Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Grado en Ingeniería Informática

Trabajo Fin de Grado

Diseño e implementación de un entorno de despliegue seguro y automatizado en la nube para la plataforma educativa De-Stance

Autor: Juan Loscos Ortega
Tutora: Jelena Bobkina
Cotutora: Elena Domínguez Romero

Madrid, enero 2026

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

Trabajo de Fin de Grado
Grado en Ingeniería Informática
Título: Diseño e implementación de un entorno de despliegue seguro y automatizado en la nube para la plataforma educativa De-Stance

Madrid, enero 2026

Autor: Juan Loscos Ortega

Tutora:
Jelena Bobkina
Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid

Cotutora:
Elena Domínguez Romero
Universidad Complutense de Madrid

Resumen

Este Trabajo de Fin de Grado se enmarca en el desarrollo y consolidación de De-Stance, una plataforma educativa web creada en el contexto del proyecto de investigación RACISMMAFF, orientada al fomento de competencias interculturales y al análisis del posicionamiento discursivo en estudiantes universitarios europeos. La plataforma combina contenidos formativos de tipo MOOC con actividades reflexivas e interactivas que permiten trabajar dimensiones epistémicas, emocionales y actitudinales relacionadas con temas socialmente sensibles como la inmigración y el racismo.

El presente trabajo aborda la plataforma desde una perspectiva técnica centrada en la experiencia de usuario y la coherencia de contenidos.

A partir de los documentos de diseño y contenido proporcionados por la tutoría académica, se reestructura la navegación principal, se actualizan las páginas informativas (About y FAQ) y se reorganiza el flujo de los micro-MOOCs para dar prioridad al vídeo educativo. La sección Concepts se mantiene como en el despliegue realizado en la máquina de GCP por Javier Hernández Contreras, sin cambios funcionales [1].

Además, se reduce la restricción del mínimo de palabras en tareas de escritura, añadiendo recomendaciones para mejorar la calidad del feedback. En paralelo, se refuerzan los flujos de alta y recuperación de contraseña trasladando la verificación por código y el envío de correos al backend.

El trabajo mantiene el objetivo de despliegue seguro y mantenible en un entorno cloud académico (CeSvImA), dejando definida la línea de operación y automatización necesaria para la puesta en producción.

Como resultado, la plataforma ofrece un recorrido más claro, coherente y accesible para el alumnado, manteniendo la escalabilidad del sistema y facilitando futuras ampliaciones de contenidos o formatos multimedia, así como su operación en la nube.

Abstract

This Final Degree Project is framed within the development and consolidation of De-Stance, a web-based educational platform created in the context of the RACISMMAFF research project, oriented toward fostering intercultural competences and analyzing discursive positioning in European university students. The platform combines MOOC-type learning content with reflective and interactive activities that allow work on epistemic, emotional, and attitudinal dimensions related to socially sensitive topics such as immigration and racism.

The present work addresses the platform from a technical perspective focused on user experience and content coherence.

Based on the design and content documents provided by the academic supervision, the main navigation is restructured, the informational pages (About and FAQ) are

updated, and the micro-MOOC flow is reorganized to prioritize the educational video. The Concepts section remains as in the deployment carried out on the GCP machine by Javier Hernández Contreras, without functional changes [1].

In addition, the minimum word restriction in writing tasks is reduced, adding recommendations to improve feedback quality. In parallel, signup and password recovery flows are reinforced by moving code verification and email sending to the backend.

The work maintains the goal of secure and maintainable deployment in an academic cloud environment (CeSvImA), leaving the operational and automation line defined and needed for production.

As a result, the platform offers a clearer, more coherent and accessible path for students, maintaining system scalability and facilitating future expansions of content or multimedia formats, as well as its operation in the cloud.

Glosario de siglas

ACME Protocolo de emisión automática de certificados (Certbot/Let's Encrypt).

ACAIA Herramienta externa de feedback ACAIA.

API Application Programming Interface.

ASVS Application Security Verification Standard (OWASP).

AWS Amazon Web Services.

BD Base de datos.

BIMI Brand Indicators for Message Identification.

CDN Red de distribución de contenido.

CeSvImA Centro de Supercomputación y Visualización de Madrid [2].

CI/CD Integración continua y despliegue continuo.

CAPTCHA Completely Automated Public Turing test to tell Computers and Humans Apart.

CNAME Registro DNS de nombre canónico.

CORS Cross-Origin Resource Sharing.

CPU Unidad central de procesamiento.

CSRF Cross-Site Request Forgery.

CSS Cascading Style Sheets.

DDNS DNS dinámico.

DKIM DomainKeys Identified Mail.

DMARC Domain-based Message Authentication, Reporting and Conformance.

DevOps Desarrollo y operaciones.

DNS Sistema de nombres de dominio.

E2E Pruebas extremo a extremo.

ECS Elastic Container Service (AWS).

ETSI Escuela Técnica Superior de Ingenieros.

EUR Euro.

FAQ Frequently Asked Questions.

FQDN Fully Qualified Domain Name.

GB Gigabyte.

GAMMA Grupo de Aplicaciones Multimedia y Acústica.

GCP Google Cloud Platform.

GDPR General Data Protection Regulation.

GPU Unidad de procesamiento gráfico.

HTML HyperText Markup Language.

HSTS HTTP Strict Transport Security.

HTTP Hypertext Transfer Protocol.

HTTPS Hypertext Transfer Protocol Secure.

IA Inteligencia artificial.

ID Identificador.

IaC Infraestructura como Código.

IONOS Proveedor de dominios y hosting IONOS.

IP Dirección IP.

JWT JSON Web Token.

MOOC Massive Open Online Course.

OWASP Open Web Application Security Project.

OVH Proveedor de VPS y hosting OVH.

PaaS Platform as a Service.

RAM Memoria de acceso aleatorio.

RACISMMAFF Estrategias de posicionamiento en el discurso del racismo y la inmigración: análisis y aplicaciones en prácticas afectivas de aprendizaje.

REST Representational State Transfer.

RF Requisito funcional.

SMTP Simple Mail Transfer Protocol.

SPF Sender Policy Framework.

SSD Unidad de estado sólido.

SSH Secure Shell.

TFG Trabajo Fin de Grado.

TLS Transport Layer Security.

TPU Tensor Processing Unit.

TTL Time To Live.

TXT Registro TXT (texto) en DNS.

UPM Universidad Politécnica de Madrid.

URL Uniform Resource Locator.

USD Dólar estadounidense.

VM Máquina virtual.

VPS Servidor privado virtual.

WAF Web Application Firewall.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción y objetivos

1.1. Contexto y motivación

De-Stance es una plataforma web educativa desarrollada en el marco del proyecto RACISMMAFF [3]. La plataforma se ha desarrollado en equipo, con colaboración continuada entre distintas personas y líneas de trabajo. Su objetivo es fomentar la competencia intercultural y el análisis crítico del posicionamiento discursivo mediante micro-MOOCs, cuestionarios y actividades reflexivas.

Este TFG se centra en mejorar la experiencia de usuario y la estructura educativa de la plataforma, siguiendo las guías de la dirección acdémica. Además, aborda el despliegue seguro en la nube, con una infraestructura reproducible y un flujo de despliegue fiable.

El trabajo se desarrolla en un entorno colaborativo continuo. Existe coordinación con el equipo de lingüistas del proyecto RACISMMAFF y con el grupo de investigación GAMMA de la Universidad Politécnica de Madrid [4], que orientan criterios pedagógicos y de contenido. La colaboración con este grupo de investigación fue clave para disponer de la VPS académica actual en CeSvImA, gracias a la ayuda de Nicolás Sáenz Lechón, miembro de GAMMA. También hay una línea de colaboración con estudiantes de la UPM de cursos anteriores y del curso actual, cuyos TFGs y aportaciones han ido consolidando la plataforma De-Stance. Entre estos trabajos previos se encuentran los de Gonzalo Álvarez García, Daniel Ramón Robertson, Ana Clara Murillo García y Javier Hernández Contreras [1], [5]-[7]. Esta contextualización ayuda a entender el origen de decisiones técnicas y de diseño y a situar este TFG como una contribución individual integrada en un esfuerzo académico colectivo.

