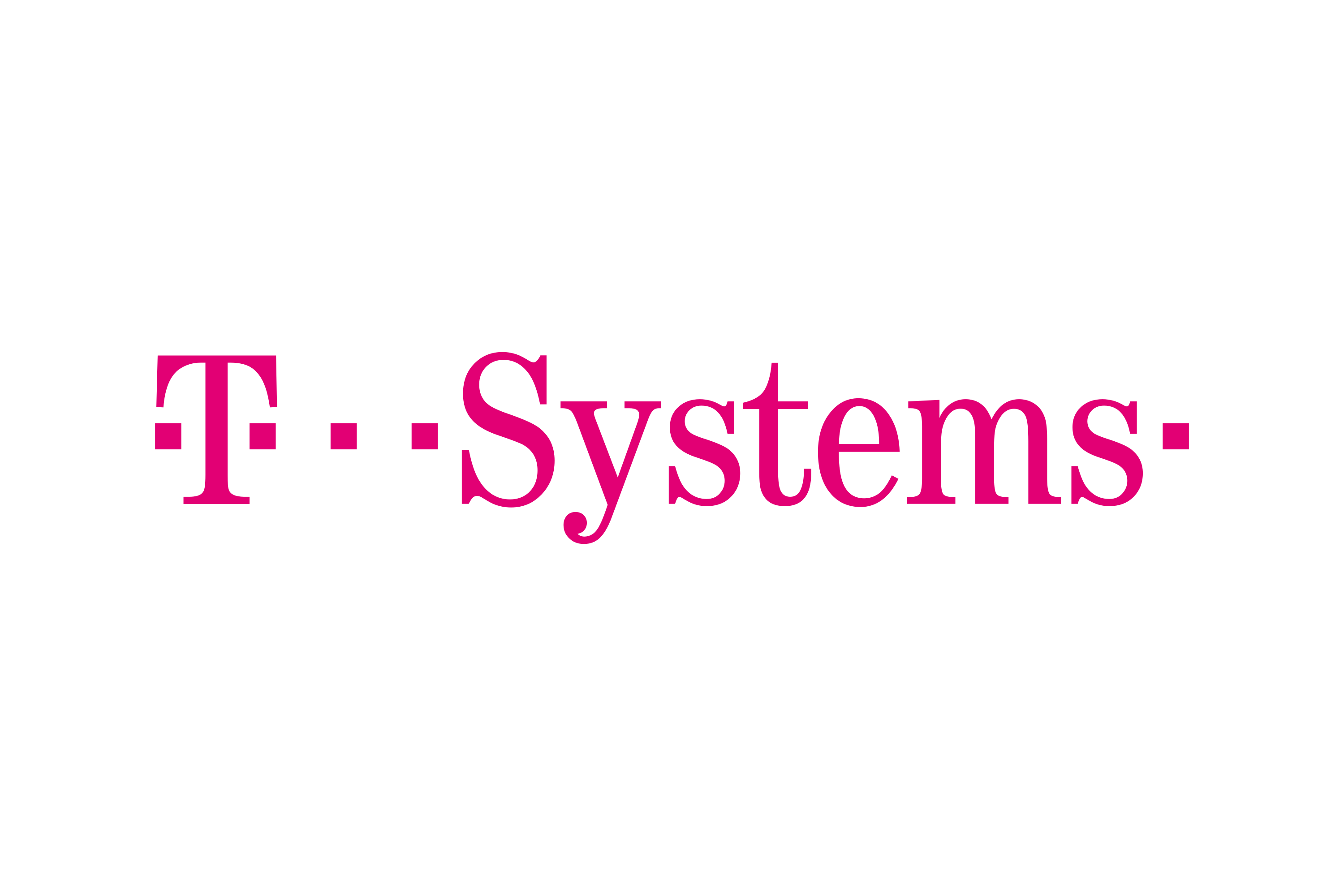
**Cristian Castilla Vílchez**

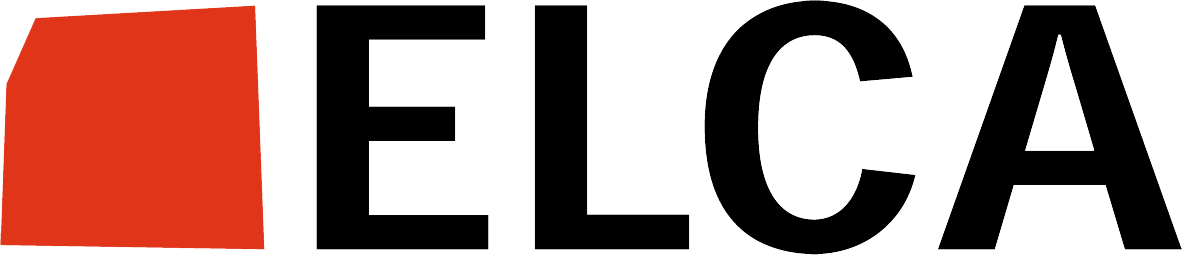
**David Pérez García**

**Javier Baena Tro**

**Antonio Garrido Machado**

**Francisco Simón Moreno Balaguer**



****

**RESUMEN**

El mundo de la programación es un campo en continua evolución en el mundo de las Tecnologías de la Información y Comunicación. El siguiente Proyecto de Fin de Máster refleja el desarrollo de una aplicación interactiva que pretende reforzar el aprendizaje de futuros alumnos del Máster con el objetivo de asentar conocimientos y promover un aprendizaje más dinámico mediante juegos educativos.

Esta herramienta podrá ser implementada en futuras ediciones como complemento de los diferentes módulos impartidos en el Máster, ofreciendo a los alumnos una manera práctica y entretenida de consolidar los conocimientos dados en clase.

Además de su valor didáctico, la herramienta permite evaluar el progreso continuo de los alumnos, permitiendo a los docentes realizar un seguimiento del aprendizaje.

