Universidad Politécnica
de Madrid

**Escuela Técnica Superior de
Ingenieros Informáticos**

Grado en Ingeniería Informática

Trabajo Fin de Grado

Desarrollo de Herramienta de Compilación y Almacenamiento de Corpus para la Exploración de Estrategias Discursivas Interculturales en Estudiantes Universitarios Europeos del Siglo XXI

Autor: Chencheng Zhan

Tutor(a): Jelena Bobkina

Madrid, Junio 2024

Este Trabajo Fin de Grado se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Grado
Grado en Ingeniería Informática
Título: Desarrollo de Herramienta de Compilación y Almacenamiento de Corpus para la Exploración de Estrategias Discursivas Interculturales en Estudiantes Universitarios Europeos del Siglo XXI
Junio 2024

Autor: Chencheng Zhan

Tutor: Jelena Bobkina
Departamento de Lingüística Aplicada a la Ciencia y a la Tecnología
ETSI Informáticos
Universidad Politécnica de Madrid

Resumen

El proyecto forma parte del proyecto RACISMMAFF (Estrategias de Posicionamiento en el Discurso del Racismo y la Inmigración: Análisis y Aplicaciones en Prácticas Afectivas de Aprendizaje) con fondos de Ministerio de Ciencia e Innovaciónl (MCIN) y Fondo Europeo de Desarrollo Regional (FEDER), liderada por las profesoras Elena Domínguez Romero de la Universidad Complutense de Madrid y Jelena Bobkina de la Universidad Politécnica de Madrid.

El objetivo del proyecto se trata de diseñar una estructura para la compilación, almacenamiento y organización eficiente de diferentes subcorpus de estudiantes europeos escritos en distintas lenguas, antes y después de completar un Curso Online Masivo y Abierto (MOOC). La estructura del proyecto se trata de una aplicación web que permite la administración y almacenamiento de subcorpus escritos de manera segura y eficiente para su posterior análisis e interpretación.[1]

La aplicación web diseñada tendrá funcionalidades claves como el registro de usuarios para facilitar el análisis de resultados y la recopilación de respuestas a preguntas abiertas y cerradas. Además, contará con un panel de control (dashboard) para los administradores, donde se podrán visualizar los resultados del análisis de las respuestas de todos los usuarios.

Por otra parte, la aplicación permitirá la modificación de las preguntas por parte de los administradores, lo que facilitará la evolución y adaptación de la plataforma a futuras investigaciones y necesidades.

Todo esto se desarrollará utilizando la técnica de MEAN Stack, con MongoDB como bases de datos, Firebase para el almacenamiento de las imágenes, HTML y Angular para el Front-End, y Express.js y Node.js para el Back-End.

Abstract

This project is part of RACISMMAFF (Stance Strategies in Immigration and Racism-Related Discourse: Analysis and Applications in Affective Learning Practices) funded by Spanish Ministry of Science and Innovation and European Regional Development Fund (ERDF), led by professors Elena Domínguez Romero from the Complutense University of Madrid and Jelena Bobkina from the Polytechnic University of Madrid. The project's objective is to design a structure for the efficient compilation, storage, and organization of various subcorpus written by European students in different languages, before and after completing a Massive Open Online Course (MOOC). The project structure involves a web application that facilitates the secure and efficient management and storage of written subcorpus for subsequent analysis and interpretation.

The designed web application will feature key functionalities such as user registration to facilitate result analysis and the collection of responses to open and closed questions. Additionally, it will be including an administrator dashboard, where the results of the user response analysis can be visualized. The application will also allow administrators to modify the questions, enabling the platform's evolution and adaptation to future research and needs.

The entire project will develop using the MEAN Stack technique, with MongoDB as the database, Firebase for image storage, HTML and Angular for the Front-End, and Express.js and Node.js for the Back-End.

Tabla de contenidos

1 Introducción

Este proyecto forma parte del proyecto RACISMMAFF, una iniciativa liderada por las profesoras Elena Domínguez Romero de la Universidad Complutense de Madrid y Jelena Bobkina de la Universidad Politécnica de Madrid. Junto con mi compañera Grettell Umpierrez, nos unimos a este proyecto con el objetivo de diseñar y desarrollar una aplicación web dedicada a investigar y aplicar estrategias de aprendizaje afectivas para abordar las problemáticas sobre el racismo en Europa. [2]

Figura 1. Icono RACISMMAFF

1.1 Motivación del proyecto

En la actualidad, los discursos negativos hacia la inmigración y el racismo constituyen un problema real y omnipresente en nuestras sociedades. Las palabras cargadas de prejuicios se difunden a través de múltiples canales, desde los medios de comunicación hasta las redes sociales, perpetuando estereotipos y fomentando divisiones. Esta problemática no solo afecta a los individuos que son objeto de discriminación, sino que también debilita el tejido social en su conjunto, creando barreras y desconfianza entre diferentes grupos. [3]

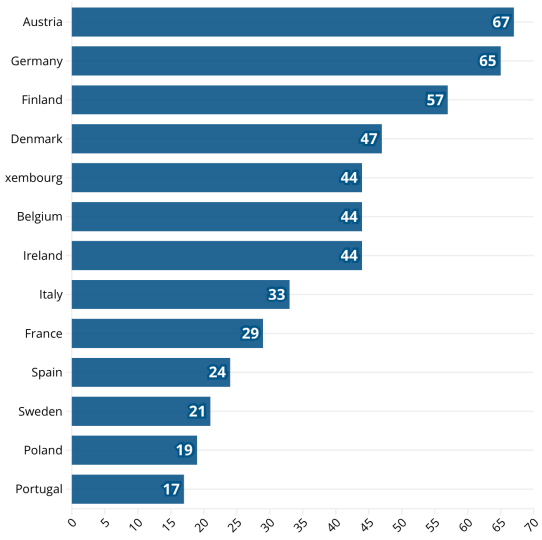


Figura 2. El pocentaje de las personas que sufrieron discriminación en los últimos 12 meses en países europeos

En la Figura 2 muestra el porcentaje de personas que han sufrido discriminación en los últimos 12 meses en varios países europeos, destacando

la urgencia de abordar estos problemas. Nuestra motivación surge de la necesidad de entender y contrarrestar estos discursos negativos. La creación de herramientas y estrategias educativas que fomentan la empatía y la comprensión es crucial para desafiar y transformar estas narrativas perjudiciales.

Como estudiantes universitarios y futuros educadores, reconocemos nuestro papel importante en la construcción de una sociedad donde todos sean respetados y valorados. Nos impulsa la idea de ser agentes de cambio positivo, promoviendo un dialogo más inclusivo y empático sobre la inmigración y el racismo.

1.2 Objetivos del proyecto

El objetivo central de nuestro proyecto es desarrollar una aplicación web que simplifique la recopilación, almacenamiento y análisis de datos lingüísticos de estudiantes europeos antes y después de su participación en un Curso Online Masivo y Abierto (MOOC). La aplicación permitirá:

- **Registro de Usuarios**: Facilitar la inscripción de los usuarios para el análisis posterior de resultados.

- **Recopilación de Respuestas:** Almacenar respuestas a preguntas abiertas y cerradas en una base de datos segura.

- **Panel de Control**: Presentar los resultados de análisis de a través de un Dashboard dirigido a los administradores.

- **Acceso a Contenidos Educativos**: Alojar videos y materiales educativos disponibles para los usuarios, para observar los cambios en sus percepciones sobre el racismo.

2 Estado del arte

Antes de diseñar la aplicación se realizó un estudio de los productos similares en el mercado actual con el objetivo de recopilar las ideas del diseño y encontrar áreas de mejora para nuestra aplicación web. A continuación, se describen algunas de las herramientas más relevantes y se analiza su utilidad y las limitaciones que presentan.

2.1 NVivo

NVIVO es una aplicación para el análisis cualitativo de datos que nos permite organizar, almacenar y analizar los datos textuales, incluidos documentos, encuestas, etc. NVico proporciona funcionalidades avanzadas para la codificacion de texto, el análisis temático y la visualización de datos mediante graficos y tablas. Su interfaz de usuario facilita el acceso y la manipulación de grandes volúmenes de datos, y su capacidad para integrar diversos tipos de datos la convierte en una herramienta poderosa para investigadores [4].

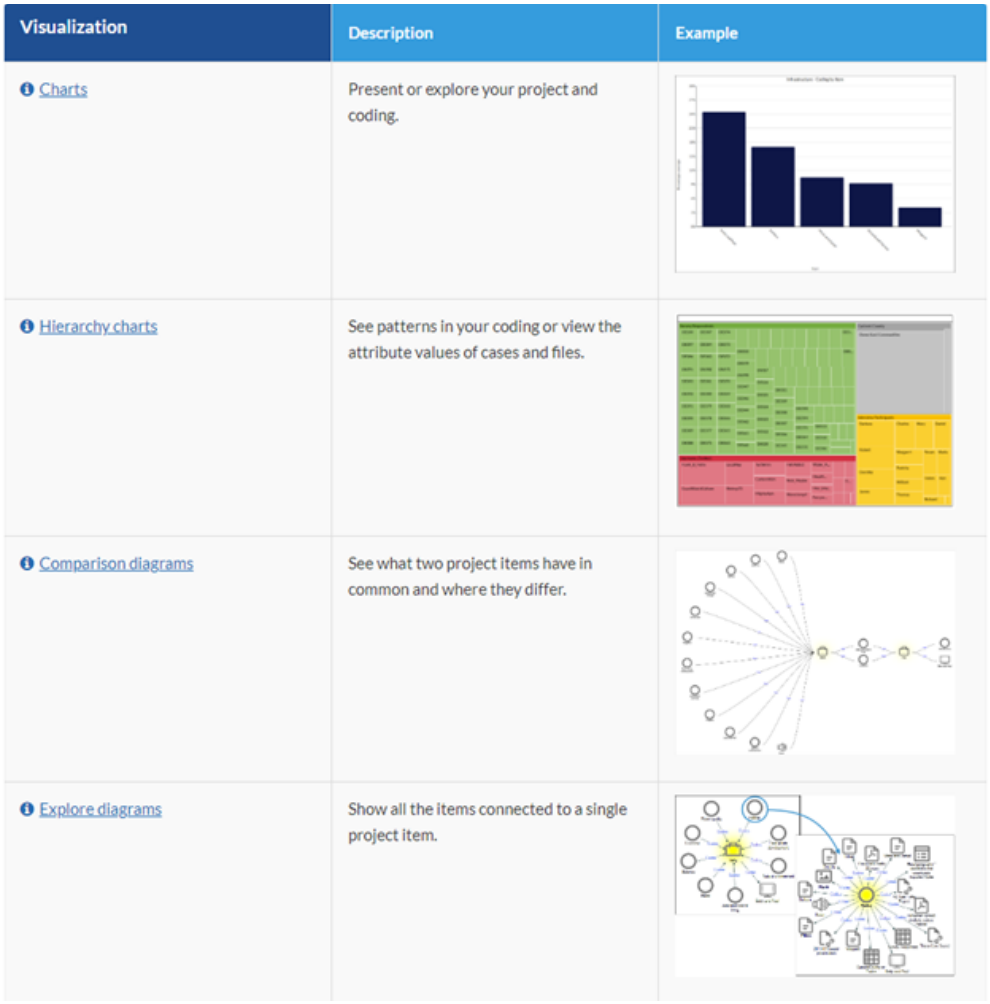


Figura 3. NVivo

Sin embargo, NVivo es una herramienta de pago, lo que limita su accesibilidad para instituciones y usuarios con recursos financieros limitados. Además, su curva de aprendizaje puede ser más alta para usuarios sin experiencia previa en análisis cualitativo, lo que podría dificultar su adopción en un entorno educativo más amplio.

2.2 Leximancer

Leximancer es una herramienta de análisis de texto automatizado que utiliza técnicas de minería de texto para identifica patrones y temas en grandes conjuntos de datos textuales lo que facilita al usuario las palabras claves del texto, así podrán saber cuál será lo más importante del texto en sí. [5]

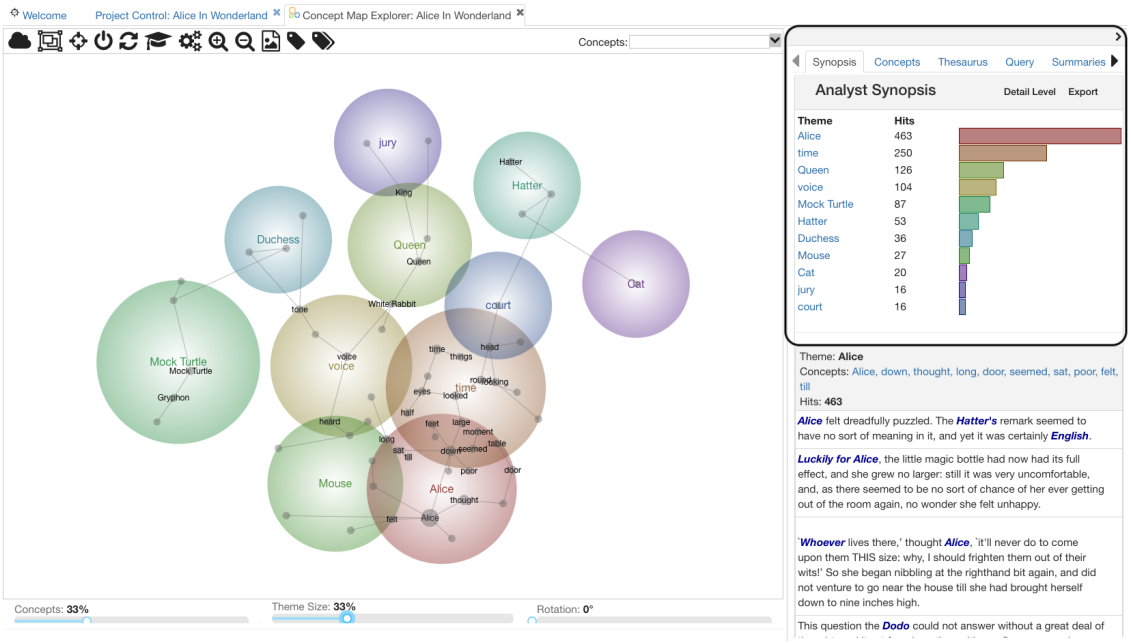


Figura 4. Leximancer

A pesar de sus capacidades avanzadas, Leximancer también es una herramienta de pago y ofrece solo un mes de uso gratuito, lo cual no es suficiente para investigaciones a largo plazo o para su integración en programas educativos continuos. Además, su enfoque automatizado puede no capturar todas las sutilezas del análisis cualitativo que un investigador podría identificar manualmente.

2.3 Linguistic Inquiry and Word Count (LIWC)

LIWC es un software que analiza automáticamente los textos en función de las palabras utilizadas y las categorías lingüísticas, como emociones o estilos de expresión. Es especialmente útil para estudios psicológicos y lingüísticos, proporcionando una visión detallada de la frecuencia y el contexto de diferentes tipos de palabras [6].

Affective or emotional processes	**Abbrev (affect)**	**Examples (happy, ugly, bitter)**	**# Words 615**
Positive emotions	Posemo	happy, pretty, good	261
Positive feelings	Posfeel	happy, joy, love	43
Optimism and energy	Optim	certainty, pride, win	69
Negative emotions	Negmo	hate, worthless, enemy	345
Anxiety or fear	Anx	nervous, afraid, tense	62
Anger	Anger	hate, kill, pissed	121
Sadness or depression	Sad	grief, cry, sad	72
Time	**Abbrev (time)**	**Examples (hour, day, clock)**	**# Words 113**
Past tense verb	Past	walked, were, had	144
Present tense verb	Present	walk, is, be	256
Future tense verb	Future	will, might, shall	14
Leisure activity	**Abbrev (leisure)**	**Examples (house, TV, music)**	**# Words 113**
Home	Home	house, kitchen, lawn	26
Sports	Sports	football, game, play	28
Television and movies	TV	TV, sitcom, cinema	19
Music	Music	tunes, song, CD	31

Figura 5. Linguistic Inquiry and Word Count

LIWC, similar a NVivo y Leximancer, es una herramienta de pago y su funcionalidad puede ser limitada en comparación con las necesidades específicas de nuestro proyecto. Si bien es excelente para el análisis de texto enfocado en categorías predefinidas, puede no ofrecer la flexibilidad necesaria para explorar nuevas áreas de investigación o para adaptar sus categorías a diferentes contextos culturales y lingüísticos.