El punto de partida tenía dos problemas claros. Por un lado, había diferencias entre los contenidos docentes y lo que el alumnado veía en pantalla; por otro, el despliegue dependía de configuraciones locales difíciles de repetir por nuevos integrantes. Por eso, el trabajo combina mejoras educativas (recorrido y feedback) con mejoras técnicas para que el sistema se pueda ejecutar de forma fiable y sin exponer datos sensibles.

Esto explica que el trabajo mezcle cambios de contenido y navegación con decisiones técnicas como mover validaciones al backend, mejorar el correo y estabilizar el despliegue. El objetivo es una experiencia clara para el alumnado y una base fácil de mantener para el equipo.

1.2. Objetivos alcanzados

Este apartado resume los objetivos definidos para el trabajo y diferencia los solicitados en el proyecto y los propuestos por el autor.

1.2.1. Objetivos solicitados en el trabajo

- Adaptar la estructura y los contenidos de la plataforma a los documentos de diseño y contenido (Diseño, About y FAQ), manteniendo Concepts sin cambios respecto al despliegue en GCP.
- Mejorar la experiencia de navegación en los módulos MOOC (orden, accesibilidad y continuidad de tareas).
- Reducir barreras en las actividades de escritura sin comprometer la calidad del feedback.
- Mantener la línea de despliegue en entorno cloud académico (CeSvImA) con vistas a automatización y operación futura.
- Documentar de forma trazable los cambios (problema, solución, beneficio actual y valor futuro).

1.2.2. Objetivos propuestos por el autor

- Reforzar los flujos de autenticación y recuperación de contraseña mediante verificación y correo transaccional en servidor.

Estos objetivos se agrupan en tres líneas principales: adecuación pedagógica y de contenidos, robustez funcional para usuarios y administración, y preparación del despliegue y la operación del sistema. El orden de prioridad se ha centrado en lo que percibe el alumnado en primer lugar y en aquello que garantiza que los datos y resultados se conserven de forma fiable.

1.3. Alcance

El trabajo se centra en cambios de interfaz, navegación y contenidos: flujo de los MOOC, página de inicio y páginas informativas. Incluye la reorganización de secciones, la coherencia visual y la adaptación del contenido a las directrices docentes, con especial atención a la claridad del itinerario y a reducir barreras en las tareas de escritura.

En lo funcional, se refuerza la seguridad con mejoras en autenticación y recuperación de contraseña, se traslada el envío de correo al backend, se garantiza la persistencia de escritos por usuario y se integra la gestión de conceptos, cuestionarios y resultados desde administración. Esto mejora la administración de la plataforma. También se estabiliza el análisis de textos mediante un proxy en servidor y se preparan materiales descargables para los MOOC.

Además, se incluye una guía operativa para ejecutar la plataforma en local y en CeSvImA, con variables de entorno, publicación de imágenes y actualizaciones sin perder datos. La guía está pensada para que nuevos integrantes puedan reproducir el despliegue sin depender del autor original.

En infraestructura, el alcance cubre la contenedorización y el acceso unificado con proxy inverso, el despliegue en CeSvImA con HTTPS y la monitorización básica, en fase de implementación avanzada pero no terminada. Todo ello se realiza para cubrir el objetivo principal del TFG: asegurar un despliegue seguro y mantenible en la nube. La coordinación institucional permitió abrir los puertos 80/443 para emitir certificados y exponer el servicio de forma segura. También se dejó preparado un pipeline de CI/CD con secretos en GitHub.

Quedan fuera del alcance la automatización completa de pruebas, la provisión de infraestructura con IaC (Terraform), la ejecución local de modelos de IA medianos o grandes y un endurecimiento avanzado de seguridad (WAF, HSTS estricto, rotación automatizada y rate limiting). Todo esto queda como trabajo futuro por tiempo y recursos.

1.4. Metodología

Se siguió un enfoque incremental:

- Revisión de requisitos con las supervisoras.
- Contraste con la plataforma existente.
- Implementación de mejoras.
- Priorización de cambios visibles.
- Validación con pruebas manuales.

Además, se aplicó priorización por valor docente y feedback continuo con las responsables académicas, sin adoptar un marco formal como Scrum. En la vertiente operativa se siguieron prácticas DevOps: control de versiones, contenedorización y preparación de un pipeline CI/CD en GitHub Actions para construir imágenes y desplegar en CeSvImA. Para el control de versiones y la colaboración se trabajó en GitHub, con desarrollo por ramas y merges entre ramas para integrar cambios.

El trabajo se organizó en iteraciones cortas, combinando revisión de documentación, implementación en el repositorio y verificación en local y en Docker. Cada iteración añadió ajustes técnicos y de contenido, cuidando que la navegación no

se rompiera. La coordinación con la dirección del TFG se reflejó en la reordenación de contenidos y la actualización de textos, y las mejoras funcionales se probaron con casos de uso reales (registro, cuestionarios y writings).

1.5. Planificación del proyecto

La planificación inicial planteaba un alcance reducido: aplicar pequeños cambios en la plataforma y completar el despliegue en CeSvImA. A medida que avanzó el trabajo, surgieron necesidades adicionales y propuestas del autor que ampliaron el esfuerzo previsto. Aun así, la línea de despliegue se mantuvo como objetivo desde el inicio y se ejecutó según lo planificado.

1.5.1. Cronograma y tareas

La ejecución se organizó en bloques de trabajo consecutivos, tomando como referencia la memoria de seguimiento. La Tabla 1.1 resume los bloques principales y sus tareas asociadas.

Tabla 1.1: Cronograma resumido y tareas principales.

Bloque (periodo)	Tareas principales
Preparación local (sep)	Set up en local y lectura de TFGs anteriores para entender la plataforma.
Cambios en local (oct–nov)	Cambios funcionales y de contenido en local, verificación en backend y correo transaccional.
Dockerización y ajustes (nov–dic)	Dockerización, resolución de fallos detectados y ajustes de última hora.
Cierre y despliegue (dic–ene)	Dominio y correo profesional, CI/CD, despliegue en CeSvImA y monitorización (en fase de implementación avanzada pero no terminada).

1.5.2. Lista de tareas

La Tabla 1.2 detalla el estado de las tareas al cierre del proyecto.

Tabla 1.2: Lista de tareas y estado al cierre del proyecto.

ID	Tarea	Estado	Observaciones
T1	Configuración local y revisión documental.	Completada	Entorno local y lectura de TFGs previos.
T2	Cambios funcionales en local.	Completada	Reordenación MOOC y ajustes de navegación.

ID	Tarea	Estado	Observaciones
T3	Páginas informativas.	Completada	About y FAQ actualizados; Concepts se mantiene como en GCP (Javier Hernández Contreras) [1].
T4	Verificación en backend.	Completada	Códigos y tokens en servidor.
T5	Correo transaccional (Mailjet).	Completada	Servicio de correo en backend.
T6	Dockerización de la plataforma.	Completada	Validación en contenedor.
T7	Ajustes tras dockerización.	Completada	Fixes de última hora y proxy de análisis.
T8	Privacidad en dashboards.	Completada	Eliminación de filtros sensibles.
T9	Dominio y correo profesional.	Completada	IONOS, destance.com y buzón.
T10	CI/CD y despliegue en CeSvImA.	Completada	Nginx, HTTPS y monitorización en fase de implementación avanzada pero no terminada.
T11	Reestructuración del MOOC.	Completada	Vídeo, pre/post y materiales.
T12	Costes y documentación.	Completada	Guía operativa y estimación de costes.
T13	Monitorización y observabilidad (en fase de implementación avanzada pero no terminada).	En fase de implementación avanzada pero no terminada	Implementación avanzada pendiente de cierre y validación.