La aplicación se llama “Duocoding” y consiste en una herramienta interactiva con diferentes juegos como tipos test, espacios que completar, código para corregir,etc que el usuario deberá ir superando para poder seguir avanzando por los diferentes módulos que se impartirán en el curso, recibiendo un feedback positivo o negativo de cada uno de ellos basado en puntuaciones y premios.

El público objetivo hacia el que está orientada la aplicación serán los alumnos del Máster de futuras ediciones, así como cualquier alumno actual o persona que quiera recordar conceptos de la programación de manera más entretenida, puesto que conocemos el contexto a la perfección por experiencia propia, sabemos lo densas que pueden resultar las clases en ocasiones y creemos que nuestra herramienta puede aportar un aire fresco necesario en el sector de la programación.

**Palabras clave:** Programación, aprendizaje, aplicación, alumnos, Máster.

**ABSTRACT**

The world of programming is a constantly evolving field within the broader domain of Information and Communication Technologies. This Master's Final Project presents the development of an interactive application aimed at reinforcing the learning process of future master's students, with the goal of consolidating knowledge and promoting a more dynamic learning experience through educational games.

This tool can be implemented in future editions of the program as a complementary resource to the various modules taught in the Master's, offering students a practical and engaging way to reinforce the content covered in class.

In addition to its educational value, the tool enables continuous assessment of student progress, allowing instructors to monitor and support the learning process effectively.

The application is called “Duocoding” and consists of an interactive tool with different games such as multiple-choice tests, fill-in-the-blank exercises, code to correct, etc., which the user must complete to advance through the different modules taught in the course. Each activity provides positive or negative feedback based on scores and rewards.

The target audience for the application will be students of the Master’s program in future editions, as well as any current student or person who wants to review programming concepts in a more entertaining way. Since we know the context perfectly from our own experience, we understand how dense the classes can sometimes be, and we believe our tool can bring a necessary breath of fresh air to the programming field.

**Key words:** Programming, learning, application, students, Máster.

**AGRADECIMIENTOS**

Quisiéramos expresar nuestro más sincero agradecimiento a todas las personas que han contribuido, directa o indirectamente, a la realización de este Proyecto de Fin de Máster.

En primer lugar, agradecemos a nuestro tutor Jose Alberto Rosales Navarro por su orientación constante, su disponibilidad y sus valiosas sugerencias durante todo el proceso. Su experiencia y criterio han sido fundamentales para el desarrollo y mejora de este trabajo.

También nos gustaría agradecer al equipo docente del Máster que ha sido partícipe en la Escuela de Dirección y Altos Estudios por haber compartido sus conocimientos y haber fomentado un entorno académico riguroso y enriquecedor. Cada asignatura y cada profesor/a han aportado una pieza clave en nuestra formación.

A mis compañeros y compañeras de promoción, gracias por el apoyo, el intercambio de ideas y la piña que han hecho de esta etapa una experiencia aún más valiosa.

.

**Índice del PFM**

[1. Introducción 1](#_Toc200963899)

[2. Análisis del problema y objetivos 1](#_Toc200963900)

[3. Revisión del estado del Arte 4](#_Toc200963901)

[4. Diseño 5](#_Toc200963902)

[5. Desarrollo 7](#_Toc200963903)

[6. Despliegue 10](#_Toc200963904)

[7. Evaluación y resultados 12](#_Toc200963905)

[8. Conclusiones 12](#_Toc200963906)

[9. Referencias 13](#_Toc200963907)

[10. Anexos 13](#_Toc200963908)

1. Introducción

La motivación del proyecto surge de la carencia que hemos detectado a lo largo de las clases impartidas de un método o herramienta que permita a los alumnos poner en práctica los conocimientos adquiridos en cada sesión y recibir un feedback diario sobre lo aprendido en cada materia. Todo esto, combinado con la complejidad que puede presentar la programación en determinadas ocasiones, nos llevó a desarrollar una aplicación que hiciera el aprendizaje más didáctico y, sobre todo, accesible para cualquier alumno en todo momento.

Actualmente, podemos observar que tanto en los colegios como en las universidades son necesarias herramientas digitales que apoyen el aprendizaje. Con el paso del tiempo, esta necesidad ha ido creciendo. Los métodos manuales, como los exámenes en papel, no permiten que los alumnos reciban una retroalimentación instantánea. Además, en ocasiones, los estudiantes necesitan contar con recursos que les permitan repasar desde cualquier lugar y a cualquier hora, especialmente en momentos en los que no disponen de un profesor.

Del mismo modo, los docentes pueden enfrentar impedimentos o dificultades para cumplir con ciertos plazos y corregir decenas de exámenes o tareas, sobre todo en determinados meses del curso.

Teniendo en cuenta todo esto, el uso de herramientas digitales puede ser una solución a estos problemas o inconvenientes, así como una forma de agilizar y optimizar la enseñanza.

Esperamos que en el futuro la aplicación traspase la barrera del máster y, por qué no, pueda llegar a personas que están aprendiendo programación de manera autodidacta, a profesionales relacionados con este mundo o incluso que se utilice en universidades públicas como una herramienta complementaria.

Por último, para llevar a cabo este proyecto decidimos aplicar la metodología Scrum, hemos realizado sprints semanales, llevando a cabo una planificación del mismo, diariamente se han realizado reuniones para tratar lo esencial, hemos realizado una revisión del sprint para mostrar todo el trabajo realizado y hemos tratado de resaltar puntos de mejora.

2. Análisis del problema y objetivos

* **Necesidades del usuario**

El estudiante necesita que los ejercicios estén organizados por asignaturas, y que pueda encontrarlos rápido y sin líos. También le viene bien que, al terminar un ejercicio, reciba comentarios o correcciones al momento, sin tener que esperar. Le interesa saber cómo va progresando: sus notas, cuántos intentos ha hecho, si ha mejorado... Ese tipo de cosas. Además, como hoy en día usamos muchos dispositivos distintos, le gustaría poder entrar desde el móvil, la compu o la tablet sin problemas. Y claro, que sus datos estén seguros es clave. A veces, si hay cosas como medallas, logros o recompensas, eso le anima más a seguir practicando.