2.4 Conclusión

El estudio de estas aplicaciones revela una amplia variedad de funciones disponibles, pero también destaca algunas limitaciones significativas. Todas las aplicaciones mencionadas son de pago, lo cual limita su accesibilidad para un uso educativo y de investigación sin grandes recursos financieros. Además, la necesidad de una curva de aprendizaje y la falta de flexibilidad en algunas de estas herramientas impiden su reutilización directa para nuestro proyecto.

Nuestro proyecto busca desarrollar una solución personalizada que combine las mejores características de estas herramientas (como el panel de control de NVivo y el análisis basado en minería de texto de Leximancer) con un enfoque más accesible y adaptable a las necesidades específicas de la investigación educativa sobre el racismo en Europa.

3 Requisitos

3.1 Introducción

Una vez fijados los objetivos de la aplicación, en este capítulo se va a presentar los requisitos principales de la plataforma. El desarrollo del proyecto está bajo el esfuerzo conjunto de la compañera Grettell Umpierrez y el propio autor, los requisitos indicados serán los conjuntos de la aplicación.

3.2 Requisitos Funcionales

3.2.1 Consulta de funcionalidades de DeStance

3.2.1.1 Requisitos por parte de la compañera Grettell Umpierrez

1) Consulta. Bienvenida. Prioridad: Alta. El sistema proporcionará a los usuarios una descripción de la herramienta y una guía de uso mediante pasos intuitivos.

2) Consulta. Conceptos. Prioridad: Alta. El sistema ofrecerá a los usuarios explicaciones detalladas de los siguientes conceptos: Postura Epistémica, Postura Efectiva, Postura Emocional, Empatía, Enseñanza y Aprendizaje Afectivo, Competencia Intercultural y Rutinas de Pensamiento Socioemocional.

3) Consulta. About the tool. Prioridad: Alta. El Sistema presentará a los usuarios una serie de respuestas a las preguntas más frecuentes.

4) Consulta. Cuestionarios. Preguntas. Prioridad: Alta. Inicialmente, el sistema ofrecerá a los usuarios tres pre-cuestionarios y tres post-cuestionarios, cada uno con una serie de preguntas que tienen cinco opciones de respuesta, evaluadas en una escala del 1 al 5.

5) Consulta. Cuestionarios. Respuestas. Prioridad: Alta. El sistema mostrará las respuestas a cada pregunta de los cuestionarios completados por el usuario. Además, se presentará el título del resultado obtenido junto con una descripción del contenido asociado.

6) Consulta. Dashboard. Cuestionarios. Prioridad: Alta. El sistema proporcionará a los administradores estadísticas de cada pre y post cuestionario basadas en las respuestas de los usuarios almacenadas en la Base de Datos. Además, se incluirán estadísticas adicionales según los criterios de nacionalidad, edad, género, lengua materna y el idioma de respuesta del Curso Online Masivo y Abierto, así como su nivel de competencia.

7) Consulta. Autorización. Prioridad Alta. Las opciones de administración como dashboards y mantenimiento de cuestionarios, estarán deshabilitadas para los usuarios que no sean administradores, por lo que es necesario diferenciar los tipos de usuarios en la Base de Datos.

3.2.1.2 Requisitos por parte del propio autor

8) Consulta. MOOC. Dashboard. Prioridad: Alta: El sistema mostrará las respuestas de los usuarios y facilitará en gráfica los tipos de palabras basado en minería de texto.

9) Consulta. MOOC. Cursos. Prioridad: Alta. Permite a los usuarios ver detalles específicos de cada curso, incluyendo contenidos del curso y las preguntas.

3.2.2 Comunicación bidireccional con los usuarios

3.2.2.1 Requisitos por parte de la compañera Grettell Umpierrez

1) Cuestionarios. Respuesta. Prioridad: Alta. El sistema permitirá a los usuarios responder cada pregunta de los cuestionarios seleccionando la opción que consideren adecuada. Una vez completado un cuestionario, las respuestas se almacenarán en la Base de Datos utilizando una escala del 1 al 5. Además, se inferirá una conclusión basada en la media de las respuestas. Cada cuestionario solo podrá ser respondido una vez.

2) Cuestionarios. Mantenimiento. Automatización. Prioridad: Baja. El sistema permitirá a los usuarios administradores modificar y eliminar cuestionarios existentes.

3) Usuarios. Datos. Prioridad: Alta. El sistema creará una cuenta para cada usuario al registrarse recopilando nombre, correo electrónico, contraseña, fecha de nacimiento, género, nacionalidad, lengua materna, lenguaje que utilizará para responder el MOOC y nivel de competencia que tiene en dicho lenguaje, de A1 a C2.

    a. Usuarios. Datos. Modificación. Prioridad: Baja. El sistema permitirá a los usuarios modificar sus datos personales introducidos anteriormente al crear su cuenta.

    b. Usuarios. Datos. Contraseña. Recuperación. Prioridad: Baja. El sistema permitirá a los usuarios recuperar su contraseña. Para ello deberán introducir un correo electrónico de recuperación que coincida con el registrado. El sistema enviará un código al correo proporcionado que deberá ser usado para validar la nueva contraseña.

3.2.2.2 Requisitos por parte del propio autor

4) MOOC. Respuestas. Mostrar Preguntas. Prioridad: Alta. El sistema mostrará las preguntas específicas en diferentes etapas de cada curso.

5) MOOC. Respuestas. Enviar Respuestas. Prioridad: Alta. El sistema permitirá a los usuarios enviar sus respuestas a las preguntas del curso.

6) MOOC. Respuestas. Validación de Respuestas. Prioridad: Alta. Validar que la respuesta cumple el número mínimo de palabras.

7) MOOC. Respuestas. Guardar Respuestas. Prioridad: Alta. El sistema permitirá almacenar las respuestas de usuarios en la Base de Datos.

8) MOOC. Respuestas. Mostrar Feedback. Prioridad: Alta. El sistema dispone de retroalimentación al usuario sobre el análisis de su respuesta.

9) MOOC. Curso. Descargar Archivos. Prioridad: Alta. Los archivos de cada curso estarán disponibles para los usuarios.

3.2.3 Requisitos de Interfaces Externos

3.2.3.1 Interfaces de Usuario

Prioridad: Alta. La interfaz de usuario debe ser accesible a través de un navegador web y diseñada de manera atractiva para asegurar una buena experiencia de usuario.

3.2.3.2 Interfaces de Software

Prioridad: Baja. Es necesario conectar el sistema con la aplicación externa SendGrid [7] para el envío de correos electrónicos de recuperación de contraseña.

3.2.3.3 Interfaces de Comunicación

Prioridad: Alta. Utilización de los servicios e infraestructura proporcionada por Firebase [8] para el alojamiento de los servidores web (frontend y backend) de DeStance.

3.2.4 Requisitos de Desarrollo

Prioridad: Alta. El ciclo de vida de desarrollo elegido para el producto será iterativo incremental [9], permitiendo la fácil incorporación de cambios y nuevas funciones. Se definirá el orden de implementación de las funcionalidades en cada ciclo.

3.3 Requisitos No Funcionales

3.3.1 Requisito de Disponibilidad y Rendimiento

Prioridad: Alta. El sistema estará disponible 24/7. Debe tener un tiempo de respuesta inferior a 2 segundos para el 95% de las solicitudes y soportar al menos 100 usuarios concurrentes sin degradación significativa del rendimiento.

3.3.2 Requisito de Mantenibilidad

Prioridad: Alta. El sistema debe requerir el mínimo mantenimiento posible. El código debe estar estructurado de manera modular para facilitar las actualizaciones y el mantenimiento.

3.3.3 Requisito de Accesibilidad

Prioridad: Alta. La interfaz de usuario debe ser intuitiva y fácil de usar, con una navegación clara y coherente. Además, debe cumplir con los estándares WCAG 2.1 AA [10]. Para asegurar la accesible a usuarios con discapacidades.

3.3.4 Requisito de Seguridad

Prioridad: Alta, Implementar autenticación mediante email y contraseña, y permisos basados en roles de usuario y administrador para asegurar que solo los usuarios autorizados accedan a determinadas funcionalidades. Todos los

datos sensibles deben estar encriptados tanto en tránsito (usando HTTP) como en reposo. La aplicación debe cumplir con las regulaciones pertinentes como el GDPR [11], y otros estándares de privacidad y seguridad de datos aplicables.

3.3.5 Requisito de Compatibilidad

Prioridad: Alta. La aplicación debe ser compatible con las versiones actuales y anteriores de los navegadores más utilizados (Chrome, Firefox, Safari, Edge).

3.3.6 Requisito de Idioma

Prioridad: Alta. La aplicación debe estar disponible en inglés y soportar la traducción ofrecida por los navegadores.

4 Tecnologías

Dados los requisitos anteriormente planteados, en esta sección se va a indicar las diversas tecnologías utilizadas en el proyecto.

Este proyecto sigue el patrón de desarrollo del MEAN Stack [12], el término "MEAN" es un acrónimo que representa MongoDB, Express.js, Angular, y Node.js. A continuación, explicaremos en detalle cada una de estas tecnologías, así, como otras herramientas auxiliares que se emplean en el desarrollo de esta aplicación.

4.1 Entorno de Trabajo

El entorno de trabajo comprende todas las herramientas, frameworks, bibliotecas, y servicios utilizados para desarrollar, probar y desplegar la aplicación. Se incluye tanto el software utilizado de desarrollo como los servicios de infraestructura.

4.1.1 Sistema Operativo

La mayoría de los desarrolladores de códigos utilizan sistemas operativos basados en Unix [13], como macOS o Linux, debido a su compatibilidad nativa con herramientas de desarrollo web y de línea de comandos. Sin embargo, en nuestro proyecto hemos elegido al Windows para el desarrollo local porque no quisiésemos desarrollarlo en una Máquina Virtual de Linux para evitar problemas con la capacidad del computador.

Figura 6: Sistema Operativo: Windows

4.1.2 Control de versiones

Para el desarrollo de proyecto en equipo siempre es necesario utilizar control de versiones, conocidos como "control de código fuente" [14], son herramientas de software que facilitan a los equipos de desarrollo a gestionar los cambios en el código fuente del desarrollo. En nuestro caso hemos elegido Git [15] por su compatibilidad con la plataforma GitHub.

4.1.2.1 Git

El control de versiones nos manejamos con Git, y el repositorio se aloja en la plataforma GitHub. Git permite a los desarrolladores colaborar de manera efectiva, gestionando ramas, revisiones de código y fusiones de ramas [16].

Figura 7: Control de Versiones: Git

4.1.3 Editores de Código y Entornos de Desarrollo Integrados (IDE)

4.1.3.1 Visual Studio Code

Visual Studio Code [17] es el editor de código más utilizado debido a su soporte extensible para diversos lenguajes de programación y su integración con herramientas de desarrollo web.

VS Code es muy popular entre la comunidad frontend por su adaptación a lenguajes como HTML, CSS y JavaScript. También soporta una amplia variedad de frameworks frontend a través de extensiones como en nuestro caso a Angular Framework [18].

Figura 8: Editor de Código: VS Code

4.1.4 Gestión de Dependencias

4.1.4.1 Node Package Manager (NPM)

El gestor de paquetes NPM [19] se utiliza para manejar las dependencias de Node.js y Angular. NPM es una herramienta esencial en el desarrollo con JavaScript, su función principal es gestionar las dependencias, es decir, las librerías y módulos que los desarrolladores utilizan en sus proyectos para evitar empezar desde cero.

Permite agregar fácilmente nuevas librerías y módulos a un proyecto y además facilita la publicación de módulos propios para que otros desarrolladores puedan utilizarlos [20].

Figura 9: NPM

4.1.5 Frameworks

Un framework es un esquema o marco de trabajo que proporciona una estructura base para desarrollar un proyecto con objetivos específicos. Es como una plantilla que facilita la organización y el desarrollo de software, ofreciendo un punto de partida estructurado [21].

4.1.5.1 Node.js

Node.js es un entorno de ejecución para JavaScript del lado del servidor que utiliza un modelo de entrada y salida sin bloque, controlado por eventos, es ligero y eficiente. Este modelo permite gestionar las solicitudes (entrada) y las respuestas (salida) de manera efectiva, aplicable a diversas operaciones, como leer o escribir archivos y realizar solicitudes HTTP [22].

Figura 10: Node.js

4.1.5.2 Express.js

Express.js es el framework backend ampliamente utilizado para Node.js. Está diselado para desarrollar aplicaciones web de una sola página, multipágina e híbridas, y se ha convertido en el estándar para la creación de aplicaciones backend con Node.js [23].

Figura 11: Express.js

4.1.5.3 Angular

Angular es un framework frontend de JavaScript de código abierto escrito en TypeScript desarrollado por Google, que permite la creación de aplicaciones web dinámicas y escalables [24].

Figura 12: Angular.js

4.1.6 Base de Datos

Una base de datos es una herramienta que recopila datos, los organiza y los relaciona para que se puedan realizar búsquedas rápidas y recuperarlos de manera eficiente. Se pueden diferenciar dos tipos de bases de datos, los de datos relacionales, que utilizan tablas para organizar datos, como por ejemplo MySQL o PostgreSQL, o los de datos No Relacionales, que están diseñadas para manejar datos no estructurados y semiestructurados, como, por ejemplo, MongoDB [25].

4.1.6.1 MongoDB

MongoDB es una base de datos No Relacional (NoSQL) de código abierto que destaca por su capacidad para procesar datos estructurados, semiestructurados y no estructurados. MongoDB utiliza un modelo de datos no relacional orientado a documentos y un lenguaje de consulta no estructurado, lo que le confiere una gran flexibilidad y capacidad para manejar diversos tipos de datos [26].

Figura 13: MongoDB

4.1.7 Herramientas de Prueba

4.1.7.1 Postman

Postman es una herramienta de desarrollo para interactuar y probar servicios web y aplicaciones. Proporciona una interfaz gráfica intuitiva para enviar solicitudes y recibir respuestas, gestionar entornos de desarrollo, organizar solicitudes en colecciones y realizar pruebas automatizadas. Es fundamental para gestionar el ciclo de vida de las APIs y organizar el proceso de diseño y desarrollo [27].

Figura 14: Postman

4.2 Lenguaje de programación

El proyecto utiliza varios lenguajes de programación y lenguaje de marcado, cada uno adecuado para diferentes partes del sistema.

4.2.1 JavaScript y TypeScript

JavaScript es el lenguaje de programación más utilizado para el desarrollo web [28]. JavaScript es crucial para ambos lados del desarrollo web, tanto en el cliente (frontend) como en el servidor [29].

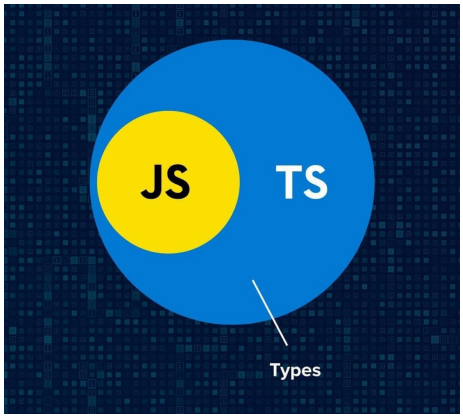


Figura 15: Diferencia entre JS y TS

4.2.1.1 Frontend

En la parte del Frontend, Angular utiliza TypeScript, es un superconjunto de JavaScript [30]. TypeScript añade tipado estático y otras características avanzadas a JavaScript, mejorando la productividad del desarrollador y la mantenibilidad del código.