1.5.3. Diagrama de Gantt

En este apartado se incluye un pequeño diagrama de Gantt sobre las tareas realizadas y cómo se han distribuido a lo largo del trabajo, recogido en la Figura 1.1.

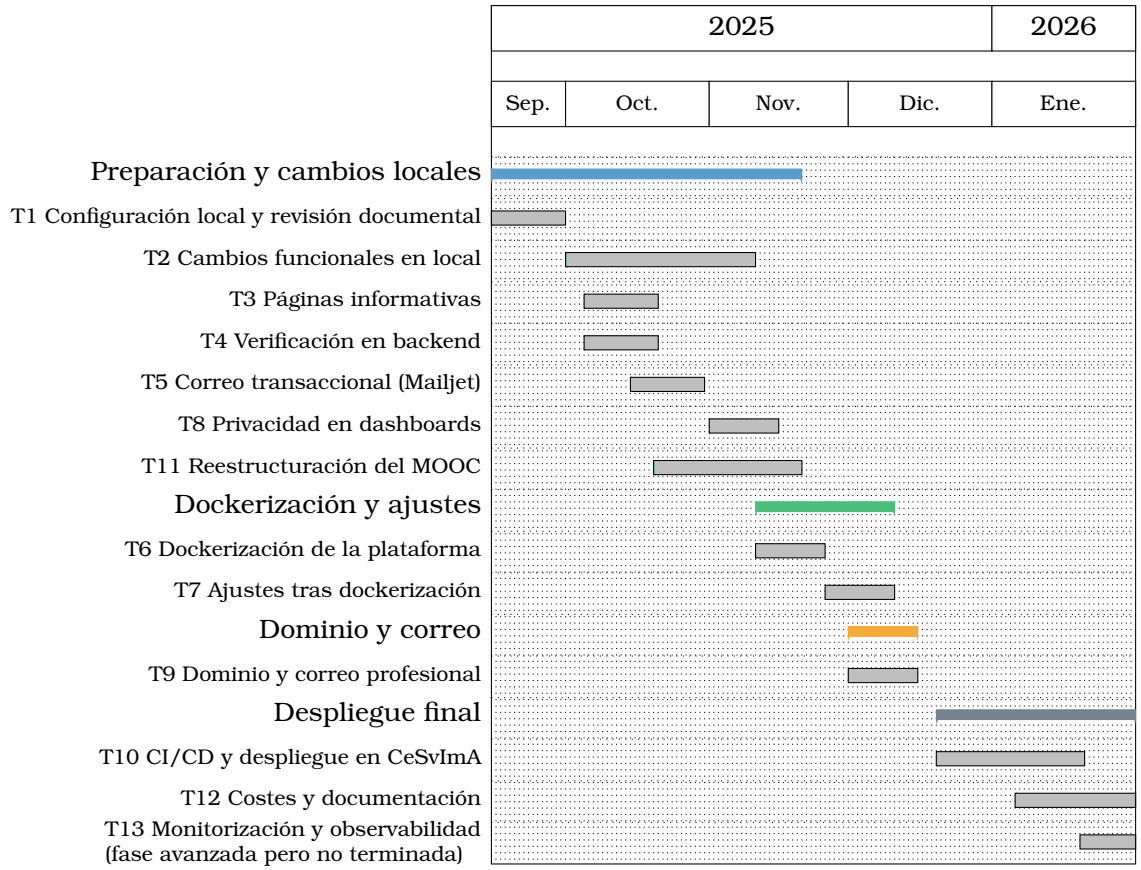

Figura 1.1: Diagrama de Gantt.

1.5.4. Hitos

- Entorno local operativo y requisitos revisados.
- Flujos de registro, verificación y recuperación en backend con correo transaccional.
- Recorrido MOOC reorganizado y contenidos alineados con las directrices docentes.
- Despliegue en CeSvImA con proxy inverso y HTTPS.
- Guía operativa y base de CI/CD documentadas.

1.5.5. Desviaciones y ajustes

El alcance real creció respecto a la idea inicial de cambios mínimos.

Además de las tareas planificadas, surgieron actividades que no estaban reflejadas en el Gantt inicial y que se incorporaron como desviaciones:

- Actualización de páginas informativas (About y FAQ) y ajustes de contenido docente; Concepts se mantiene como en GCP [1].
- Revisión de privacidad en dashboards: eliminación de filtros por lengua materna, nacionalidad o género.
- Proxy de análisis en backend para evitar CORS (restricciones de acceso entre distintos orígenes) y estabilizar el funcionamiento en Docker.
- Separación de docker-compose local y despliegue, con Certbot (cliente para emitir y renovar certificados), HTTPS y rutas de monitorización (en fase de implementación avanzada pero no terminada).

Además, la salida de un compañero redujo capacidad del equipo y obligó a redistribuir tareas, lo que ralentizó algunas fases. La estrategia de despliegue se mantuvo desde el inicio y no cambió, pero se priorizó cerrar los cambios funcionales antes de automatizaciones avanzadas (IaC y pruebas automáticas), que quedaron como trabajo futuro.

1.6. Estructura del documento

La presente memoria se organiza en diez capítulos que describen de forma progresiva el desarrollo del proyecto, desde su contexto inicial hasta la implementación y los resultados obtenidos.

Capítulo 1: Introducción y objetivos – Expone el contexto general del proyecto De-Stance, su relación con el proyecto de investigación RACISMMAFF y el grupo GAMMA de la UPM. Se detallan la motivación, los objetivos generales y específicos, el alcance del trabajo, la metodología empleada, la planificación temporal y los hitos alcanzados.

Capítulo 2: Estado del arte y trabajos previos – Presenta los antecedentes técnicos y académicos sobre los que se fundamenta este TFG. Se revisan las memorias previas de Gonzalo Álvarez García, Daniel Ramón Robertson, Ana Clara Murillo García y Javier Hernández Contreras, así como otros TFG recientes relacionados con la automatización del despliegue y las metodologías DevOps en entornos educativos y cloud [1], [5]-[9]. Además, se analizan las brechas técnicas detectadas en la plataforma De-Stance y las buenas prácticas aplicadas en plataformas educativas web.

Capítulo 3: Especificación de requisitos – Define los requisitos funcionales y no funcionales del sistema, la trazabilidad entre ellos y las pruebas asociadas. Este capítulo establece las bases sobre las que se diseña e implementa la nueva versión de la plataforma, garantizando coherencia entre objetivos, funcionalidades y validaciones.

Capítulo 4: Tecnologías utilizadas – Describe las tecnologías y herramientas empleadas tanto en frontend como en backend, bases de datos y despliegue: SvelteKit, Tailwind CSS, Node.js, Express, MongoDB, Docker, Nginx, Mailjet, Certbot, Prometheus y Grafana, entre otras. También se justifican las decisiones

técnicas adoptadas en función de la estabilidad, mantenibilidad y compatibilidad con CeSvImA.

Capítulo 5: Diseño del sistema – Expone la arquitectura funcional, técnica y de despliegue de la plataforma, incluyendo la evaluación comparativa de proveedores cloud (AWS, Azure, GCP, CeSvImA) y la justificación del entorno adoptado. Se presentan las decisiones sobre contenedorización, proxy inverso, monitorización (en fase de implementación avanzada pero no terminada), modelo de datos y flujo principal de información, así como una propuesta de arquitectura IaC para futuras migraciones.

Capítulo 6: Implementación – Detalla los cambios funcionales y estructurales realizados durante el desarrollo, diferenciando los solicitados por las responsables académicas y los propuestos por el autor. Se documentan los flujos de registro, autenticación y recuperación de contraseña, el correo transaccional, la ejecución en Docker, el despliegue en CeSvImA con HTTPS y proxy Nginx, la automatización de entornos y la gobernanza de secretos en GitHub Actions.