El profesor, por su parte, busca algo práctico para crear ejercicios de diferentes materias. Quiere poder elegir si serán tipo test, verdadero o falso, o preguntas abiertas. También le interesa poder definir si un ejercicio es fácil o difícil, según lo que necesite enseñar. Ver cómo les va a sus alumnos, tanto uno por uno como en grupo, es importante. Y si los estudiantes le dejan comentarios o dudas, necesita poder responder fácilmente. Algo que valora mucho es que el sistema corrija los ejercicios automáticamente, porque así ahorra tiempo.

Para ambos usuarios es importante que tengan un inicio de sesión rápido, intuitivo y seguro, así como una interfaz fácil de usar, rápida y que no presente cuelgues o fallos recurrentes.

* **Carencias en soluciones actuales**

En estos momentos las plataformas que hay para practicar programación online de forma gratuita son bastante deficientes o bien carecen de contenido suficiente o no tienen ningún tipo de feedback ni por parte del alumno ni del administrador o profesor que las lleva.

Otra carencia común es la falta de diversidad en cuanto a ejercicios se refiere, muchas plataformas te brindan un pdf con ejercicios a resolver por tu cuenta o te ponen únicamente un formato de ejercicio, lo cual es insuficiente para los que desean practicar y ampliar su conocimiento.

Uno de los problemas más graves es la falta de automatización en cuanto a la corrección de ejercicios, existiendo tiempos de demora que van desde unos días hasta el punto de no existir nunca una respuesta. Además, usando esas plataformas los profesores no tienen ocasión de hacer un seguimiento sobre los rendimientos del alumnado, quedando el trabajo de este en anecdótico.

Por otro lado, tenemos la falta de feedback en las plataformas, la mayoría no dejan escribir dudas o comentarios en los ejercicios y materias, lo cual deja a los alumnos completamente desamparados en la realización de ejercicios, haciendo que recurran más de lo necesario a herramientas como la inteligencia artificial. 7

Para finalizar podemos exponer que gran parte de la oferta de la competencia no funciona bien siendo multiplataforma,cuando hoy en día es muy útil poder practicar desde el móvil mientras viajas en metro o desde la tablet cuando tienes un rato en el sofá de tu casa.

* **Objetivos funcionales y no funcionales:**

**Objetivos funcionales:**

En esta plataforma, los usuarios podrán registrarse como alumnos o profesores. Los alumnos tendrán acceso a ejercicios organizados por asignaturas, mientras que los profesores podrán crear, modificar o eliminar esos ejercicios cuando lo necesiten. Además, podrán asignarlos a las materias correspondientes.

Los ejercicios que sean de tipo test o de verdadero/falso se corregirán de forma automática, para que el alumno pueda ver su nota al instante. Cada estudiante podrá revisar su progreso y tener un historial con los resultados de los ejercicios que haya hecho.

También se podrá dejar comentarios en los ejercicios, lo que permitirá compartir opiniones o sugerencias. Habrá un apartado donde se muestren todas las asignaturas disponibles junto con sus ejercicios.

Dependiendo del tipo de usuario, se tendrá acceso a unas funciones u otras. Por ejemplo, solo los profesores podrán editar los ejercicios. Además, el sistema mostrará estadísticas generales y personales, para que se pueda ver cómo ha sido la participación y el rendimiento tanto de cada persona como del grupo en general.

**Objetivos no funcionales:**

La aplicación podrá usarse desde cualquier navegador actual y también funcionará bien en móviles, para que sea accesible desde cualquier lugar. Todo debería cargar y responder rápidamente, idealmente en menos de dos segundos en la mayoría de los casos.

La información de los usuarios, especialmente las contraseñas, se guardará de forma segura. Además, la plataforma será fácil de usar, con una interfaz intuitiva y pensada para que cualquier persona pueda usarla sin complicaciones.

Se espera que esté disponible todo el tiempo, con un nivel de funcionamiento medio, y que pueda manejar a muchos usuarios conectados al mismo tiempo sin que eso afecte la velocidad o el rendimiento.

El sistema también está pensado para crecer con el tiempo, por lo que será posible añadir nuevas funciones sin necesidad de rehacerlo todo. En cuanto a los datos, la base de datos se encargará de que todo se guarde correctamente y sin errores, manteniendo la información bien organizada. Por último, se protegerá el contenido de los ejercicios para evitar que alguien los modifique sin autorización.

3. Revisión del estado del Arte

En esta sección y con el fin de mostrar qué soluciones y tecnologías ya existen se están usando en este sector, realizaremos una revisión de las principales plataformas educativas que propongan modelos similares al nuestro para realizar una comparativa.

Concretamente podemos encontrar ciertos paralelismos con las plataformas enfocadas en el aprendizaje de idiomas como, por ejemplo, Duolingo, Babbel o Quizlet. Todas estas proponen modelos educativos interactivos basados en el aprendizaje a partir de la realización del test. Es decir, estas plataformas utilizan preguntas de opción múltiple, verdadero o falso y ejercicios autoevaluables, para reforzar ideas básicas, así como vocabulario o gramática.

En primer lugar, Duolingo es una de las plataformas más reconocidas a nivel mundial con relación al aprendizaje de idiomas. Su modelo educativo está basado en la gamificación de contenidos.  Muchos de sus ejercicios están basados en modelos de elección múltiple, verdadero o falso, emparejar palabras o audio o escritura. Para nosotros será de gran relevancia los apartados de elección múltiple o verdadero y falso ya que serán el grueso de nuestros ejercicios.  Además, cabe destacar que este modelo permite una corrección automática e inmediata de cada actividad. Así mismo, algo fundamental de esta plataforma que debemos de tener en cuenta en todo momento es el proceso gamificador mediante sistema de logros y recompensas que actúa como motor motivacional en el alumno.

