uc3m | Universidad Carlos III de Madrid

Grado Universitario en Ingeniería Informática

2021-2022

Trabajo Fin de Grado

"APLICACIÓN WEB PARA APRENDER INGLÉS CON GESTIÓN Y CREACIÓN DE EJERCICIOS INTERACTIVOS"

Luis Alberto Rasquin Marra

Tutor:

Alejandro Rey López

Leganés, 23 de junio del 2022

RESUMEN

El proyecto que se presenta en este documento pretende proporcionar una herramienta, a los profesores particulares de inglés, con el fin de que esta les permita crear ejercicios personalizados para sus alumnos, y que estos últimos los puedan resolver. En el presente documento se indicarán los objetivos y motivos de la realización de este proyecto, además de estudiar las posibles soluciones tecnológicas de las que se disponen para la elaboración de esta herramienta, se definirán las funcionalidades de esta misma y cómo se podrá utilizar. Dicha herramienta, es una aplicación web, que está construida con React, Node, Express y MongoDB.

Finalmente, se culminará con conclusiones sobre el resultado obtenido y el futuro del trabajo planteado.

Palabras clave: Node, React, Express, MongoDB, API, REST, Desarrollo web, Clases de inglés, Ejercicios personalizados.

AGRADECIMIENTOS

En primer lugar, me gustaría dar las gracias a mis padres Luis y Antonina, y a mi pareja Andrea, por siempre aconsejarme y brindarme su máximo apoyo.

También agradecer a mi familia y amigos, por su apoyo.

Por último, agradecer a mi tutor Alejandro, por aceptar mi propuesta, guiarme y ayudarme a lo largo del proyecto.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

En el siguiente apartado se realizará una breve descripción de la motivación, detrás del desarrollo de este trabajo, así como los objetivos generales del mismo.

1.1. Motivación

Muchas personas toman la decisión de impartir clases particulares de inglés, por el hecho de que es un trabajo flexible, y se suele adaptar con mayor facilidad al horario del que estas disponen. Siendo este mi caso y el de muchos individuos de mi entorno, una dificultad notable a la hora de llevar a cabo esta tarea es que no abundan aplicaciones que se ajusten a las necesidades de estos profesores.

La falta de herramientas útiles para el desarrollo de esta actividad docente deja pocas opciones como alternativa. Entre las opciones disponibles se encuentran los editores de texto colaborativos como pueden ser Google Documents y Microsoft Word. El inconveniente de estas herramientas es que el proceso de compartir, acceder y manejar un documento puede resultar complicado para individuos con poca familiaridad con el uso de la tecnología, como pueden ser niños o personas mayores. Esto se presenta como un obstáculo para el proceso de adaptación a la época tecnológica en la cual nos encontramos, donde muchas de estas clases se imparten de manera online, especialmente tras la pandemia. También existen algunas páginas web que contienen ejercicios de inglés predefinidos o permiten crear preguntas, pero aun así es insuficiente, ya que no ofrecen una gran variedad de alternativas para generar clases dinámicas y personalizadas.

Por tanto, a los alumnos (especialmente aquellos muy jóvenes) el proceso de integrarlos y motivarlos al estudio es tedioso y complejo, debido a que no se cuenta con instrumentos versátiles y dinámicos para generar ejercicios y poder tomar constancia del desarrollo intelectual del alumnado.

A pesar de los avances tecnológicos, se siguen impartiendo clases de manera presencial, donde se emplean métodos tradicionales como lo es el papel y el lápiz. Sin embargo, esto conlleva a encontrarse con la dificultad de almacenar y llevar el seguimiento de las actividades asignadas, un problema que se agudiza aún más si cada profesor ha de gestionar a varios alumnos. El tiempo que conlleva la elaboración y corrección manual de los ejercicios, ralentiza el proceso de enseñanza, ya que puede resultar en distracción y desmotivación, que acaba afectando al instructor y al instruido.

Debido a estas razones, la elaboración del trabajo en cuestión tiene como principal motivo crear una herramienta valiosa que permita el provecho eficiente del tiempo que se tiene con el alumno. Apoyarse en la agilidad que brinda la tecnología resulta en la ganancia de minutos que, en contraste, se invierten en el beneficioso factor humano que trae consigo la enseñanza, y que conlleva a conseguir mejores resultados académicos para el aprendiz.

1.2. Objetivo del proyecto

Como objetivos principales de este proyecto, se tienen:

Objetivos principales:

- Desarrollar una aplicación con la capacidad de crear ejercicios de inglés variados, así como la gestión, compartición y automatización en la corrección de los mismos.
- Desarrollar una aplicación de fácil acceso y uso para que alumnos puedan resolver ejercicios personalizados de inglés.

Otros objetivos:

- Adquirir nuevos conocimientos y profundizar en los ya obtenidos, en base al desarrollo web, concretamente en tecnologías como React o Node.

1.3. Estructura del documento

El documento estará compuesto por 10 capítulos diferentes de la siguiente manera:

- Capítulo 1, Introducción: se expondrá los objetivos y motivos principales del desarrollo del proyecto en cuestión.
- Capítulo 2, Estado del arte: se estudiará la situación actual del entorno relativo al proyecto, así como las posibles soluciones que se puedan emplear. Se repasarán diferentes modelos de bases de datos, lenguajes de programación y metodologías de trabajo.
- Capítulo 3, Planteamiento del problema: definirá el alcance del proyecto, qué puede hacer la aplicación y cómo funciona, estableciendo los requisitos y casos de uso.

- Capítulo 4, Diseño de la solución: establecerá qué herramientas se seleccionaron para llevar a cabo el desarrollo, y definirá la arquitectura que tendrá la aplicación.

- Capítulo 5, Evaluación: determinará el conjunto de pruebas para comprobar el funcionamiento de la aplicación.

- Capítulo 6, Detalle de la solución: precisará las diferentes interfaces que posee la aplicación, mediante una descripción que incluirá las funcionalidades correspondientes a cada interfaz.

- Capítulo 7, Planificación: contendrá la metodología de trabajo seleccionada para el desarrollo del proyecto, como un diagrama de GANTT ilustrando la planificación del mismo.

- Capítulo 8, Marco regulador: repasará las leyes que ha de cumplir la aplicación en función de su comportamiento, y las licencias de las herramientas utilizadas en el proyecto.

- Capítulo 9, Entorno socioeconómico: definirá los costes de los recursos empleados para la realización del proyecto, tanto humanos como materiales, además de una descripción sobre el impacto que puede causar la aplicación en la sociedad.

- Capítulo 10, Conclusiones: para cerrar el documento, se expondrá los resultados conseguidos una vez haya culminado el proyecto, así como las pretensiones para el futuro del mismo.

1.4. Idea general

La idea de este proyecto es elaborar un portal web, donde el usuario, en este caso un profesor, pueda acceder al mismo mediante sus credenciales. En él podrá crear diferentes tipos de ejercicios personalizados, modificarlos y compartirlos con sus alumnos para que los puedan resolver.

Además, optará a una visión general de los resultados obtenidos por sus alumnos, facilitando así el seguimiento de los mismos. Por otro lado, el usuario que hace de alumno, podrá acceder al ejercicio mediante un enlace que le ha compartido su profesor sin la necesidad de registrarse en la plataforma, así agiliza y facilita el proceso de cara al alumno.

En resumen, se pretende desarrollar una herramienta que nos permita sustituir el típico libro de ejercicios que se utiliza en las clases tradicionales, por uno online, elaborado por el profesor, y adaptado al alumno.

2. ESTADO DEL ARTE

2.1. Soluciones Existentes

Hoy en día es posible encontrar un sinfín de aplicaciones y páginas web, capaces de realizar una variedad inmensa de tareas. Aplicaciones o páginas que sirven como herramientas de apoyo para impartir clases de un tema, también existen varias, pero en este caso se hará mención a algunas de las más conocidas:

Kahoot [1]: Es una herramienta muy popular, seguramente aún más tras la pandemia, en la cual se le permite a un usuario realizar un test de selección simple con un número de preguntas determinado. Los usuarios invitados pueden acceder a la sesión a la cual mediante un enlace, y dispondrán de cuatro botones para seleccionar la respuesta correcta, mientras que la pregunta aparece en la pantalla del usuario que ha creado la sesión. A la vez, durante cada pregunta de la sesión hay un tiempo asignado a cada pregunta y una puntuación acumulada tras cada pregunta, donde la persona que acierte de manera más rápida acumulará más puntos por pregunta.

Learn English British Council [2]: Es un portal web de British Council, donde se ofrecen lecciones sobre diferentes aspectos del inglés: vocabulario, gramática, lectura, etc. Hay diversos temas, y en cada uno de ellos se incluyen ejercicios básicos predeterminados con diferentes niveles de dificultad, para que el usuario pueda poner a prueba sus conocimientos.

Wooclap [3]: Es una plataforma conocida, la cual está orientada a presentaciones, pero aún así tiene apartados para crear ejercicios de tipo test, preguntas de asociación de elementos o nubes de palabras. Un usuario crea una sesión y crea el contenido de la misma, comparte un código con los usuarios a invitar, y estos se unen, pudiendo responder y participar en las actividades que ha determinado el usuario dueño de la sesión.

Editores de Texto: A la hora de escribir un ejercicio, siempre existe la posibilidad de utilizar un editor de texto. En el caso de ser colaborativo como lo es Google Documents, puede compartirse el documento con los ejercicios escritos para que otra persona los resuelva. También existe la posibilidad de utilizar aplicaciones como Google Meet para crear una conferencia, compartir pantalla y así mostrar al alumno los ejercicios escritos en algún editor.

2.2. Necesidad en el mercado

Actualmente a pesar de existir diferentes aplicaciones, como las mencionadas anteriormente, no hay una que se ajuste a las necesidades de los profesores de inglés, puesto que son poco flexibles y variadas. En el caso de Learn English British Council los ejercicios pueden resultar muy repetitivos y básicos. Wooclap y Kahoot, permiten realizar ejercicios de tipo test, lo cual limita al usuario a ese tipo de ejercicio únicamente. Por su parte, Wooclap es la que más variedad ofrece, ya que también tiene preguntas con respuesta abierta o asociación de elementos, pero aun así sigue siendo insuficiente para clases específicamente de inglés.

Por otro lado, está la opción de escribir ejercicios en un editor de texto. El proceso de compartir un documento o compartir la pantalla para mostrar un ejercicio, puede resultar lento y complicado, sobre todo en el caso de usuarios poco experimentados con estas aplicaciones.

Para un profesor de inglés, sería ideal disponer de una única herramienta que le permita diseñar diferentes tipos de ejercicios, con variedad de opciones, que él mismo pueda ajustar a las necesidades de su alumno, independientemente de la edad o nivel de inglés que tenga. Esto le permitirá crear una clase más dinámica y productiva.

2.3. Tecnologías

Una aplicación web puede tener el objetivo de mostrar información cuyo estado no varía, y a esto se le conoce como páginas web estáticas. Sin embargo, si nuestro objetivo es mostrar información que está contenida en una base de datos, es imprescindible contar con dos partes, el front-end y el back-end.

2.3.1. Front-End

El front-end es la parte de una aplicación web que se encarga de construir una interfaz gráfica para el usuario, de manera que la pueda ver e interactuar con ella. Lo podemos denominar como “la cara de la aplicación web”. Los lenguajes estándar para construir la interfaz de una web son HTML [4], CSS [5] y JavaScript [6]. HTML se utiliza para construir el esqueleto de la web, CSS para embellecerla y JavaScript para darle funcionalidad. Sin embargo, existen frameworks [7] que ayudan en esta parte del desarrollo. Algunos de los más comunes son:

React [8]: desarrollado por Facebook, es uno de los frameworks de JavaScript más utilizados en la actualidad. Utiliza JSX [9], que es una extensión de JavaScript para leer lenguajes de etiquetas como XML [10] o HTML, y basado en Babel [11], que es un

compilador capaz de convertir JSX en JavaScript [12]. Se utiliza para construir Single Page Apps [13], de manera dinámica.

Angular [14]: es una librería actual de JavaScript desarrollada por Google, que al igual que React se utiliza para crear Single Page Apps y de manera dinámica [15, 16]. Está desarrollada por Google.

Vue [17]: es una librería de JavaScript para construir interfaces de usuario, que a diferencia de las anteriores no está desarrollada por una gran empresa. Presta atención a lo que opinan los usuarios, y tarda más en publicar nuevas versiones [18, 19].

Todas estas tecnologías tienen el mismo principio, crear componentes que tienen un "estado" y cuando este estado cambia, el componente se vuelve a cargar para mostrar un cambio, sin la necesidad de recargar la página entera. Esto permite construir interfaces de usuario dinámicas de manera mucho más sencilla que si se hace con JavaScript puro. Además, permiten implementar lógica, donde indicar cuándo se tienen que realizar cambios en la interfaz o cuándo tiene que cargarse un componente.

2.3.2. Back-End

El back-end es como se denomina frecuentemente a todo lo relativo a la lógica de una aplicación web. En él se sitúa el código de una API [20], o el código para realizar la conexión a una base de datos, y cómo tratar la información que esta le suministra. Se le podría denominar como el "servidor" de una aplicación web. Para el desarrollo del mismo existen numerosas tecnologías, como pueden ser:

Node [21]: es una versión de JavaScript, que, en vez de ejecutarse en el navegador, es capaz de correr en una máquina [22]. Cuenta con una gran variedad de frameworks, entre ellos Express, una librería que facilita la creación de una API en un servidor de Node, facilitando la creación de endpoints [23] y poniendo a disposición diferentes herramientas para el middleware [24], lo que lo convierte en ideal para la creación de servidores de plataformas [25].

Django [26]: es un framework del lenguaje de programación Python [27], ideal para construir el servidor de una aplicación web, basado en el modelo-vista-controlador [28]. Además, cuenta con una gran variedad de características que hacen el desarrollo más fácil y rápido, como autenticación, seguridad, librerías fáciles de integrar, entre otros [29].

Spring Boot [30]: es un framework que permite al desarrollador crear aplicaciones Java Enterprise [31], simples de ejecutar y listas para poner en producción [32]. Forma parte de Spring [33], pero siendo una versión más sencilla, ya que nos facilita la creación de microservicios [34] y las configuraciones [35].

En el front-end, además de mostrar información estática, también nos puede interesar información dinámica. Para realizar esto último, existen dos maneras a las que se les conoce como “Client-Side Rendering” y “Server-Side Rendering.

2.3.3. Client-Side Rendering

Conocido como “renderizado del lado del cliente” en español, Client-Side Rendering consiste en realizar una petición desde el cliente (navegador web) a un servidor, y tras la respuesta del servidor, la aplicación que se está ejecutando en el lado del cliente, es la encargada de realizar el tratamiento apropiado de los datos para luego decidir cómo mostrarlos en la interfaz. En este caso, el lenguaje de programación que domina a la hora de construir este tipo de aplicaciones, es JavaScript. Sin embargo, este lenguaje dispone de diversos frameworks para facilitar el desarrollo. Algunos de los más conocidos son los mencionados anteriormente: React, Angular y Vue.

2.3.4. Server-Side Rendering

Conocido como “renderizado del lado del servidor” en español, Server-Side Rendering consiste en un procedimiento en el que el cliente (navegador web) envía una petición al servidor, y este último, tras recibirla, extrae los datos requeridos de la base datos, y a continuación se encarga de elaborar la página a mostrar al usuario, con la información estática y dinámica correspondiente, para enviárselo como respuesta de la petición al cliente. De esta manera, el cliente solo se tiene que encargar de mostrar la página por pantalla, que obtuvo como respuesta del servidor. Algunas tecnologías para ello son: JSP [36] o Ejs [37], que permiten trabajar con fragmentos estáticos de HTML (o “plantillas”) a las que se añade el código generado dinámicamente en el servidor.

2.4. Bases de datos

Existen dos tipos de bases de datos, la lógica más tradicional, como lo son las bases de datos relacionales, y una idea más nueva como lo son las bases de datos no relacionales.

2.4.1. Bases de datos relacionales

Este tipo de base de datos, parte de la idea del modelo relacional, el cual tiene como objetivo mantener la consistencia entre los diferentes datos. Las principales propiedades [38] de esta idea son:

- Atomicidad: asegura que, si se realiza un cambio, debe llevar a cabo la modificación completa o, no hacer nada.
- Consistencia: tras una transacción, asegura que los datos se encuentren en un estado correcto o válido.
- Aislamiento: asegura, que mientras una sesión está tratando unos datos, el resto de sesiones no tienen acceso a esas modificaciones, hasta que la sesión modificadora lo haga público.
- Durabilidad: una vez se complete una transacción y esta se ha confirmado, los datos permanecerán almacenados en la base de datos, incluso ante posibles fallos.

Las bases de datos relacionales, como lo indica su nombre, mantienen una relación lógica entre los datos. Los datos están estructurados en tablas, cuyas columnas contienen los diferentes atributos de un elemento. Estas tablas contienen claves primarias, que son únicas, para poder identificar registros en una tabla, y claves foráneas para relacionar un registro de una tabla con otro registro de otra tabla [39]. Veamos un ejemplo práctico: un cliente de una tienda web quiere comprar tres artículos y los añade a su carrito. Este carrito tendrá que referenciar al cliente que lo posee, y a los artículos que el cliente ha añadido a su carrito.

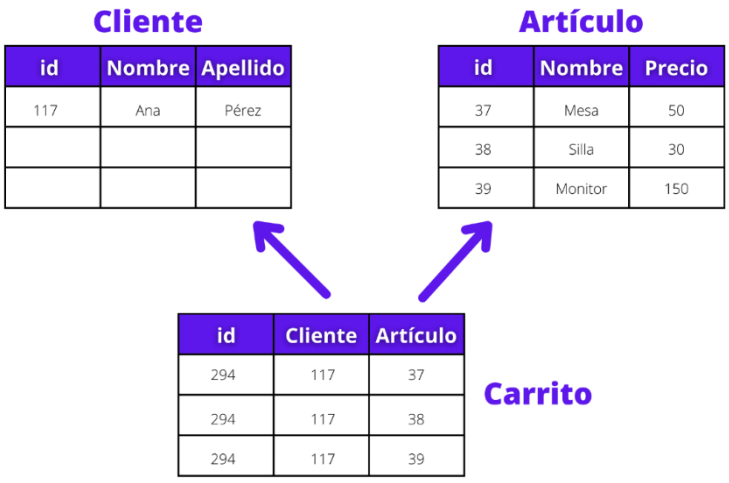


Fig. 1. Ilustración de bases de datos relacionales

Algunas de las bases de datos relacionales más populares son: Oracle, IBM Db2, MariaDB, MySQL, entre otras.

2.4.2. Bases de datos no relacionales

También conocidas como NoSQL, tiene un modelo libre de almacenamiento, ajustándose a las necesidades de la información que se está almacenando en ella. En este caso no suelen seguir un esquema, en el que se tenga que indicar qué se va a almacenar y en qué tipos de datos, aunque si se desea se puede hacer. El no seguir un esquema, proporciona mayor flexibilidad a la hora de almacenar datos, ya que cada objeto que se guarde puede tener unos atributos diferentes a otro, o menos atributos que otro [40]. Las bases de datos no relaciones siguen el teorema CAP cuyas propiedades [41, 42] son:

- Consistencia: al insertar o consultar la base de datos, se debe obtener la misma información o datos, independientemente del nodo al que se accedió.
- Disponibilidad: aunque se haya caído el sistema, los usuarios pueden seguir leyendo o escribiendo en la base de datos.
- Tolerancia a particiones: al ser un sistema distribuido, es probable que esté dividido en particiones, por lo que el sistema tiene que continuar su funcionamiento, aunque algunos nodos no estén operativos.