Capítulo 7: Pruebas y validación – Recoge el conjunto de pruebas funcionales, de persistencia y de despliegue efectuadas para verificar el correcto funcionamiento de la plataforma. También se describen las incidencias detectadas, las soluciones aplicadas y las propuestas de automatización de pruebas como línea de mejora futura.

Capítulo 8: Resultados – Resume los resultados obtenidos, mostrando las evidencias visuales de los cambios implementados en la interfaz, el recorrido educativo y los componentes principales de De-Stance. Se analizan los efectos de las mejoras sobre la usabilidad, la claridad del flujo y la experiencia de aprendizaje.

Capítulo 9: Análisis de impacto – Evalúa el impacto del trabajo desde una perspectiva técnica y social, considerando la experiencia de usuario, la mantenibilidad, la escalabilidad, la seguridad y la operación en la nube. Asimismo, se reflexiona sobre la sostenibilidad y la alineación del proyecto con los Objetivos de Desarrollo Sostenible (ODS) de la Agenda 2030.

Capítulo 10: Conclusiones – Presenta las conclusiones finales del proyecto, las limitaciones identificadas y las líneas de trabajo futuro. También se incluye una estimación preliminar de costes de infraestructura, la capacidad de escalado y un agradecimiento a las personas y entidades que han contribuido al desarrollo del trabajo.

Finalmente, los Anexos recogen información técnica complementaria: el informe de originalidad mediante Turnitin, inventario de endpoints, rutas del frontend y listado de secrets empleados en GitHub Actions, útiles para la reproducibilidad y el mantenimiento del sistema.

Capítulo 2

Estado del arte y trabajos previos

2.1. Punto de partida del proyecto

Al inicio del trabajo, De-Stance ya incluía micro-MOOCs, cuestionarios pre y post, perfiles de usuario y un panel de resultados. La arquitectura está basada en un frontend SvelteKit, backend Node/Express y una base de datos MongoDB. El reto principal no era crear funcionalidades desde cero, sino consolidar la experiencia educativa, ajustar el contenido a las pautas docentes y mejorar la navegación.

El frontend ya ofrecía páginas informativas, acceso a cuestionarios y módulos de aprendizaje, pero coexistían rutas y textos que no reflejaban la nomenclatura docente. En paralelo, el backend disponía de un API REST para usuarios, cuestionarios, cursos y dashboards, además de un panel de administración (AdminJS) para editar contenidos sin tocar código. La base de datos guardaba tanto contenido editorial (conceptos y cuestionarios) como evidencias de aprendizaje (escritos y resultados), lo que permitía recuperar el progreso por usuario.

En operación, el equipo contaba con ejecución local y contenedores, pero la configuración estaba dispersa y algunas piezas dependían del cliente: el envío de correos y parte de las validaciones se realizaban en el navegador, y el análisis automático se conectaba a servicios externos sin pasar por el servidor.

Este punto de partida explica por qué el trabajo se orienta a centralizar flujos críticos en backend, estabilizar el despliegue y alinear el recorrido educativo con las guías entregadas.

2.2. Trabajos previos relacionados

Se revisaron las memorias de los Trabajos Fin de Grado centrados en la plataforma De-Stance, desarrollados por Gonzalo Álvarez García, Daniel Ramón Robertson, Ana Clara Murillo García y Javier Hernández Contreras, con el objetivo de alinear la estructura de la presente memoria y situar este trabajo dentro de la evolución de la plataforma [1], [5]-[7]. Estos proyectos constituyen una base sólida sobre la que se apoya el desarrollo actual, aportando contexto sobre

el rediseño de la interfaz, los módulos interactivos, la analítica educativa y las herramientas de retroalimentación.

En concreto, Gonzalo Álvarez García [5] aborda la plataforma en su conjunto, consolidando su arquitectura y la coherencia visual; Daniel Ramón Robertson [6] desarrolla módulos interactivos orientados a la participación activa del alumnado; Ana Clara Murillo García [7] amplía el sistema con mecanismos de análisis lingüístico y feedback automático; y Javier Hernández Contreras [1] documenta la administración y los paneles de control orientados a la monitorización (en fase de implementación avanzada pero no terminada en el despliegue actual) y la gestión de usuarios.

El presente TFG toma como punto de partida una plataforma ya funcional, con panel de administración, cuestionarios y primeros módulos de análisis y retroalimentación. Sobre esa base, se reestructura la arquitectura para reforzar la consistencia del contenido, la experiencia de usuario y los flujos de autenticación y notificación. Asimismo, se extiende el ámbito técnico hacia la línea de despliegue en entorno cloud académico (CeSvImA), integrando elementos de seguridad, proxy inverso y correo transaccional desde el backend.

Del mismo modo, se heredan las prácticas de despliegue y ejecución local documentadas en los trabajos anteriores, que en esta memoria se consolidan en una guía operativa más clara, con separación explícita de servicios, parametrización mediante variables de entorno y automatización reproducible. El objetivo no es reemplazar la labor previa, sino cerrar el ciclo entre el diseño docente, la implementación técnica y la operación sostenible, facilitando la continuidad del proyecto y su mantenimiento en entornos reales.

Además, para situar este trabajo dentro de un contexto técnico más amplio, se analizaron otros Trabajos Fin de Grado y tesis recientes que abordan la automatización del despliegue en la nube y la adopción de metodologías DevOps en entornos académicos y profesionales. Entre ellos, el trabajo de Santiago Darío Montero Cabezas (Universidad Politécnica de Madrid, 2024), que desarrolla una solución de integración continua y despliegue en servicios cloud mediante un pipeline CI/CD completo basado en AWS y GitHub Actions [8], y el de José Andrés Pérez López (Universidad de Alicante, 2023), que implementa una arquitectura de integración y despliegue continuo para una aplicación web, incluyendo la evaluación comparativa de distintos proveedores de nube [9].

Estos trabajos reflejan una tendencia consolidada en el ámbito académico hacia la automatización del despliegue, la reproducibilidad y la trazabilidad en entornos cloud. Siguiendo esa línea, el presente TFG aplica dichas prácticas al ecosistema De-Stance, trasladando las estrategias de CI/CD y despliegue seguro al contexto de una plataforma educativa universitaria. Este enfoque refuerza la coherencia metodológica del proyecto y garantiza que las contribuciones técnicas no solo sean funcionales, sino también sostenibles, escalables y alineadas con las buenas prácticas de ingeniería de software en entornos cloud institucionales como CeSvImA.

2.3. Análisis de la situación inicial y brechas técnicas

El punto de partida del despliegue estaba orientado a pruebas locales y a un funcionamiento básico en contenedores, con acceso por HTTP en el puerto 8080. Esta configuración era suficiente para validar el producto, pero no para una puesta en producción con requisitos de seguridad: no existía cifrado TLS activo, las cookies no podían marcarse como seguras y el acceso dependía de puertos no estándar. Además, la emisión de certificados era inviable mientras no se habilitaban los puertos 80/443 en el firewall institucional.

En operación, las actualizaciones dependían de builds en la propia máquina y de ejecuciones manuales, lo que aumentaba el riesgo de inconsistencias entre versiones y dificultaba el rollback. Tampoco había una separación clara entre entorno local y despliegue, lo que generaba configuraciones mezcladas y variaciones difíciles de reproducir por nuevas personas del equipo.

La observabilidad era limitada: los diagnósticos se apoyaban en logs puntuales sin métricas ni paneles, por lo que detectar problemas de rendimiento o saturación requería inspección manual; su mejora está en fase de implementación avanzada pero no terminada. A nivel de red, el dominio debía resolverse mediante CNAME hacia el FQDN de CeSvImA debido a la variabilidad de IPs, lo que obligaba a gestionar el redireccionamiento del dominio raíz en el registrador.

Estas brechas justifican la necesidad de activar HTTPS, estabilizar el proxy, separar despliegue de build y añadir una monitorización que, en el despliegue actual, está en fase de implementación avanzada pero no terminada.