Por otro lado, podemos tener en cuenta otras aplicaciones similares, pero con importantes diferencias a nivel estético como es el caso de Babble, esta plataforma más formal y está menos enfocado en la gamificación a pesar de contar con actividades muy parecidas a Duolingo. En esta plataforma podemos destacar que además de las pruebas podemos encontrar un abanico de contenidos más amplios de perfil teórico, esto es, repasos gramaticales, vocabulario o incluso podcast de refuerzo, algo que tal vez se podría incluir en futuras versiones de nuestra aplicación. En definitiva, Babbel es una plataforma de corte más académica y formal que Duolingo y sin embargo su forma de presentar los contenidos organizados por temas y contextos, así como su presentación más austera y clara de presentar los diferentes ejercicios nos puede servir de referencia visual para crear nuestra propia aplicación.

Finalmente, además de las plataformas enfocadas en el aprendizaje de idiomas existen otras herramientas generalistas que merecen nuestra atención y que son utilizadas ampliamente en el ámbito de la docencia. Se trata de herramientas como Kahoot! o Quizizz que permiten realizar cuestionarios interactivos con preguntas de opción múltiple, verdadero/falso y otras variantes, enfocadas tanto a la enseñanza como a la evaluación formativa en el aula. Estas herramientas están pensadas fundamentalmente para fomentar actividades en grupos sin embargo la versatilidad y agilidad que ofrece a los profesores a la hora de crear las actividades será un aspecto a tener en cuenta en nuestra aplicación.

A modo de conclusión podemos decir que este modelo de aprendizaje basado en tipo test es ampliamente utilizado en plataformas educativas especialmente en aquellas dirigidas al aprendizaje de nuevas lenguas y cuyo aprendizaje puede resultar interesante desde el punto de vista del aprendizaje autónomo integrado en entornos educativos. Sin embargo, podemos encontrar ciertas limitaciones ya que muchas de estas plataformas no permiten el control o seguimiento por parte del docente. Por esta razón, nuestra aplicación buscará una solución intermedia donde podremos encontrar un catálogo de pruebas interactivas, con corrección automática bajo una supervisión tanto de los contenidos como del progreso del alumno por parte del profesorado.

4. Diseño

**4.1. Arquitectura General**

La arquitectura de **DuoCoding** se basa en un modelo cliente-servidor desacoplado, estructurado en tres capas:

**Capa de Presentación (Frontend):**

* Desarrollada con **Angular 19**, un framework moderno para SPA.
* Se encarga de la interfaz de usuario, validación de formularios, navegación y comunicación con el backend.
* Comunicación con el servidor mediante peticiones HTTP (REST).

**Capa de Lógica de Negocio (Backend):**

* Implementada con **Spring Boot 3.x**, que permite crear APIs RESTful de manera sencilla.
* Gestiona autenticación, autorización, lógica de negocio, validaciones y errores.

**Capa de Persistencia (Base de Datos):**

* Utiliza **MySQL** (o PostgreSQL) como sistema relacional.
* El acceso a datos se realiza mediante **Spring Data JPA**, usando entidades JPA para mapear tablas.

**4.2. Diagrama de Componentes**

**Frontend (Angular):**

**Componentes**:

* LoginComponent, RegistroComponent, DashboardComponent, AsignaturasComponent
* EjerciciosComponent, ResultadosComponent, ComentariosComponent, AdminPanelComponent

**Servicios**:

* UsuarioService: autenticación y gestión de usuarios
* AsignaturaService, EjercicioService, ResultadoService, ComentarioService

**Backend (Spring Boot):**

**Controladores REST**:

* UsuarioController, AsignaturaController, EjercicioController, etc.

**Servicios**:

* Encapsulan lógica de negocio por entidad

**Repositorios**:

* Interfaces que permiten operaciones CRUD mediante Spring Data JPA

**Base de Datos**:

* Tablas: usuarios, asignaturas, ejercicios, resultados, comentarios

**4.3. Casos de Uso**

* **UC01 – Registro**: Usuario se registra (nick, email, contraseña).
* **UC02 – Login**: Inicia sesión y accede al dashboard.
* **UC03 – Navegar por asignaturas**: Selecciona módulos y ve contenidos.
* **UC04 – Resolver ejercicio**: Contesta y envía respuestas. Se calcula la puntuación.
* **UC05 – Ver progreso**: Consulta estadísticas, logros, ranking.
* **UC06 – Gestión de contenido (admin)**: Admin crea/modifica/elimina contenido.
* **UC07 – Feedback**: Estudiante comenta sobre los ejercicios.

**4.4. Mockups o Wireframes**

* **Login**: Email, contraseña, enlace a “Recuperar contraseña”.
* **Registro**: Nick, email, contraseña y confirmación.
* **Dashboard**: Tarjetas de asignaturas, progreso, botón de entrar.
* **Ejercicios**: Lista de ejercicios, preguntas, botón de envío.
* **Perfil**: Información de usuario, opciones para cambiar contraseña/email.
* **AdminPanel**: Gestión CRUD de asignaturas y ejercicios, moderación de comentarios.