4.2.1.2 Backend

En cuanto al Backend, Node.js permite ejecutar JavaScript en el lado del servidor. Express.js, un framework minimalista para Node.js, facilita la creación de aplicaciones web y APIS.

4.2.2 HTML 5

HTML5 es el lenguaje de marcado estándar para crear páginas web y aplicaciones web. HTML5 ofrece nuevas funcionalidades para estructuras semánticas y capacidades multimedia sin la necesidad de plugins adicionales [31].

Figura 16: Lenguaje HTML

4.2.3 CSS

CSS3 Es un lenguaje utilizado para el diseño y la presentación visual de las páginas web. CSS3 introduce nuevas características como flexbox, grid layouts y animaciones, que mejoran significativamente el diseño responsivo y dinámico [32].

Figura 17: Lenguaje CSS

4.2.4 MongoDB Query Language (MQL)

MQL utilizado para interactuar con MongoDB, permite realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) y otras consultas complejas en una base de datos NoSQL [33].

5 Diseño

El diseño es una fase crucial en el desarrollo de software, ya que establece la estructura y las relaciones entre los componentes del sistema [34]. Esta sección proporciona una descripción detallada de los actores involucrados, los casos de uso, la arquitectura del sistema y las interfaces necesarias para la implementación del proyecto.

5.1 Actores

Los actores son entidades externas que interactúan con el sistema, cada uno con roles y responsabilidades específicos. En el contexto de nuestro proyecto, los actores serán los usuarios y sistemas externos que se comunican con nuestra plataforma.

5.1.1 Usuarios

5.1.1.1 Usuario Registrado

Los usuarios registrados disponen de accesos funcionalidades personalizadas, pueden crear, leer y actualizar sus propios datos. Podrán acceder a características avanzadas según su perfil.

Las funcionalidades limitadas en este perfil de usuario pueden ver contenidos públicos, pueden realizar los cursos y responder las preguntas.

5.1.1.2 Usuario Administrador

Los usuarios administradores podrán acceder a todas las funciones del sistema, podrán gestionar los permisos de usuarios. Podrán acceder a las respuestas de los resultados, mantenimiento y administración de las preguntas.

5.1.2 Sistemas Externos

5.1.2.1 Bases de Datos Externas

La plataforma se comunica con sistemas de bases de datos para la importación o exportación de datos.

5.2 Diagrama de Casos de Uso

El diagrama de casos de uso visualiza las interacciones entre los actores y el sistema, mostrando las funciones que cada actor puede realizar [35]. Este diagrama es fundamental para comprender las funcionalidades y alcance del sistema desde el punto de vista del usuario.

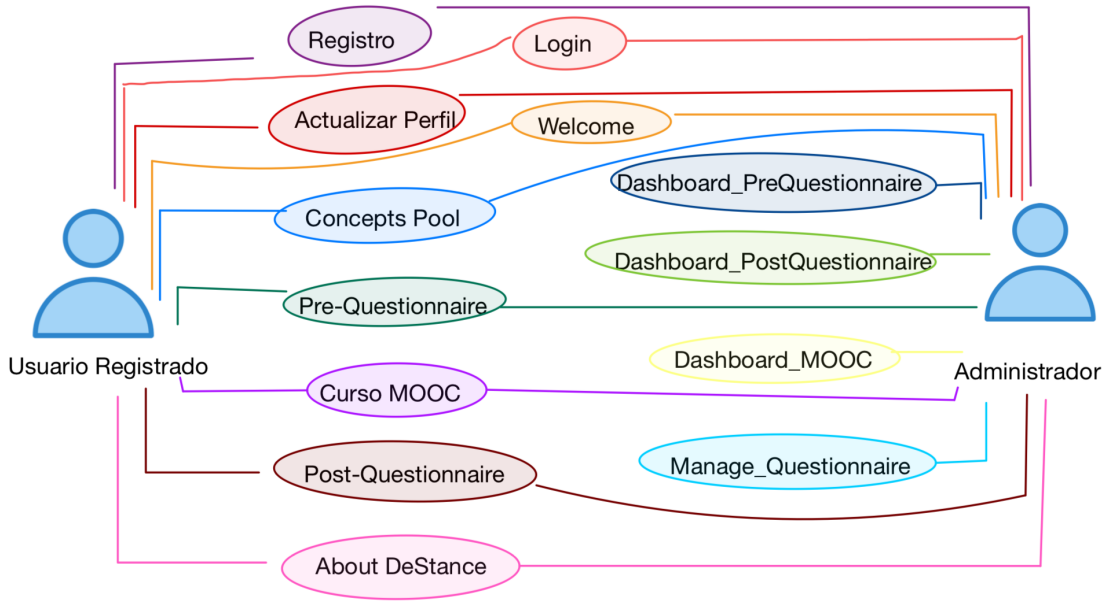


Figura 18: Diagrama de Casos de uso

5.3 Arquitectura

La arquitectura del sistema define la estructura general del software, especificando cómo se organizan los componentes y cómo se interactúan entre ellos [36]. Para este proyecto, adoptamos una arquitectura basada en microservicios, que ofrece escalabilidad y flexibilidad.

5.3.1 Componentes Principales

5.3.1.1 Frontend

El componente frontend está desarrollado con Angular framework que nos permite desarrollar una interfaz de usuario dinámica y responsiva. Este componente se interactúa con el backend a través de API RESTful.

5.3.1.2 Backend

El componente backend está implementado con Node.js y Express.js. Proporciona la lógica de la plataforma y se comunica con la base de datos MongoDB.

5.3.1.3 Base de Datos

En el base de datos tenemos a MongoDB como base de datos NoSQL para el almacenamiento de datos, es flexible y capaz de manejar grandes volúmenes de datos.

5.3.1.4 Middleware

El componente middleware son manejadores de solicitudes que aplican lógica antes de que lleguen a las rutas del backend, especificadamente vamos a usar para la autenticación, validación de datos y manejo de errores.

5.3.2 Diagrama de Arquitectura

La diagrama de arquitectura para los proyectos que siguen el patrón de desarrollo MEAN stack es lo siguiente [37]:

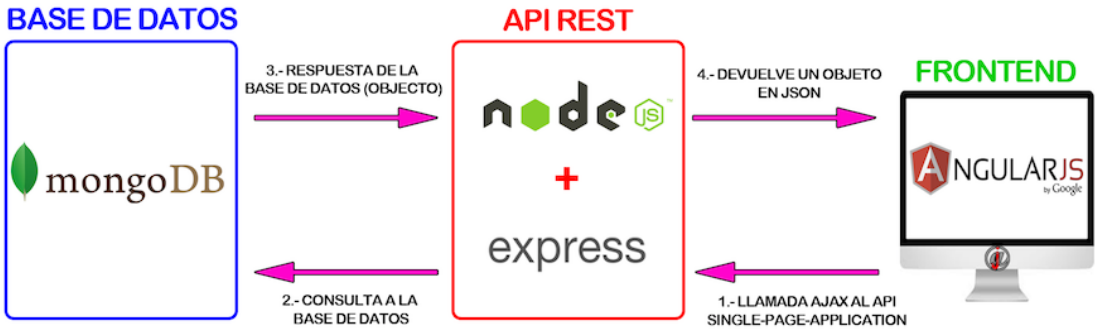


Figura 19: Diagrama de Arquitectura en general de MEAN Stack

En nuestro caso, añadiremos el componente Middleware que servirá de la autenticación y autorización, que se quedaría de la siguiente forma:

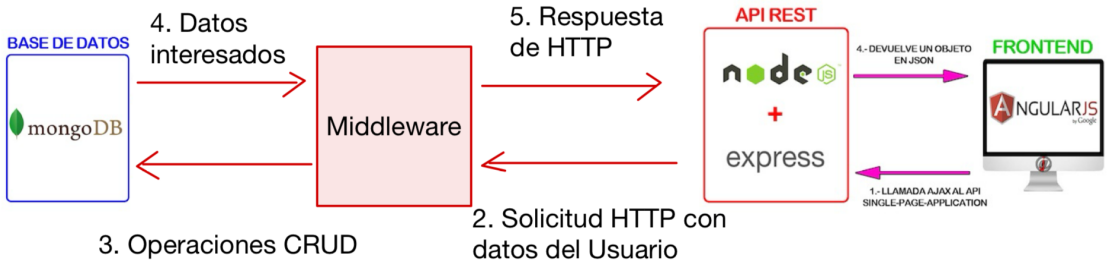


Figura 20: Diagrama de Arquitectura en nuestro caso

5.4 Interfaces

Las interfaces de usuario y de sistema son para garantizar la interacción efectiva entre los usuarios y el sistema, así como diferentes componentes del sistema.

5.4.1 Interfaces de Usuario

5.4.1.1 Panel de Administración

En el panel de administración dispondrá de los usos del:

- Gestión de Preguntas: interfaz para crear, editar, eliminar las preguntas.
- Administración de Dashboards: panel de control que muestra los resultados de cada pregunta, los detalles de los usuarios registrados y tendrán también el acceso a las respuestas de los usuarios.

5.4.1.2 Portal de Usuarios Registrados

En el portal de usuarios registrados tendrán las funcionalidades de:

- Gestión de Perfil: Interfaz para manejar los datos personales.
- Gestión de Cuestionarios: Interfaz que muestra las preguntas y tendrán de responder seleccionando una respuesta.
- Gestión de Cursos: Interfaz que muestra la imagen, el video, las preguntas del curso, y los materiales del curso.
- Visualización de Contenidos Informativos: Interfaz que muestra la información detallada de la plataforma.

5.4.2 Interfaces del Sistema Externos

5.4.2.1 API RESTful

En la interfaz de API RESTful se dispone de ciertas funcionalidades [38]:

- Endpoints CRUD: rutas para manejar operaciones de crear, leer, actualizar y eliminar datos.
- Autenticación y Autorización: Rutas para manejo de autenticación y autorización de usuarios.
- Gestión de Sesiones: Endpoints específicos para la gestión de permisos y roles de usuarios.
- Manejo de Errores: Rutas y respuestas estándar para el manejo de errores, proporcionando mensajes de error y códigos de estado HTTP adecuados.

5.4.2.2 SendGrid

La conexión con la plataforma SendGrid para el uso de envío de correos de confirmación de la recuperación de contraseña.

5.4.2.3 Firebase

El uso de la plataforma Firebase para el alojamiento del servidor y almacenamiento de las imágenes e iconos.

5.4.3 Diagrama de Interfaces

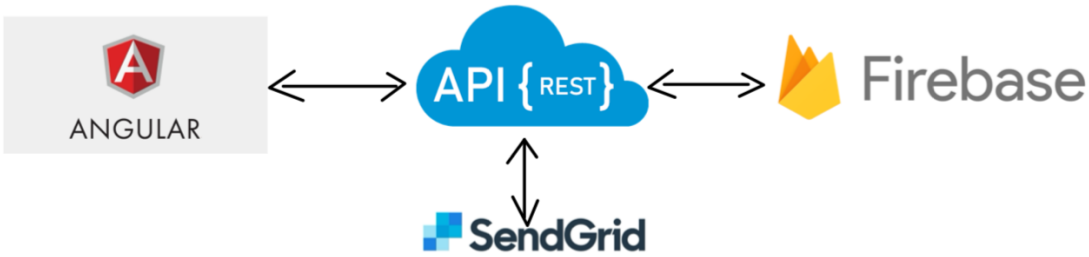


Figura 21: Diagrama de Interfaces

6 Implementación

En esta sección se detalla la implementación de la aplicación web, incluyendo la estructura del proyecto, el código relevante y las imágenes de las funcionalidades para proporcionar una comprensión visual de cómo funciona el sistema.

6.1 Estructura

En esta subsección se describe la estructura del proyecto de la aplicación, detallando cómo se organizan los archivos y directorios principales.

- Root/
    - Backend/: Contiene los códigos del servidor backend
        - Controllers/: Controladores que manejan la lógica de las rutas
            - About-us.js: Controlador para manejar las rutas y lógica de About-us.
            - Auth.js: Controlador para manejar las rutas y lógica de Authentication.
            - Concept-pool.js: Controlador para manejar las rutas y lógica de Concept-Pool.
            - Dashboards.js: Controlador para manejar las rutas y lógica de Dashboards
            - Questionnaires.js: Controlador para manejar las rutas y lógica de Questionnaires.
            - User-questionnaires.js: Controlador para manejar las rutas y lógica de User-Questionnaires.
            - Course1.js: Controlador para manejar las rutas y lógica de Course1.
            - Course2.js: Controlador para manejar las rutas y lógica de Course2.
            - Course3.js: Controlador para manejar las rutas y la lógica de Course3.
            - User.js: Controlador para manejar las rutas y la lógica de los usuarios.
        - Middleware/: Lógica del autenticación y autorización.
            - Check-admin.js: Middleware para verificar si el usuario es administrador.

        - Check-auth.js: Middleware para verificar la autenticación del usuario.

- Models/: Definición de los modelos de datos.
    - About-us.js: Modelo de datos para About-us.
    - Auth.js: Modelo de datos para Auth.
    - Concept-pool.js: Modelo de datos para Concept-pool.
    - Dashboards.js: Modelo de datos para Dashhboards.
    - Questionnaires.js: Modelo de datos para Questionnaires.
    - User-questionnaires.js: Modelo de datos para User-Questionnaires.
    - Course1.js: Modelo de datos para Course1.
    - Course2.js: Modelo de datos para Course2.
    - Course3.js: Modelo de datos para Course3.
    - User.js: Modelo de datos para los usuarios.
- Routes/: Definición de las rutas de la API.
    - About-us.js: Rutas de la API para About-us.
    - Auth.js: Rutas de la API para Auth.
    - Concept-pool.js: Rutas de la API para Concept-pool.
    - Dashboards.js: Rutas de la API para Dashboards.
    - Questionnaires.js: Rutas de la API para Questionnaires.
    - User-questionnaires.js: Rutas de la API para User-Questionnaires.
    - Course1.js: Rutas de la API para Course1.
    - Course2.js: Rutas de la API para Course2.
    - Course3.js: Rutas de la API para Course3.
    - User.js: Rutas de la API para usuarios.
- Shared/: Contiene funciones externas para facilitar la llamada a estas.

        - Auxiliar-function.js: Funciones auxiliares compartidas en la aplicación.

    - App.js: Archivo principal del servidor

- Src/: Código fuente de la aplicación
    - App/: Contiene los componentes principales de la aplicación
        - About-us/: Componentes relacionados con la pagina de About-us.
        - Authentication/ Componentes para el manejo de la autenticación de usuarios.
        - Concepts-pool/: Componentes relacionados con el módulo de conceptos.
        - Dashboards/: Componentes para los Dashboards.
        - Foote/: Componentes para el pie de página.
        - Header/: Componentes para el encabezado de la aplicación.
        - Not-Found/: Componentes para la página de Not-Found.
        - Questionnaires/: Componentes para manejar los cuestionarios.
        - Shared/: Componentes compartidos entre diferentes partes de la aplicación.
        - User/: Componentes para el manejo de usuarios.
        - Welcome/: Componentes para la página de Welcome.
        - Dashboard-mooc/: Componentes específicos para los Dashboards de los cursos.
        - Mooc/: Componentes específicos para el módulo de cursos.
        - Sidebar/: Componentes para la barra lateral de navegación
        - Angular-material.module.ts: Configuración y módulos de Angular Material.
        - App-routing.module.ts: Configuracion de las rutas principales de la aplicación.