2.4. Buenas prácticas en plataformas educativas

En los trabajos previos revisados se insiste en una navegación clara, reducción de barreras en tareas y contenidos coherentes con el objetivo pedagógico. En plataformas con reflexión escrita y feedback, es clave permitir avanzar sin bloqueos excesivos y ofrecer recomendaciones para mejorar la calidad de las respuestas [5], [6].

A nivel de diseño instruccional, resulta efectivo estructurar el aprendizaje en pasos cortos y visibles, priorizar el recurso principal (por ejemplo, el vídeo) y acompañarlo de instrucciones concretas antes de las tareas. También se considera buena práctica guardar el progreso de forma incremental, de modo que el usuario pueda retomar la actividad sin pérdida de información y con continuidad narrativa.

En usabilidad y accesibilidad se buscan textos claros, contraste adecuado, jerarquía tipográfica estable y llamadas a la acción consistentes. En ética y privacidad se prefiere mostrar resultados por usuario y evitar filtros que puedan identificar grupos pequeños, sobre todo en contextos sensibles. Estas pautas se reflejan en la reorganización del recorrido, el ajuste del mínimo de palabras y el control de acceso a resultados.

Capítulo 3

Especificación de requisitos

3.1. Requisitos funcionales

Los requisitos funcionales describen lo que la plataforma debe permitir hacer al usuario y qué operaciones son imprescindibles para completar el recorrido educativo. Se expresan en términos de acciones concretas y de resultados observables, y recogen los requisitos identificados durante el análisis.

- Registro con verificación por código y recuperación de contraseña mediante correo transaccional desde backend.
- Inicio de sesión con emisión de token y sesión persistente para consumo seguro de la API.
- Navegación principal clara: Home, Concepts, Pre-course Questionnaires, Micro-MOOCs, Post-course Questionnaires, My Dashboard, About, FAQ y Contact.
- Integración de MOOC con tres pasos: Video Lesson, Pre-Writing y Post-Writing, y materiales descargables por módulo.
- Guardado incremental de escritos y rehidratación al volver a entrar en cada sección.
- Permitir avanzar entre pasos sin bloqueos innecesarios por límite de palabras, guardando cada avance.
- Actualizar textos y estructura de About y FAQ según documentación docente; Concepts se mantiene sin cambios [1].
- Incluir enlaces a herramientas externas (ACAIA) o mostrar estado Work in Progress.
- Persistencia de resultados de cuestionarios y escritos MOOC por usuario, visibles en My Dashboard.

- Panel de administración para gestionar contenidos (conceptos, cuestionarios, cursos y dashboards) sin intervención técnica.
- Análisis automático de textos accesible vía backend para evitar dependencias directas del navegador.
- Formulario de contacto operativo con envío a cuenta de soporte institucional.

Estos requisitos se han priorizado porque afectan directamente a la comprensión del itinerario y a la continuidad del aprendizaje.

3.2. Requisitos no funcionales

Los requisitos no funcionales recogen cualidades del sistema relacionadas con la experiencia, la seguridad y el mantenimiento, más allá de funciones concretas.

- Coherencia visual con las guías de diseño entregadas.
- Accesibilidad básica (tipografía legible, contraste y navegación directa).
- Mantenibilidad del contenido (estructura clara y reutilizable).
- Seguridad en comunicaciones sensibles (correo y análisis automático) centralizadas en backend.
- Autenticación y autorización basadas en JWT, con control de rol administrador.
- Privacidad: resultados por usuario y ausencia de filtros sensibles en vistas agregadas.
- Observabilidad básica: healthchecks, logs y métricas para diagnóstico.
- Despliegue reproducible con separación de servicios y proxy inverso.
- Portabilidad entre ejecución local, Docker y entorno VPS.
- Resiliencia ante fallos de servicios externos mediante proxy y control de errores.

Su cumplimiento permite que la plataforma pueda operar de forma estable y que futuros cambios no comprometan la calidad del servicio, tomando como referencia guías de verificación de seguridad en aplicaciones web como OWASP ASVS [10].

3.3. Trazabilidad de requisitos

La Tabla 3.1 conecta requisitos clave con las partes del sistema donde se implementaron y las pruebas que los validaron.

Tabla 3.1: Trazabilidad de requisitos, implementación y validación.

Requisito	Implementación	Pruebas
RF-01 Registro y recuperación con correo transaccional	Autenticación en backend y envío de correo con Mailjet.	Pruebas funcionales.
RF-02 Inicio de sesión y sesión persistente	JWT, cookies seguras y control de sesión en backend.	Pruebas funcionales.
RF-03 Navegación principal y páginas informativas	Reordenación de rutas y actualización de About/FAQ; Concepts se mantiene como en GCP [1].	Pruebas funcionales.
RF-04 MOOC en tres pasos y recursos descargables	Reestructuración del flujo Video–Pre–Post y materiales.	Pruebas funcionales.
RF-05 Guardado incremental y rehidratación	Persistencia de writings y recuperación por usuario.	Pruebas de persistencia.
RF-06 Avance sin bloqueos por mínimo de palabras	Ajuste de límites y mensajes de orientación.	Pruebas funcionales.
RF-07 Resultados por usuario en My Dashboard	Panel de resultados y control de acceso por usuario.	Pruebas de persistencia.
RF-08 Administración de contenidos	AdminJS y rutas /admin integradas en proxy.	Pruebas funcionales.
RF-09 Análisis automático vía backend	Proxy de análisis en servidor para evitar CORS.	Pruebas de despliegue.
RF-10 Formulario de contacto operativo	Endpoint de contacto en backend y correo institucional.	Pruebas funcionales.

Capítulo 4

Tecnologías utilizadas

La capa de interfaz se implementa con SvelteKit y Tailwind CSS. SvelteKit aporta un enrutado claro y mantiene coherencia con la evolución del proyecto descrita en trabajos previos [1]. Tailwind se utiliza para asegurar consistencia visual y rapidez en la maquetación, lo que facilita mantener la alineación con las guías docentes.

En el backend se utiliza Node.js con Express, una combinación que permite definir rutas de API de forma directa, gestionar autenticación y centralizar la lógica de negocio. MongoDB se emplea como base de datos por su flexibilidad para almacenar respuestas y resultados heterogéneos (cuestionarios, writings y análisis), algo ya explorado en fases anteriores del proyecto [5].

Para el envío de correos se adoptó Mailjet como servicio transaccional. Esta elección responde a la necesidad de centralizar la verificación en servidor y mantener un flujo estable de registro y recuperación, con un volumen gratuito mayor que alternativas basadas en cliente.

La contenedorización se realiza con Docker y el acceso unificado se gestiona con Nginx. Docker es una plataforma abierta para desarrollar, distribuir y ejecutar aplicaciones, separando la aplicación de la infraestructura para acelerar la entrega de software [11]. En este trabajo, permite reproducir entornos en local y en VPS con configuraciones consistentes, mientras que Nginx actúa como punto único de entrada y distribuye el tráfico hacia frontend, backend y panel de administración.

El análisis de textos se apoya en Hugging Face a través de un cliente externo, siguiendo la línea de trabajo de análisis automático y feedback documentada en memorias anteriores [7]. Hugging Face es una empresa y comunidad abierta que ofrece herramientas, modelos y plataformas de machine learning centradas en ciencia de datos y procesamiento del lenguaje natural [12]. Esta decisión evita la carga de modelos pesados en la infraestructura local y mejora la estabilidad en entornos con recursos limitados. Como soporte multimedia se integran recursos externos como Genially, que proporcionan contenidos interactivos sin necesidad de alojarlos en el servidor propio.

La automatización de la entrega se apoya en GitHub Actions para ejecutar pruebas, construir imágenes y desplegar por SSH, y en Docker Hub como registro de imágenes versionadas. Esta combinación permite separar construcción y despliegue, mantener trazabilidad por commit y evitar builds en la VPS.