Sin embargo, como estas bases de datos son escalables y distribuidas, no pueden seguir los tres principios, por lo que cada solución opta por dos de estos tres principios.

El enfoque que sigue una base de datos NoSQL, puede variar, puede estar orientada a documentos, a grafos, a columnas o almacenamiento de clave-valor. Uno de los sistemas más comerciales es MongoDB, el cual sigue un enfoque orientado al almacenamiento de documentos. Un ejemplo, para este sistema, sería:

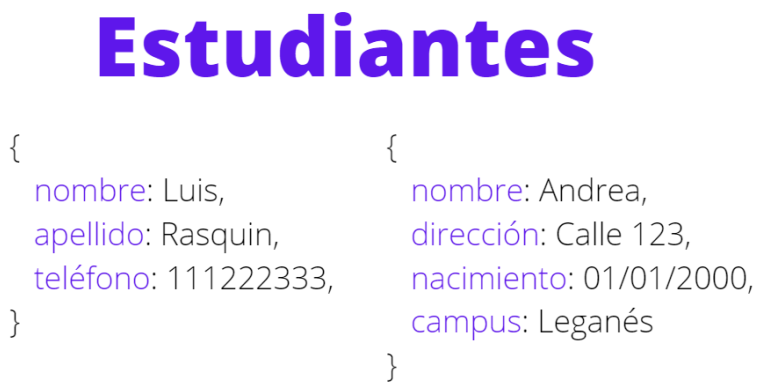


Fig. 2. Ilustración bases de datos no relacionales

Tal y como se puede apreciar en la Fig. 2, nos permite guardar dos estudiantes distintos, con diferentes atributos y cantidad de los mismos.

Además de MongoDB, existen otras bases de datos NoSQL como lo son: Cassandra, Redis, Amazon DynamoDB, Neo4j o Google BigTable.

2.4.3. Comparación

En primer lugar, las bases de datos relacionales siguen un esquema tabular, definiendo de manera cerrada el número de atributos y aquellos que determinan a un elemento. Mientras que, en NoSQL, disponemos de una flexibilidad que nos permite guardar elementos con un número variable de atributos, sin la obligación de indicar todos los atributos definidos previamente. Además, en lugar del esquema tabular, puede variar su orientación entre clave-valor, documentos, grafos o columnas.

Las bases de datos relacionales utilizan el lenguaje SQL (structured query language, “lenguaje de consulta estructurada” en español). Puede que existan pequeñas diferencias en la sintaxis de los diferentes motores SQL, pero en rasgos generales son iguales. Para el caso de NoSQL, la sintaxis para realizar las consultas varía entre las diferentes bases de datos.

NoSQL escala horizontalmente, lo cual implica que puede mejorar su rendimiento al distribuir la información entre varios servidores. En el caso de las relacionales escalan verticalmente, lo que significa que su rendimiento mejorará tras potenciar elementos del servidor como su CPU, memoria RAM o disco.

Las relacionales están basadas en ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad), pero en el caso de NoSQL se basan en el teorema CAP (Consistencia, Disponibilidad y Tolerancia a particiones) [43].

2.5. Metodologías de trabajo tradicionales y ágiles

En el desarrollo de software, se denomina como metodologías de trabajo, al conjunto de técnicas y métodos que se utilizan para organizar a un equipo frente al diseño y desarrollo de una solución de software. Esto ayuda a los integrantes del equipo a ser más eficientes a la hora de emplear su tiempo y realizar una tarea [44]. Dentro de estas metodologías podemos diferenciar dos tipos: tradicionales y ágiles.

2.5.1. Metodologías Tradicionales

Entre las metodologías tradicionales disponemos de las siguientes:

Cascada

Consiste en una secuencia de actividades a llevar a cabo durante el proceso de desarrollo, siempre respetando el orden de las diferentes etapas. Las principales etapas son: elaboración de requisitos, diseño, implementación, verificación y mantenimiento. Es una metodología bastante rígida, y en la cual no suelen variar los requisitos iniciales. Nos permite encontrar errores en las primeras etapas del proyecto [44].

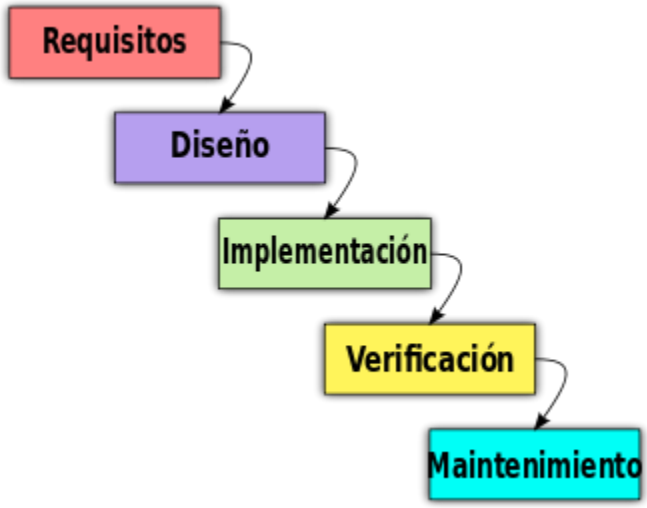


Fig. 3. Modelo en cascada; fuente: [68]

Incremental

Está metodología nos permite ir construyendo el producto final, sobre la marcha. Se parte de una versión básica, y en cada etapa se van agregando nuevas funcionalidades. Se puede hacer uso del software sin la necesidad de que llegue a su estado final [44]. Es una versión reducida y más flexible del modelo en cascada, cada iteración es un mini modelo en cascada.

Espiral

Toma en consideración el análisis del riesgo. Las diferentes actividades se conforman en una espiral, donde cada iteración representa una nueva versión. Las cuatro actividades principales son: planificación, análisis de riesgo, desarrollo y evaluación por parte del cliente. Para pasar a la siguiente actividad, se debe haber finalizado la actual. Mientras más iteraciones, mayor complejidad tendrá el proyecto y más definido estará [45] [44].

Fig. 4. Modelo en espiral; fuente: [69]

Prototipo evolutivo

Se parte de una idea inicial sobre la cual se construye un prototipo. Hay 4 etapas claves: idea inicial, construcción del prototipo, revisión y corrección del prototipo y producto final. Sobre todo, destaca la construcción y revisión y corrección del prototipo, ya que son en esas etapas donde se discute qué está mal o qué se desea agregar, y se corrige o añade. Una vez el prototipo esté lo suficientemente avanzado, llegará a su versión final y se entregará [46] [44].

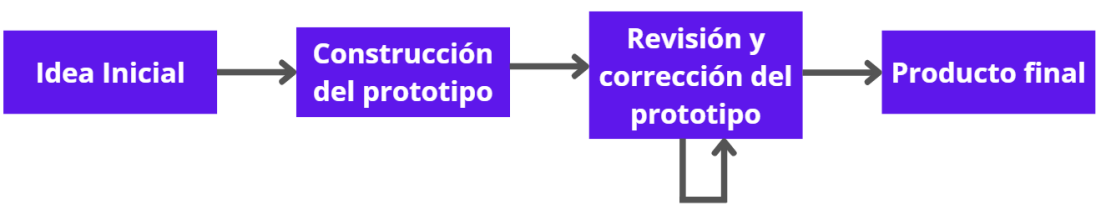


Fig. 5. Modelo prototipo evolutivo

2.5.2. Metodologías Ágiles

Dentro de las metodologías ágiles contamos con los siguientes modelos:

Kanban

Esta metodología divide el trabajo en pequeñas tareas, y las organiza en un tablero con las columnas “listo”, “en proceso” y “pendiente”. Esto ayuda a visualizar el estado del proyecto, la carga de trabajo y las prioridades [47] [44].

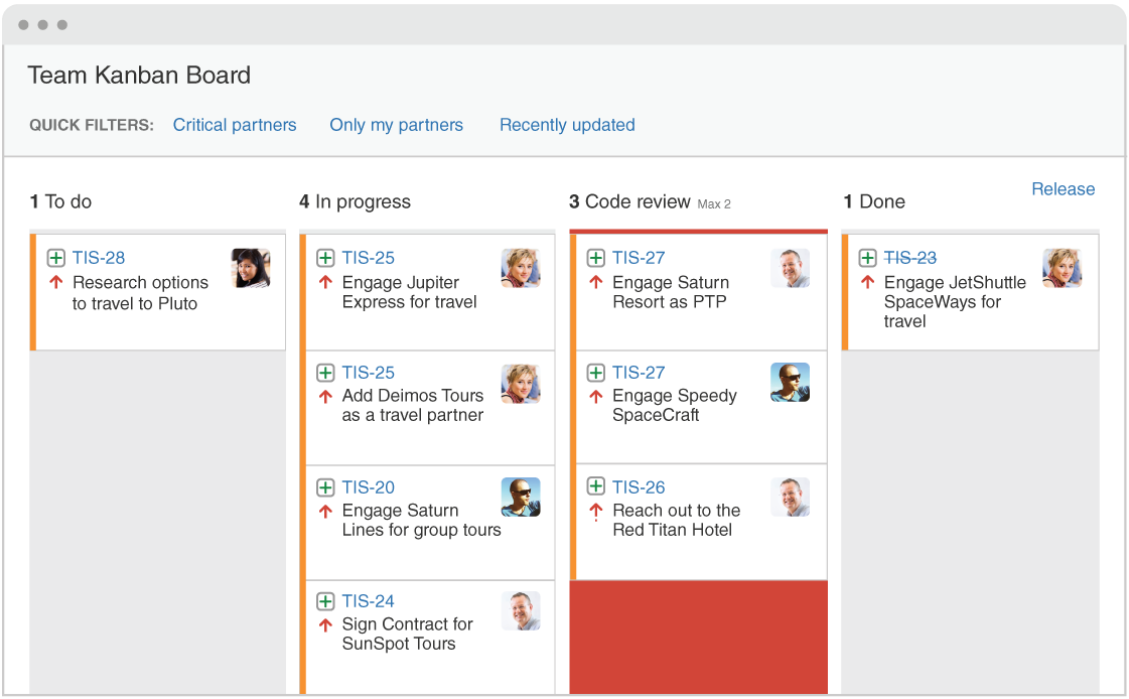


Fig. 6. Tablero Kanban; fuente: [47]

Scrum

Sigue la filosofía de la metodología de Kanban, pero con iteraciones de dos a cuatro semanas. A estas iteraciones también se les conoce como “sprints”. Las principales fases de un sprint son: planificación del sprint, ejecución, reunión diaria y revisión de los resultados. La idea es ir avanzando en base a los resultados obtenidos en el sprint anterior [48] [44].

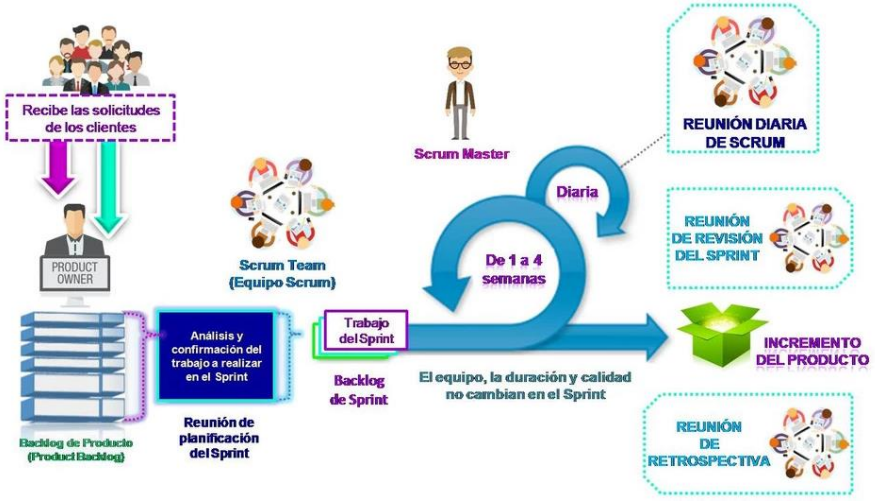


Fig. 7. Metodología scrum; fuente: [70]

XP (Programación Extrema)

Está basado en el feedback constante y el trabajo en equipo, pero sobre todo en la simplicidad. Entre sus características principales tenemos: desarrollo iterativo, pruebas de manera continua, refactorización del código, corrección de errores regularmente y programación en parejas, entre otras [44].

Fig. 8. Metodología programación extrema; fuente: [70]

2.6. Resumen del estado del arte

A lo largo de este capítulo se ha estudiado y detallado la situación actual del entorno para el desarrollo de este proyecto. Primero se han visto las alternativas que existen hoy en día en el mercado, a la aplicación que se desea desarrollar (apartado 2.1). Posteriormente, se detalló cuál es la necesidad que existe en el mercado y cómo esta aplicación puede saciar dicha necesidad (apartado 2.2).

Luego, se procedió a estudiar un pilar del proyecto, el entorno tecnológico. En primer lugar, se definió qué es el Front-End y algunas de las tecnologías más famosas para su desarrollo, como son: React, Angular y Vue (apartado 2.3.1). En segundo lugar, se definió qué es el Back-End y se mencionaron algunas de las tecnologías más conocidas, como son: Node, Django y Spring Boot (apartado 2.3.2). Posteriormente se define Client-Side Rendering, mencionando algunas de sus tecnologías, como son: React, Angular y Vue (apartado 2.3.3), y Server-Side Rendering, con tecnologías como: EJS y JSP (apartado 2.3.4). Por último, se procede a hablar de los diferentes tipos de bases de datos, las relacionales como pueden ser: Oracle, MariaDB o MySQL (apartado 2.4.1), y las no relacionales, como pueden ser: MongoDB, Redis o Cassandra (apartado 2.4.2), para finalmente hacer una comparación entre ambos tipos (apartado 2.4.3).

En la última sección de este capítulo, se procede a hablar de las diferentes metodologías (apartado 2.5) de trabajo para proyectos de desarrollo de software. En primer lugar, están las metodologías tradicionales como: cascada, espiral o prototipo evolutivo (apartado 2.5.1), y en otro lugar las metodologías ágiles, como son: SCRUM o Kanban (apartado 2.5.2).

3. PLANTEAMIENTO DEL PROBLEMA

3.1. Alcance de la solución

La aplicación detallada a continuación, tiene como objetivo facilitar la creación y compartición de ejercicios de inglés. Para poder definir las funcionalidades de las que va a disponer, primero se deben definir los diferentes tipos de usuario que harán uso de ella. Se va a distinguir entre los siguientes tipos de usuario:

- Profesor: el profesor es el tipo de usuario que accede a la aplicación por iniciativa propia y hace un uso avanzado de ella. Deberá poseer una cuenta para acceder a la aplicación. Una vez dentro, dispondrá de un panel donde podrá crear ejercicios, ver todos los ejercicios que ha creado, editarlos y compartirlos.

- Alumno: el alumno está representado por todo aquel individuo que accede a la aplicación mediante el enlace de un ejercicio que le haya compartido un profesor. Así podrá resolver dicho ejercicio.

3.2. Casos de Uso

A continuación, se mostrará un diagrama UML [49] en el cual se indican los casos de usos para esta aplicación. Estos representarán las acciones que pueden realizar los diferentes tipos de usuario de manera general, para brindar una mayor comprensión del funcionamiento de la aplicación. Se pueden observar las relaciones que existen entre los diferentes casos de uso, “include” y “extend”.

Por un lado, las relaciones “include” indican que existe una dependencia con un caso de uso base, por lo que si se ejecuta la acción del caso de uso base, los casos de usos “include” relativos a este también lo harán. Por otro lado, las relaciones “extend” implican que si un caso de uso base se ejecuta, un caso de uso “extend” relativo a este, se ejecutará solo en algunos casos.

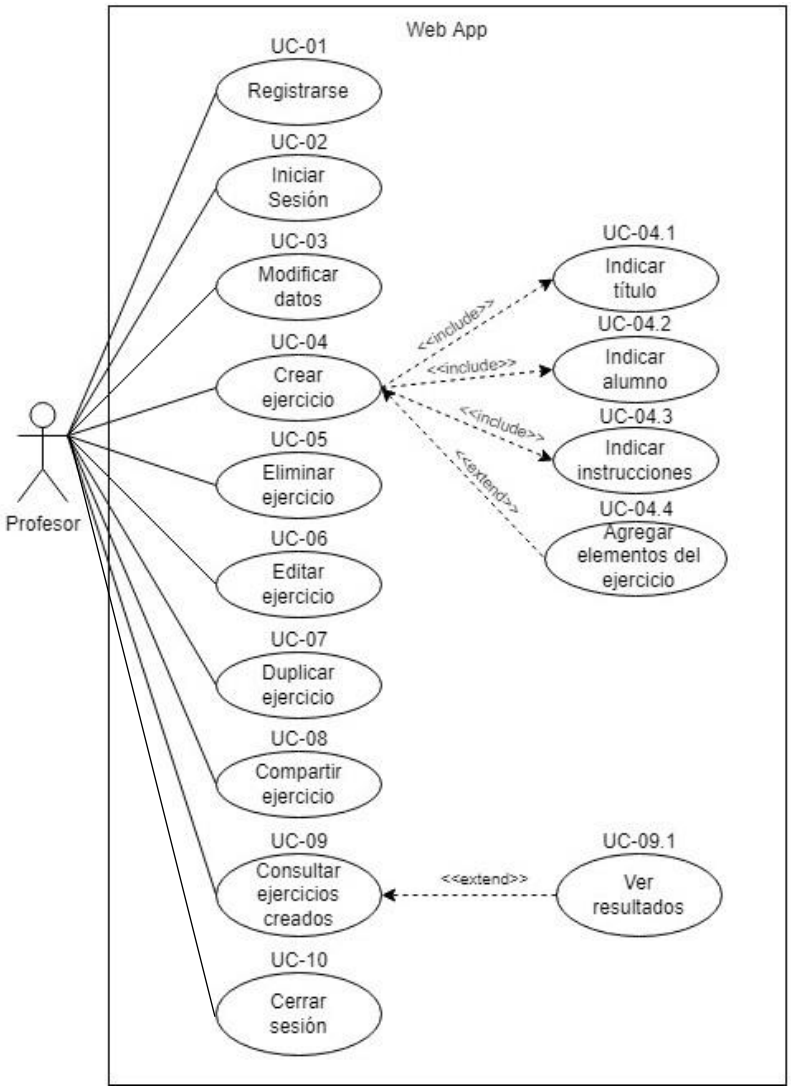


Fig. 9. Diagrama casos de uso para profesor

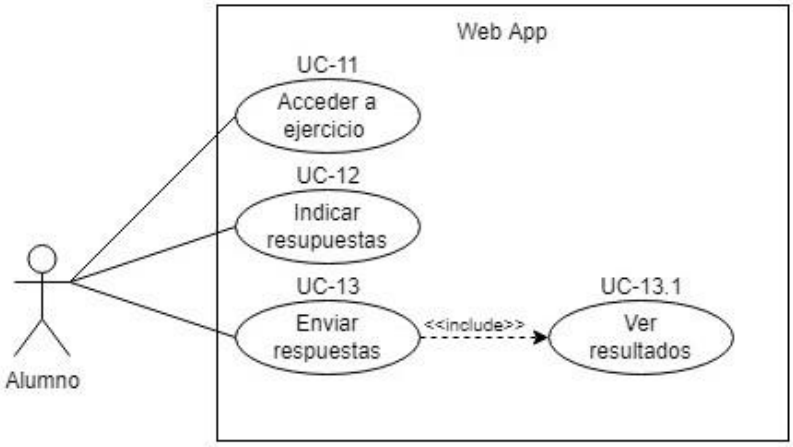


Fig. 10. Diagrama casos de uso para alumno

3.3. Requisitos

A la hora de definir requisitos, debemos diferenciar entre dos tipos: requisitos de usuario y requisitos de software. Los requisitos de usuario se centran en definir qué puede hacer el usuario con el software. Los requisitos de software indican que debe hacer el sistema y qué restricciones tiene, por ello se dividen en funcionales y no funcionales respectivamente.