- App.component.html: Plantilla principal del componente de la aplicación
- App.module.ts: Módulo principal de la aplicación
- App.component.css: Estilos CSS para el componente principal de la aplicación.
- Error.Interceptor.ts: Interceptor para manejar errores globales en la aplicación.

▪ Assets/: Archivo donde suelen guardar las imágenes o iconos.

▪ Environments/: Determina en qué puerto vamos a trabajar

o Node_modules/: Contiene las dependencias de Node.js

o Package.json: Archivo de configuración de las dependencias del proyecto.

6.2 Códigos

6.2.1 Backend

6.2.1.1 Controllers

Los controladores gestionan la lógica de negocio y las rutas del servidor. A continuación, se presentan ejemplos de controladores:

```
const Course1 = require('../models/course1');
const User = require('../models/users');

exports.createCourse1 = async (req, res) => {
    try {
        const motherTongue = req.body.motherTongue;
        const pre1 = req.body.pre1;
        const post1 = req.body.post1;

        const course = new Course1({ motherTongue, pre1, post1 });
        await course.save();
        res.status(201).json(course);
    } catch (error) {
        res.status(400).json({ message: error.message });
    }
};

exports.getCourses1 = async (req, res) => {
    try {
        const courses = await Course1.find();
        res.status(200).json(courses);
    } catch (error) {
        res.status(404).json({ message: error.message });
    }
};
```

Figura 22: Ejemplo de Controller, en este caso es el de Course1.js

En el ejemplo podemos observar las rutas de las actividades en Course1. En ella, la función createCourse1 se usa para guardar las respuestas y la lengua materna del usuario para los futuros análisis.

6.2.1.2 Models

Los Models son los modelos de datos definidos para crearlo en MongoDB. A continuación, se presentan ejemplos de Models:

```
const mongoose = require('mongoose');

const Course1Schema = new mongoose.Schema({
    motherTongue: {type: String, required: true},
    pre1: {type: String, required: true},
    post1: { type: String, required: true}
});

module.exports = mongoose.model('Course1', Course1Schema);
```

Figura 23: Ejemplo de Models, en este caso es el de Course1.js

6.2.1.3 Middleware

El middleware maneja la autenticación y autorización. A continuación, se presentan ejemplos de middleware:

```
const jwt = require("jsonwebtoken");

module.exports = (req, res, next) => {
  const token = req.headers.authorization.split(" ")[1];
  const tokenDec = jwt.decode(token);
  if (!tokenDec.admin) {
    return res
      .status(401)
      .json({ message: "Auth failed!. User is not an admin" });
  }
  next();
};
```

Figura 24: Ejemplo de Middleware, en este caso es el de Check-admin.js

En el ejemplo de check-admin, verificamos si el usuario es administrador, si no lo es, devuelve un estatus 401.

6.2.1.4 Routes

Las rutas definen los endpoints de la API. A continuación, se presentan ejemplos de rutas:

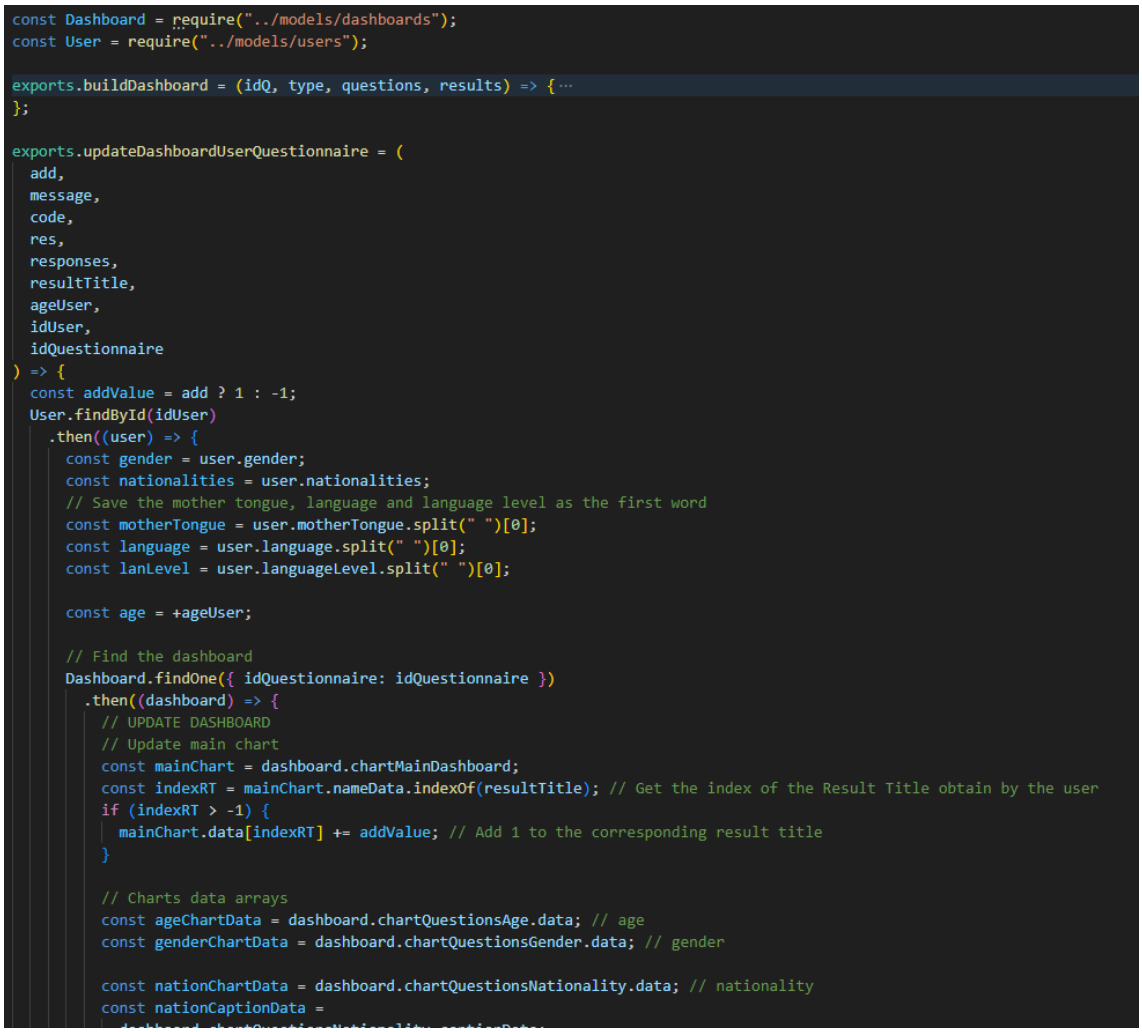
```
const express = require("express");
const router = express.Router();

const Course1Controller = require("../controllers/course1");

router.post("", Course1Controller.createCourse1);
router.get("", Course1Controller.getCourses1);

module.exports = router;
```

Figura 25: Ejemplo de Routes, en este caso es el de Course1.js

En el ejemplo de Course1, en la ruta definimos las acciones de HTTP que sería POST y GET que utilizan las funciones anteriormente implementadas en el controller.

6.2.1.5 Shared

Funciones auxiliares compartidas en la aplicación:



```
const Dashboard = require("../models/dashboards");
const User = require("../models/users");

exports.buildDashboard = (idQ, type, questions, results) => {···
};

exports.updateDashboardUserQuestionnaire = (
  add,
  message,
  code,
  res,
  responses,
  resultTitle,
  ageUser,
  idUser,
  idQuestionnaire
) => {
  const addValue = add ? 1 : -1;
  User.findById(idUser)
    .then((user) => {
      const gender = user.gender;
      const nationalities = user.nationalities;
      // Save the mother tongue, language and language level as the first word
      const motherTongue = user.motherTongue.split(" ")[0];
      const language = user.language.split(" ")[0];
      const lanLevel = user.languageLevel.split(" ")[0];

      const age = +ageUser;

      // Find the dashboard
      Dashboard.findOne({ idQuestionnaire: idQuestionnaire })
        .then((dashboard) => {
          // UPDATE DASHBOARD
          // Update main chart
          const mainChart = dashboard.chartMainDashboard;
          const indexRT = mainChart.nameData.indexOf(resultTitle); // Get the index of the Result Title obtain by the user
          if (indexRT > -1) {
            mainChart.data[indexRT] += addValue; // Add 1 to the corresponding result title
          }

          // Charts data arrays
          const ageChartData = dashboard.chartQuestionsAge.data; // age
          const genderChartData = dashboard.chartQuestionsGender.data; // gender

          const nationChartData = dashboard.chartQuestionsNationality.data; // nationality
          const nationCaptionData =
```

Figura 26: Ejemplo de funciones externas que comparte en el archivo Shared

Disponemos varias funciones compartidas en el cual se utilizan para actualizar el Dashboard.

6.2.1.6 App.js

Archivo principal del servidor:

```
mongoose
  .connect(
    "mongodb+srv://" +
      process.env.MONGO_ATLAS_USER +
      ":" +
      process.env.MONGO_ATLAS_PASSWORD +
      "@clusterracismmaff.gz2qarq.mongodb.net/racismmaff-db?retryWrites=true&w=majority&appName=ClusterRACISMMAFF"
  )
  .then(() => {
    console.log("Connected to database!");
  })
  .catch(() => {
    console.log("Connection failed!");
  });

app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));

app.use((req, res, next) => {
  res.setHeader("Access-Control-Allow-Origin", "*");
  res.setHeader(
    "Access-Control-Allow-Headers",
    "Origin, X-Requested-With, Content-Type, Accept, Authorization"
  );
  res.setHeader(
    "Access-Control-Allow-Methods",
    "GET, POST, PATCH, PUT, DELETE, OPTIONS"
  );
  next();
});

// Routes for about-us content
app.use("/api/about-us", aboutUsRoutes);
// Routes for auth
app.use("/api/auth", authRoutes);
// Ruotes for concept-pool content
app.use("/api/concept-pool", conceptPoolRoutes);
// Routes for dashboards
app.use("/api/dashboards", dashboardsRoutes);
// Routes for questionnaires
app.use("/api/questionnaires", questRoutes);
// Routes for users and users/questionnaires
app.use("/api/users", usersRoutes);
// Routes for MOOC
app.use("/api/course1", course1Routes);
app.use("/api/course2", course2Routes);
app.use("/api/course3", course3Routes);
module.exports = app;
```

Figura 27: Una parte de los códigos de App.js

En el archivo App.js realizamos la conexión con MongoDB a través del mongoose. Una vez conectada con la Base de Datos tenemos por dónde alojar los datos de cada API, en la imagen se demuestra el uso a través del app.use para declarar cada API.

6.2.2 Src

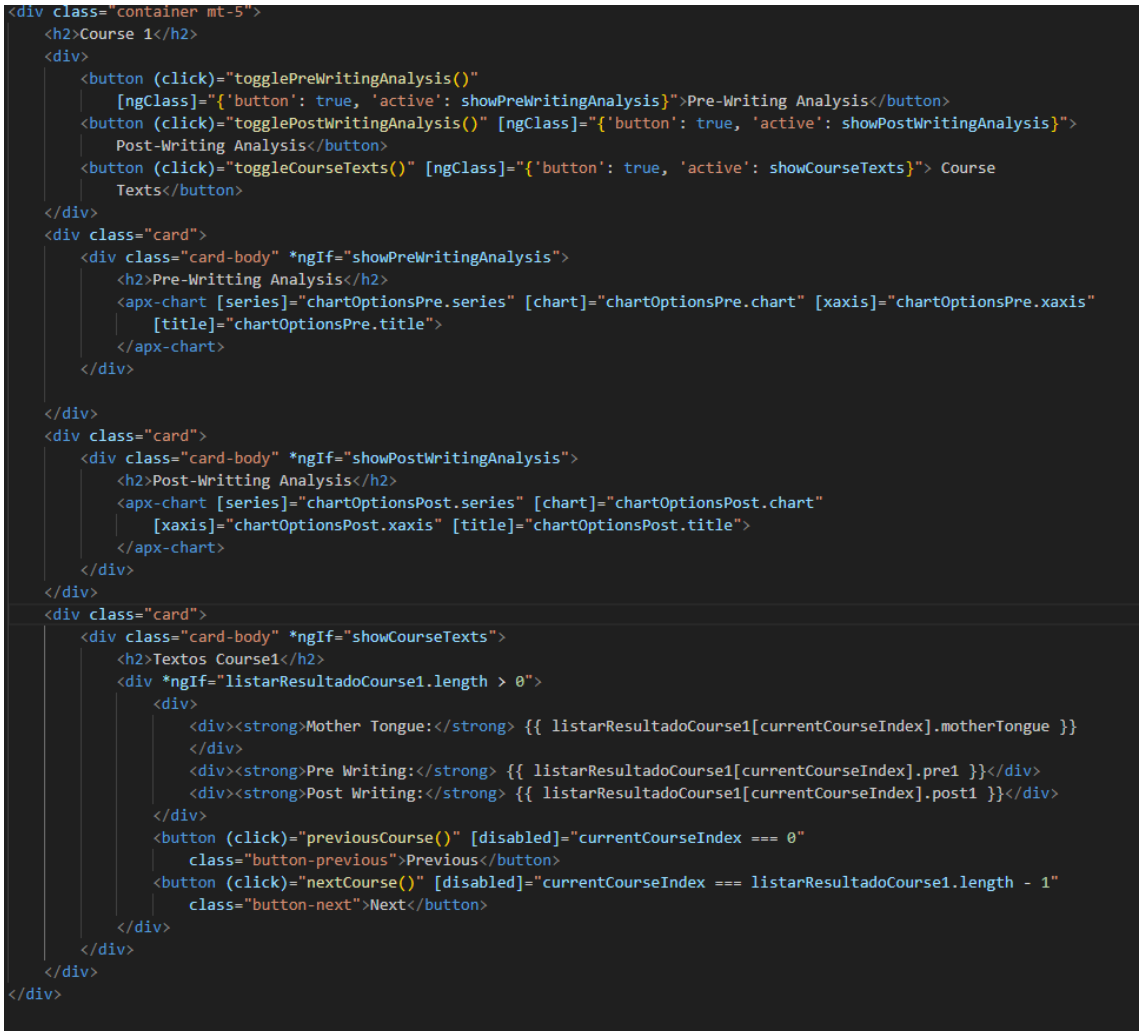
6.2.2.1 Dashboard-MOOC

Componentes específicos para el panel de control del módulo de cursos y la página principal del Dashboard-MOOC sería lo siguiente:

```
<div class="container-fluid">
    <div class="row">
        <div class="col-md-5 col-sm-10">
            <div class="row">
                <mat-card class="col-xs-12">
                    <mat-card-header>
                        <mat-card-title><h2>Dashboards for <i>MOOC</i></h2></mat-card-title>
                        <mat-card-subtitle><h3></h3></mat-card-subtitle>
                    </mat-card-header>

                    <mat-card-content>
                                <mat-action-list>
                                    <button mat-list-item routerLink="/dashboard-mooc/course1">
                                        <span matListItemTitle>Course 1</span>
                                        <span matListItemIcon><mat-icon>radio_button_unchecked</mat-icon></span>
                                    </button>
                                    <button mat-list-item routerLink="/dashboard-mooc/course2">
                                        <span matListItemTitle>Course 2</span>
                                        <span matListItemIcon><mat-icon>radio_button_unchecked</mat-icon></span>
                                    </button>
                                    <button mat-list-item routerLink="/dashboard-mooc/course3">
                                        <span matListItemTitle>Course 3</span>
                                        <span matListItemIcon><mat-icon>radio_button_unchecked</mat-icon></span>
                                    </button>
                                </mat-action-list>
                    </mat-card-content>
                </mat-card>
            </div>

        </div>
        <div class="col-md-8 col-sm-15 detail ">
            <router-outlet></router-outlet>
        </div>
    </div>
</div>
```