Para la seguridad de comunicaciones se utiliza Certbot (Let's Encrypt) como cliente ACME y Nginx como proxy inverso, lo que habilita HTTPS con renovación automática. En observabilidad se emplean Prometheus y Grafana junto con node_exporter y cAdvisor; esta línea está en fase de implementación avanzada pero no terminada y aporta métricas del host y de los contenedores con paneles accesibles desde el mismo dominio.

Capítulo 5

Diseño del sistema

5.1. Arquitectura funcional

La plataforma se estructura en módulos de aprendizaje (MOOC) y páginas informativas. Cada MOOC está compuesto por secciones con tipos de contenido (texto, imagen, vídeo y tareas de escritura). Esta estructura facilita la reutilización de componentes y el mantenimiento de contenidos.

La arquitectura se organiza en tres capas: presentación (frontend), lógica de negocio (backend) y datos (base de datos). Esta separación mejora la claridad, el mantenimiento y la evolución del sistema sin afectar al resto de componentes. Además, el uso de certificados y HTTPS refuerza la seguridad del acceso al cifrar la comunicación entre usuarios y plataforma.

La arquitectura de despliegue se resume en la Figura 5.1. El proxy Nginx concentra el acceso externo, el backend gestiona lógica y persistencia, y la monitorización se apoya en Prometheus y Grafana con métricas del host y de contenedores. Certbot automatiza la emisión y renovación de certificados TLS.

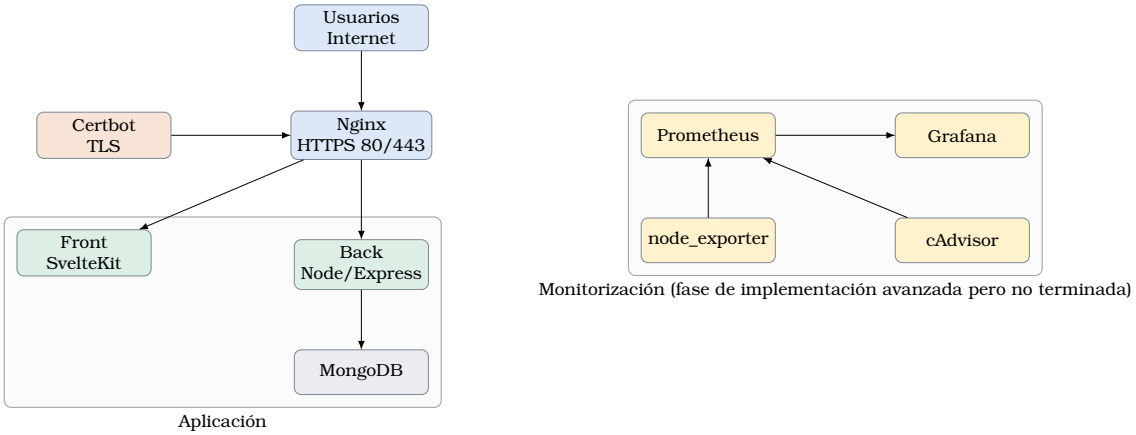

Figura 5.1: Arquitectura de despliegue en CeSvImA con proxy, servicios de aplicación y monitorización en fase de implementación avanzada pero no terminada.

5.2. Diseño de la experiencia de usuario

Se siguió la estructura indicada en los documentos de diseño: cabecera con rutas clave, inicio en cuatro pasos y páginas informativas con textos revisados. La prioridad fue que el recorrido educativo fuera claro y sin bloqueos.

5.3. Línea de despliegue en entorno cloud

El despliegue inicialmente se planteó en un proveedor cloud (AWS, GCP o Azure). Antes de cerrar ese estudio, se solicitó un VPS en CeSvImA y fue concedido. Una evaluación breve mostró que era suficiente para desplegar De-Stance, sobre todo porque el uso actual es reducido.

El despliegue en CeSvImA se hace con frontend y backend separados, proxy inverso y certificados TLS. La apertura de los puertos 80/443 permitió activar HTTPS y emitir certificados con Certbot, mejorando la seguridad al cifrar el acceso y las sesiones. Esta línea marca la configuración y la documentación técnica, y sirve como base para futuros entornos.

5.4. Evaluación de proveedores cloud

Este apartado se incluye pensando en un uso más extendido de la plataforma o en despliegues fuera de CeSvImA. Como referencia para un posible despliegue fuera de CeSvImA se han considerado AWS, Azure y Google Cloud. La idea no es decir cuál es "mejor", sino qué opción encaja por coste y complejidad.

La comparación entre Amazon Web Services (AWS), Google Cloud Platform (GCP) y Microsoft Azure se ha realizado considerando su aplicación al despliegue de la plataforma De-Stance, evaluando aspectos como coste, facilidad de uso, escalabilidad y adecuación al contexto académico.

AWS. AWS dispone de un ecosistema consolidado y muy amplio que permite evolucionar desde entornos básicos hasta arquitecturas gestionadas más complejas.

Amazon Lightsail ofrece un entorno similar a un VPS con precios fijos, adecuado para proyectos educativos que requieren estabilidad y configuración sencilla.

Además, servicios como Elastic Container Service (ECS) o App Runner permiten desplegar contenedores con escalado automático, mientras que CloudFront proporciona una red global de distribución de contenidos (CDN) de baja latencia, útil en escenarios con usuarios distribuidos [13], [14]. Su principal inconveniente es la complejidad del catálogo de servicios y la posibilidad de sobrecoste si no se controla bien el uso de recursos [13].

Google Cloud. Google Cloud destaca por su enfoque en la simplicidad y la automatización.

El servicio Cloud Run permite ejecutar contenedores con escalado automático y modelo de pago por uso, sin necesidad de gestionar infraestructura, lo que

facilita la continuidad del servicio en entornos académicos como De-Stance. Su red global y las herramientas integradas de monitorización ayudan a mantener un rendimiento estable [13]; en este proyecto, la monitorización está en fase de implementación avanzada pero no terminada.

Como desventaja, su menor presencia en entornos universitarios y corporativos tradicionales puede requerir una mayor curva de adaptación [13].

Azure. Microsoft Azure se integra fácilmente con herramientas del ecosistema Microsoft, habituales en entornos educativos e institucionales.

Azure Container Apps ofrece un despliegue serverless basado en contenedores, mientras que Azure Front Door y Azure CDN proporcionan distribución global de contenido, funcionalidades equivalentes a CloudFront en AWS [15], [16].

Su principal inconveniente es la amplitud del catálogo y la complejidad para estimar costes sin recurrir a la calculadora oficial [15].

En cuanto a costes, los tres proveedores utilizan un modelo de facturación pay-as-you-go, donde el precio final depende del uso de recursos como cómputo, almacenamiento y transferencia.

En el caso de De-Stance, cuyo uso actual se limita a un entorno académico controlado con pocos usuarios simultáneos, una máquina virtual contenedorizada proporciona un equilibrio adecuado entre estabilidad, mantenimiento y control. Si la plataforma aumenta su número de usuarios, podrían valorarse servicios gestionados de contenedores y bases de datos para mejorar la escalabilidad y la disponibilidad.

Google Cloud dispone además de un nivel gratuito con una instancia e2-micro, suficiente para pruebas iniciales o despliegues de bajo consumo [17]. Como referencia se toma un servidor virtual similar al actual (2 vCPU, 4 GB RAM, almacenamiento persistente y Linux). La Tabla 5.1 resume servicios equivalentes a un VPS y un coste orientativo. En proveedores comerciales el precio cambia por región y modelo de facturación, por lo que se recomienda usar sus calculadoras oficiales [18]-[20].

Tabla 5.1: Soluciones equivalentes a un VPS de referencia.