3.3.1. Plantilla

La plantilla que se va a utilizar para la definición de los requisitos es la siguiente:

RX-NN	
ID	NN
Tipo	{Usuario, Funcional, No Funcional}
Descripción	Texto
Prioridad	{Baja, Media, Alta}

Tabla 1. Plantilla de requisitos

- Título: será el nombre del requisito, donde R significa requisito, X tendrá un valor del conjunto {U, S} que determinará si es de usuario o de software, y NN que representa el número identificativo del requisito.
- ID: NN representa el número identificativo del requisito.
- Tipo: indica qué tipo de requisito es. Tendrá el valor usuario si es un requisito de usuario, funcional si es de software funcional y no funcional si es de software no funcional.
- Descripción: texto donde se explica el requisito.
- Prioridad: Indica el nivel de prioridad.

3.3.2. Requisitos de usuario

RU-01	
ID	01
Tipo	Usuario
Descripción	El profesor podrá registrarse en la plataforma.
Prioridad	Alta

Tabla 2. RU-01

RU-02	
ID	02
Tipo	Usuario
Descripción	El profesor podrá iniciar sesión en la plataforma.
Prioridad	Alta

Tabla 3. RU-02

RU-03	
ID	03
Tipo	Usuario
Descripción	El profesor podrá modificar sus datos.
Prioridad	Alta

Tabla 4. RU-03

RU-04	
ID	04
Tipo	Usuario
Descripción	El profesor podrá crear un ejercicio en la plataforma.
Prioridad	Alta

Tabla 5. RU-04

RU-05	
ID	05
Tipo	Usuario
Descripción	El profesor podrá eliminar un ejercicio.
Prioridad	Alta

Tabla 6. RU-05

RU-06	
ID	06
Tipo	Usuario
Descripción	El profesor podrá modificar los datos de un ejercicio.
Prioridad	Alta

Tabla 7. RU-06

RU-07	
ID	07
Tipo	Usuario
Descripción	El profesor podrá duplicar un ejercicio.
Prioridad	Alta

Tabla 8. RU-07

RU-08	
ID	08
Tipo	Usuario
Descripción	El profesor podrá compartir un ejercicio mediante un enlace.
Prioridad	Alta

Tabla 9. RU-08

RU-09	
ID	09
Tipo	Usuario
Descripción	El profesor podrá visualizar sus ejercicios creados.
Prioridad	Alta

Tabla 10. RU-09

RU-10	
ID	10
Tipo	Usuario
Descripción	El profesor deberá poder ver los resultados obtenidos por el alumno para cada ejercicio.
Prioridad	Alta

Tabla 11. RU-10

RU-11	
ID	11
Tipo	Usuario
Descripción	El alumno que posea un enlace podrá resolver un ejercicio.
Prioridad	Alta

Tabla 12. RU-11

RU-12	
ID	12
Tipo	Usuario
Descripción	El alumno podrá enviar sus respuestas para un ejercicio.
Prioridad	Alta

Tabla 13. RU-12

RU-13	
ID	13
Tipo	Usuario
Descripción	El alumno podrá ver si su respuesta es correcta para cada pregunta, tras terminar el ejercicio.
Prioridad	Alta

Tabla 14. RU-13

RU-14	
ID	14
Tipo	Usuario
Descripción	El alumno podrá ver si su respuesta es incorrecta para cada pregunta, tras terminar el ejercicio.
Prioridad	Alta

Tabla 15. RU-14

RU-15	
ID	15
Tipo	Usuario
Descripción	El alumno podrá ver todas las respuestas correctas del ejercicio, tras haberlo terminado.
Prioridad	Alta

Tabla 16. RU-15

RU-16	
ID	16
Tipo	Usuario
Descripción	El alumno podrá ver la puntuación obtenida tras terminar el ejercicio.
Prioridad	Alta

Tabla 17. RU-16

RU-17	
ID	17
Tipo	Usuario
Descripción	El alumno podrá ver la nota final del ejercicio tras haberlo terminado.
Prioridad	Alta

Tabla 18. RU-17

RU-18	
ID	18
Tipo	Usuario
Descripción	El profesor podrá cerrar sesión.
Prioridad	Alta

Tabla 19. RU-18

3.3.3. Requisitos de software

RS-01	
ID	01
Tipo	Funcional
Descripción	El sistema permitirá a un profesor registrarse en la plataforma.
Prioridad	Alta

Tabla 20. RS-01

RS-02	
ID	02
Tipo	Funcional
Descripción	El sistema permitirá a un profesor iniciar sesión en la plataforma.
Prioridad	Alta

Tabla 21. RS-02

RS-03	
ID	03
Tipo	Funcional
Descripción	El sistema permitirá a un profesor modificar sus datos.
Prioridad	Alta

Tabla 22. RS-03

RS-04	
ID	04
Tipo	Funcional
Descripción	El sistema permitirá a un profesor crear un ejercicio.
Prioridad	Alta

Tabla 23. RS-04

RS-05	
ID	05
Tipo	Funcional
Descripción	El sistema mostrará por pantalla los elementos correspondientes al tipo de ejercicio que se crea.
Prioridad	Alta

Tabla 24. RS-05

RS-06	
ID	06
Tipo	Funcional
Descripción	El sistema permitirá a un profesor eliminar un ejercicio.
Prioridad	Alta

Tabla 25. RS-06

RS-07	
ID	07
Tipo	Funcional
Descripción	El sistema permitirá a un profesor modificar los datos de un ejercicio.
Prioridad	Alta

Tabla 26. RS-07

RS-08	
ID	08
Tipo	Funcional
Descripción	El sistema permitirá a un profesor duplicar un ejercicio.
Prioridad	Alta

Tabla 27. RS-08

RS-09	
ID	09
Tipo	Funcional
Descripción	El sistema generará un enlace único por ejercicio para que este pueda ser compartido.
Prioridad	Alta

Tabla 28. RS-09

RS-10	
ID	10
Tipo	Funcional
Descripción	El sistema mostrará los ejercicios creados por un profesor.
Prioridad	Alta

Tabla 29. RS-10

RS-11	
ID	11
Tipo	Funcional
Descripción	El sistema mostrará los resultados obtenidos por el alumno para cada ejercicio.
Prioridad	Alta

Tabla 30. RS-11

RS-12	
ID	12
Tipo	Funcional
Descripción	El sistema mostrará el ejercicio listo para ser resuelto si se accede al ejercicio mediante un enlace.
Prioridad	Alta

Tabla 31. RS-12

RS-13	
ID	13
Tipo	Funcional
Descripción	El sistema deberá guardar la resolución de un ejercicio enviada por un alumno.
Prioridad	Alta

Tabla 32. RS-13

RS-14	
ID	14
Tipo	Funcional
Descripción	El sistema mostrará si una respuesta es correcta, una vez se haya enviado la resolución del ejercicio.
Prioridad	Alta

Tabla 33. RS-14

RS-15	
ID	15
Tipo	Funcional
Descripción	El sistema mostrará si una respuesta es incorrecta, una vez se haya enviado la resolución del ejercicio.
Prioridad	Alta

Tabla 34. RS-15

RS-16	
ID	16
Tipo	Funcional
Descripción	El sistema mostrará las respuestas correctas, una vez se haya enviado la resolución del ejercicio.
Prioridad	Alta

Tabla 35. RS-16

RS-17	
ID	17
Tipo	Funcional
Descripción	El sistema mostrará la puntuación obtenida por el alumno, una vez se haya enviado la resolución del ejercicio.
Prioridad	Alta

Tabla 36. RS-17

RS-18	
ID	18
Tipo	Funcional
Descripción	El sistema mostrará la nota final del ejercicio obtenida por el alumno, una vez se haya enviado la resolución del ejercicio.
Prioridad	Alta

Tabla 37. RS-18

RS-19	
ID	19
Tipo	Funcional
Descripción	El sistema comprobará que las credenciales son correctas para iniciar sesión.
Prioridad	Alta

Tabla 38. RS-19

RS-20	
ID	20
Tipo	Funcional
Descripción	El sistema permitirá a un profesor cerrar su sesión en la plataforma.
Prioridad	Alta

Tabla 39. RS-20

RS-21	
ID	21
Tipo	No Funcional
Descripción	El formulario de registro contendrá los siguientes campos: -Nombre -Apellido -Correo (obligatorio) -Contraseña (obligatorio)
Prioridad	Alta

Tabla 40. RS-21

RS-22	
ID	22
Tipo	No Funcional
Descripción	El formulario para iniciar sesión contendrá los siguientes campos: -Correo (obligatorio) -Contraseña (obligatorio)
Prioridad	Alta

Tabla 41. RS-22

RS-23	
ID	23
Tipo	No Funcional
Descripción	El formulario para crear un ejercicio contendrá los siguientes campos: -Título (obligatorio) -Tipo (obligatorio) -Nombre del alumno (obligatorio) -Instrucciones (obligatorio) -Elementos relativos al tipo de ejercicio.
Prioridad	Alta

Tabla 42. RS-23

3.4. Matriz de trazabilidad: Requisitos de Usuario vs Requisitos de Software

		REQUISITOS DE USUARIO																	
		RUS-01	RUS-02	RUS-03	RUS-04	RUS-05	RUS-06	RUS-07	RUS-08	RUS-09	RUS-10	RUS-11	RUS-12	RUS-13	RUS-14	RUS-15	RUS-16	RUS-17	RUS-18
REQUISITOS DE SOFTWARE	RS-01	x																	
	RS-02		x																
	RS-03			x															
	RS-04				x														
	RS-05				x														
	RS-06					x													
	RS-07						x												
	RS-08							x											
	RS-09								x										
	RS-10									x									
	RS-11										x								
	RS-12											x							
	RS-13												x						
	RS-14													x					
	RS-15														x				
	RS-16															x			
	RS-17																x		
	RS-18																	x	
	RS-19		x																
	RS-20																		x
	RS-21	x																	
	RS-22		x																
	RS-23				x														

Fig. 11. Matriz de trazabilidad RS vs RU

3.5. Matriz de trazabilidad: Requisitos de Software vs. Casos de Uso

		CASOS DE USO												
		CU-01	CU-02	CU-03	CU-04	CU-05	CU-06	CU-07	CU-08	CU-09	CU-10	CU-11	CU-12	CU-13
REQUISITOS DE SOFTWARE	RS-01	x												
	RS-02		x											
	RS-03			x										
	RS-04				x									
	RS-05				x									
	RS-06					x								
	RS-07						x							
	RS-08							x						
	RS-09								x					
	RS-10									x				
	RS-11									x				
	RS-12											x	x	
	RS-13													x
	RS-14													x
	RS-15													x
	RS-16													x
	RS-17													x
	RS-18													x
	RS-19		x											
	RS-20										x			
	RS-21	x												
	RS-22		x											
	RS-23				x									

Fig. 12. Matriz de trazabilidad RS vs CU

4. DISEÑO DE LA SOLUCIÓN

4.1.Elección de Tecnologías

Para el desarrollo de la aplicación, se considera que la mejor solución es utilizar el "MERN Stack". Así es como se le conoce al conjunto de tecnologías formado por MongoDB, Express, React y Node.

La principal razón detrás de esta elección se debe a que son un conjunto de tecnologías que trabajan muy bien entre sí, ya que todas ellas están unificadas por el mismo lenguaje de programación: JavaScript. Esto facilita el proceso a la hora de programar si el desarrollador posee conocimientos de este lenguaje (como es el caso). Respecto a la base datos, MongoDB está basada en documentos con formato JSON [50] "JavaScript Object Notation", es decir en objetos del lenguaje de JavaScript, además de que permite incluir código de dicho lenguaje a la hora de realizar consultas.

Por otro lado, Node cuenta con una librería para MongoDB, denominada Mongoose [51], que permite hacer uso del driver de MongoDB para Node de manera más sencilla, y permite estructurar mejor la información a almacenar en la base de datos, sin llegar a ser tan rígida como lo son las bases de datos relacionales.

Aparte de lo anteriormente mencionado, se considera que MongoDB, es una excelente solución para este desarrollo debido a la flexibilidad que ofrece. Esto último es ideal por el hecho de que, a la hora de crear un ejercicio, podrá ser almacenado sin importar el número de elementos que este contenga. Por ejemplo, un ejercicio con 10 preguntas será almacenado al igual que lo será un ejercicio con 20 preguntas.

Por otra parte, React es la tecnología más popular a la hora de construir interfaces de usuarios interactivas en la actualidad, por lo que el desarrollo de este proyecto es una oportunidad para profundizar en ella. Cabe destacar que esta tecnología está respaldada por una gran compañía como lo es Facebook, está en constante evolución y al ser tan popular cuenta con una gran comunidad. Por último, mencionar que existe una previa y pequeña experiencia, por parte del autor, en el uso de React en comparación con las otras opciones.

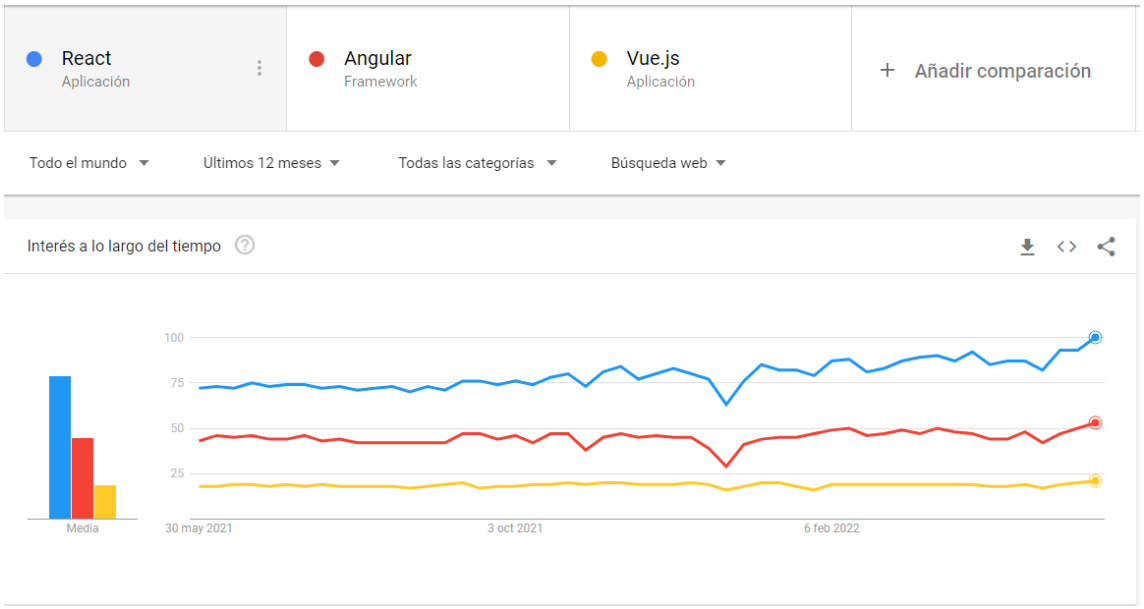


Fig. 13. Gráfica React vs Angular vs Vue; fuente: [71]

En la Fig. 13, se ve representado en una escala del 0 al 100 el interés de cada término (a mayor el valor, mayor el interés y viceversa), basado en búsquedas de todo el mundo datadas en los últimos 12 meses. En la misma, se percibe claramente una mayor inclinación por React en comparación con los otros términos.

4.2. Arquitectura

Finalmente, la arquitectura que seguirá la aplicación es el modelo cliente-servidor [52] de la siguiente manera: optar por client-side rendering utilizando React en el lado del cliente. Desde React se realizarán peticiones HTTP [53] al servidor construido en Node y Express. Este servidor contendrá una API REST [54] con diferentes endpoints, los cuales devolverán el conjunto de datos correspondiente a la petición en formato JSON. Para que el servidor sea capaz de devolver los datos correspondientes, antes de enviar la respuesta, se conectará a la base de datos en MongoDB y extraerá la información mediante la consulta más adecuada.

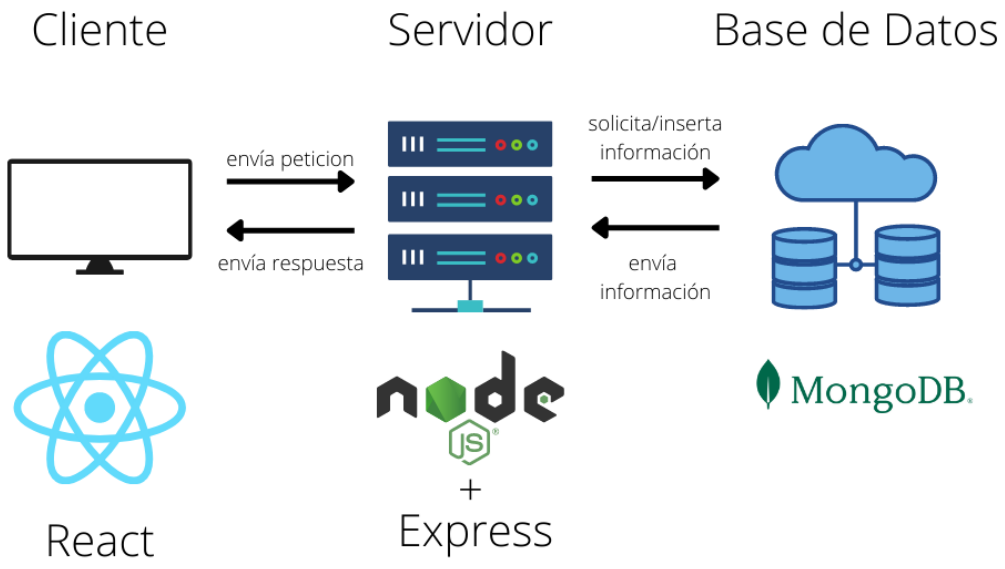


Fig. 14. Diagrama de la arquitectura de la aplicación

4.2.1. Plantilla endpoints

La plantilla para definir los diferentes endpoints será la siguiente:

"Método"	**"Endpoint"**
Descripción	Texto
Parámetros en URL	Lista de parámetros
Body	Lista de parámetros
Códigos de respuesta	Lista de códigos de respuesta

Tabla 43. Plantilla Endpoints.