Figura 28: La página principal de Dashboard-MOOC, en este caso es el dashboard-mooc.component.html

En la página principal se disponen de tres opciones a elegir, cada una de las opciones llevan a su correspondiente página de Dashboard. A continuación, mostraremos un ejemplo de ello:



```
<div class="container mt-5">
    <h2>Course 1</h2>
    <div>
        <button (click)="togglePreWritingAnalysis()"
            [ngClass]="{'button': true, 'active': showPreWritingAnalysis}">Pre-Writing Analysis</button>
        <button (click)="togglePostWritingAnalysis()" [ngClass]="{'button': true, 'active': showPostWritingAnalysis}">
            Post-Writing Analysis</button>
        <button (click)="toggleCourseTexts()" [ngClass]="{'button': true, 'active': showCourseTexts}"> Course
            Texts</button>
    </div>
    <div class="card">
        <div class="card-body" *ngIf="showPreWritingAnalysis">
            <h2>Pre-Writting Analysis</h2>
            <apx-chart [series]="chartOptionsPre.series" [chart]="chartOptionsPre.chart" [xaxis]="chartOptionsPre.xaxis"
                [title]="chartOptionsPre.title">
            </apx-chart>
        </div>

    </div>
    <div class="card">
        <div class="card-body" *ngIf="showPostWritingAnalysis">
            <h2>Post-Writting Analysis</h2>
            <apx-chart [series]="chartOptionsPost.series" [chart]="chartOptionsPost.chart"
                [xaxis]="chartOptionsPost.xaxis" [title]="chartOptionsPost.title">
            </apx-chart>
        </div>
    </div>
    <div class="card">
        <div class="card-body" *ngIf="showCourseTexts">
            <h2>Textos Course1</h2>
            <div *ngIf="listarResultadoCourse1.length > 0">
                <div>
                    <div><strong>Mother Tongue:</strong> {{ listarResultadoCourse1[currentCourseIndex].motherTongue }}
                    </div>
                    <div><strong>Pre Writing:</strong> {{ listarResultadoCourse1[currentCourseIndex].pre1 }}</div>
                    <div><strong>Post Writing:</strong> {{ listarResultadoCourse1[currentCourseIndex].post1 }}</div>
                </div>
                <button (click)="previousCourse()" [disabled]="currentCourseIndex === 0"
                    class="button-previous">Previous</button>
                <button (click)="nextCourse()" [disabled]="currentCourseIndex === listarResultadoCourse1.length - 1"
                    class="button-next">Next</button>
            </div>
        </div>
    </div>
</div>
```

Figura 29: Página principal del dashboard1, en este caso, es el dashboard1.component.html

El código HTML lo utilizamos para organizar los componentes en vista del usuario.

```css
.button-previous {
    background-color: #4CAF50; /* Verde */
    color: white;
    border: none;
    padding: 10px 20px;
    text-align: center;
    text-decoration: none;
    display: inline-block;
    font-size: 16px;
    margin: 4px 2px;
    cursor: pointer;
}

.button-next {
    background-color: #008CBA;
    color: white;
    border: none;
    padding: 10px 20px;
    text-align: center;
    text-decoration: none;
    display: inline-block;
    font-size: 16px;
    margin: 4px 2px;
    cursor: pointer;
}
.button {
    padding: 10px 20px;
    margin-right: 10px;
    background-color: deepskyblue;
    color: white;
    border: none;
    cursor: pointer;
    transition: background-color 0.3s ease;
}

.button.active {
    background-color: gray;
    color: white;
}
.button-previous:disabled, .button-next:disabled {
    background-color: #cccccc;
    cursor: not-allowed;
}
```

Figura 30: Estilo del dashboard, en este caso es el dashboard1.component.css

Con el código CSS definimos los estilos y detalles.

```typescript
@Component({
  selector: 'app-dashboard1',
  templateUrl: './dashboard1.component.html',
  styleUrl: './dashboard1.component.css'
})
export class Dashboard1Component implements OnInit {
  @ViewChild("chartPre") chartPre: ChartComponent;
  @ViewChild("chartPost") chartPost: ChartComponent;
  public chartOptionsPre: Partial<ChartOptions>;
  public chartOptionsPost: Partial<ChartOptions>;
  showPreWritingAnalysis: boolean = false;
  showPostWritingAnalysis: boolean = false;
  showCourseTexts: boolean = false;

  currentCourseIndex = 0;
  listarResultadoCourse1: Course1[] = [];
  allPre1: string = '';
  allPost1: string = '';


  keywordGroups = {
    NonPersonalcognitiveFactives: [
      "the truth that", "the truth is that", "the fact that", "the fact is that", "the reality is that", "in truth, in fact, in reality", "really, it is true",
      "it remains true", "it is known that", "it's known that", "it is well known that", "it's well known that", "everyone knows that", "the international community
      "one knows, one is aware", "one remembers", "one forgets", "who will deny that", "is well known"
    ],
    PersonalCognitiveFactives:[
      "I know, we know", "I remember", "we remember", "I recall", "we recall", "I forget", "we forget", "I recognize", "we recognize",
      "I accept", "we accept", "I acknowledge", "we acknowledge", "I admit", "we admit", "remind me", "reminds me", "remind us", "reminds us"
    ],
    NonPersonalCognitiveAttitude: [
      "one thinks", "it is thought", "it' s thought", "it is believed", "it's believed", "one understands", "it is understood", "it's understood",
      "one discovers", "one finds out", "it is discovered", "it's discovered", "it can be said", "it could be said", "it might be said", "supposedly",
      "presumably", "naturally", "frankly", "admittedly", "no doubt", "undoubtedly", "doubtless", "conceivably", "it is conceivable", "it's conceivable",
      "it is unarguable", "it's unarguable", "it is plausible", "it's plausible", "of course", "for sure", "indeed, predictably",
    ],
```

Figura 31: La lógica del dashboard1, en este caso es el dashboard1.component.ts

Y finalmente con el TypeScript añadimos lógica para distintos eventos de la página.

6.2.2.2 MOOC

Componentes específicos para los cursos, disponemos de una página principal con tres opciones para cada curso.

```html
<div class="container-fluid">
    <div class="row">
        <div class="col-md-5 col-sm-10">
            <div class="row">
                <mat-card class="col-xs-12">
                    <mat-card-header>
                        <mat-card-title><h2>Welcome back to <i>MOOC</i></h2></mat-card-title>
                        <mat-card-subtitle><h3></h3></mat-card-subtitle>
                    </mat-card-header>

                    <mat-card-content>
                            <mat-action-list>
                                <button mat-list-item routerLink="/mooc/Course1">
                                    <span matListItemTitle>Session 1 - Epistemic Stance</span>
                                    <span matListItemIcon><mat-icon>radio_button_unchecked</mat-icon></span>
                                </button>
                                <button mat-list-item routerLink="/mooc/Course2">
                                    <span matListItemTitle>Session 2 - Emotional Stance</span>
                                    <span matListItemIcon><mat-icon>radio_button_unchecked</mat-icon></span>
                                </button>
                                <button mat-list-item routerLink="/mooc/Course3">
                                    <span matListItemTitle>Session 3 - Effective Stance</span>
                                    <span matListItemIcon><mat-icon>radio_button_unchecked</mat-icon></span>
                                </button>
                            </mat-action-list>
                    </mat-card-content>
                </mat-card>
            </div>

        </div>
        <div class="col-md-8 col-sm-15 detail ">
            <router-outlet></router-outlet>
        </div>
    </div>
</div>
```

Figura 32: Página principal del MOOC, mooc.component.html

Una vez seleccionado el curso, entremos en detalle de la página del curso.

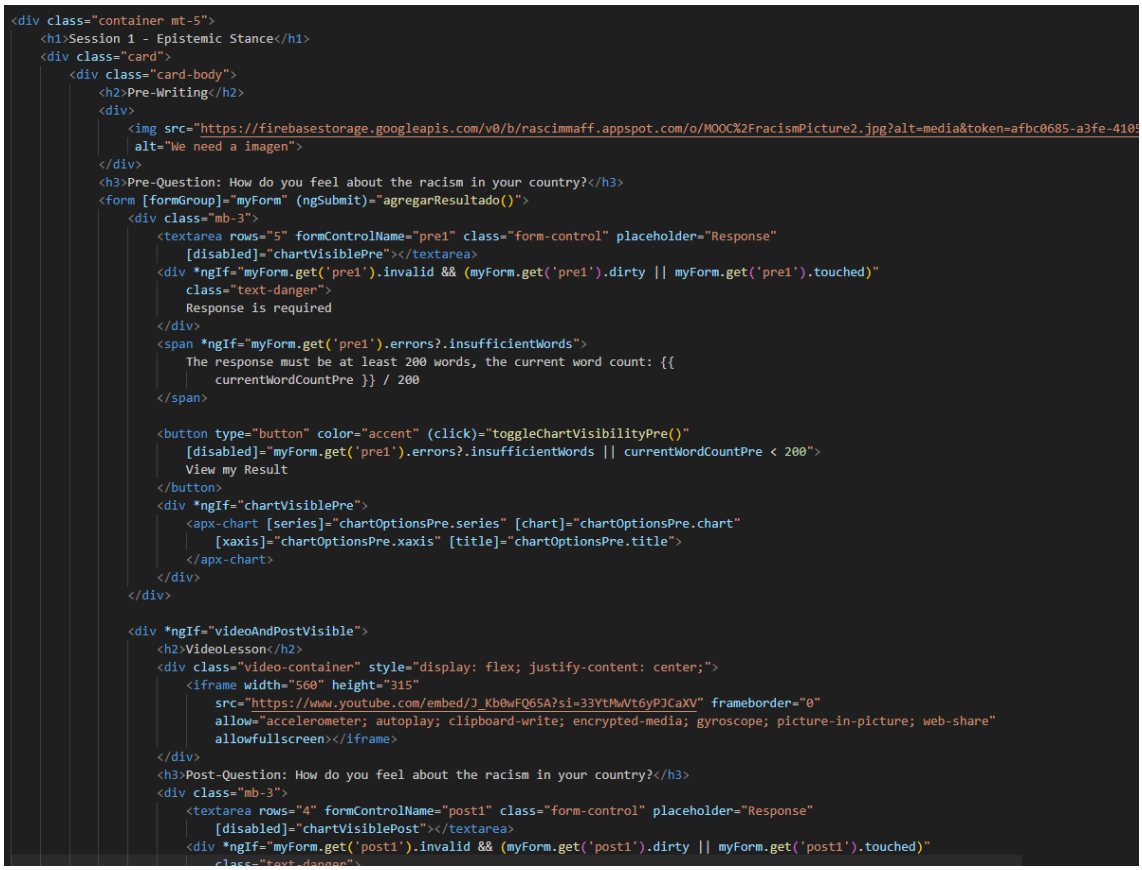


Figura 33: Página principal del curso1, course1.component.html

En la página principal de cada curso, dispondrán de una imagen con la respectiva pregunta, y un cuadro de texto para sus respuestas.

```
  countKeywords(text: string): Record<string, number> {
    return Object.keys(this.keywordGroups).reduce((acc, key) => {
      acc[key] = this.keywordGroups[key].reduce((count, keyword) => {
        const regex = new RegExp(`\\b${keyword.replace(/[.*+?^${}()|[\]\\]/g, '\\$&')}\\b`, 'gi');
        return count + (text.match(regex) || []).length;
      }, 0);
      return acc;
    }, {});
  }

  updateChart(counts: Record<string, number>, type: 'pre1' | 'post1') {
    const data: number[] = Object.values(counts);
    const options = type === 'pre1' ? this.chartOptionsPre : this.chartOptionsPost;
    options.series = [{
      name: 'Keyword Frequency',
      data: data
    }];
    options.xaxis.categories = Object.keys(counts);
    const chart = type === 'pre1' ? this.chartPre : this.chartPost;
    if (chart) {
      chart.updateOptions(options, true);
    }
  }

validateWordCount(type: string, control) {
  const value = control.value;
  if (value) {
      const wordCount = value.trim().split(/\s+/).length;
      if (type === 'pre') {
          this.currentWordCountPre = wordCount;
      } else {
          this.currentWordCountPost = wordCount;
      }
      return wordCount >= 200 ? null : { insufficientWords: true };
  }
  return null;
}

  toggleChartVisibilityPre(): void {
    this.chartVisiblePre = !this.chartVisiblePre;
    this.videoAndPostVisible = true;
    this.myForm.controls['pre1'].disable();  // Consider if you really need to disable this after viewing the chart
}
```

Figura 34: Lógica del curso1, course1.component.ts

Y con el archivo TypeScript añadimos la lógica a las actividades.

6.2.2.3 Sidebar

Componentes para la barra lateral de navegación:

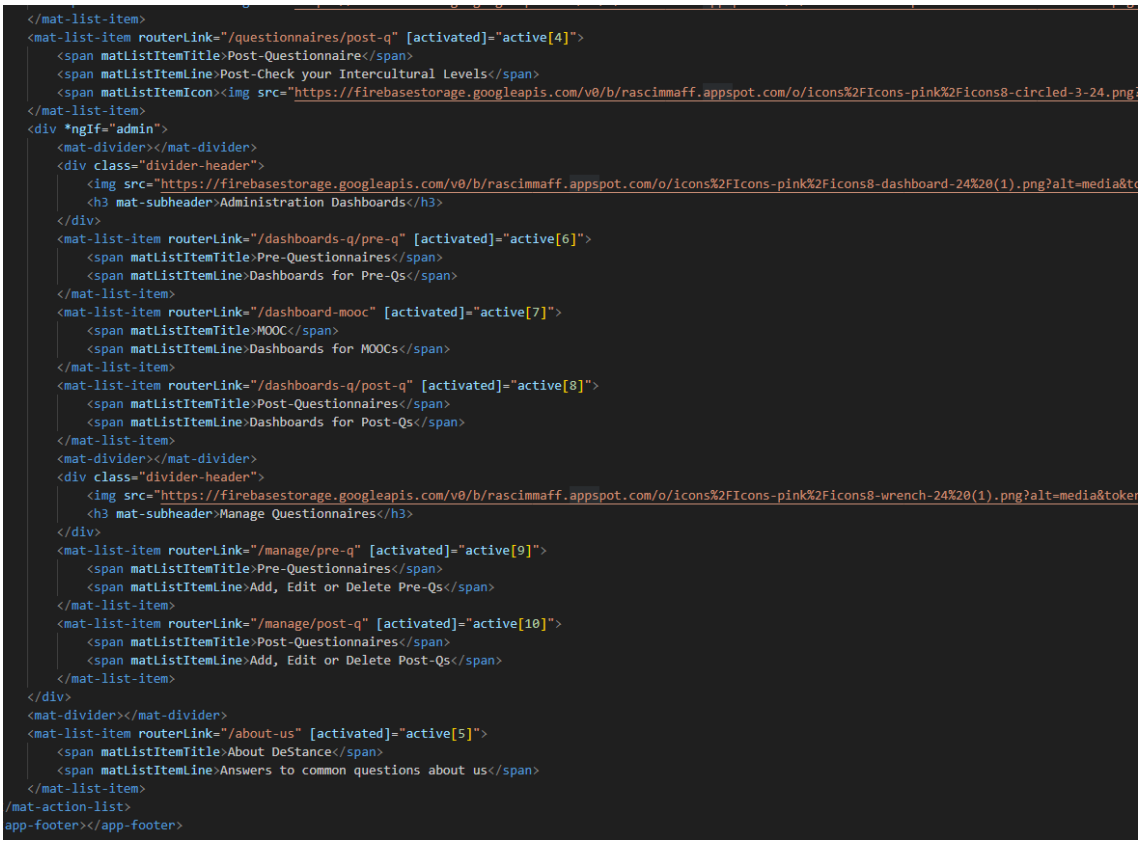


Figura 35: barra lateral de navegación, sidebar.component.html

En la parte de Sidebar hay operación de condición, si es administrador, dispondrán de todas las funcionalidades, y en caso contrario, dispondrán de funcionalidades limitadas. A través del archivo TypeScript realizamos el control de administrador:

6.2.2.4 App-routing.module.ts

Configuración de las rutas principales de la aplicación:

```
const routes: Routes = [
  { path: '', redirectTo: '/welcome', pathMatch: 'full' },
  { path: 'welcome', component: WelcomeComponent },
  { path: 'about-us', component: AboutUsComponent },
  { path: 'concepts-pool', component: ConceptsPoolComponent },
  {
    path: 'user-details',
    component: UserComponent,
    canActivate: [AuthGuard],
    children: [{ path: ':edit', component: UserEditComponent }],
  },
  {
    path: 'auth',
    component: AuthenticationComponent,
    children: [
      { path: 'login', component: LoginComponent },
      { path: 'sign-up', component: SignUpComponent },
      { path: 'reset-password', component: ResetPasswordComponent },
      { path: '', redirectTo: '/login', pathMatch: 'full' },
    ],
  },
  {
    path: 'questionnaires/:type',
    component: QuestionnariesComponent,
    canActivate: [AuthGuard],
    children: [
      { path: '', component: QuestionnariesStartComponent, pathMatch: 'full' },
      { path: ':id', component: QuestionnarieDetailComponent },
    ],
  },
```

Figura 36: Las rutas principales definidas en: App-routing.module.ts

6.2.2.5 App.component.html

Plantilla principal del componente de la aplicación que incluye los componentes header, sidebar, footer y router para cambiar a los componentes de uno a otro.

```
<div class="app-container" [class.example-is-mobile]="mobileQuery.matches">
    <app-header (clickedSB)="snav.toggle()"></app-header>
    <mat-sidenav-container class="app-sidenav-container"
                           [style.marginTop.px]="mobileQuery.matches ? 56 : 0">
      <mat-sidenav #snav [(opened)]="opened" [mode]="mobileQuery.matches ? 'over' : 'side'"
                   [fixedInViewport]="mobileQuery.matches" fixedTopGap="56">
        <mat-nav-list>
          <app-sidebar></app-sidebar>
        </mat-nav-list>
      </mat-sidenav>
      <mat-sidenav-content>
        <router-outlet></router-outlet>
      </mat-sidenav-content>
    </mat-sidenav-container>
</div>
```

Figura 37: Página principal de la aplicación, App.component.html

6.3 Imágenes de las Funcionalidades

A continuación, mostraremos algunas imágenes correspondientes al funcionamiento de la aplicación:

- **Login**: para navegar la aplicación lo primero es login del usuario, el sistema mostrará todos los contenidos si el usuario tiene permiso de administrador.

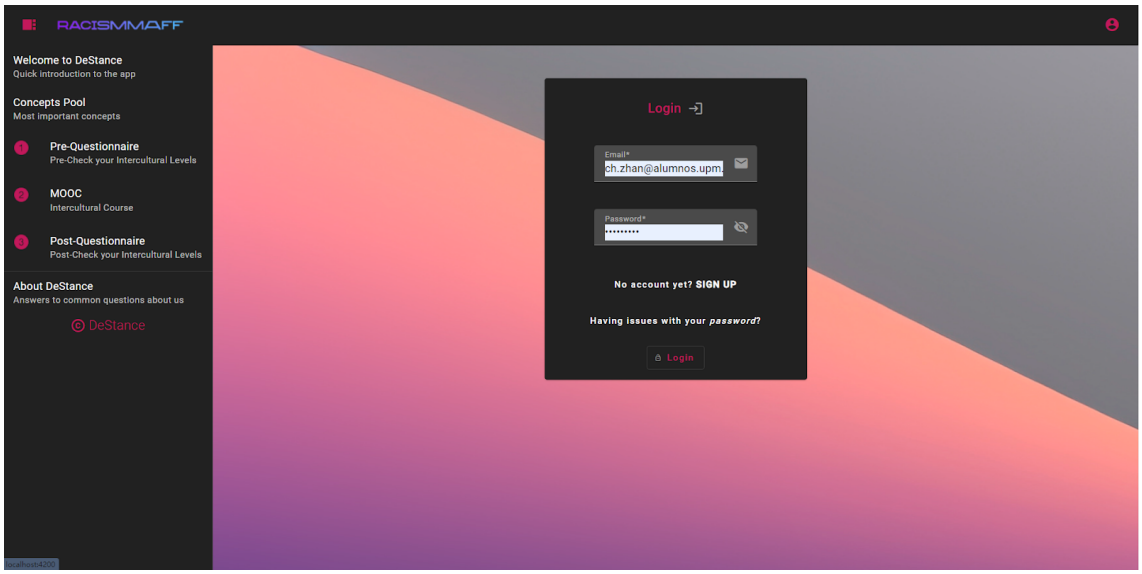


Figura 38: Página del login de los usuarios registrados

- **Página Principal (normal)**: La página principal de la aplicación para los usuarios registrados sin permiso de administrador es limitado.

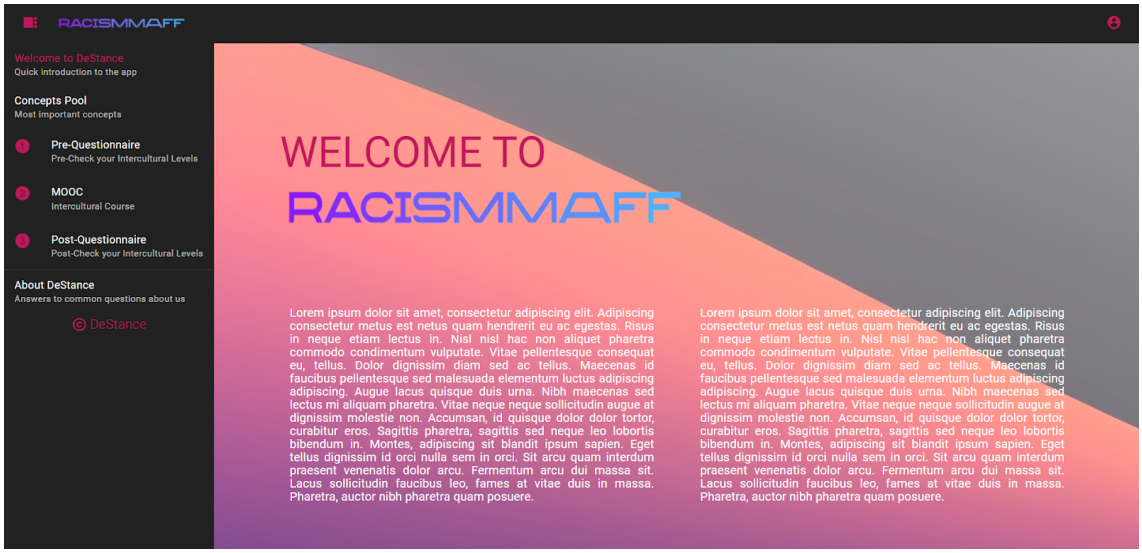


Figura 39: Página principal de la aplicación para los usuarios registrados

- **Página Principal (Administrador)**: La página principal de la aplicación para los administradores muestra todas las funcionalidades, incluyendo el Dashboard y el mantenimiento de los cuestionarios.

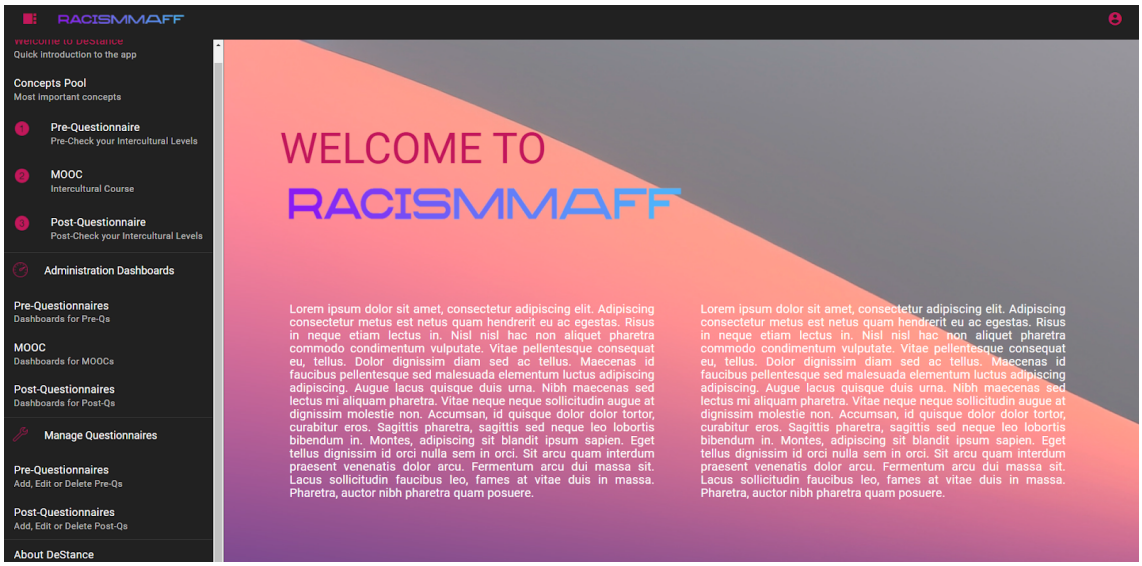


Figura 40: Página principal de la aplicación para los administradores

- **MOOC**: Acceso a la página principal de las sesiones online.

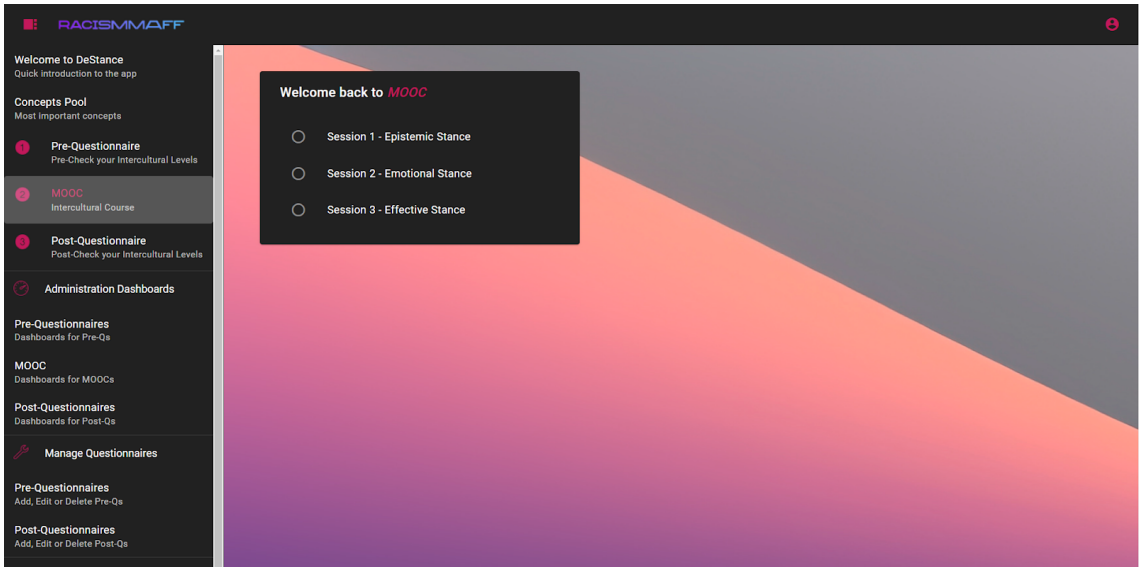


Figura 41: Página principal para acceder a los cursos online

- **Sesión 1**: Dentro de los cursos dispondrán de una imagen y cuadro de textos a responder.

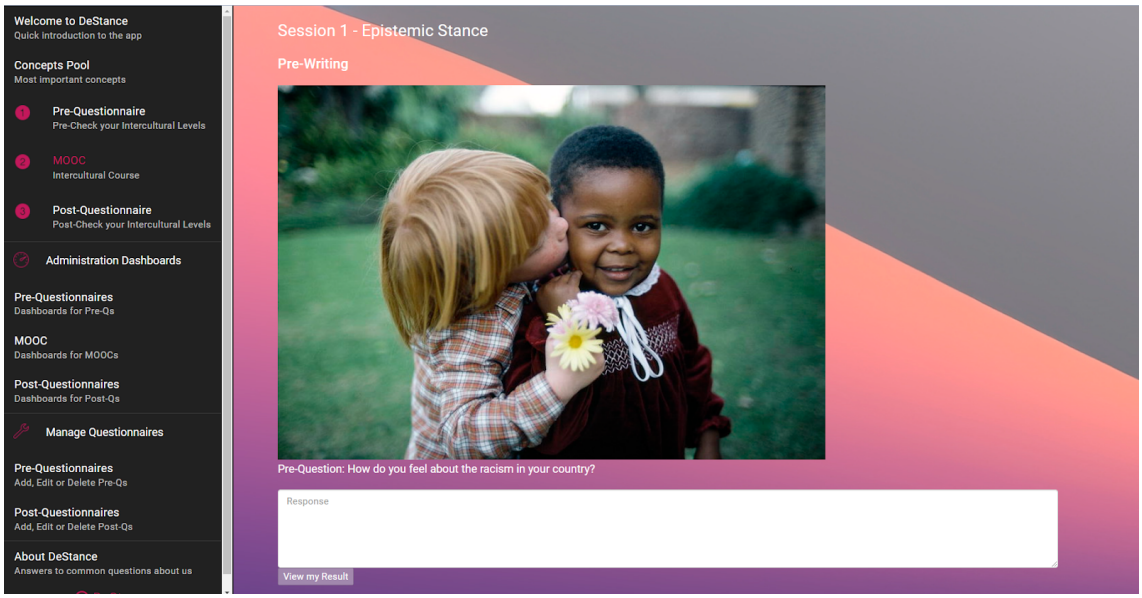


Figura 42: Imagen y el cuadro de respuesta

- **Feedback**: Una vez respondido la pregunta dispondrá de la gráfica de retroalimentación.

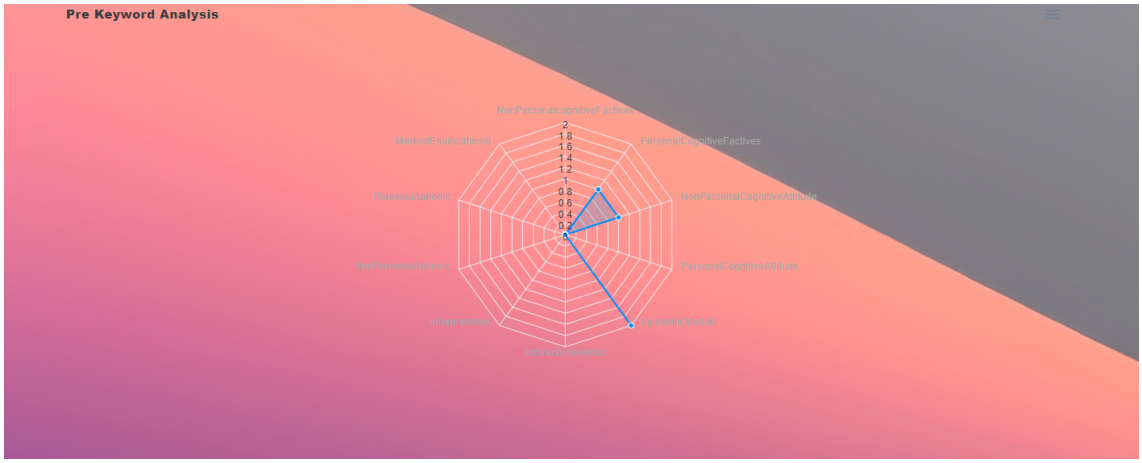


Figura 43: Retroalimentación de la respuesta

- **Video**: Se dispondrán de un video corto y deberán de responder la pregunta a continuación.