Proveedor	Servicio	Configuración aproximada	Precio orientativo
AWS	Lightsail	2 vCPU, 4 GB RAM, 80 GB SSD	24 USD/mes
Google Cloud	Compute Engine (E2)	2 vCPU, 4 GB RAM	˜62,03 USD/mes (ejemplo)
Azure	Virtual Machines (A2 v2)	2 vCPU, 4 GB RAM	68,74 USD/mes
CeSvImA (académico)	VPS	2 vCPU, 4 GB RAM, 100 GB	0 EUR

En este escenario, las soluciones gestionadas reducen significativamente la carga operativa frente a un servidor VPS tradicional, aunque implican una mayor dependencia del proveedor.

Para De-Stance, a medio plazo resulta razonable combinar un servicio PaaS para el backend, una base de datos gestionada y un CDN para el frontend, de forma que se mantenga un equilibrio entre escalabilidad, sencillez de mantenimiento y coste.

Servicios como Cloud Run en Google Cloud, App Runner en AWS o App Service en Azure encajan adecuadamente en este esquema, al ofrecer despliegue automático y escalado transparente sin necesidad de administrar servidores [13]-[15]. Aunque Kubernetes permitiría un mayor control sobre la infraestructura, su complejidad y requisitos de mantenimiento resultan excesivos para el tamaño y el objetivo actual del proyecto.

La Tabla 5.2 resume los servicios más representativos de cada proveedor en un escenario avanzado con contenedores gestionados y redes de distribución de contenido (CDN).

Tabla 5.2: Escenario avanzado con contenedores gestionados y CDN.

Proveedor	Servicio principal	Modelo	Uso recomendado
AWS	App Runner + CloudFront	PaaS + CDN	Escalado automático y distribución global
Google Cloud	Cloud Run + Cloud CDN	Contenedores serverless	Pago por uso y despliegue rápido
Azure	Container Apps + Front Door	Contenedores serverless	Integración empresarial y edge

En cuanto al rendimiento, la latencia global no puede eliminarse completamente, aunque puede reducirse mediante el uso de CDN, balanceo geográfico y réplicas regionales de servicios críticos.

Si en el futuro De-Stance se extiende a un público internacional, una estrategia viable consistiría en servir el frontend desde una red CDN y desplegar el backend en varias regiones, con bases de datos replicadas o de lectura distribuida [16].

En conclusión, la comparación entre Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP) y CeSvImA se ha realizado teniendo en cuenta el coste, la escalabilidad, la facilidad de uso y la adecuación al contexto académico del proyecto.

Los proveedores comerciales ofrecen gran capacidad de escalado y una amplia variedad de servicios, pero su modelo de facturación por consumo y la complejidad de configuración los hacen menos adecuados para un entorno universitario. Aunque disponen de niveles gratuitos y créditos educativos, estos son limitados y no garantizan un despliegue continuo durante todo el desarrollo [13], [17].

Por su parte, CeSvImA proporciona un entorno gratuito, estable y con soporte

técnico cercano, lo que facilita el mantenimiento del despliegue y asegura la continuidad del servicio sin depender de políticas comerciales externas [2]. Actualmente cubre todas las necesidades de De-Stance, pensada para un número reducido de usuarios, y seguirá siendo adecuada mientras el uso permanezca moderado.

Si en el futuro la plataforma aumenta su número de usuarios y requiere disponibilidad continua a nivel global, resultará razonable considerar una migración progresiva hacia un entorno cloud comercial con mayor capacidad de escalado y automatización.

En conjunto, CeSvImA representa la opción más equilibrada para el estado actual de De-Stance, al ofrecer una infraestructura estable, sin coste y con soporte institucional. Además, proporciona una base sólida y sostenible sobre la que evolucionar conforme la plataforma crezca en alcance y complejidad.

5.5. Ejecución de los servicios de inteligencia artificial

En esta sección se analizan las distintas alternativas para ejecutar los servicios de inferencia de los modelos de inteligencia artificial utilizados en la plataforma. En concreto, se estudian dos enfoques: la ejecución mediante contenedores gestionados y la ejecución en entornos serverless. El objetivo es evaluar las diferencias entre ambas opciones en términos de funcionamiento, rendimiento y coste, y justificar la decisión adoptada en el proyecto.

5.5.1. Ejecución mediante contenedores

Según la documentación oficial de Microsoft Azure Machine Learning, es posible ejecutar modelos de inteligencia artificial como servicios de inferencia utilizando contenedores personalizados [21]. En este enfoque, el modelo se incluye dentro de un contenedor junto con todo lo necesario para su ejecución, como librerías, dependencias y código. Este contenedor se despliega como un servicio accesible a través de una dirección web, mientras que la plataforma se encarga de ponerlo en marcha y mantenerlo operativo.

En este tipo de despliegue, el desarrollador decide cómo se ejecuta el modelo y qué recursos se le asignan, como la cantidad de memoria o el número de instancias que van a ejecutar el servicio. Al tratarse de un servicio que permanece activo, el modelo puede mantenerse cargado en memoria, lo que evita tiempos de espera al recibir nuevas peticiones y proporciona un comportamiento más estable en cuanto al tiempo de respuesta.

5.5.2. Ejecución en entornos serverless

Por otro lado, la documentación oficial de Amazon Web Services describe Amazon SageMaker Serverless Inference como una forma de ejecutar modelos de inteligencia artificial sin necesidad de gestionar servidores [22]. En este enfoque, el servicio solo se activa cuando llegan peticiones y se apaga cuando no hay uso,

adaptándose automáticamente a la carga.

Este tipo de ejecución resulta especialmente adecuado cuando el número de peticiones es irregular, ya que el coste depende únicamente del uso real del servicio y no se incurre en gastos durante los periodos de inactividad. No obstante, la propia documentación de AWS indica que este tipo de soluciones puede introducir un retraso inicial cuando el servicio se activa después de un periodo sin uso, dependiendo del tamaño del modelo y del tiempo necesario para ponerlo en funcionamiento. Para reducir este efecto, AWS permite mantener el servicio preparado de forma permanente, aunque esta opción implica un coste adicional.

5.5.3. Comparación de rendimiento y coste

Desde el punto de vista del rendimiento, el uso de contenedores gestionados ofrece un comportamiento más predecible, ya que el servicio permanece activo y el modelo está listo para responder en todo momento. En cambio, las soluciones serverless priorizan la flexibilidad y el ahorro de costes, pero pueden presentar tiempos de respuesta más elevados en las primeras peticiones si el servicio no estaba activo previamente.

En cuanto al coste, el despliegue basado en contenedores suele implicar un gasto más constante, relacionado con los recursos que se mantienen asignados al servicio. Por el contrario, la ejecución serverless permite reducir costes cuando no hay actividad, ya que solo se paga cuando el modelo se utiliza.

Es importante matizar que, aunque la integración de Hugging Face con Azure no conlleva un coste adicional como servicio mientras se encuentra en versión preliminar, la ejecución de los modelos de inteligencia artificial sí implica el consumo de recursos de computación de Azure, los cuales pueden generar costes en función del uso [23]. En la documentación oficial de Azure Machine Learning no se especifica un modelo de precios independiente para el uso de modelos de Hugging Face, ya que estos se ejecutan sobre recursos estándar de la plataforma. En consecuencia, el coste asociado depende exclusivamente del tipo de infraestructura utilizada para el despliegue (instancias de computación, contenedores gestionados, endpoints online, etc.), y no del modelo en sí. Esta falta de una referencia de precios directa para modelos de Hugging Face obliga a estimar los costes a partir de los servicios subyacentes empleados para la inferencia.

En este contexto, Azure ofrece capas gratuitas en algunos de sus servicios de ejecución de contenedores. Por ejemplo, en Azure Container Apps, los primeros millones de solicitudes mensuales, así como una cantidad limitada de uso de CPU y memoria, se incluyen sin coste dentro de la capa gratuita. Bajo configuraciones moderadas —como un número reducido de solicitudes simultáneas y tiempos de ejecución acotados— es posible que un servicio experimental o de baja carga no genere costes económicos directos. Aunque Azure Container Apps está orientado principalmente a la ejecución de microservicios y no específicamente al despliegue de modelos de inteligencia artificial, esta referencia resulta útil para obtener una estimación aproximada del coste de ejecutar servicios de inferencia

en contenedores gestionados. No obstante, para modelos de mayor tamaño y consumo de recursos, como los empleados en este proyecto, es previsible que se superen los umbrales gratuitos y se incurra en costes asociados al uso efectivo de la infraestructura.