- Método: será sustituido por el método http del endpoint. {GET, PUT, POST, DELETE}
- Endpoint: será sustituido por la ruta del endpoint.
- Descripción: contiene una breve descripción del propósito del endpoint.
- Parámetros en URL: indicará los parámetros de la petición que se envían a través de la URL.

- Body: indicará los parámetros de la petición que se envían a través del cuerpo de la petición.
- Códigos de respuesta: indicará el código de respuesta http para la petición recibida, en el caso correcto y en el caso de error, acompañados de un breve mensaje.

4.2.2. Endpoints

La API REST alojada en el servidor deberá contener los siguientes endpoints:

POST	**/users**
Descripción	Registra un usuario en la base de datos.
Parámetros en URL	Ninguno
Body	• Nombre: nombre del usuario. • Apellido: apellido del usuario. • Correo: correo del usuario. • Contraseña: contraseña del usuario.
Códigos de respuesta	• 201. User registered. • 400. Bad request. • 409. Conflict. User already exists.

Tabla 44. Endpoint 1

POST	/users/login
Descripción	Comprueba la existencia de un usuario y que sus credenciales son correctas, para concederle acceso a la aplicación.
Parámetros en URL	Ninguno
Body	• Correo: correo del usuario. • Contraseña: contraseña del usuario.
Códigos de respuesta	• 200. Ok. • 400. Bad request.

Tabla 45. Endpoint 2

PUT	/users
Descripción	Actualiza los datos de un usuario.
Parámetros en URL	Ninguno
Body	• Id: id del usuario. • Nombre: nombre del usuario. • Apellido: apellido del usuario. • Correo: correo del usuario. • Contraseña: contraseña del usuario.
Códigos de respuesta	• 200. Ok. • 400. Bad request. • 404. User not found.

Tabla 46. Endpoint 3

GET	/users/{id}
Descripción	Busca un usuario y devuelve sus datos.
Parámetros en URL	• Id: id del usuario.
Body	Ninguno
Códigos de respuesta	• 200. Ok. • 400. Bad request. • 404. User not found.

Tabla 47. Endpoint 4

GET	/users/{id}/students
Descripción	Busca un usuario y devuelve sus estudiantes.
Parámetros en URL	• Id: id del usuario.
Body	Ninguno
Códigos de respuesta	• 200. Ok. • 400. Bad request. • 404. User not found.

Tabla 48. Endpoint 5

POST	/exercises
Descripción	Crea un ejercicio en la base de datos.
Parámetros en URL	Ninguno
Body	• User: id del usuario que crea el ejercicio. • Title: título del ejercicio. • Type: tipo del ejercicio. • Student: nombre del alumno. • Instructions: instrucciones del ejercicio. • Sentences: preguntas del ejercicio. • Answers: respuestas del profesor para el ejercicio. • Nanswers: lista con el número de respuestas para cada pregunta. • Result: puntuación obtenida por el alumno.
Códigos de respuesta	• 201. Exercise created. • 400. Bad request.

Tabla 49. Endpoint 6

POST	/exercises/duplicate
Descripción	Genera una copia de un ejercicio previamente creado.
Parámetros en URL	Ninguno
Body	• id
Códigos de respuesta	• 201. Exercise duplicated. • 404. Exercise not found.

Tabla 50. Endpoint 7

PUT	/exercises
Descripción	Actualiza los datos de un ejercicio.
Parámetros en URL	Ninguno
Body	• Id: id del ejercicio • User: id del usuario que crea el ejercicio. • Title: título del ejercicio. • Type: tipo del ejercicio. • Student: nombre del alumno. • Instructions: instrucciones del ejercicio. • Sentences: preguntas del ejercicio. • Answers: respuestas del profesor para el ejercicio. • Nanswers: lista con el número de respuestas para cada pregunta. • Result: puntuación obtenida por el alumno.
Códigos de respuesta	• 200. Ok. • 400. Bad request. • 404. Exercise not found.

Tabla 51. Endpoint 8

PUT	/exercises/solution
Descripción	Almacena la solución enviada por un alumno para un ejercicio previamente creado.
Parámetros en URL	Ninguno
Body	• Id: id del ejercicio. • Result: puntuación obtenida por el alumno. • Grade: nota obtenida, en base a diez, en el ejercicio. • Solved: indica si el ejercicio ha sido resuelto.
Códigos de respuesta	• 200. Ok. • 400. Bad request. • 404. Exercise not found.

Tabla 52. Endpoint 9

GET	/exercises/{id}
Descripción	Busca un ejercicio y devuelve sus datos.
Parámetros en URL	• Id: id del ejercicio.
Body	Ninguno
Códigos de respuesta	• 200. Ok. • 404. Exercise not found.

Tabla 53. Endpoint 10

GET	/exercises/id={id}&student={student}
Descripción	Busca un ejercicio asociado a un alumno, y devuelve sus datos.
Parámetros en URL	• Id: id del ejercicio. • Student: nombre del alumno.
Body	Ninguno
Códigos de respuesta	• 200. Ok. • 404. Exercise for student not found.

Tabla 54. Endpoint 11

5. EVALUACIÓN

Para comprobar el correcto funcionamiento de la aplicación, se ha diseñado un plan de pruebas que se encarga de verificar los requisitos previamente definidos. A continuación, se encuentra la plantilla que se va a utilizar para cada prueba, y las pruebas definidas.

5.1. Plantilla de pruebas

La plantilla que se va a utilizar para definir todas las pruebas del plan es la siguiente:

P-NN	
Procedimiento	Texto
Requisitos	Lista de requisitos
Interfaz	Interfaz
Resultado esperado	Texto

Tabla 55. Plantilla de pruebas

- Título: la P denomina prueba y NN son los dígitos del número identificativo de la prueba.
- Procedimiento: incluirá la explicación de cómo realizar la prueba.
- Requisitos: indica los requisitos que cubre dicha prueba
- Interfaz: las interfaces serán definidas en la sección 6.1 “Interfaces”, sin embargo, en este apartado serán mencionadas para indicar el nombre de la interfaz donde se realiza la prueba.
- Resultado esperado: indica lo que se espera obtener tras realizar la prueba, ya sea error o un resultado correcto.

5.2. Plan de pruebas

P-01	
Procedimiento	Se rellena y envía el formulario de registro de manera correcta.
Requisitos	RU-01, RS-01, RS-21
Interfaz	Register
Resultado esperado	Se crea un nuevo usuario en la base de datos y se redirige a la interfaz Login.

Tabla 56. P-01

P-02	
Procedimiento	Se rellena y envía el formulario de registro, dejando el campo de correo vacío.
Requisitos	RU-01, RS-01, RS-21
Interfaz	Register
Resultado esperado	Error. Sale un mensaje diciendo “Please fill required fields”.

Tabla 57. P-02

P-03	
Procedimiento	Se rellena y envía el formulario de registro, dejando el campo de contraseña vacío.
Requisitos	RU-01, RS-01, RS-21
Interfaz	Register
Resultado esperado	Error. Sale un mensaje diciendo “Please fill required fields”.

Tabla 58. P-03

P-04	
Procedimiento	Se rellena y envía el formulario de registro, con un correo que no cumple el formato "*algo@algo.algo*"
Requisitos	RU-01, RS-01, RS-21
Interfaz	Register
Resultado esperado	Error. Sale un mensaje diciendo "Invalid email".

Tabla 59. P-04

P-05	
Procedimiento	Se rellena y envía el formulario de registro, con una contraseña de longitud < 8 caracteres.
Requisitos	RU-01, RS-01, RS-21
Interfaz	Register
Resultado esperado	Error. Sale un mensaje diciendo "Password must be 8 or more characters long".

Tabla 60. P-05

P-06	
Procedimiento	Se rellena y envía el formulario de inicio de sesión de manera correcta.
Requisitos	RU-02, RS-02, RS-19, RS-22
Interfaz	Login
Resultado esperado	El profesor inicia sesión y se le redirige a la interfaz Dashboard. En esta interfaz verá los ejercicios que haya creado y los resultados de cada uno, si ya han sido resueltos.

Tabla 61. P-06

P-07	
Procedimiento	Se rellena y envía el formulario de inicio de sesión con un email no existente.
Requisitos	RU-02, RS-02, RS-19, RS-22
Interfaz	Login
Resultado esperado	Error. Sale un mensaje diciendo “Wrong login credentials”.

Tabla 62. P-07

P-08	
Procedimiento	Se rellena y envía el formulario de inicio de sesión con una contraseña incorrecta.
Requisitos	RU-02, RS-02, RS-19, RS-22
Interfaz	Login
Resultado esperado	Error. Sale un mensaje diciendo "Wrong login credentials".

Tabla 63. P-08

P-09	
Procedimiento	Se rellena y envía el formulario de inicio de sesión, dejando el campo del correo vacío.
Requisitos	RU-02, RS-02, RS-19, RS-22
Interfaz	Login
Resultado esperado	Error. Sale un mensaje diciendo "Please fill required fields".

Tabla 64. P-09

P-10	
Procedimiento	Se rellena y envía el formulario de inicio de sesión, dejando el campo de la contraseña vacío.
Requisitos	RU-02, RS-02, RS-19, RS-22
Interfaz	Login
Resultado esperado	Error. Sale un mensaje diciendo "Please fill required fields".

Tabla 65. P-10

P-12	
Procedimiento	Se rellena y envía el formulario para editar el perfil, correctamente.
Requisitos	RU-03, RS-03
Interfaz	Edit Profile
Resultado esperado	Se hacen los cambios correspondientes en los datos del usuario. Al refrescar la interfaz de Edit Profile deben aparecer los datos cambiados.

Tabla 66. P-12

P-13	
Procedimiento	Se rellena y envía el formulario de para editar el perfil, con el campo del correo vacío.
Requisitos	RU-03, RS-03
Interfaz	Edit Profile
Resultado esperado	Error. Sale un mensaje diciendo “Please fill required fields”.

Tabla 67. P-13

P-14	
Procedimiento	Se rellena y envía el formulario para editar el perfil, con el campo de la contraseña vacío.
Requisitos	RU-03, RS-03
Interfaz	Edit Profile
Resultado esperado	Error. Sale un mensaje diciendo “Please fill required fields”.

Tabla 68. P-14

P-15	
Procedimiento	Se rellena y envía el formulario para editar el perfil, con un correo que no cumple el formato "*algo@algo.algo*"
Requisitos	RU-03, RS-03
Interfaz	Edit Profile
Resultado esperado	Error. Sale un mensaje diciendo "Invalid email".

Tabla 69. P -15

P-16	
Procedimiento	Se rellena y envía el formulario para editar el perfil, con una contraseña de longitud < 8 caracteres.
Requisitos	RU-03, RS-03
Interfaz	Edit Profile
Resultado esperado	Error. Sale un mensaje diciendo "Password must be 8 or more characters long".

Tabla 70. P-16

P-17	
Procedimiento	Se introduce el título del ejercicio, nombre del alumno, instrucciones y se selecciona un tipo de ejercicio, en el formulario para crear un ejercicio.
Requisitos	RU-04, RS-04, RS-05, RS-23
Interfaz	Create Exercise
Resultado esperado	Se crea un nuevo ejercicio. Aparecerá en la interfaz Dashboard.

Tabla 71. P-17

P-18	
Procedimiento	Se rellena y envía el formulario para crear un ejercicio, dejando el título del ejercicio vacío.
Requisitos	RU-04, RS-04, RS-05, RS-23
Interfaz	Create Exercise
Resultado esperado	Error. Sale un mensaje diciendo “Input a title”.

Tabla 72. P-18

P-19	
Procedimiento	Se rellena y envía el formulario para crear un ejercicio, dejando el nombre del alumno vacío.
Requisitos	RU-04, RS-04, RS-05, RS-23
Interfaz	Create Exercise
Resultado esperado	Error. Sale un mensaje diciendo “Input the name of the student”.

Tabla 73. P-19

P-20	
Procedimiento	Se rellena y envía el formulario para crear un ejercicio, dejando las instrucciones vacías.
Requisitos	RU-04, RS-04, RS-05, RS-23
Interfaz	Create Exercise
Resultado esperado	Error. Sale un mensaje diciendo “Input the instructions”.

Tabla 74. P-20

P-21	
Procedimiento	Se rellena y envía el formulario para editar un ejercicio, sin seleccionar el tipo.
Requisitos	RU-06, RS-07
Interfaz	Edit Exercise
Resultado esperado	Error. Sale un mensaje diciendo “Select the type of exercise”.

Tabla 75. P-21

P-22	
Procedimiento	Se modifican los campos deseados correctamente del formulario para editar un ejercicio, y se envía el ejercicio.
Requisitos	RU-06, RS-07
Interfaz	Edit Exercise
Resultado esperado	Se actualizan los datos del ejercicio en la base de datos. Aparecerán los nuevos datos en el ejercicio.

Tabla 76. P-22

P-23	
Procedimiento	Se rellena y envía el formulario para editar un ejercicio, dejando el campo del título vacío.
Requisitos	RU-06, RS-07
Interfaz	Edit Exercise
Resultado esperado	Error. Sale un mensaje diciendo “Input a title”.

Tabla 77. P-23

P-24	
Procedimiento	Se rellena y envía el formulario para editar un ejercicio, dejando el campo del nombre del alumno vacío.
Requisitos	RU-06, RS-07
Interfaz	Edit Exercise
Resultado esperado	Error. Sale un mensaje diciendo “Input the name of the student”.

Tabla 78. P-24

P-25	
Procedimiento	Se rellena y envía el formulario para editar un ejercicio, dejando el campo de las instrucciones vacío.
Requisitos	RU-06, RS-07
Interfaz	Edit Exercise
Resultado esperado	Error. Sale un mensaje diciendo “Input the instructions”.

Tabla 79. P-25

P-26	
Procedimiento	Presionar el botón “Delete” de un ejercicio.
Requisitos	RU-05, RS-06
Interfaz	Dashboard
Resultado esperado	Se elimina el ejercicio. Desaparece de la interfaz.

Tabla 80. P-26

P-27	
Procedimiento	Presionar el botón “Share” de un ejercicio.
Requisitos	RU-08, RS-09
Interfaz	Dashboard
Resultado esperado	Sale una ventana emergente para elegir si copiar o cancelar. Si le damos a copiar, copiará en el portapapeles un enlace único para acceder a resolver el ejercicio.

Tabla 81. P-27

P-28	
Procedimiento	Presionar el botón “Duplicate” de un ejercicio.
Requisitos	RU-07, RS-08
Interfaz	Dashboard
Resultado esperado	Se genera una copia del ejercicio al cual se le dió “duplicate”. Aparecerá el nuevo ejercicio en la interfaz.

Tabla 82. P-28

P-29	
Procedimiento	Presionar el botón “Edit” de un ejercicio.
Requisitos	RU-06, RS-07
Interfaz	Dashboard
Resultado esperado	Redirige al usuario a la interfaz Edit Exercise. Aparecen los elementos del ejercicio con los campos rellenados con la información correspondiente guardada en la base de datos.

Tabla 83. P-29

P-30	
Procedimiento	Presionar el botón “Result” de un ejercicio.
Requisitos	RU-10, RS-11
Interfaz	Dashboard
Resultado esperado	Redirige al usuario a la interfaz Exercise Result. Aparecen los elementos del ejercicio con las respuestas del alumno, indicando si son correctas o incorrectas. También se mostrará la puntuación del alumno y la nota final en base a diez.

Tabla 84. P-30

P-31	
Procedimiento	Presionar el botón “Logout” en la barra lateral.
Requisitos	RU-18, RS-20
Interfaz	Dashboard, Create Exercise, Edit Exercise, Exercise Result, Edit Profile
Resultado esperado	Cierra la sesión actual del usuario y lo redirige a la interfaz Login.

Tabla 85. P-31

P-32	
Procedimiento	Enviar las respuestas de un ejercicio.
Requisitos	RU-10, RU-12, RU-13, RU-14, RU-15, RU-16, RU-17, RS-11, RS-13, RS-14, RS-15, RS-16, RS-17, RS-18
Interfaz	Solve Exercise.
Resultado esperado	Se guardarán las respuestas en la base de datos. Al enviar las respuestas los campos de las respuestas no se pueden editar. Se indicará si una respuesta es correcta e incorrecta, además de aparecer la respuesta correcta que indicó el profesor, la puntuación y la nota final. El profesor deberá dirigirse al ejercicio y presionar el botón “Result”, que lo redirigirá a la interfaz Exercise Result, y aparecerá la solución del alumno.