**4.5. Modelo de Datos / Diagrama ER**

**Usuario**

* id: Long, nick: String, email: String, password: String, fechaRegistro: LocalDateTime, admin: Boolean, progreso: Double

**Asignatura**

* id: Long, nombre: String, descripcion: String, fechaCreacion: Date

**Ejercicio**

* id: Long, titulo: String, tipo: String, dificultad: String, contenido: Text, asignaturaId (FK)

**Resultado**

* id: Long, usuarioId (FK), ejercicioId (FK), puntuación: Double, fechaRealizacion: Date

**Comentario**

* id: Long, usuarioId (FK), ejercicioId (FK), contenido: Text, fechaComentario: Date

5. Desarrollo

**5.1. Tecnologías utilizadas.**

Para llevar a cabo el desarrollo de este Proyecto Fin de Máster hemos utilizado diversas tecnologías a la orden del día en el sector de la programación que garantizan el buen funcionamiento de la herramienta y la correcta navegación del usuario dentro de ella. Estas tecnologías utilizadas las podemos ver a continuación, divididas por cada una de las partes que componen la herramienta:

**Frontend:**

Angular es un *framework* que permite construir aplicaciones web dinámicas y robustas. Ofrece una estructura sólida para el desarrollo de aplicaciones mediante la combinación de tecnologías como **HTML**, **CSS** y **TypeScript**, este último siendo un superconjunto de JavaScript que incorpora tipado estático, a diferencia del tipado dinámico propio de JavaScript.

Entre las características más relevantes de Angular se destaca el uso de **componentes**, que permiten construir la interfaz de usuario mediante bloques modulares que encapsulan lógica, plantillas y estilos. Además, Angular implementa el mecanismo de **data binding** bidireccional, lo que significa que los cambios realizados en la interfaz de usuario se reflejan automáticamente en los datos, y viceversa.

Otro aspecto clave del framework es su sistema de **inyección de dependencias**, que facilita la gestión eficiente de los distintos servicios y recursos que una aplicación requiere. A ello se suma su **sistema de enrutamiento integrado**, que permite crear aplicaciones de una sola página (SPA), y el soporte para **formularios reactivos y basados en plantillas**, lo que facilita la gestión y validación de formularios complejos. También destaca la integración con **RxJS**, una biblioteca que permite gestionar flujos de datos asíncronos mediante el uso de *Observables*.

En definitiva, gran parte de nuestra motivación para elegir este framework se basa en su capacidad para construir aplicaciones web modernas, escalables y eficientes con relativa facilidad.

Por otra parte, en lo que respecta a los estilos visuales, tanto **TailwindCSS** como **Bootstrap** se presentan como opciones adecuadas para el desarrollo del proyecto. Su utilización permite acceder a componentes predefinidos como botones, barras de navegación (*navbars*) o modales, con estilos visuales consistentes y personalizables. La principal ventaja radica en la agilidad con la que se puede lograr una apariencia uniforme y profesional en las interfaces de usuario.

Finalmente, en cuanto a las herramientas de pruebas, se considera que **Jasmine** sería la más útil, ya que proporciona una sintaxis clara y descriptiva que permite estructurar pruebas de manera legible y comprensible, favoreciendo así el aseguramiento de la calidad del desarrollo frontend.

**Backend:**

Para construir nuestro backend emplearemos Java, uno de los lenguajes de programación orientado a objetos más ejecutados en dispositivos en todo el mundo. Este lenguaje es la base en sistemas operativos de smartphones,software empresarial y muchos otros programas conocidos. A pesar de tener una trayectoría de más de 20 años hoy en día sigue siendo uno de los lenguajes de programación más populares en el desarrollo de aplicaciones.

A continuación, expondremos algunas de las principales razones por las cuales hemos seleccionado esta tecnología.

En primer lugar, se trata de un lenguaje de programación orientado a objetos, es decir, nos va a permitir organizar el código en torno a clases y objetos, en lugar de funciones y comandos, esto hace que Java sea un lenguaje versátil y relativamente fácil de aprender

Por otro lado, se trata de un lenguaje multiplataforma, es decir, su código escrito para una plataforma (como el sistema operativo Windows) puede transferirse fácilmente a otra plataforma (como el sistema operativo de un teléfono móvil) y viceversa, sin necesidad de reescribir por completo. A continuación, nos centraremos en explicar cómo funciona Java tomando como referencia la documentación oficial de Microsoft.

El proceso es el siguiente, se código Java se escribe en primer lugar en un kit de desarrollo Java disponible para varios sistemas como Windows o Linux y luego este kit traduce a código informático que cualquier dispositivo puede leer con el software adecuado. Esto se consigue con un software llamado compilado, este compilador toma el código informático de alto nivel, como Java, y lo traduce a un lenguaje que los sistemas operativos entienden, llamado código de bytes. Es entonces cuando entra en juego un elemento fundamental, el intérprete, conocido como Máquina virtual Java (JVM) que procesa el código de bytes. Las JMV están disponibles para la mayoría de las plataformas de software y hardware y esto les permite transferir el código Java de un dispositivo a otro. Para ejecutar, las JVM cargán el código, lo comprueban y proporcionan un entorno en tiempo de ejecución.

Por otro lado, para nuestro proyecto será de vital importancia el framework Spring. Se trata de un popular marco de trabajo empresarial de código abierto que permite crear aplicaciones autónomas de producción que se ejecutan en una máquina virtual Java (JVM).

Según podemos encontrar en el recurso web de IBM Java Spring Boot (Spring Boot) es una herramienta que nos va a permitir acelerar y simplificar el desarrollo de microservicios y aplicaciones web con Spring FrameWork. Esto es debido fundamentalmente a tres funcionalidades: la configuración automática, un enfoque de configuración obstinado y la capacidad de crear aplicaciones autónomas. Estas funciones combinadas nos permitirán configurar una aplicación basada en Spring con el mínimo de instalación y configuración.

Para ser más concreto debemos mencionar tres de los módulos más importantes para nuestro ecosistema de Spring Framework en Java, Spring Web, Spring Security y Spring Data.JPA.