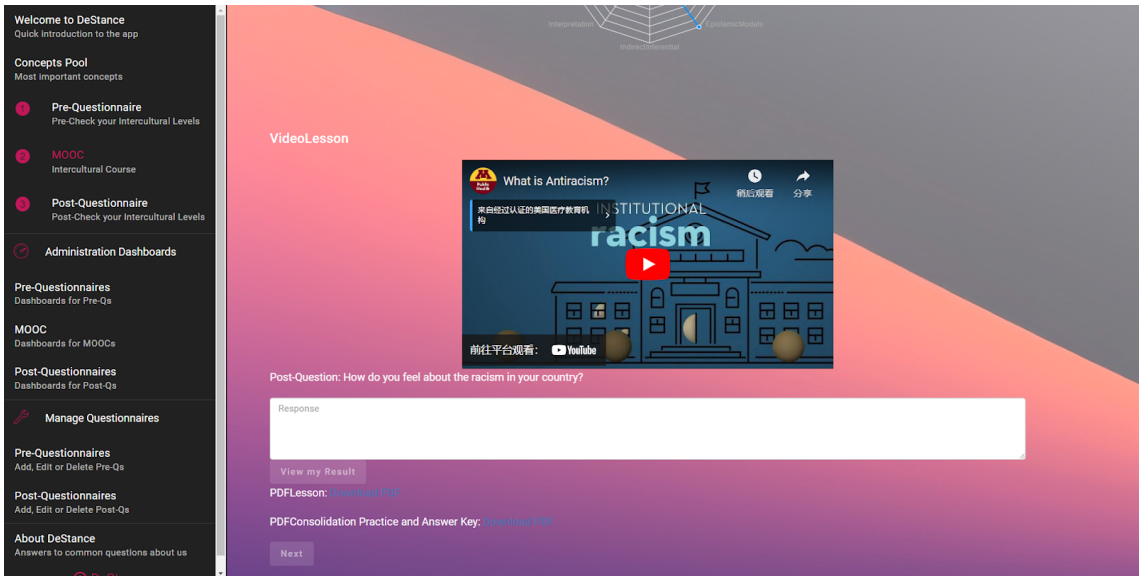


Figura 44: Video del curso online, el cuadro de respuestas y descarga de archivos

- **Descarga del Archivo:** Podrán descargar los archivos de resumen de la sesión.

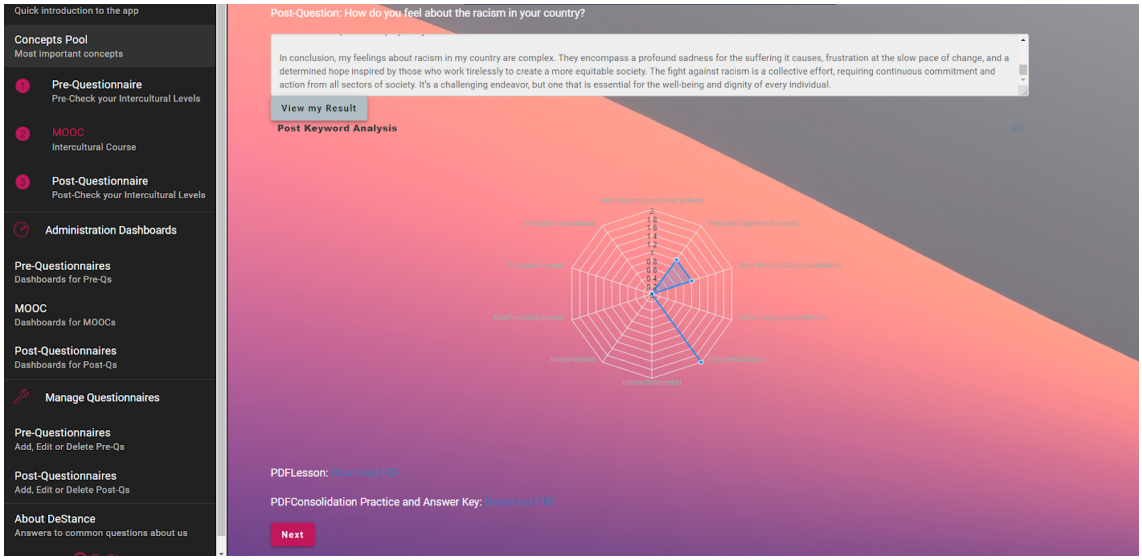


Figura 45: Retroalimentación de la respuesta

- **Dashboard MOOC**: La página principal del Dashboard-MOOC donde puede seleccionar la sesión interesada.

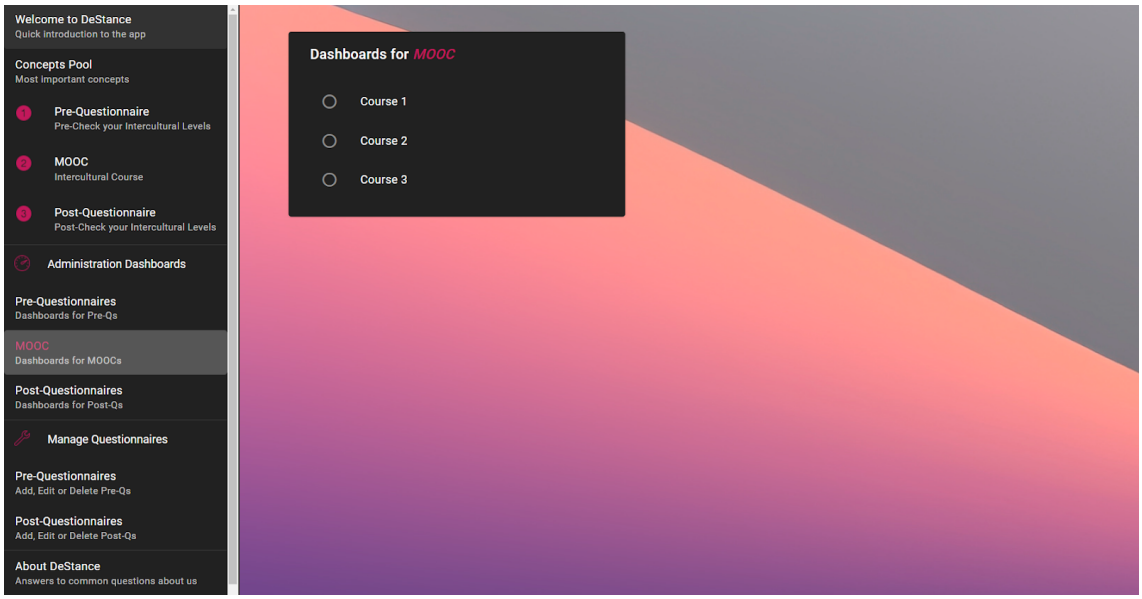


Figura 46: Página principal de Dashboard-MOOC

- **Dashboard MOOC.Course1**: La página principal del Dashboard course1 dispone de tres opciones a seleccionar, el análisis de pre/post writting y respuestas en texto.

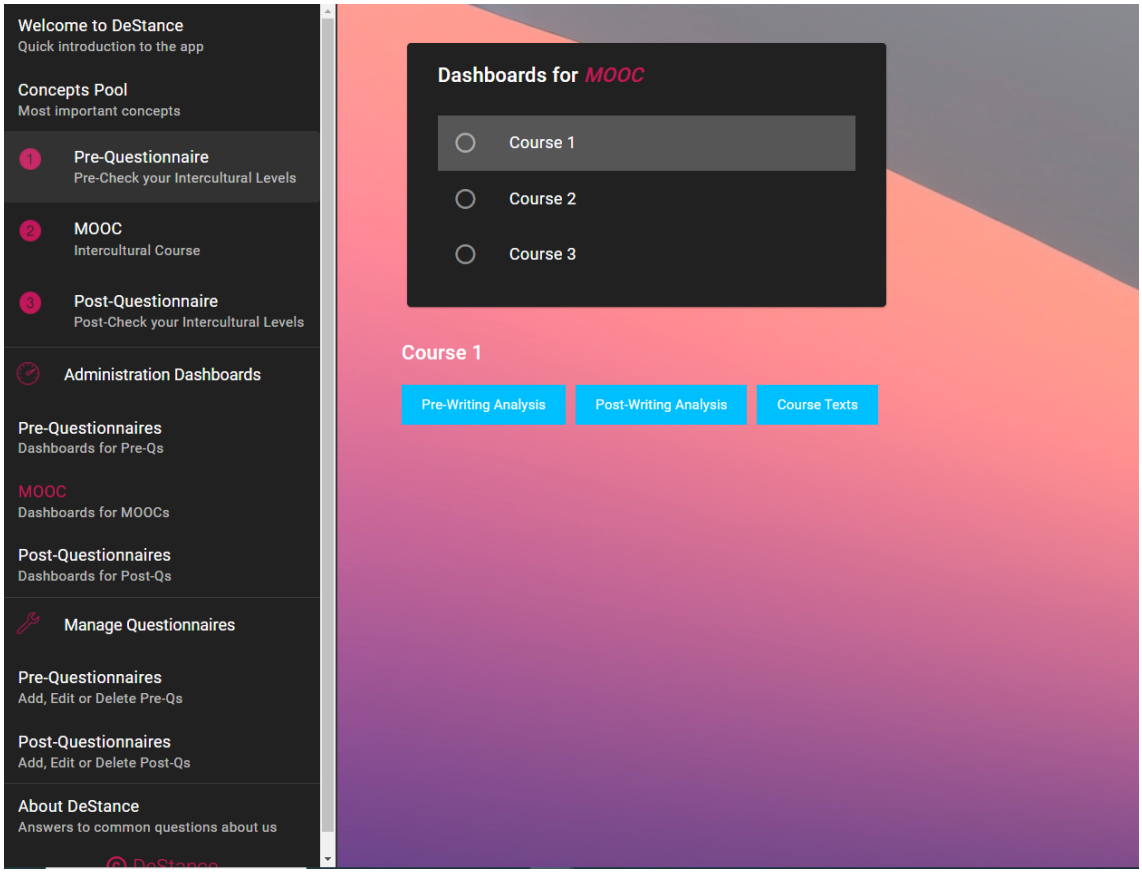


Figura 47: Página principal del dashboard del Curso1

- **Análisis de Pre-Writting**: El Dashboard-MOOC muestra la estadística de los resultados recogidos del Pre-Writting.

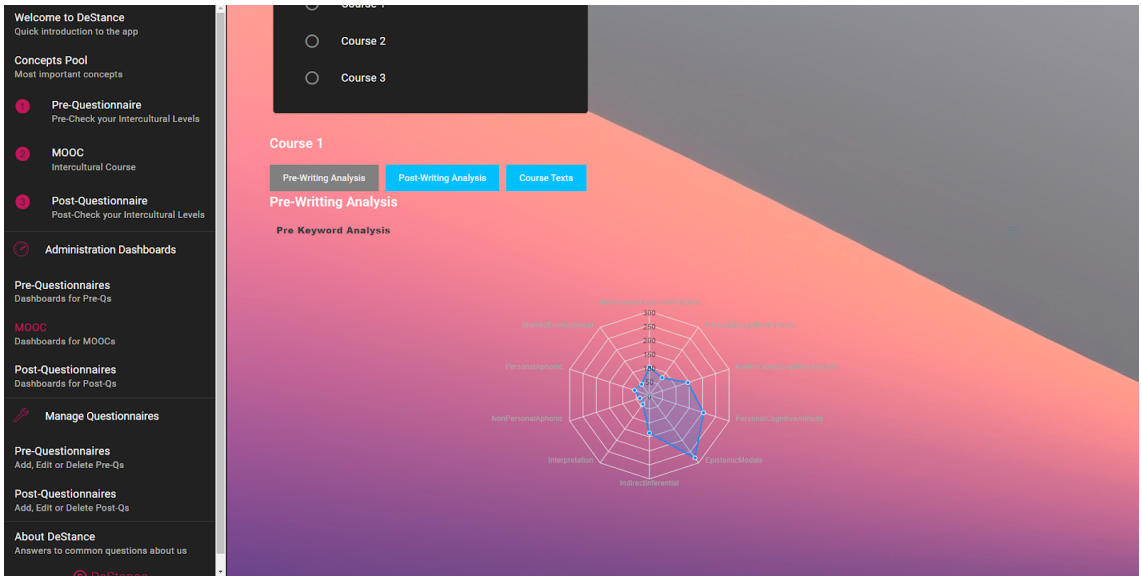


Figura 48: Análisis de las respuestas del pre-writting en el Curso 1

- **Análisis de Post-Writting**: El Dashboard-MOOC muestra la estadística de los resultados recogidos del Post-Writting.

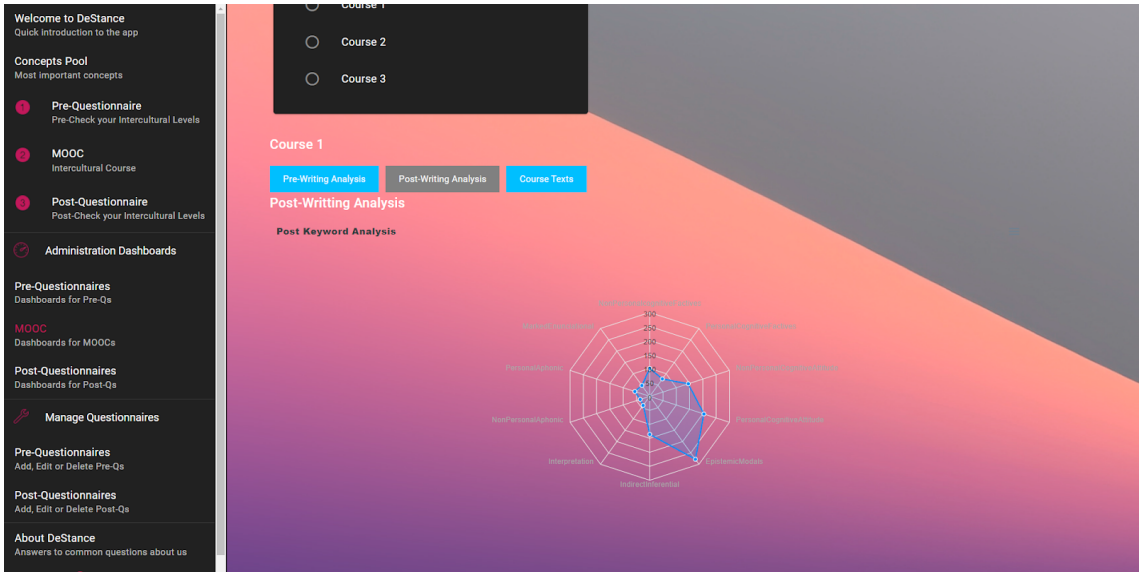


Figura 49: Análisis de las respuestas del post-writting en el Course1

- **Mostrar Textos**: El Dashboard-MOOC muestra los resultados en texto y la lengua materna.