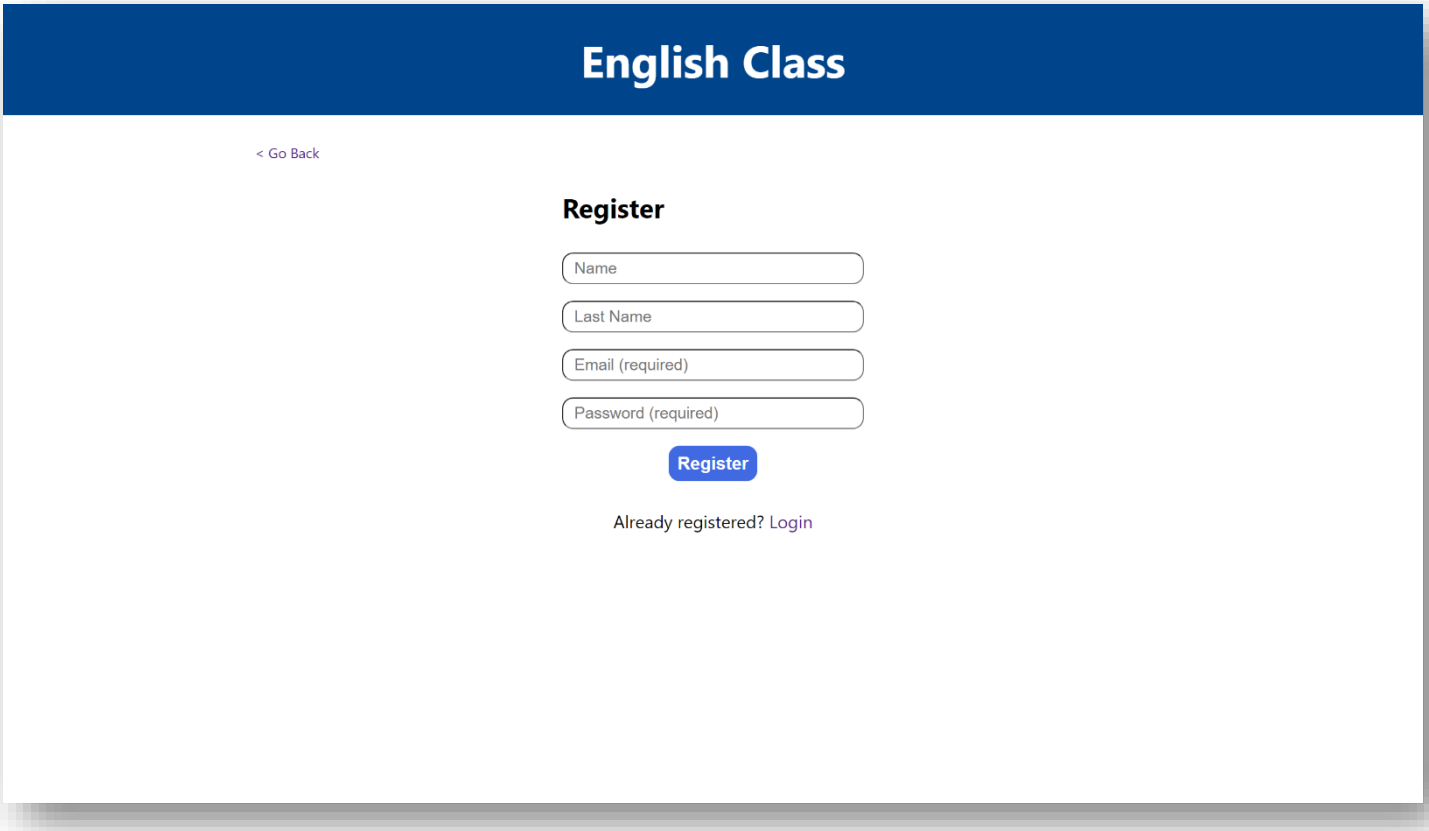
Tabla 86. P-32

6. DETALLE DE LA SOLUCIÓN

6.1. Interfaces

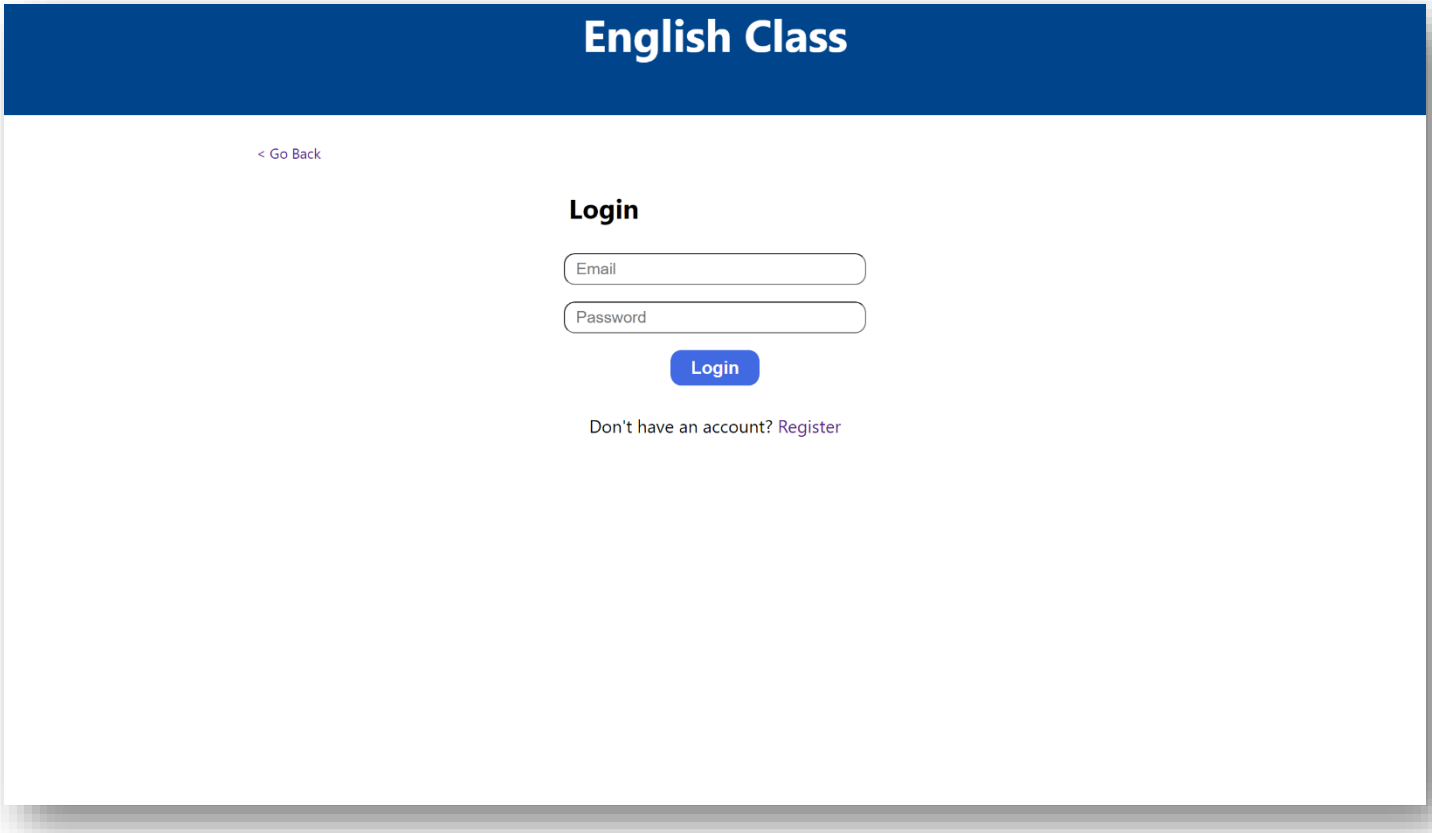
A continuación, se mostrarán capturas de las diferentes interfaces o pantallas que posee la aplicación, cada una acompañada de una breve descripción de la interfaz y sus funcionalidades. Cabe destacar que estas interfaces pueden cambiar en el futuro, conforme se continúe desarrollando la aplicación.

Register



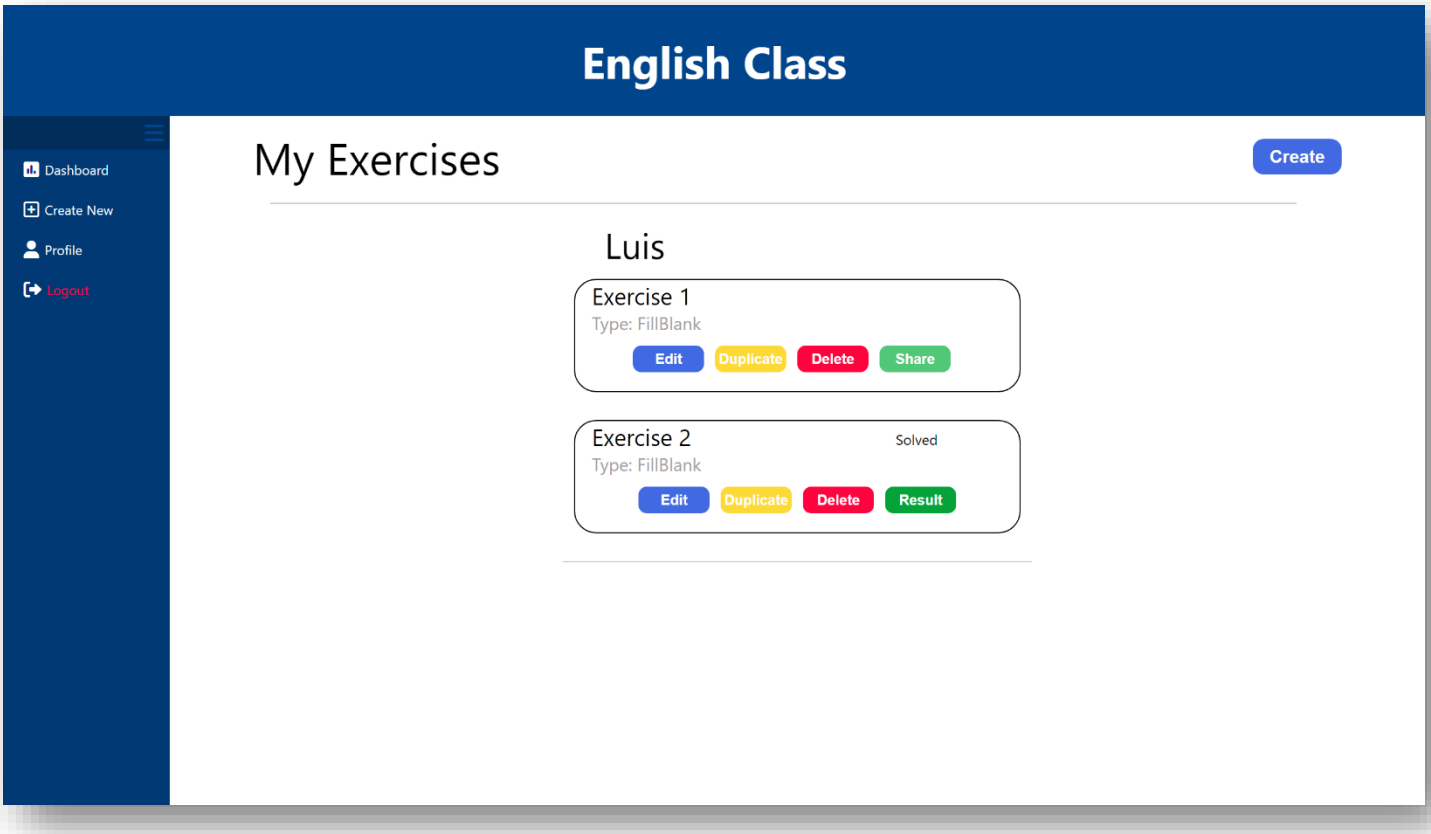
Esta interfaz corresponde al proceso de registro de un usuario en la aplicación. Contiene el formulario de registro donde será necesario indicar el correo y la contraseña. Opcionalmente se podrá indicar el nombre y el apellido. Contiene un botón “register” que se encarga de ejecutar la acción de enviar el formulario al servidor, para crear un nuevo usuario en la base de datos. Por último, cuenta con dos enlaces, uno en la parte superior izquierda para ir a la página anterior, y otro debajo del formulario para acceder al inicio de sesión.

Login



Esta interfaz corresponde al proceso de inicio de sesión en la plataforma. Contiene un formulario donde será necesario indicar el correo y la contraseña. Contiene un botón “login” que se encarga de ejecutar la acción de enviar el formulario al servidor para validar las credenciales. Por último, cuenta con dos enlaces, uno en la parte superior izquierda para ir a la página anterior, y otro debajo del formulario para acceder a la interfaz del registro. Una vez se inicie sesión, se podrá acceder a las vistas de un profesor.

Dashboard



Esta interfaz corresponde a toda la información y funcionalidades de los ejercicios creados. En primer lugar, tenemos, agrupados por estudiantes, todos los ejercicios creados para un estudiante. Cada ejercicio muestra como información el título, el tipo, y "solved" si ha sido resuelto. Además, cada ejercicio cuenta con cuatro botones:

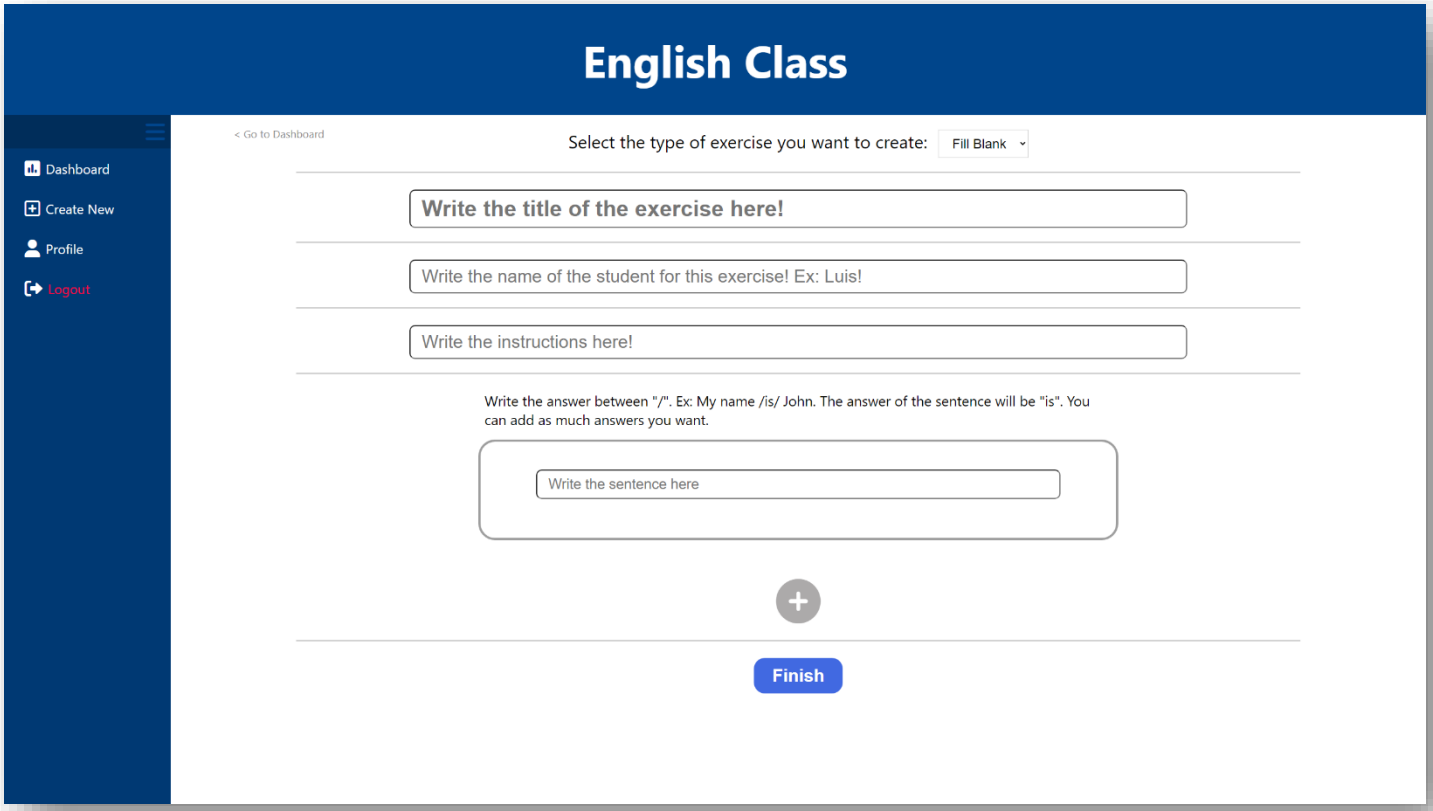
- Edit: redirige al usuario a la interfaz Edit Exercise, para proceder a modificar los datos del usuario.
- Duplicate: genera una copia del ejercicio y refresca la página.
- Delete: elimina el ejercicio. Tras presionar el botón sale una ventana modal para prevenir errores, y así confirmar que realmente se quiere eliminar el ejercicio.
- Share: muestra una ventana modal con un botón "copy", que copiará en el portapapeles la url para acceder al ejercicio, que estará listo para ser resuelto. El botón share aparecerá cuando el ejercicio aún no haya sido resuelto.
- Result: redirige al usuario a la interfaz Exercise Result, para poder visualizar la solución del ejercicio enviada por el alumno. Este botón solo aparecerá cuando el ejercicio haya sido resuelto.

Por otro lado, existe un menú en el lado izquierdo con el formato de barra lateral, que contiene accesos directos:

- Dashboard: redirige al usuario a la interfaz Dashboard.
- Create New: redirige al usuario a la interfaz Create Exercise, para crear un ejercicio.
- Profile: redirige al usuario a la interfaz Edit Profile, para visualizar y modificar los datos de su perfil.
- Logout: finaliza la sesión actual del usuario y redirige a login.

Por último, existe un botón “create”, que redirige al usuario a la interfaz Create Exercise, para crear un ejercicio.

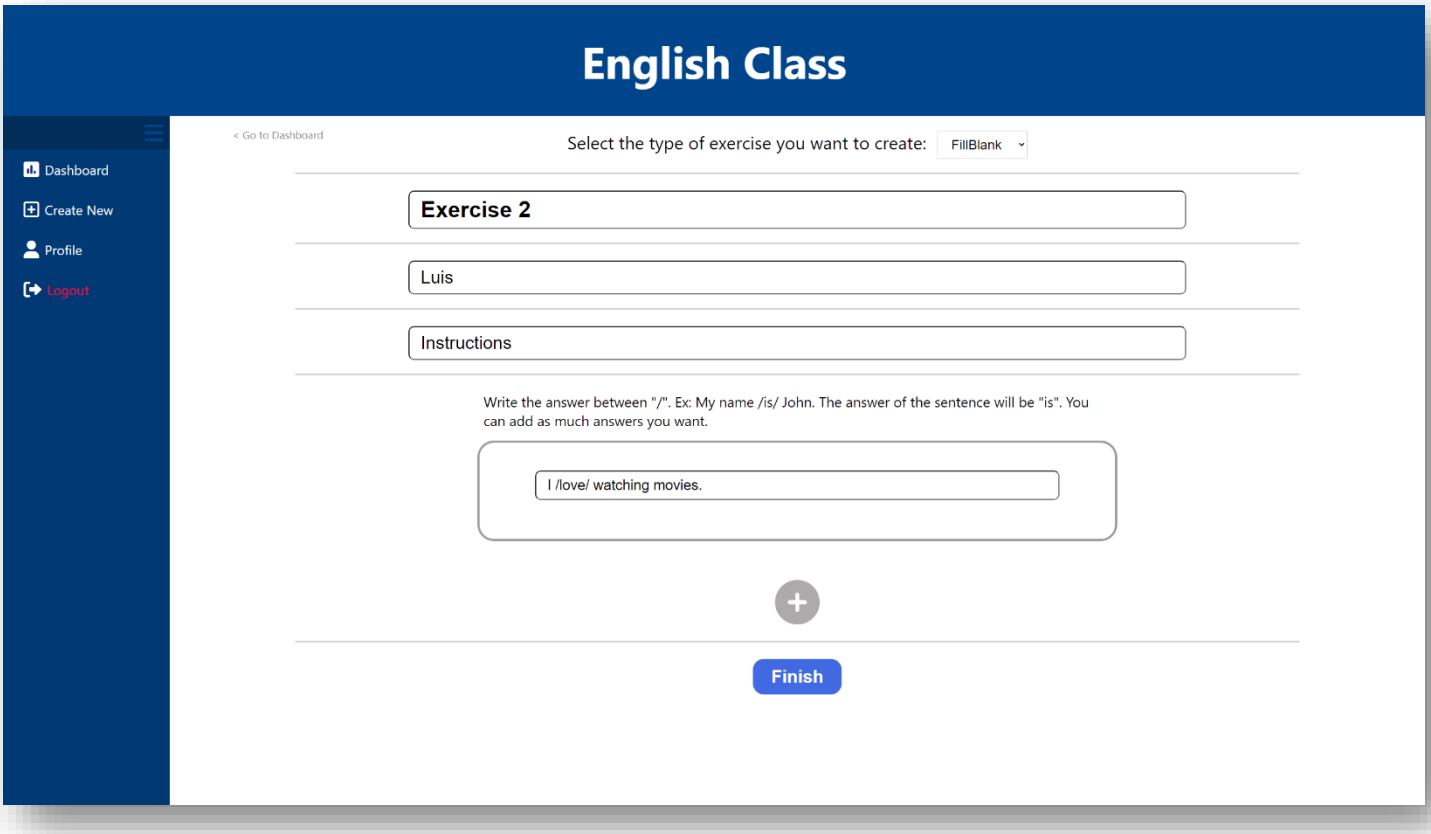
Create Exercise



Esta interfaz corresponde con el proceso de crear un ejercicio. En primer lugar, en la parte superior cuenta con un selector para indicar el tipo de ejercicio que se va a crear. A continuación, aparece un formulario para indicar el título del ejercicio, el alumno para el que se crea y las instrucciones. Posteriormente cuenta con un botón circular “+” para añadir los elementos correspondientes al tipo de ejercicio seleccionado. Al final del formulario hay un botón “finish”, para enviar el ejercicio al servidor y así almacenarlo en la base de datos.