En primer lugar, Spring web es clave para poder crear aplicaciones web tradicionales o RESTful. Spring Web forma parte de Spring MCV (Model-View-controller). Esto es, este módulo nos permitirá utilizar anotaciones de tipo @RestController o @RequestMapping, así como prestarnos soporte para manejar las solicitudes HTTP.Por su partem Spring Security nos permitirá encargarnos de la autenticación y autorización de usuarios en la aplicación. En este sentido, debemos señalar la importancia de la autenticación basada en formularios o tokens JWT o la autorización de roles y permisos.Por último, Spring Data JPA, nos facilitará el acceso a bases de datos utilizando JPA (Java Persistence Api), esto nos plantea una serie de ventajas como la posibilidad de definir nuestros repositorios con interfaces sin la necesidad de escribir implementaciones así como la opción de realizar consultas automáticas o la integración con bases de datos como MySQL, O PostgreSQL.

Finalmente, para cerrar el apartado del backend vamos a señalar una serie de herramientas que nos serán de gran utilidad durante nuestro trabajo. En primer lugar, debemos subrayar el empleo de Postman, esta herramienta básicamente nos permite realizar peticiones de una manera simple para testear **APIs** de tipo **REST** propias, es decir, nos va a facilitar testear colecciones o catálogos de APIs tanto para Frontend como para Backend. Además, nos permitirá realizar otra serie de funciones como gestionar el ciclo de vida (conceptualización y definición, desarrollo, monitoreo y mantenimiento) de nuestra **API,** así como organizar en carpetas, funcionalidades y módulos los servicios web.

Con Postman podremos realizar los métodos básicos de un CRUD, esto es, crear, leer, actualizar y borrar y podremos recibir información sobre nuestros errores al realizar las peticiones. A modo de resumen, sin la respuesta se encuentra dentro del rango de “200” querrá decir que nuestra petición fue exitosa, si se encuentra dentro del rango de “400” harán referencia a errores vinculados con el cliente y aquellos dentro del rango “500” tendrán que ver con fallos en el servidor.

Otra de las herramientas que usaremos será Lombok. Se trata de una biblioteca de código abierto que elimina la necesidad de escribir código repetitivo. Nos permitirá simplificar la creación de getters y setters así como implementar constructores de forma rápida, generar automáticamente los métodos equals, hashCode o toString(). En definitiva, Lombok, aumentará nuestro rendimiento a la vez que dejará nuestras clases más limpias, sin código repetitivo lo cual mejorará su legibilidad y el tamaño de estas clases básicas a lo largo del proyecto.

Finalmente, tenemos que comentar el uso de Flyway. Esta herramienta se utiliza en relación a las migraciones de bases de datos y básicamente nos facilitará el trabajo tanto a la hora de mantener como de montar un entorno. En resumidas cuentas, cuando tenemos que replicar un entorno en un equipo, o estamos realizando la configuración inicial de nuestra aplicación en un servidor en el que estamos desplegándola, tenemos que realizar una minuciosa labor de configuración de la base de datos. Mediante Flyway podemos automatizar todos estos pasos en un script SQL para homogeneizar la configuración y evitar fallos por despistes.

**Base de Datos:**

Finalmente, para cerrar este apartado debemos desarrollar el uso y elección de nuestra base de datos. Tanto PostgresSQL y MySQL son muy similarese entre ellas y ambas serían buenas opciones para nosotros, sin embargo, finalmente nos decantamos por esta última ya que al estar todos los miembros del equipo más familiarizados con esta podremos alcanzar un mayor rendimiento a la hora de elaborar nuestro proyecto.

MySQL es una de las bases de datos de código abierto más populares en todo el mundo y fundamentalmente nos permitirá realizar entregas de forma rentable de aplicaciones de comercio electrónico, procesamiento de transacciones en línea y de bases de datos integradas de forma segura y escalable. En definitiva, MySQL nos facilita el uso, escalabilidad y rendimiento a la vez que un conjunto completo de controladores de bases de datos y herramientas visuales que nos ayudará como programadores a administrar las bases de datos.

Por último, y para complementar el apartado de base de datos debemos comentar otra herramienta fundamental para poder relacionarnos con esta desde nuestro programa. Concretamente para realizar consultas a la base de datos emplearemos una herramienta ORM (Object-Relational Mapping) como Hibernate. Esta herramienta nos permitirá interactuar con la base de datos de una manera orientada a objetos. JPA (Java Persistence ) funciona como una capa de abstracción que se coloca sobre las implementaciones ORM, como Hibernate, para estandarizar la forma en que interactuamos con la base de datos.

**5.2. Organización del código**

Como apuntábamos en el anterior punto, para construir nuestro backend será de vital importancia el uso del framework Spring Boot. Este jugará un rol fundamental a la hora de elaborar nuestra aplicación web permitiéndonos construir servicios web robustos y estructurados de forma modular, esto será una de las claves para poder exponer una de una manera clara y global de la estructura que presentará nuestro backend.

Dentro del directorio: src/main/java/com.codingtrainers.duocoding podemos visualizar la estructura de nuestro backend, una estructura que sigue la denominada arquitectura por capas, esto es, una estructura modular que fomenta la encapsulación y la separación de responsabilidades. De esta forma, podemos encontrar las siguientes carpetas:

**Entities**: una Entity o entidad será una clase que representará una entidad del dominio del sistema, en otras palabras, podemos entender que representará un objeto importante para el negocio. Esta entidad tendrá identidad única que será constante, además estás clases están mapeadas directamente a las tablas de la base de datos mediante anotaciones JPA.

 Finalmente, también podemos destacar que estas clases podrían llegar a tener métodos que expresan comportamientos de la propia entidad.

**Repositories**: En esta capa encontraremos una serie de interfaces donde emplearemos JpaRepository. Aquí podemos ver como entra en juego Spring Data JPA, lo que nos permitirá acceder a la base de datos de forma abstracta. JPA (Java Persistence API) es una especificación de Java que se emplea para el mapeo objeto-relacional (ORM) y básicamente permite mapear clases Java (Entities) a tablas de base de datos sin escribir SQL de forma manual. En definitiva, estas interfaces se implementarán de forma automática durante la ejecución y básicamente nos permitirá realizar operaciones CRUD de una manera más fácil. En definitiva, podemos entender esta capa como una especie de “puerta de entrada” a la base de datos.  Gracias a esta capa, otras capas como Services o Controller podrán trabajar con Entities al margen de cómo estas se guardan en la base de datos.