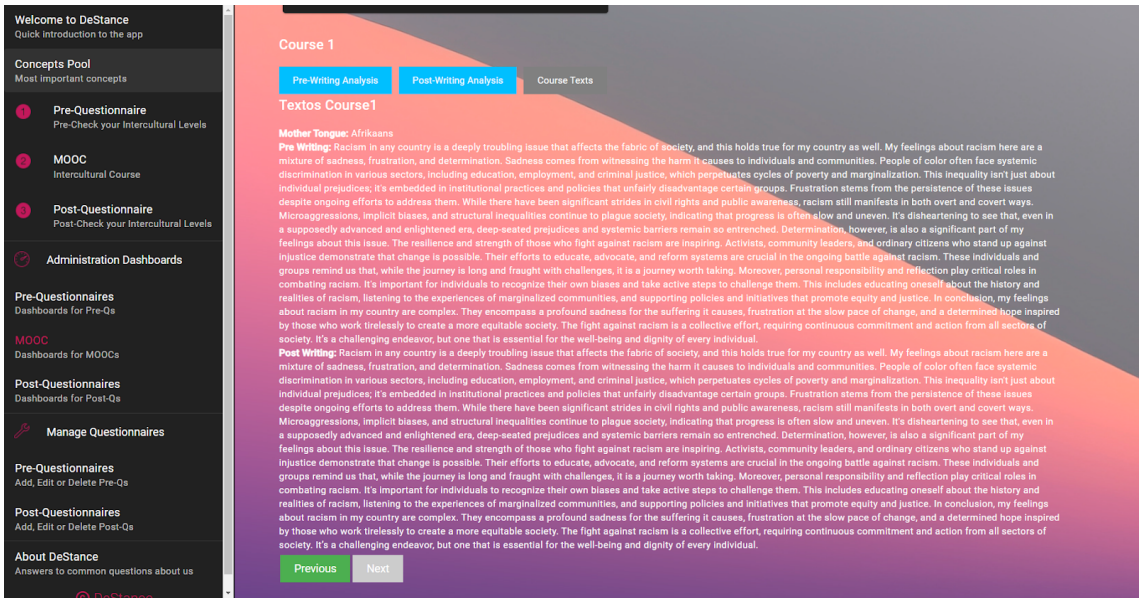


Figura 50: Respuesta en texto del Curso 1

6.4 Dificultades durante el desarrollo

El trabajo en equipo es fundamental para el éxito de cualquier proyecto de desarrollo de software, pero también puede ser una fuente de desafíos. A lo largo del proyecto, enfrentamos varias dificualtades relacionadas con la coordinación y la colaboración entre los miembros del equipo.

6.4.1 Dificultades

- Coordinación de horarios: Coordinar reuniones y sesiones de trabajo conjunto fue complicado debido a los diferentes horarios y compromisos personales.

- Feedback Tardío: La retroalimentación no siempre se proporcionó de manera oportuna. Esto resultó en retrasos en el desarrollo y en la identificación y resolución de problemas.

- Dependencias entre Tareas: Muchas tareas del proyecto dependían de la finalización de otras. Gestionar estas dependencias y asegurarse de que las tareas previas se completaran a tiempo para no retrasar las siguientes fue un desafío constante.

6.4.2 Estrategias para Superar las Dificultades

Para abordar estas dificultades, adaptamos a varias estrategias que resultaron ser efectivas:

- Reuniones Regulares: Establecimos reuniones periódicas semanales para revisar el progreso, discutir problemas y planificar las siguientes etapas del proyecto. Esto ayudó a mejorar la coordinación y la comunicación.

- Retroalimentacion Continua: Fomentamos una cultura de Feedback continuo y oportuno, lo que permitió a los miembros del equipo mejorar su desempeño y abordar problemas de manera proactiva.

A pesar de los desafíos, estas estrategias nos permitieron avanzar y completar el proyecto de manera exitosa. La experiencia adquirida en la gestión de un equipo y la comunicación efectiva será invaluables para los futuros proyectos.

7 Conclusiones

El proyecto de desarrollo de la aplicación web de cursos en línea ha proporcionado una valiosa experiencia y varios aprendizajes claves. A continuación, se presentan las conclusiones derivadas del proyecto:

7.1 Eficiencia en el Desarrollo

7.1.1 Frameworks y Herramientas

La elección de las herramientas y frameworks como Node.js, Express.js, Angular y MongoDB ha sido fundamental para el desarrollo del proyecto. Estas tecnologías permiten un desarrollo rápido y eficiente gracias a su alta compatibilidad y escalabilidad. Node.js y Express.js han facilitado la creación de un backend robusto, mientras que Angular ha sido crucial para desarrollar una interfaz de usuario dinámica y responsiva. MongoDB, con su naturaleza flexible y escalable, ha sido esencial para gestionar la base de datos de manera eficaz.

7.1.2 Automatización

La utilización de herramientas de automatización como Postman ha mejorado nuestra capacidad de realizar pruebas y garantizar la calidad del código, esto ha reducido el tiempo necesario para el desarrollo y el lanzamiento del producto. La automatización ha permitido identificar y corregir errores de manera eficiente, asegurando una aplicación más estable y confiable.

7.2 Escalabilidad y Mantenimiento

La clara separación de responsabilidades entre los controladores, rutas, modelos, middleware y componentes frontend permite que los desarrolladores trabajen en diferentes partes del proyecto sin interferencias significativas y la escritura de código limpio. Esta arquitectura modular facilita el mantenimiento y la escalabilidad de la aplicación, permitiendo futuras mejoras y expansiones sin comprometer la integridad del sistema.

7.3 Experiencia de Usuario

El diseño de una interfaz de usuario intuitiva y responsiva ha sido una prioridad para que los usuarios pueden navegar fácilmente por la aplicación y acceder a los cursos desde cualquier dispositivo. Un enfoque centrado en el usuario ha permitido crear una experiencia de usuario positiva, mejorando la satisfacción y el compromiso de los usuarios con la plataforma.

7.4 Seguridad

La protección de datos ha sido un aspecto crucial del proyecto. La implementación de medidas de seguridad como la autenticación basada en JWT y la autorización robusta garantiza que los datos de los usuarios estén protegidos. Estas medidas aseguran que solo los usuarios autorizados puedan acceder a información sensible, manteniendo la integridad y la confidencialidad de los datos almacenados.

7.5 Limitaciones de la Aplicación

A pesar de los logros alcanzados, la aplicación presenta algunas limitaciones. La dependencia de la conexión a internet puede ser una barrera para usuarios en áreas con acceso limitado o inestable. Además, la capacidad de la aplicación

para manejar un gran volumen de usuarios aún no ha sido completado, lo cual, podría afectar el rendimiento en escenarios de alta demanda. Y por último, otra de las limitaciones es la falta de integración con los demás sistemas educativos y plataforma de aprendizaje que podrían enriquecer la experiencia del usuario.

7.6 Futuras Líneas de Desarrollo

Para abordar estas limitaciones, nos incluimos las futuras líneas de desarrollo:

- **Optimización de Rendimiento**: Mejorar la capacidad de la aplicación para manejar múltiples usuarios simultáneos.

- **Compatibilidad Offline:** Desarrollar funcionalidades que permitan el acceso a ciertos contenidos sin necesidad de estar conectado a internet.

- **Integración con Otras Plataformas:** Expandir la compatibilidad con otros sistemas educativos y herramientas de aprendizaje para ofrecer una experiencia más completa.

- **Funciones Avanzadas de Análisis de Datos:** Implementar técnicas de análisis de datos más avanzadas para proporcionar insights más profundos y personalizados.

8 Análisis de Impacto

El desarrollo de la aplicación web de cursos online contribuye varios Objetivos de Desarrollo Sostenible (ODS) establecidos por las Naciones Unidas. A continuación, se explicará en detalle el análisis de impacto de este proyecto en relación con algunos de los ODS [39].

8.1 ODS 4, Educación de Calidad

La aplicación de cursos online proporciona el acceso a los videos educativos de alta calidad a un público amplio, independientemente de su ubicación geográfica. Este acceso universal a la educación contribuye directamente al ODS4 [40], que intentamos de garantizar una educación equitativa de calidad.

8.2 ODS 10, Reducción de las Desigualdades

En esta plataforma se dispondrán de videos en contra del racismo, el acceso a esta plataforma online puede ayudar a reducir las desigualdades debido a los contenidos educativos. El objetivo está alineado con el ODS 10 [41], que busca reducir la desigualdad dentro y entre los países.

8.3 ODS 17, Alianzas para Lograr los Objetivos

El desarrollo y la implementación de esta plataforma pueden involucrar la colaboración con diversas instituciones educativas, organizaciones no gubernamentales fomentando alianzas y colaboraciones que contribuyen al ODS 17 [42], que busca fortalecer los medios de implementación y revitalizar la alianza mundial para el desarrollo sostenible.

8.4 Conclusión

El impacto de la aplicación web es notable desde la educación de calidad y la reducción de las desigualdades. A través de la implementación continua de mejoras y la incorporación de Feedback de los usuarios, la plataforma tiene el potencial de seguir creciendo, y además como este proyecto está liderada por el Ministerio de Educación y llevando a cabo por investigadores de UCM y de UPM, podrá haber colaboraciones educativas contribuyendo positivamente a los objetivos de desarrollo sostenible a nivel global.

9 Bibliografía

[1] Universidad Politécnica de Madrid. (2023). Plan de estudios: 10II- Grado en Ingeniería Informática[En línea]. Available:

https://www.fi.upm.es/ccfi/tfgm/alumnos/detalles_oferta.php?id_oferta=7829

[2] Universidad Complutense de Madrid. (2023). Estrategias de posicionamiento en el discurso del racismo y la inmigración: análisis y aplicaciones en prácticas afectivas de aprendizaje (RACISMMAFF) [En línea]. Available:

https://www.ucm.es/racismmaff/

[3] Euronews. (2023). ¿Qué países europeos son los más racistas? [En línea]. Available:

https://es.euronews.com/2023/10/25/que-paises-europeos-son-los-mas-racistas

[4] NVivo. (2024). Nvivo Spain [En línea]. Available:

https://www.nvivo-spain.com/que-es-nvivo/

[5] Leximancer. (2024). Leximancer [En línea]. Available:

https://www.leximancer.com

[6] LIWC. (2023). Welcome to LIWC-22 [En línea]. Available:

https://www.liwc.app

[7] SendGrid. (2024). Event Webhook Security [En línea]. Available: https://sendgrid.com/en-us/solutions/email-api

[8] Firebase. (2024). Web App Development Platform. Available: https://firebase.google.com/?hl=es-419

[9] Ciclo de vida Incremental. (2018). ¿Cuáles son los tipos de ciclo de vida de un proyecto?. Available: https://grctools.software/2018/06/28/cuales-son-los-tipos-de-ciclo-de-vida-de-un-proyecto/

[10] Web Content Accessibilty guidelines 2.1. (2023). Available: https://www.w3.org/TR/WCAG21/

[11] General Data Protection Regulation. (2024). Available:

https://gdpr-info.eu

[12] IBM. (2024). ¿Qué es la pila MEAN? Available:

https://www.ibm.com/es-es/topics/mean-stack

[13] Xataka. (2022). Resulta que 2022 sí fue la el año de linux en el escritorio Available: https://www.xataka.com/aplicaciones/resulta-que-2022-fue-para-algunos-ano-linux-escritorio#:~:text=Windows%20sigue%20siendo%20el%20más,07%25)%2C%20algo%20realmente%20sorprendente.

[14] Atlassian. (2024). Qué es control de versiones. Available: https://www.atlassian.com/es/git/tutorials/what-is-version-

control#:~:text=Aunque%20se%20puede%20desarrollar%20software,ningún%20equipo%20profesional%20debería%20aceptar.

[15] Atlassian. (2024). Por qué usar git. Available: https://www.atlassian.com/es/git/tutorials/why-git

[16] Git. (2024). Acerca de control de versiones git. Available: https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Acerca-del-Control-de-Versiones

[17] Visual Studio Code. (2024). Code editing. Available: https://code.visualstudio.com/learn

[18] Arsys. (2024). ¿Qué es visual studio code y cuáles son sus ventajas?. Available: https://www.arsys.es/blog/que-es-visual-studio-code-y-cuales-son-sus-ventajas

[19] NPM. (2024). Abour NPM. Available: https://docs.npmjs.com/about-npm

[20] Arsys. (2024). ¿Qué es NPM? JavaScript para principiantes. Available: https://www.arsys.es/blog/que-es-npm-javascript-para-principiantes#Que_es_NPM_Node_Package_Manager

[21] UNIR FP. (2022). Framework: qué es, para qué sirve. available:

https://unirfp.unir.net/revista/ingenieria-y-tecnologia/framework/

[22] Node.js. (2024). About Node.js. Available: https://nodejs.org/en/about

[23] Kinsta. (2024). ¿Qué es Express.js? todo lo que debes saber. Available: https://kinsta.com/es/base-de-conocimiento/que-es-express/

[24] Angular Hispano. (2024). Angular. Available: https://docs.angular.lat

[25] Ticportal. (2023). Base de datos: ¿qué tipos hay y cuantó cuesta? Available: https://www.ticportal.es/glosario-tic/base-datos-database

[26] MongoDb. (2024). ¿Qué es MongoDB?. Available: https://www.mongodb.com/es/company/what-is-mongodb

[27] Formadores IT. (2023). ¿Qué es Postman? ¿Cuáles son sus principales ventajas? Available: https://formadoresit.es/que-es-postman-cuales-son-sus-principales-ventajas/

[28] Agencia Rumpelstinski. (2023). Los lenguajes de programación web más usados en 2023. Available: https://www.rumpelstinski.es/actualidad/5-lenguajes-de-programación-web-más-usados-en-2023

[29] Blog de HubSpot. (2023). Qué es JavaScript, para qué sirve y cómo funciona. Available: https://blog.hubspot.es/website/que-es-javascript

[30] W3Schools. (2024). TypeScript Introduction. Available: https://www.w3schools.com/typescript/typescript_intro.php

[31]LenguajeHTML. (2024). ¿Qué es HTML? Available: https://lenguajehtml.com/html/introduccion/que-es-html/

[32] Escuela IT. (2024). CSS3. Available: https://escuela.it/materias/css3#:~:text=CSS3%20es%20la%20especificación%20más,JavaScript%20disponibles%20en%20el%20navegador.

[33] Quest Software. (2020). How to work with data using MongoDB Query Language. Available: https://blog.quest.com/how-to-work-with-data-using-mongodb-query-language/

[34] Evotic. (2024). Etapas y modelos del Ciclo de Vida de un Software. Available: https://evotic.es/software-a-medida/ciclo-de-vida-del-software/

[35] DiagramasUML. (2024). Diagrama de casos de uso. available: https://diagramasuml.com/casos-de-uso/#google_vignette

[36] Edrawsoft. (2024). Diagrama de arquitectura de sistemas. Available: https://www.edrawsoft.com/es/article/system-architecture-diagram.html#:~:text=La%20arquitectura%20de%20sistemas%20es,componentes%20del%20sistema%20en%20general.

[37] Jarroba. (2014). MEAN, Ejemplo de Aplicacion web. Available: https://jarroba.com/mean-mongo-express-angular-node-ejemplo-de-aplicacion-web-parte-ii/

[38] Red Hat. (2023). ¿Qué es una API REST? Available: https://www.redhat.com/es/topics/api/what-is-a-rest-api

[39] United Nations. (2024). Objetivos y Metas de Desarrollo Sostenible. Available: https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/

[40] United Nations. (2024). Objetivo 4, Educacion de Calidad. Available: https://www.un.org/sustainabledevelopment/es/education/

[41] United Nations. (2024). Objetivo 10, Reducción de las Desigualdades. Available: https://www.un.org/sustainabledevelopment/es/inequality/

[42] United Nations. (2024). Objetivo 17, Alianzas para Lograr los Objetivos. Available: https://www.un.org/sustainabledevelopment/es/globalpartnerships/

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=ETS Ingenieros Informaticos - UPM, C=ES
	Fecha/Hora	Mon Jun 03 22:32:06 CEST 2024
	Emisor del Certificado	EMAILADDRESS=camanager@etsiinf.upm.es, CN=CA ETS Ingenieros Informaticos, O=ETS Ingenieros Informaticos - UPM, C=ES
	Numero de Serie	561
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)