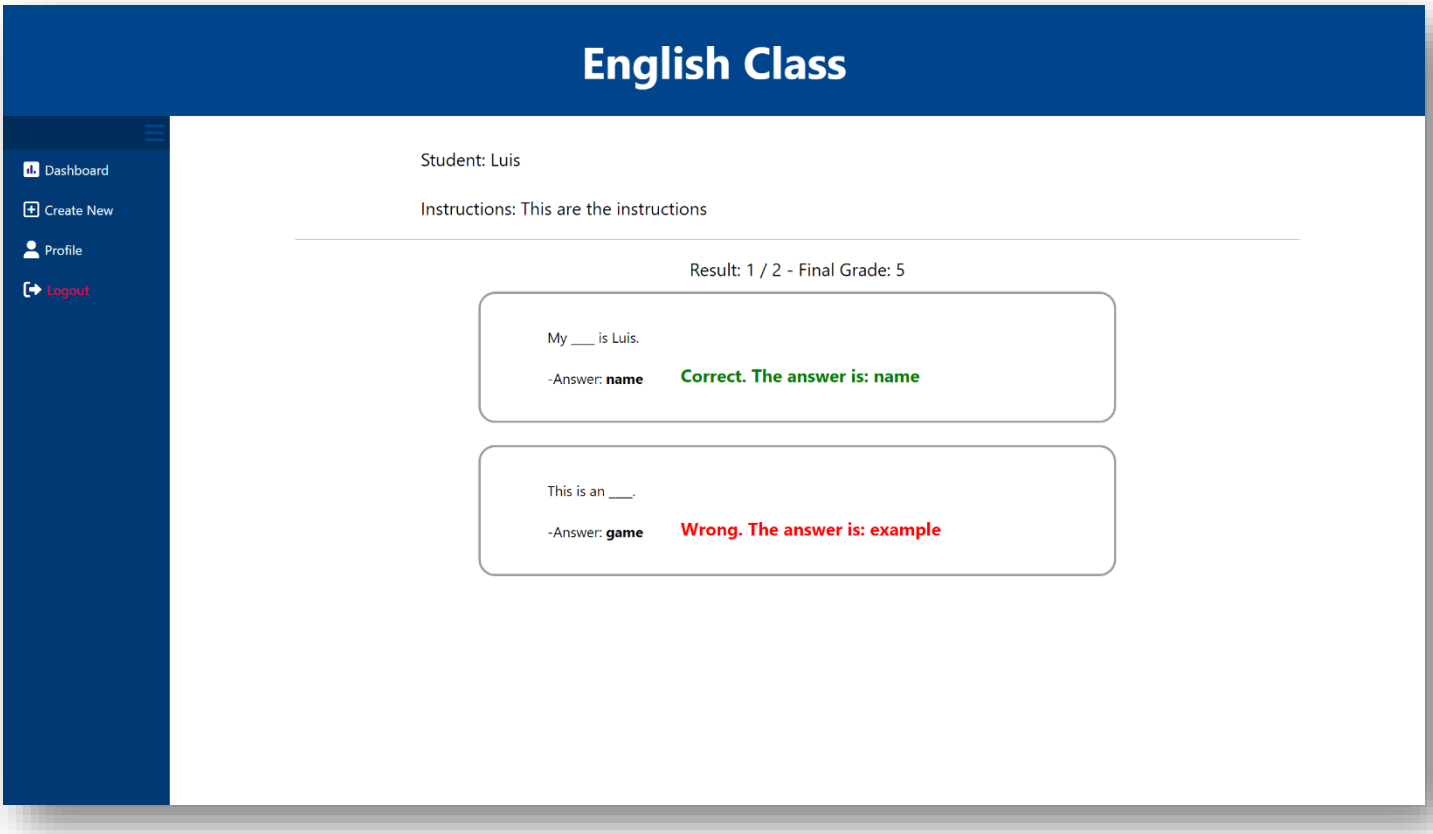
Por último, existe un menú en el lado izquierdo con el formato de barra lateral (previamente definido en la interfaz Dashboard).

Edit Exercise



Esta interfaz corresponde al proceso de editar los datos de un ejercicio previamente creado. La interfaz es exactamente la misma que la del escenario de crear un ejercicio (Create Exercise), con la única diferencia de que los campos se encuentran rellenados con la información correspondiente contenida en la base de datos.

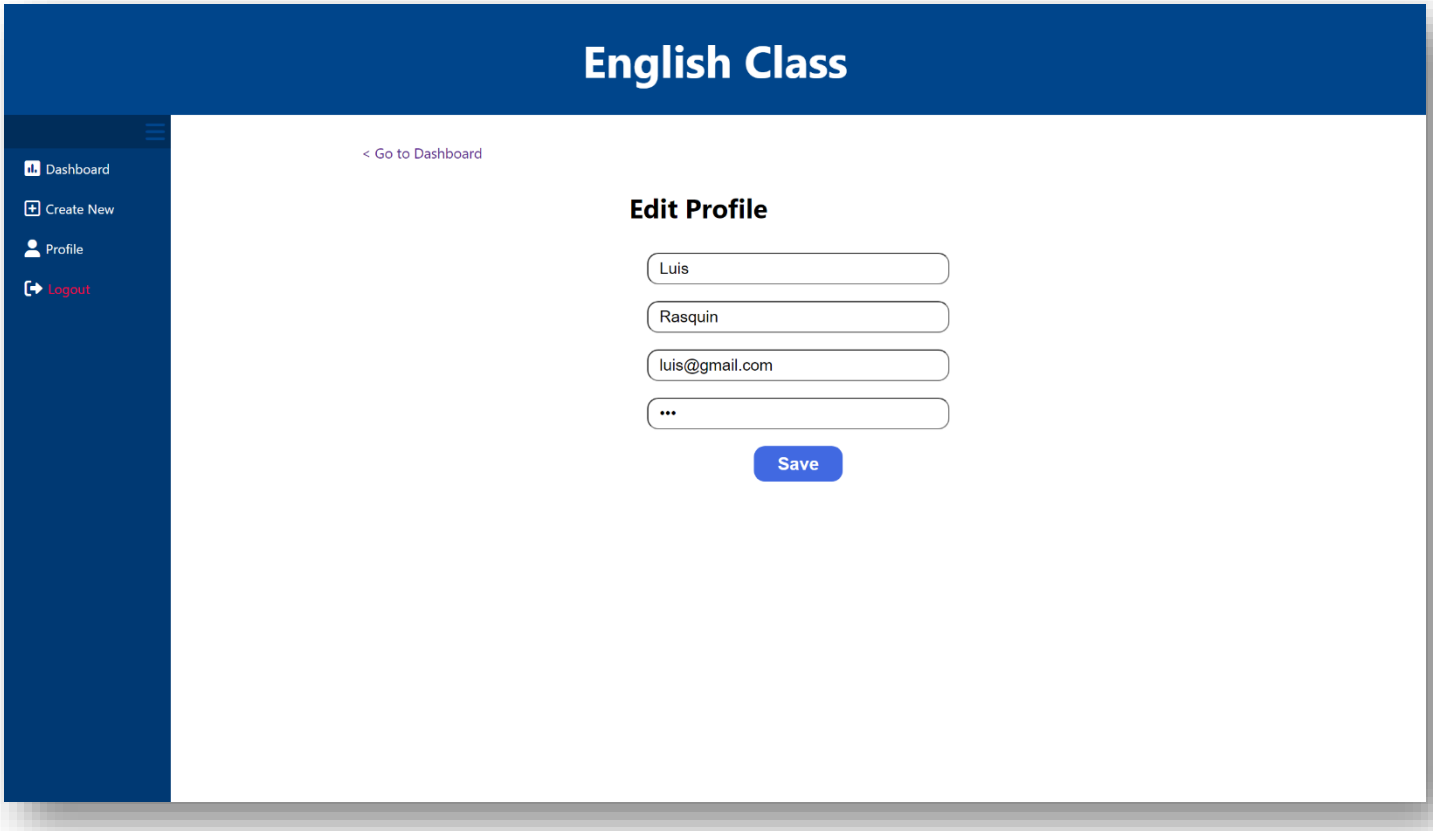
Exercise Result



Esta interfaz corresponde a ver la solución enviada por el alumno. En primer lugar, se muestra el nombre del alumno y las instrucciones del ejercicio. A continuación, se muestra el resultado obtenido por el alumno, así como su nota final. Debajo se encuentran los elementos correspondientes al tipo de ejercicio, indicando si la respuesta es correcta o incorrecta, y mostrando la solución correcta.

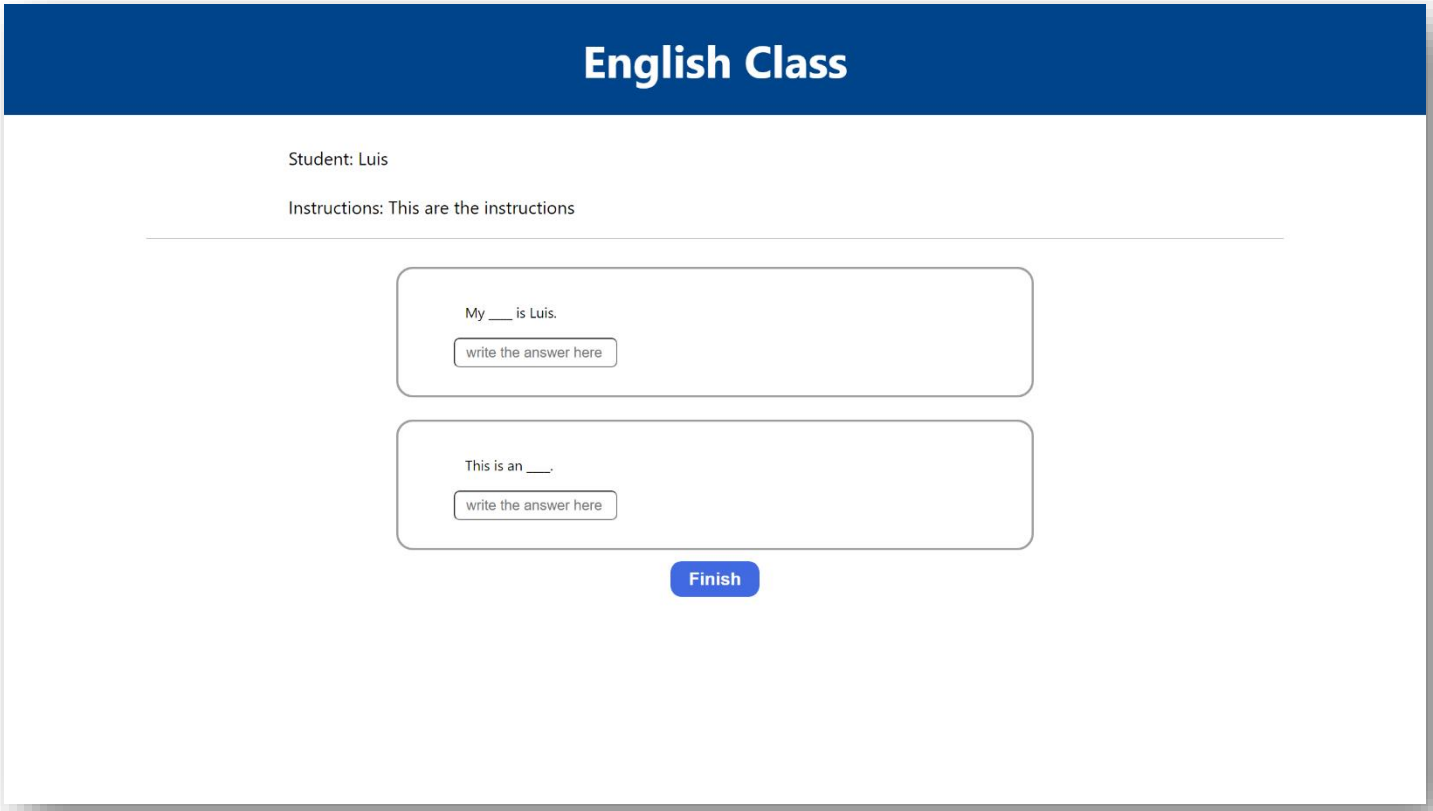
Por último, existe un menú en el lado izquierdo con el formato de barra lateral (previamente definido en la interfaz Dashboard).

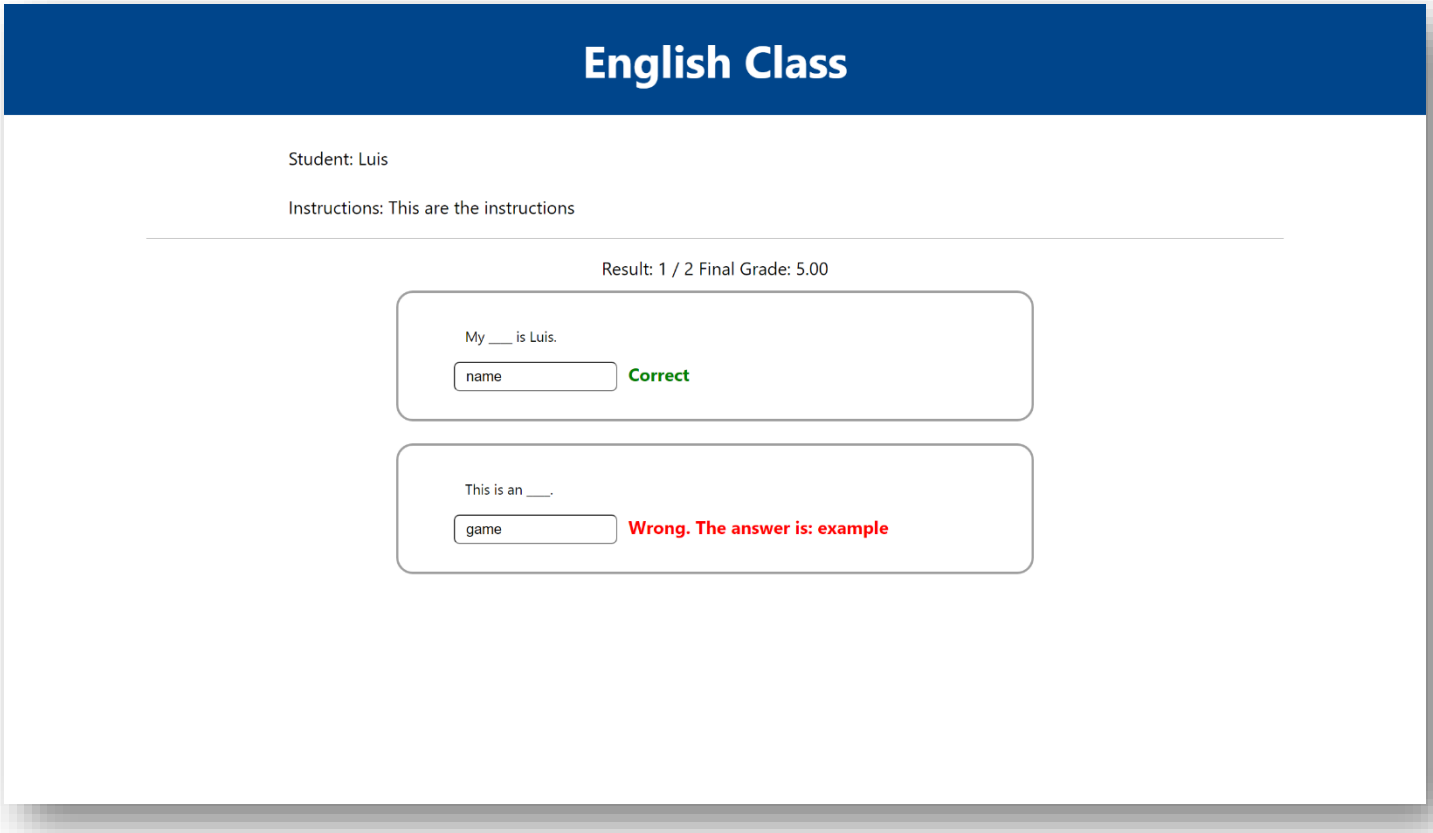
Edit Profile



Esta interfaz permite al usuario visualizar sus datos y modificarlos. Cuenta con un formulario con los campos nombre (opcional), apellido (opcional), correo y contraseña, que ya contendrán la información del usuario. Al final hay un botón “save” para enviar la modificación al servidor y que se cambien los datos en la base de datos. Simultáneamente, el botón, refrescará la página.

Solve Exercise





Esta interfaz corresponde al proceso de resolver un ejercicio. En primer lugar, se muestran las instrucciones y el nombre del alumno. A continuación, se encuentran los elementos correspondientes al tipo de ejercicio que se haya creado. Al final hay un botón “send”, para enviar la solución al servidor que será almacenada en la base de datos. Al presionar el botón, primero aparecerá una ventana modal con el fin de prevenir errores y confirmar que se desea enviar la solución.

Una vez enviada la solución desaparecerá el botón “finish” y las respuestas no se podrán modificar. Además, debajo de las instrucciones, aparecerá el resultado obtenido por el alumno, así como su nota final. En cada elemento del ejercicio, se indicará si la respuesta es correcta o incorrecta, y mostrando la solución correcta.

7. PLANIFICACIÓN

7.1. Metodología

Durante el proceso de desarrollo del actual proyecto, la metodología seguida fue y es Scrum. Esto se debe a que hacían planificaciones en periodos de entre dos y cuatro semanas, que serían lo que se conoce como sprints. En cada sprint se establecían cuáles eran los objetivos a alcanzar una vez este llegara a su fin. Una vez finalizado el sprint se llevaba a cabo una reunión con el tutor, donde se enseñaban los resultados y se proporcionaba un feedback sobre los mismos.

Por otro lado, se utilizó un tablero en la plataforma Trello, para establecer las tareas que estaban “listas”, “pendientes” o “en proceso”, y así tener una mejor perspectiva del estado del proyecto y mayor facilidad para establecer prioridades.

7.2. Diagrama de GANTT

Para esta planificación, la semana 1 comienza el día 17 de enero del 2022, y la semana 23 empieza el 20 de junio y acaba el 23 de junio del 2022, puesto que es el día final para realizar la entrega de este documento. La recogida de requisitos no se representa en el diagrama, ya que al tener una idea clara de qué se iba a desarrollar antes de tener la reunión para matricular el trabajo de fin de grado, no forma parte del periodo planificado.

La duración del proyecto, en diferentes unidades, es:

- 5 meses y 3 días.
- 111 días laborales
- 444 horas de trabajo (suponiendo que la jornada de trabajo es de 4 horas).