**Services**: probablemente la capa más importante de nuestra estructura se trata de la capa donde implementaremos la lógica de negocio de la aplicación. La capa de los services recibirá peticiones desde la capa Controllers a continuación en los services se procesa la lógica con la información recibida y se comunicará con la base de datos a través de los repositorios anteriormente explicados.

**Controllers**:  En esta capa es la responsable de realizar las peticiones HTTP entrantes. Se comunicará con la capa services para obtener o enviar información. Podemos visualizarla como la capa de entrada a las peticiones del cliente recibiendo las solicitudes HTTP (GET, POST, PUT, DELETE, etc.) tras ello, llamará a su correspondiente servicio y devolverá una respuesta HTTP adecuada.

**DTO (Data Transfer Object)**: esta capa juega un rol clave a la hora de transmitir información. una DTO será un objeto plano que tendrá el objeto de transportar datos entre capas, fundamentalmente podemos destacar la comunicación entre el cliente y el servidor. Esta capa no contiene ningún tipo de lógica de negocio solo busca transportar información de forma clara y segura.

Por otro lado, también podemos destacar una pieza muy importante en nuestra estructura de nuestro backend, hablamos concretamente de Flyway, que no es tanto un elemento estructural como una herramienta. Esta herramienta nos permitirá gestionar versiones y migraciones a la base de datos. Dentro de nuestro directorio resources/migration podremos encontrar los scripts SQL que definirán nuestra estructura de la base de datos, así como sus posteriores modificaciones.

Para cerrar este punto podríamos concluir que este modelo de estructuras está enfocado en la búsqueda de un desarrollo más limpio, mantenible y escalable.  En los próximos apartados buscaremos centrarnos de forma más exhaustiva en cada uno de estos componentes para poder ver cómo este sistema se implementa para la creación y gestión de tests por parte de los usuarios.

**5.3. CRUDs implementados (explicar cada uno)**

1. **Usuarios**:
   * Funciones: registro, login, editar, eliminar
   * Seguridad: contraseñas cifradas, validación de email
2. **Asignaturas**:
   * Solo accesible por administradores
   * Asociación con ejercicios
3. **Ejercicios**:
   * Test o V/F, asignables a módulos
   * CRUD completo con contenido enriquecido
4. **Resultados**:
   * Guarda puntuaciones de cada intento
   * Historial visible por usuario
5. **Comentarios**:
   * Feedback de estudiantes
   * Moderación de admins

**5.4. Funcionalidades adicionales (login, roles, estadísticas, etc.)**

* **Login y roles**:
  + Autenticación vía email/contraseña
  + Roles: Estudiante y Admin
  + Seguridad JWT o sesión
* **Estadísticas y logros**:
  + Progreso por módulo
  + Ranking, badges y niveles
* **Panel Admin**:
  + Gestión total de contenido
  + Visualización de comentarios
* **Diseño adaptativo (responsive)**:
  + Compatible con móviles, tablets y escritorio
* **Validaciones y errores**:
  + Frontend: validación en formularios
  + Backend: manejo centralizado de errores (401, 403, 500…)
* **Experiencia de usuario**:
  + Feedback visual, navegación intuitiva, mensajes claros

6. Despliegue

Para garantizar un acceso confiable, seguro y escalable a la plataforma, se ha optado por desplegar Duocoding en un entorno de producción basado en Microsoft Azure.

La elección de Azure como plataforma cloud permitirá aprovechar una infraestructura sólida y altamente disponible, que ofrecerá servicios gestionados para cada uno de los componentes de la aplicación: frontend, backend, base de datos y dominio. Esto nos permitirá automatizar despliegues, escalar recursos según sea necesario y centralizar la gestión de la infraestructura desde el panel de Azure.

* Entorno de producción

El entorno de producción se configurará para asegurar una experiencia óptima de usuario. Al usar azure, contamos con las siguientes características en nuestro entorno de producción:

* Alta disponibilidad: Replicación y balanceo de carga mediante los datacenter de Azure para garantizar una experiencia continua, incluso ante fallos de hardware o picos de demanda en el caso de que la app escale.
* Escalabilidad automática: Los recursos que se asignan a cada parte de la app podrían crecer o reducirse en base a la demanda, lo que nos ayudaría a optimizar costes y rendimiento.
* Seguridad: Se aplicarán medidas de seguridad a nivel de red, aplicación y datos, incluyendo firewall, cifrado de información y gestión de identidades mediante servicios integrados.
* Monitoreo y alertas: Se configurarán herramientas de Azure (como “Application Insights”) para monitorizar el estado de la aplicación, detectar errores y analizar patrones de uso, facilitando la mejora continua de nuestra app.
* Separación de entornos: Se mantendrán diferenciados los entornos de desarrollo, pruebas y producción para minimizar riesgos y asegurar la calidad en cada lanzamiento.
* Configuración del servidor / hosting

Cada componente de Duocoding se desplegará en servicios gestionados de Azure que simplificarán la administración y mejorarán la fiabilidad.

* Backend (Azure App Service): El backend desarrollado en Java 21 con Spring Boot se desplegará en Azure App Service, que ofrecerá soporte para despliegues continuos desde Github, escalado automático y gestión segura de variables de entorno y secretos. Esto permitirá realizar actualizaciones sin interrupciones y adaptarse a incrementos en el tráfico de cara a futuro.
* Frontend (Azure Static Web Apps): El frontend en Angular se alojará en Azure Static Web Apps, que está especialmente diseñado para servir contenido web estático de forma eficiente y segura.