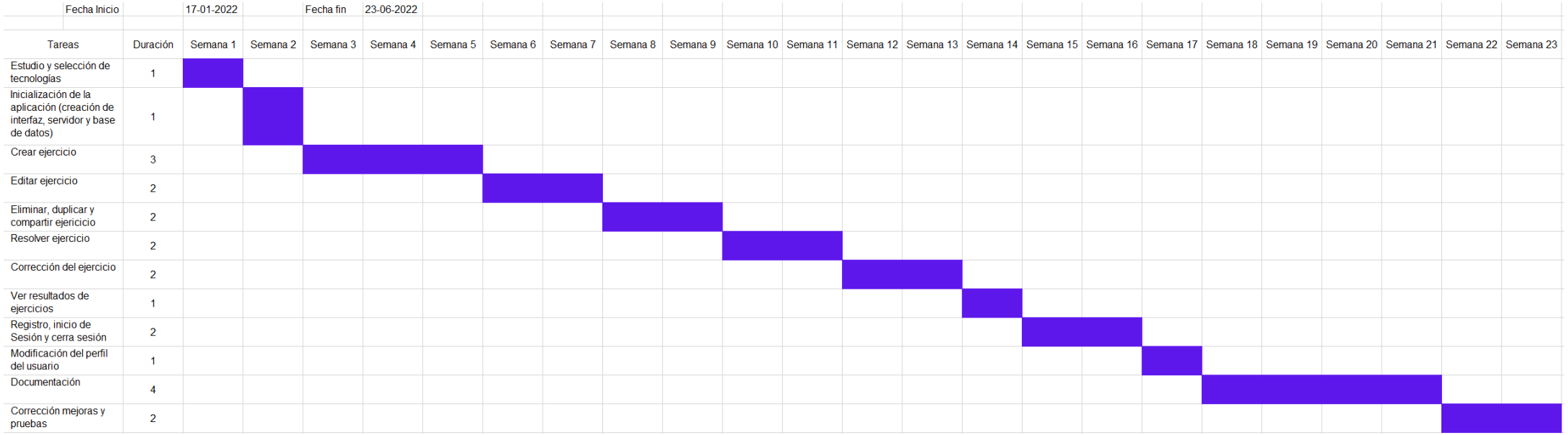


Fig. 15. Diagrama GANTT

7.3. Evolución real

A pesar de la planificación establecida, no fue posible cumplirla tal y como se había definido en un principio. El proyecto empezó el día 17 de enero del 2022, ya que tuvo lugar la primera reunión con el tutor (tras la matriculación del trabajo de fin de grado) para discutir sobre el proyecto.

Durante el mes de enero y febrero se realizaron varios avances debido a la poca exigencia de los estudios académicos. Sin embargo, con el avance del año lo contrario ocurrió tras la llegada de marzo y abril debido a que la exigencia empezó a ser más alta. Por consiguiente, una vez culminada la evaluación académica (a mediados de mayo) y la lista de responsabilidades por atender fue quedándose más corta, la labor relacionada al desarrollo del proyecto se tornó el principal ocupante del tiempo del autor alrededor del mes de junio.

El desarrollo del proyecto se llevó a cabo contemporáneamente con prácticas en empresa, asignaturas de la titulación de ingeniería informática e impartición de clases particulares de inglés, lo que implicó que el trabajo realizado se llevará a cabo en franjas temporales disponibles en el calendario del autor, que podían variar según el día, y que se ajustaban a las tareas mencionadas anteriormente.

Por otro lado, aparecieron ciertos inconvenientes ajenos al proyecto que se presentaron como obstáculos, y que como consecuencia ralentizaron el desarrollo del mismo. Todo esto conllevó a la necesidad de reajustar la planificación conforme a las circunstancias y la carga de trabajo restante con la motivación de poder abordar todos los objetivos de manera eficiente.

8. MARCO REGULADOR

Llegado el momento en que la aplicación salga a producción, es necesario que cumpla con ciertos aspectos legales, como lo pueden ser las leyes respecto a páginas web en la región correspondiente, o las licencias de uso.

8.1. Leyes

El uso de esta aplicación implica el tratamiento y almacenamiento de los datos de los usuarios, ya que necesitan facilitar un correo, contraseña, nombre y apellidos para registrarse y disfrutar del servicio que esta brinda. Es por ello que la plataforma debe cumplir con las diferentes leyes reguladoras en España y en la Unión Europea. Algunas de las leyes principales que se deben cumplir son:

- LOPD-GDD (Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales) [55]. Esta ley tiene el propósito de regular el uso de los datos de una persona física por parte de una página web, obligando a la entidad a asegurar la confidencialidad de los mismos. Un usuario tiene el derecho de acceder a sus datos, rectificarlos, limitar el uso de los mismos, eliminarlos e incluso oponerse a que sus datos sean tratados. Esta ley llegó como sustitución de la LOPD 15/1999.

- RGPD (Reglamento Europeo 2016/679 de Protección de Datos) [56]. Esta ley persigue el mismo objetivo legal que la LOPD-GDD, con la diferencia que no solo aplica en territorio español, si no que aplica en todos los países que conforman la Unión Europea.

Por otro lado, toda web debe contar con el “Aviso Legal Web”. En él se debe indicar los términos y condiciones del servicio, la política de privacidad que sigue la plataforma, la política de cookies que emplea la página y, en caso de ser oportuno, las condiciones de contratación [57, 58]. Este aviso debe ser visible y accesible por todos los usuarios. Normalmente se encuentra un enlace en el pie de la página para acceder a él.

Es importante destacar que este proyecto, al ser de uso personal, no maneja información ni datos de usuarios reales. Hasta ahora, son todos datos hipotéticos para probar las funcionalidades de la aplicación. En caso de ser publicado en Internet y comenzar a manejar datos reales, todos estos aspectos serán tomados en consideración por el autor para aplicar las medidas necesarias.

8.2. Licencias

A continuación, se listan las licencias de cada una de las tecnologías empleadas en este proyecto:

- **React**: utilizada para construir la interfaz de la aplicación. Se encuentra bajo MIT License[59]. Esta permite a cualquier usuario que tenga una copia utilizar el software sin restricciones, siempre y cuando se incluya una copia de licencia y un aviso de copyright, para preservar los derechos de autor.

- **Node**: utilizada para construir el servidor de la aplicación. Se encuentra bajo MIT License.

- **Express**: utilizada para construir la API REST contenida en el servidor de la aplicación. Esta tecnología se encuentra bajo una licencia Creative Commons Attribution-ShareAlike 3.0 United States[60]. Esto permite al usuario compartir y adaptar el material siempre y cuando se haga referencia a la licencia, se indique si se han hecho cambios, y en caso de hacerlos se debe distribuir bajo la misma licencia.

- **MongoDB**: base de datos utilizada para la persistencia de los datos de la aplicación. Se encuentra bajo Server Side Public License[61]. Esta permite a los usuarios hacer uso del software de manera libre, pero sin hacer cambios en él.

- **Canva** [62]: utilizada para elaborar algunas de las ilustraciones de este documento. En este caso los elementos utilizados en las ilustraciones cuentan con el acuerdo de licencia de medios exentos de derechos de autor, lo cual permite modificarlos y utilizarlos de manera gratuita con fines comerciales y no comerciales.

9. ENTORNO SOCIOECONÓMICO

Este capítulo tiene como objetivo detallar un presupuesto para el desarrollo del proyecto, así como una descripción del impacto que este podría tener económica y socialmente.

9.1. Presupuesto

Se elaboró un presupuesto basado en el periodo de desarrollo de la primera versión, obteniendo así los siguientes resultados:

9.1.1. Costes de personal

Asumiendo que el sueldo medio (en el momento que se escribe este documento) de un desarrollador junior full stack es de 21.343€ al año [63]:

Empleado	Tiempo	Coste / mes	Coste bruto
Desarrollador Junior Full Stack	5 meses	1.778,58 €	8.892,90 €
TOTAL			**8.892,90 €**

Tabla 87. Costes de personal

9.1.2. Costes de equipos

Equipo	Tiempo	Coste / mes	Coste bruto
Ordenador de sobremesa	5 meses	220,85 €	1.104,26 €
Monitor 1440p 27’	5 meses	72.78 €	363,90 €
Monitor 1080p 24’	5 meses	35,80 €	179,00 €
Teclado	5 meses	20,00 €	99,99 €
Ratón	5 meses	10,00 €	49,99 €
TOTAL / MES			**359,43 €**
TOTAL			**1.797,14 €**

Tabla 88. Costes de equipos

9.1.3. Costes de herramientas y servicios

A pesar de no aplicar en este momento al proyecto, puesto que aún no ha sido publicado en Internet, los costes en caso de que la aplicación saliera a producción serían:

- Según MongoDB, el coste estimado de un cluster M10 por mes es de 56,94 $ [64].
- Según Amazon Web Services [65], para una aplicación con React y Node, el costo estimado de su servicio es de 35,50$ [65, 66].

Herramienta / Servicio	Coste / mes
Amazon Web Services	33,12 €
MongoDB Atlas	53,12 €
Visual Studio Code	0 €
TOTAL / MES	**86,24 €**

Tabla 89. Costes de herramientas y servicios

9.1.4. Coste total

Tras calcular todos los costes del proyecto, faltaría calcular los costes indirectos tales como agua, luz, internet, entre otros. Normalmente este coste oscila entre el 15 y 20% de los costes previamente calculados, conocidos como costes directos. Para realizar el cálculo de los costes indirectos, se tomará el peor escenario posible: 20%.

Costes	Coste bruto
Personal	8.892,90 €
Equipos	1.797,14 €
Indirectos	2.138,01 €
TOTAL	**12.828,05 €**

Tabla 90. Coste Total

9.2. Impacto socio-económico

En primer lugar, el impacto principal de este proyecto recaerá sobre un conjunto reducido de personas, formado por: el autor del proyecto, sus alumnos, conocidos del autor y los alumnos de estos últimos. A la hora de realizar una clase, los profesores tendrán a su disposición una herramienta que les permitirá diseñar ejercicios a la medida de sus alumnos y llevar un seguimiento sobre los mismos, lo que agilizará la preparación de una clase y las hará más dinámicas. Por otro lado, de cara a los alumnos, les permitirá aprender inglés de una manera más moderna que incluso puede resultar innovadora, ya que realizan sus ejercicios a través de dispositivos tecnológicos, a diferencia de la manera tradicional de utilizar un papel y un lápiz o bolígrafo.

Debido a los motivos mencionados en el párrafo anterior, si este proyecto es subido a Internet de manera que sea de acceso y uso público, puede ser una gran innovación dentro del sector formado por personas relativas a clases de inglés, sobre todo en los profesores de clases particulares, puesto que lo más probable es que un profesor particular no cuente con los mismos recursos que un profesor de colegio o academia.

Por último, esta herramienta está pensada para hacer uso personal de la misma, por lo que de momento no existe un plan de explotación. Sin embargo, es posible que con un futuro desarrollo de mejoras y mantenimiento, sea posible hacerla pública para que otras personas hagan uso de ella, y sacar un beneficio económico de dicho uso mediante anuncios o suscripciones para obtener funcionalidades avanzadas.

10. CONCLUSIONES

En este último capítulo se abordarán las conclusiones obtenidas por el autor sobre el desarrollo del proyecto y qué ocurrirá en el futuro con el mismo.

10.1. Conclusiones del proyecto

Tras el periodo de trabajo y desarrollo del proyecto, se puede concluir que se han cumplido con los objetivos establecidos en un principio. A lo largo de este tiempo se ha desarrollado una aplicación capaz de:

- Crear y editar ejercicios personalizados de inglés.
- Compartir ejercicios mediante un enlace único.
- Visualizar los resultados obtenidos por los alumnos para llevar un seguimiento.
- Acceder a un ejercicio para resolverlo, sin la necesidad de realizar ningún registro. En otras palabras, acceso fácil y directo.

Cabe mencionar que se han profundizado en los conocimientos relativos al desarrollo web, particularmente en tecnologías como React, Node, y MongoDB.

Por último, mencionar que el proyecto ha servido para aplicar los conocimientos que se obtienen durante la carrera de ingeniería informática. Normalmente estos conocimientos se ponen en prácticas con actividades y evaluaciones que no son casos reales, sino hipotéticos. La diferencia en este caso es que todos esos conocimientos se aplican a un proyecto real, que en algunos casos puede tener un futuro.

10.2. Futuro del proyecto

En primer lugar, se van a desarrollar nuevos tipos de ejercicios, de manera que conforme pase el tiempo los profesores y los alumnos dispongan de diferentes maneras de evaluar o ser evaluados. A corto plazo se pretenden incorporar ejercicios del tipo “listening” donde el alumno debe reproducir un audio y responder a unas preguntas, y ejercicios de asociación, donde el objetivo es unir un elemento con otro que le corresponde. Por otro lado, se desarrollarán nuevas funcionalidades respecto al seguimiento de los resultados obtenidos por los alumnos, para así facilitar la evaluación de su progreso.

Puesto que ya existe una primera versión final del proyecto, se pondrá en un uso, y a medida que pase el tiempo, se descubrirán errores a los cuales se les proporcionará mantenimiento, y aparecerán nuevas ideas para incorporar en la aplicación. El objetivo

a largo plazo es brindar a los profesores y alumnos, una plataforma con un amplio catálogo de tipos de ejercicios y herramientas para la evaluación y el seguimiento.

SUMMARY

1. INTRODUCTION

Nowadays technology is used on a daily basis, since almost every person has one or more devices such as smartphones, tablets or computers. Since the pandemic, technology has been used even more for educational purposes. Schools and universities started to teach via online. This is also the case for people who work as English tutors.

What fuels and motivates this project is to offer this last group of people, a tool which they can use to develop their classes and waste less time creating and manually correcting exercises for their students. This way they can focus more on managing the class's time, to use it in a more efficient way, and help the students to understand and learn.

Main objectives

- Develop a web application which allows English tutors to create, manage and share exercises, and automate the process of correcting them.
- Develop a web application that enables students to access and use easily, in which they can solve customized exercises.

Other objectives

- Acquire new knowledge in web development, especially in technologies like React and Node.

2. STATE OF ART

Existing solutions

In these times, there are an endless number of websites for different purposes. Right now, the project will focus on the ones that allow users to create or use English exercises for their lessons. Some of them are:

- Wooclap: a website that allows the user to create and share exercises such as tests, word clouds or association of an element.
- Kahoot: a website that allows the user to create and share tests.

- Learn English British Council: a website where you can find English lessons and preset exercises to solve.
- Text editors: tools such as Microsoft Word or Google Documents enable the user to write down in a computer sentences and questions to solve, as if you do with a sheet of paper.

Market needs

For an English tutor, it would be ideal to have a tool that allows them to create different types of exercises, with a variety of options, that suits their students, regardless of their age or level of English. This will allow them to create a more dynamic and productive class.

The alternatives mentioned above are very limited. And in the case of text editors, sharing the exercise could be difficult if the student has a vague understanding of the tool used.

Technologies

In this section, a list of the technologies that could be used for the development of the web application is shown, followed by a short explanation.

- Front-end: is the part of a web application that is in charge of building the user interface, so the final user can interact with it. It is composed by HTML, CSS and JavaScript, although there are some technologies that ease the process of coding it, such as:
    - React: is a JavaScript framework developed by Facebook.
    - Angular: is a JavaScript framework developed by Google.
    - Vue: is a JavaScript framework developed by Facebook.
- Back-end: is the part of the web application that implements all the logic. It is known as the server and holds the code of an API or the connection to a database. Some technologies to build the back-end are:
    - Node: is a JavaScript framework that runs on a machine instead of on a web browser. It has a lot of libraries, such as Express, that eases the process of building and API.

- Django: is a Python framework, ideal to build a server based on the model-view-controller architecture.
- Spring boot: is a framework that allows the developer to create Java Enterprise applications, that are easy to run and ready to be deployed to production.

Besides displaying static information in the front-end, there could be the interest to also show dynamic information. For this, there is two methods:

- Client-Side Rendering: after requesting and getting the data from the server, the client (web browser) processes it and decides how it is going to be displayed in the user interface and builds it. Common technologies for this are the previously mentioned: React, Angular and Vue.
- Server-Side Rendering: after getting the request from the client (web browser), the server builds the webpage with the proper information, and then sends it to the client. Now the client will only need to display the response from the server. Technologies for this development are: JSP or EJS.

In terms of databases, there are two types:

- Relational databases: driven by the relational model, is a table-based database which needs to define how the data is structured and follows the ACID properties (Atomicity, Consistency, Isolation and Durability). Some examples are: MySQL, Oracle, MariaDB.
- Non-relational databases: also known as NoSQL databases, may be oriented in different ways such as key-pair values, graphs, document-based or wide-columns. These don't require the definition of a structure to store data and follow the CAP theorem (Consistency, Availability, Partition tolerance). Some examples are: MongoDB, Cassandra, Neo4j.

Methodologies

While talking about working methodologies to follow during a software development project, it is possible to differentiate between two types:

- Traditional: some of the most common are Cascade process, Spiral process, Evolutionary Prototyping process and Incremental process.

- Agile: some of the most common are Kanban, Scrum and XP (Extreme Programming).

3. ANALYSIS OF THE PROBLEM

First of all, to describe and understand the application, two different types of users must be defined:

- Teacher: this type of user accesses the website by self-initiative and uses it in a more complex way. They will access a dashboard where they will be able to create exercises, manage them, share them and take a general look of those that they have already created.
- Student: this user that accesses the application via the link the teacher shared with them. Once on the website, they will be able to solve the assigned exercise.

Requirements

While defining the application's requirements, there are two types:

- User requirements: they focus on defining what the user can do while using the software.
- Software requirements: at the same time, they are divided in functional and non-functional requirements. The first ones tell what the system can do, and the second ones refer to the constraints and qualities of the system.

In the original document, in section 3.3 "Requisitos", a list of the defined requirements for the software of this project will be found.

Use cases

An UML diagram for the use cases defined for this project can be found in the section 3.2 "Casos de uso" of the original document. This will help the reader understand the functionalities of the software and how the user can interact with it.

Traceability Matri

In section 3.4 and 3.5 of the original document, traceability matrix will be shown, one for user and software requirements and another one for software requirements and use cases. This helps the reader notice the relationship between them.

4. DESIGN OF THE SOLUTION

Technologies selection

What is considered as the best solution for developing this project's application is the "MERN Stack". This is how the set of technologies formed by MongoDB, Express, React and Node, is called. The main reason behind this decision is the fact that all of these technologies are based on the same programming language, JavaScript, which helps the developer code faster if they possess knowledge about it (which is the case).

Other reasons are: Mongoose, a framework for Node that eases the use of MongoDB, the flexibility that MongoDB gives due to allowing the developer to store unstructured data in the database, and the constant evolution, support and popularity of React.

Architecture

The architecture decided to be followed by the application is formed by:

- Client-side rendering using React.
- A Node server that will host a REST API, returning data in JSON format and connected to the database.
- MongoDB database.

For consulting the endpoints in the REST API, go to section 4.2.2 "Endpoints" in the original document, for a list of tables that contains the information for each one.

5. EVALUATION

This section contains an evaluation plan defined with multiple tests, to ensure the proper operation of the different functionalities that the application offers. To consult the different tests, go to section 5. "Evaluación" in the original document.

6. DETAIL OF THE SOLUTION

Interfaces

This section holds a list of the different interfaces that the application displays, followed by a brief description. The main interfaces are:

- **Register**: contains the form to sign up in the application, which submits the data to the server to create a new user.