Aunque Azure ofrece una red global de servidores CDN (Content Delivery Network) para distribuir el contenido, en este caso la configuración se centrará en optimizar la entrega para España, por lo que el contenido se almacenará en nodos CDN cercanos a España, minimizando la latencia y mejorando la velocidad de carga para los usuarios locales, garantizando una experiencia rápida y fluida, fundamental para mantener la satisfacción de los estudiantes y profesores que usen Duocoding.

* Base de datos: La base de datos MySQL se gestionará mediante Azure Database for MySQL Flexible Server, proporcionando alta disponibilidad, copias de seguridad automáticas, acceso seguro con SSL y firewall, y escalabilidad para ajustarse al volumen de usuarios.
* Dominio personalizado y certificados: El dominio personalizado se registrará y gestionará directamente desde Azure, con configuración automática de DNS y provisión de certificados SSL gratuitos que asegurarán la conexión y la confianza del usuario.

Accesibilidad de la app (web, móvil, etc).

7. Evaluación y resultados

8. Conclusiones

Tras la realización del Proyecto Fin de Máster, hemos podido llegar una serie de conclusiones sobre el trabajo realizado en las que podemos encontrar las deducciones comunes de los diferentes integrantes, aprendizajes adquiridos durante el desarrollo y posibles líneas de futuro cuando los usuarios hagan uso de ella, recibamos las primeras opiniones y posibles puntos de mejora.

1. **Herramienta complementaria para futuras ediciones del Máster**

La realización de este proyecto no solo nos ha servido para reforzar conocimientos académicos, sino también nos ha aportado un valor práctico debido al desarrollo de las distintas partes del proyecto. La aplicación que hemos desarrollado esta enfocada para que sea una herramienta complementaria en las futuras ediciones de este Máster, tanto para practicar código como para aprender conceptos teóricos sobre el mundo de la programación. El principal objetivo es fomentar un aprendizaje mas práctico para los futuros alumnos, ofreciendo distintos juegos de las distintas partes que componen una aplicación.

1. **Potencial de mejora en la jugabilidad**

A pesar de que la aplicación permite una jugabilidad 360º, pensamos que existen puntos de mejora como puede ser un mayor número de juegos, mejora de diseño mediante CSS u ofrecer una jugabilidad mas simplificada y entretenida mediante retos progresivos que aumente la satisfacción de los alumnos y realmente sientan ganas de hacer uso de ella.

1. **Aprendizaje integral del proceso de desarrollo de aplicaciones**

Una de las principales cosas que nos ha aportado el desarrollo de este proyecto ha sido conocer en primera persona todo lo que conlleva el desarrollo de una aplicación, conocer en profundidad la integración de las distintas partes, la funcionalidad de cada una de ellas, así como observar el resultado final de todo el trabajo. Este proyecto refleja todo el conocimiento adquirido durante estos meses de aprendizaje impartido en cada una de las asignaturas.

1. **Escalabilidad del proyecto**

Este proyecto se ha desarrollado alrededor de las principales herramientas utilizadas hoy en el mundo de la programación. También hemos tenido en cuenta su posible escalabilidad en un futuro, pudiendo desarrollar dentro de ella nuevas herramientas para que se mantenga actualizada en todo momento, mediante la integración de preguntas actualizadas o haciendo uso de las herramientas utilizadas en el mercado.

A pesar de que el proyecto se ha desarrollado pensando principalmente en los alumnos de las próximas ediciones del Máster, también hemos tenido en cuenta que puede escalar más allá de las clases, pudiendo ser usada por cualquier persona interesada en aprender, reforzar o, divertirse con la programación.

1. **Metodologías ágiles y trabajo en equipo**

Todo este trabajo lo hemos llevado a cabo haciendo uso de principios integrados en metodologías ágiles inspirados en algunas de ellas como Scrum, lo que nos ha permitido llevar a cabo un desarrollo organizado y funcional. Mediante reuniones diarias programadas, división del trabajo, asignación de roles y revisión del trabajo realizado hemos podido tomar mejores decisiones, adaptarnos mejor a los cambios surgidos durante el desarrollo, y hemos aprovechado mejor el tiempo disponible. Basarnos en principios sobre esta metodología también nos ha servido para trabajar en equipo, debatir distintos puntos de vista, y vivir una simulación de lo que puede ser un desarrollo real en un proyecto dentro de una empresa.

9. Referencias

* Fuentes académicas y técnicas
* Documentación de tecnologías utilizadas  
  Oracle. (2024). *MySQL 8.0 Reference Manual*.<https://dev.mysql.com/doc/>
* Angular Team. (2024). *Angular Docs (v17)*. Google LLC. <https://angular.io/docs>
* Microsoft. (2024). *TypeScript Handbook*. https://www.typescriptlang.org/docs/
* Spring Team. (2024). *Spring Boot Reference Documentation (v3.x)*. VMware.<https://docs.spring.io/spring-boot/docs/current/reference/html/>
* Spring Security Team. (2024). *Spring Security Reference Documentation (v6.x)*. VMware.<https://docs.spring.io/spring-security/reference/>
* Node.js Contributors. (2024). *Node.js v20.x Documentation*. OpenJS Foundation. <https://nodejs.org/en/docs>
* Swagger. (2024). *Swagger Tools and Documentation*. <https://swagger.io/tools/>
* Microsoft Corporation. (2024). *Azure Static Web Apps documentation*.<https://learn.microsoft.com/azure/static-web-apps/>
* Microsoft Corporation. (2024). *Azure App Service documentation*.<https://learn.microsoft.com/azure/app-service/>
* Microsoft Corporation. (2024). *Azure Database for MySQL documentation*.<https://learn.microsoft.com/azure/mysql/>
* GitHub, Inc. (2024). *GitHub Actions documentation*.<https://docs.github.com/en/actions>
* Angular Team. (2024). *Angular – One framework. Mobile & desktop*.<https://angular.io/>
* Spring.io. (2024). *Spring Boot reference documentation*.<https://spring.io/projects/spring-boot>

10. Anexos

Documentación adicional que queráis añadir (no es necesario)