- **Login**: contains the form to authenticate the user and lets him through the application. The data is submitted to server to verify the existence of the u
- **Dashboard**: this view will display the information of all the exercises created by the user. Each exercise will have different buttons that lets the user edit, delete, share or duplicate an exercise. In case the exercise is solved, instead of sharing, the teacher can review the answer of the student. This view will also have a button to create exercises, that will redirect him to the Create Exercise interface, and a sidebar with multiple shortcuts to other interfaces, like Dashboard, Create Exercise, Edit Profile or log out from the website.
- **Create Exercise**: this view contains the form to create an exercise, indicating title, type, instructions, student and the elements relative to the type of exercise. It also contains the sidebar mentioned in Dashboard.
- **Edit Exercise**: it is the as Create Exercise interface, the only difference is that the fields are already filled with the information stored.
- **Exercise Result**: this interface will display the title of the exercise and the student's name, the elements relative to the type of the exercise with the answer of the student, indicating if they are correct or wrong. Will also show the grade and result obtained by the student. It also contains the sidebar mentioned in Dashboard.
- **Edit Profile**: contains a form with the fields email, password, name, and last name filled. If the user desires to change some of the information, he can do it by submitting this form. It also contains the sidebar mentioned in Dashboard.
- **Solve Exercise**: will display the title of the exercise and the student's name, the elements relative to the type of the exercise and an element to indicate the answer. Once the solution is sent, it will appear if answers are correct or wrong and will show the grade and result obtained by the student. Now, the answers can't be changed. It also contains the sidebar mentioned in Dashboard.

Solve exercise is the only interface that corresponds to the student user, the rest are relative to the teacher. To take a look at the pictures of the interfaces, go to section 6.1 "Interfaces" in the original document.

7. Planning

Methodology

The selected methodology followed in the project was Scrum. This is because the planning was made around periods of two to four weeks, which are known as sprints. In each sprint new objectives were defined. After the end of the spring a meeting took place with the tutor, to review the results obtained.

On another note, a board from the tool Trello was used to classify tasks as “done”, “pending” or “in progress”, helping the author define priorities and keep track of the project’s evolution.

GANTT diagram

To see the project’s original planning go to section 7.2, where a GANTT diagram will be found, detailing how tasks were distributed over time.

Real Evolution

Following the initial planning was very difficult, so adjustments were made as time went by. For reasons that are not related to the project, the development suffered some delays. It is worth mentioning that the project wasn’t a full-time work, since the author was doing an internship, attending to university classes and working as a part-time English tutor, which lead to working during spare time hours that could change every day. As the author’s responsibilities decreased, more time could be spent in developing the project.

8. Legal Framework

Laws

It is important to note that this project, being for personal use, does not handle real information or data from real users. At the moment it is all hypothetical data to test the functionalities of the application. In case it is published on the Internet and starts to handle real data, all these aspects will be taken into consideration by the author to take the necessary measures.

Supposing that this project is published on the Internet and can be accessed by anyone, since it handles and stores sensitive data from users, it has to meet some Spanish and European regulations. Some of them are:

- LOPD-GDD (Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales).
- RGPD (Reglamento Europeo 2016/679 de Protección de Datos).
- Meet the “Web Legal Notice” which implies that every website must indicate its terms and conditions, its privacy policy, its cookies policy and, if appropriate, its contracting conditions. These must be visible and accessible to all users.

Licenses

The licenses for the tools that were used to develop this project are:

- MIT License for React and Node.
- Server-Side Public license for MongoDB.
- Creative Commons Attribution-Share Alike 3.0 United States for Express.
- Free Content licenses, part of the Canva's Content License Agreement.

9. SOCIO-ECONOMIC ENVIRONMENT

Project budget

For a breakdown of the total costs of the project go to section 9.1 “Presupuesto” in the original document, where each cost is addressed.

Costs	Gross cost
Human resources	8.892,90 €
Equipment	1.797,14 €
Indirect costs	2.138,01 €
TOTAL	**12.828,05 €**

Tabla 91. Total costs

Socio-economic impact

The main impact of this project will fall on a small group of people, consisting of: the author of the project, his students, the author's acquaintances and the students of the latter. One hand, teachers will have at their disposal a tool that will allow them to create customized exercises for their students and keep track of them, which will speed up the preparation of a class and make it more dynamic. On the other hand, the students will be able to learn English in a more modern way that can even be more attractive, since they use technological devices to solve their exercises, instead of the traditional way of using paper and pencil or pen.

For the same reason, if this project is published to the Internet so that it could be accessed by anyone, it can be a great innovation within the sector formed by people related to English classes, especially for English tutors, as it is most likely that a tutor does not have the same resources as a school or academy teacher.

10. CONCLUSIONS

Project conclusions

After the development period of the project, it is possible to say that the main objectives were fulfilled. The web application developed is capable of:

- Creating customized exercises and managing them.
- Share exercises with a unique link.
- Visualize the results obtained by the students, to keep track of their progress.
- Access an exercise to solve it without needing to sign up. In other words, easy and direct access.

It is also worth mentioning that new knowledge about web development was acquired.

Another important note is that this project served as an option to apply all the knowledge acquired during the degree of computer science engineering, in a real-world case, and not in a hypothetical case as it is done with activities and evaluations during the course.

Project's future

In the first place, new types of exercise will be developed, such as listening and association of elements. Also new functionalities will be added so the tracking of the results is enhanced, making it easier for teachers to evaluate the progress of the students.

Since a first final version of the software exists, it will be used and tested during classes, this way errors can be spotted and fixed, and open a space for new ideas of functionalities to come in. As a long-term goal, is offer teachers and students a platform with a wide catalog of exercises and tools for the assessment and tracking of students

Lista de referencias

[1] «Kahoot! | Learning games | Make learning awesome!» [En línea]. Disponible en: https://kahoot.com/. [Accedido: 22-jun-2022].

[2] «Learn English Online | British Council». [En línea]. Disponible en: https://learnenglish.britishcouncil.org/. [Accedido: 22-jun-2022].

[3] «Wooclap - Una plataforma interactiva que hace que el aprendizaje sea fascinante». [En línea]. Disponible en: https://www.wooclap.com/es/. [Accedido: 22-jun-2022].

[4] MDN contributors, «HTML: Lenguaje de etiquetas de hipertexto | MDN», 20-abr-2021. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/HTML. [Accedido: 19-jun-2022].

[5] MDN contributors, «CSS | MDN», 07-jul-2021. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/CSS. [Accedido: 19-jun-2022].

[6] MDN contributors, «JavaScript | MDN», 30-may-2021. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/JavaScript. [Accedido: 19-jun-2022].

[7] «Framework - Wikipedia, la enciclopedia libre». [En línea]. Disponible en: https://es.wikipedia.org/wiki/Framework. [Accedido: 19-jun-2022].

[8] «Empezando – React». [En línea]. Disponible en: https://es.reactjs.org/docs/getting-started.html. [Accedido: 19-jun-2022].

[9] «Presentando JSX – React». [En línea]. Disponible en: https://es.reactjs.org/docs/introducing-jsx.html. [Accedido: 19-jun-2022].

[10] MDN contributors, «XML: Extensible Markup Language | MDN», 15-jun-2022. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/XML. [Accedido: 19-jun-2022].

[11] «What is Babel? · Babel». [En línea]. Disponible en: https://babeljs.io/docs/en/. [Accedido: 19-jun-2022].

[12] «What is React». [En línea]. Disponible en: https://www.w3schools.com/whatis/whatis_react.asp. [Accedido: 03-jun-2022].

[13] MDN contributors, «SPA (Single-page application) - MDN Web Docs Glossary: Definitions of Web-related terms | MDN», 08-oct-2021. [En línea]. Disponible en:

https://developer.mozilla.org/en-US/docs/Glossary/SPA. [Accedido: 19-jun-2022].

[14] «Angular - Introduction to the Angular Docs». [En línea]. Disponible en: https://angular.io/docs. [Accedido: 19-jun-2022].

[15] Fireship, «Angular in 100 Seconds - YouTube», 09-sep-2020. [En línea]. Disponible en: https://www.youtube.com/watch?v=Ata9cSC2WpM. [Accedido: 03-jun-2022].

[16] «What is Angular». [En línea]. Disponible en: https://www.w3schools.com/whatis/whatis_angularjs.asp. [Accedido: 03-jun-2022].

[17] «Introduction | Vue.js». [En línea]. Disponible en: https://vuejs.org/guide/introduction.html. [Accedido: 19-jun-2022].

[18] «What is Vue.js». [En línea]. Disponible en: https://www.w3schools.com/whatis/whatis_vue.asp. [Accedido: 03-jun-2022].

[19] Fireship, «Vue.js Explained in 100 Seconds - YouTube», 03-abr-2020. [En línea]. Disponible en: https://www.youtube.com/watch?v=nhBVL41-_Cw. [Accedido: 03-jun-2022].

[20] «¿Qué es una API? - Guía sobre las API para principiantes - AWS». [En línea]. Disponible en: https://aws.amazon.com/es/what-is/api/. [Accedido: 19-jun-2022].

[21] «Documentación | Node.js». [En línea]. Disponible en: https://nodejs.org/es/docs/. [Accedido: 19-jun-2022].

[22] Programming with Mosh, «What is Node js? - YouTube». [En línea]. Disponible en: https://www.youtube.com/watch?v=uVwtVBpw7RQ. [Accedido: 03-jun-2022].

[23] «What is an API Endpoint? | API Endpoint Definition | RapidAPI». [En línea]. Disponible en: https://rapidapi.com/blog/api-glossary/endpoint/. [Accedido: 19-jun-2022].

[24] «El concepto de middleware», 21-mar-2018. [En línea]. Disponible en: https://www.redhat.com/es/topics/middleware. [Accedido: 19-jun-2022].

[25] «Express - Infraestructura de aplicaciones web Node.js». [En línea]. Disponible en: https://expressjs.com/es/. [Accedido: 03-jun-2022].

[26] «Django documentation | Django». [En línea]. Disponible en: https://docs.djangoproject.com/en/4.0/. [Accedido: 19-jun-2022].

[27] «Python 3.10.5 documentation». [En línea]. Disponible en: https://docs.python.org/3/. [Accedido: 19-jun-2022].

[28] MDN contributors, «MVC - Glosario | MDN», 19-feb-2020. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Glossary/MVC. [Accedido: 03-jun-2022].

[29] IBM Technology, «What is Django? - YouTube», 27-dic-2021. [En línea]. Disponible en: https://www.youtube.com/watch?v=t_p4ZyAYyaY. [Accedido: 03-jun-2022].

[30] «Spring Boot Reference Documentation». [En línea]. Disponible en: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/. [Accedido: 19-jun-2022].

[31] «GlassFish». [En línea]. Disponible en: https://javaee.github.io/glassfish/documentation. [Accedido: 19-jun-2022].

[32] Telusko, «What is Spring Boot? | Introduction - YouTube», 23-may-2018. [En línea]. Disponible en: https://www.youtube.com/watch?v=Ch163VfHtvA. [Accedido: 03-jun-2022].

[33] «Spring Framework Documentation». [En línea]. Disponible en: https://docs.spring.io/spring-framework/docs/current/reference/html/index.html. [Accedido: 19-jun-2022].

[34] «¿Qué son los microservicios? | AWS». [En línea]. Disponible en: https://aws.amazon.com/es/microservices/. [Accedido: 19-jun-2022].

[35] R. James, «Node.js vs. Spring Boot — Which Should You Choose? | by Ryan James | Better Programming», 08-ene-2020. [En línea]. Disponible en: https://betterprogramming.pub/node-js-vs-spring-boot-which-should-you-choose-2366c2f76587. [Accedido: 03-jun-2022].

[36] V. Bhatia, «Introduction to JSP - GeeksforGeeks», 02-jul-2018. [En línea]. Disponible en: https://www.geeksforgeeks.org/introduction-to-jsp/. [Accedido: 19-jun-2022].

[37] «EJS - Embedded JavaScript templates». [En línea]. Disponible en: https://ejs.co/.

[Accedido: 19-jun-2022].

[38] L. De Seta, «ACID en las bases de datos», 19-feb-2013. [En línea]. Disponible en: https://dosideas.com/noticias/base-de-datos/973-acid-en-las-bases-de-datos. [Accedido: 03-jun-2022].

[39] «Qué es una base de datos relacional | Oracle España». [En línea]. Disponible en: https://www.oracle.com/es/database/what-is-a-relational-database/. [Accedido: 03-jun-2022].

[40] «Explicación Sobre Las Bases De Datos NoSQL | MongoDB». [En línea]. Disponible en: https://www.mongodb.com/es/nosql-explained. [Accedido: 03-jun-2022].

[41] IBM Cloud Education, «¿Qué es el teorema de CAP? - España | IBM». [En línea]. Disponible en: https://www.ibm.com/es-es/cloud/learn/cap-theorem. [Accedido: 03-jun-2022].

[42] Rubenfa, «NoSQL: Clasificación de las bases de datos según el teorema CAP», 28-ene-2014. [En línea]. Disponible en: https://www.genbeta.com/desarrollo/nosql-clasificacion-de-las-bases-de-datos-segun-el-teorema-cap. [Accedido: 03-jun-2022].

[43] Ayusharma0698, «Difference between SQL and NoSQL - GeeksforGeeks», 31-may-2021. [En línea]. Disponible en: https://www.geeksforgeeks.org/difference-between-sql-and-nosql/. [Accedido: 06-jun-2022].

[44] Santander Universidades, «Metodologías de desarrollo software | Blog Becas Santander», 21-dic-2020. [En línea]. Disponible en: https://www.becas-santander.com/es/blog/metodologias-desarrollo-software.html. [Accedido: 03-jun-2022].

[45] Udacity, «Spiral Process - Georgia Tech - Software Development Process», 23-feb-2015. [En línea]. Disponible en: https://www.youtube.com/watch?v=mp22SDTnsQQ. [Accedido: 03-jun-2022].

[46] Udacity, «Evolutionary Prototyping Process - Georgia Tech - Software Development Process - YouTube», 23-feb-2015. [En línea]. Disponible en: https://www.youtube.com/watch?v=bAEnaGG8Otc&list=PLAwxTw4SYaPkNAtqsKcFkUGpf4j67NBaz&index=30. [Accedido: 03-jun-2022].

[47] D. Radigan, «Kanban: una breve introducción | Atlassian». [En línea]. Disponible en: https://www.atlassian.com/es/agile/kanban. [Accedido: 03-jun-2022].

[48] C. Drumond, «Scrum: qué es, cómo funciona y por qué es excelente». [En línea]. Disponible en: https://www.atlassian.com/es/agile/scrum. [Accedido: 03-jun-2022].

[49] «¿Qué es el lenguaje unificado de modelado (UML)? | Lucidchart». [En línea]. Disponible en: https://www.lucidchart.com/pages/es/que-es-el-lenguaje-unificado-de-modelado-uml. [Accedido: 22-jun-2022].

[50] «Introducción a JSON». [En línea]. Disponible en: https://www.json.org/json-es.html. [Accedido: 19-jun-2022].

[51] «Mongoose v6.4.0: API docs». [En línea]. Disponible en: https://mongoosejs.com/docs/api.html#. [Accedido: 19-jun-2022].

[52] MDN Contributors, «Visión General Cliente-Servidor - Aprende sobre desarrollo web | MDN». [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Learn/Server-side/First_steps/Client-Server_overview. [Accedido: 23-jun-2022].

[53] MDN contributors, «HTTP | MDN», 08-dic-2020. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/HTTP. [Accedido: 19-jun-2022].

[54] IBM Cloud Education, «¿Qué es una API REST? - España | IBM», 06-abr-2021. [En línea]. Disponible en: https://www.ibm.com/es-es/cloud/learn/rest-apis. [Accedido: 19-jun-2022].

[55] B.O. del Estado, «Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales.», 2018. [En línea]. Disponible en: https://www.boe.es/boe/dias/2018/12/06/pdfs/BOE-A-2018-16673.pdf. [Accedido: 09-jun-2022].

[56] D. O. de la Unión Europea, «REGLAMENTO (UE) 2016/ 679 DEL PARLAMENTO EUROPEO Y DEL CONSEJO - de 27 de abril de 2016 - relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos y por el que se deroga», 2016. [En línea]. Disponible en: https://www.boe.es/doue/2016/119/L00001-00088.pdf. [Accedido: 09-jun-2022].

[57] «¿Qué aspectos legales debe cumplir una página web? –», *¿Qué aspectos legales debe cumplir una página web?*, 13-jun-2019. [En línea]. Disponible en: https://www.grupovadillo.com/aspectos-legales-pagina-web/. [Accedido: 09-jun-2022].

[58] R. Cabezudo, «Marco normativo aplicable a tu Web Site – ReDigital», *Marco normativo aplicable a tu Web Site*, 12-abr-2021. [En línea]. Disponible en: https://redigital.economistas.es/marco-normativo-aplicable-a-tu-web-site. [Accedido: 09-jun-2022].

[59] «MIT License | Choose a License». [En línea]. Disponible en: https://choosealicense.com/licenses/mit/. [Accedido: 20-jun-2022].

[60] «Creative Commons — Attribution-ShareAlike 3.0 United States — CC BY-SA 3.0 US». [En línea]. Disponible en: https://creativecommons.org/licenses/by-sa/3.0/us/. [Accedido: 20-jun-2022].

[61] «Server Side Public License (SSPL) | MongoDB». [En línea]. Disponible en: https://www.mongodb.com/licensing/server-side-public-license. [Accedido: 20-jun-2022].

[62] «Canva». [En línea]. Disponible en: https://www.canva.com/. [Accedido: 22-jun-2022].

[63] «Sueldo: Junior Developer - Full Stack (Junio, 2022) | Glassdoor». [En línea]. Disponible en: https://www.glassdoor.es/Sueldos/junior-full-stack-developer-sueldo-SRCH_KO0,27.htm. [Accedido: 22-jun-2022].

[64] «AWS MongoDB Pricing | MongoDB». [En línea]. Disponible en: https://www.mongodb.com/mongodb-on-aws-pricing. [Accedido: 22-jun-2022].

[65] «AWS | Cloud Computing - Servicios de informática en la nube». [En línea]. Disponible en: https://aws.amazon.com/es/. [Accedido: 22-jun-2022].

[66] «Precios de AWS Amplify | Servicios de frontend web y móviles | Amazon Web Services». [En línea]. Disponible en: https://aws.amazon.com/es/amplify/pricing/?nc=sn&loc=4. [Accedido: 22-jun-2022].

[67] «Servicios y costos». [En línea]. Disponible en: https://aws.amazon.com/es/getting-started/hands-on/deploy-nodejs-web-

app/services-costs/. [Accedido: 22-jun-2022].

[68] «Desarrollo en cascada - Wikipedia, la enciclopedia libre». [En línea]. Disponible en: https://es.wikipedia.org/wiki/Desarrollo_en_cascada. [Accedido: 06-jun-2022].

[69] Beao, «File:Software Development Spiral.svg - Wikimedia Commons». [En línea]. Disponible en: https://commons.wikimedia.org/wiki/File:Software_Development_Spiral.svg. [Accedido: 06-jun-2022].

[70] Y. Muradas, «Conoce las 3 metodologías ágiles más usadas | OpenWebinars», 08-mar-2018. [En línea]. Disponible en: https://openwebinars.net/blog/conoce-las-3-metodologias-agiles-mas-usadas/. [Accedido: 06-jun-2022].

[71] «React, Angular, Vue.js - Explorar - Google Trends». [En línea]. Disponible en: https://trends.google.es/trends/explore?q=%2Fm%2F012l1vxv,%2Fg%2F11c6w0ddw9,%2Fg%2F11c0vmgx5d. [Accedido: 03-jun-2022].