Por este motivo, las capas gratuitas se consideran únicamente como una orientación inicial y no como una solución garantizada para un despliegue en producción, especialmente en escenarios con modelos pesados o con un volumen de peticiones creciente.

La Tabla 5.3 resume una estimación de coste anual para el enfoque serverless en AWS, obtenida con la calculadora oficial de SageMaker [24]. En dicha calculadora la región Europa (España), eu-south-2, no ofrece esta solución, por lo que se seleccionó Europa (Irlanda), eu-west-1, donde sí está disponible.

Tabla 5.3: Comparativa de costes anuales para inferencia de IA.

Modalidad	Servicio	Región	Coste 12 meses	Notas
Serverless	AWS SageMaker Serverless Inference	Europa (Irlanda) eu-west-1	14,76 USD	Estimación calculadora AWS; eu-south-2 no disponible
Contenedores	Azure Machine Learning (contenedores)	Por definir	—	Pendiente de cálculo en calculadora de Azure

5.5.4. Decisión adoptada y justificación

En el contexto de este proyecto, los modelos empleados (RoBERTa y T5) son modelos de gran tamaño y con un consumo de recursos elevado. Teniendo esto en cuenta, una solución basada en contenedores gestionados resulta más adecuada si se desea ejecutar la inferencia dentro de la infraestructura cloud, ya que permite asignar los recursos necesarios y evitar retrasos en la respuesta.

Las soluciones serverless son más apropiadas para modelos más ligeros o escenarios con un uso muy irregular. Dadas las limitaciones del entorno académico y el estado actual del proyecto, mantener la inferencia como un servicio externo continúa siendo la opción más estable, quedando estas alternativas como posibles líneas de trabajo futuro.

5.6. Arquitectura IaC propuesta

Infraestructura como Código (IaC) consiste en describir la infraestructura con ficheros versionados para poder reproducirla y auditarla. En esta fase del proyecto no se aplica IaC; este apartado se presenta como propuesta para trabajo futuro. En la propuesta se define una red virtual con subred pública para proxy y aplicación, y subred privada para la base de datos. La aplicación se ejecuta en contenedores, Nginx actúa como proxy inverso y se habilita TLS con certificados

gestionados. La base de datos iría en un servicio gestionado o en una VM aislada con copias de seguridad. El pipeline de CI/CD construye imágenes, las publica en un registro privado y el despliegue se aplica con IaC (Terraform o equivalente) para tener control de cambios y rollback.

Esta arquitectura mantiene la separación de servicios ya usada en CeSvImA y añade reproducibilidad y automatización. Aunque Terraform no se usa en esta fase, se incluye para dejar preparada una futura migración a cloud propio.

5.7. Arquitectura técnica

La plataforma sigue una arquitectura de tres capas: frontend SvelteKit, backend Node/Express y base de datos MongoDB.

Como decisión técnica se comparó el uso de un proxy inverso dedicado frente a un proxy interno integrado en el backend. Un proxy interno puede ser suficiente durante el desarrollo, pero añade carga a la aplicación y limita el control sobre la terminación TLS, las cabeceras de seguridad, el filtrado inicial del tráfico y el reparto básico de peticiones.

En el despliegue, se optó por Nginx como proxy inverso para unificar el acceso, servir la web y enrutar la API y el panel de administración. Esta separación entre la capa de red y la lógica de negocio mejora la seguridad, la escalabilidad y el mantenimiento del sistema [25], [26].

Se descartó operar un SMTP propio por la complejidad de configuración y los riesgos de reputación de envío. Se eligió un proveedor de correo transaccional con dominio profesional (@destance.com) para mejorar entregabilidad, trazabilidad y fiabilidad. Esta integración automatiza el envío de correos y mantiene la gestión del correo separada de la lógica de la aplicación.

El frontend gestiona la navegación, el backend concentra la lógica, validación y correo, y MongoDB guarda usuarios, cuestionarios, cursos y resultados.

En simple: el frontend es la parte visible, el backend decide y guarda, y la base de datos conserva la información. Nginx recibe las peticiones y las dirige al servicio correcto para que el usuario no tenga que conocer puertos, mejorando el mantenimiento del sistema.

5.8. Modelo de datos y persistencia

El modelo de datos se organiza alrededor de usuarios y actividad educativa. Cada escrito o cuestionario se asocia al usuario, lo que permite recuperar textos y resultados entre sesiones y ver el progreso real. El modelo también incluye entidades para verificación de alta y recuperación de contraseña, con caducidad controlada en servidor.

Las colecciones principales son usuarios, cuestionarios, cursos MOOC y resultados de dashboards. Se añaden entidades de soporte (conceptos, verificaciones

temporales, respuestas de laboratorio).

Esta separación mantiene el contenido didáctico por un lado y las evidencias de aprendizaje por otro, y evita mezclar datos sensibles con contenido público.

En este modelo, ConceptPool se considera una entidad editorial: representa unidades de contenido independientes, desacopladas de la lógica de negocio y gestionadas con su propio ciclo de vida. La Figura 5.2 resume las entidades principales y sus relaciones.

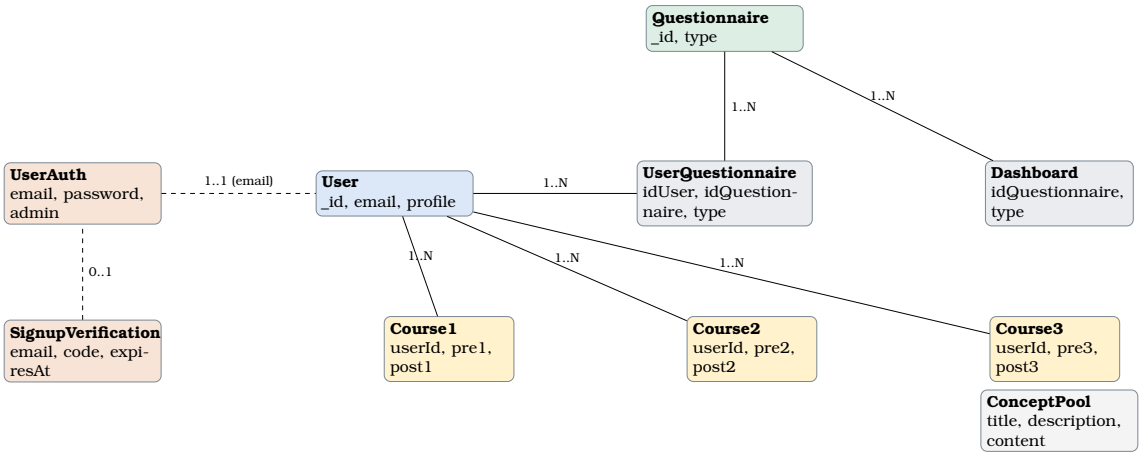

Figura 5.2: Modelo de datos simplificado y relaciones principales.

5.9. Flujo de datos principal

Tras el inicio de sesión, el backend guarda el JWT en una cookie segura (HttpOnly). El frontend no maneja ese token directamente; consulta /api/auth/session para saber si hay sesión activa y el backend valida la cookie. La cabecera Authorization queda como opción secundaria para herramientas o pruebas. Al completar un cuestionario o un escrito, el frontend envía los datos al backend, que valida y guarda en MongoDB. La recuperación de resultados se realiza por usuario, lo que permite rehidratar los textos en los MOOC y poblar los dashboards con información específica.

Este flujo mejora la seguridad del sistema, evita que el navegador tome decisiones críticas y centraliza el control en el servidor. Además, simplifica la privacidad, ya que cada resultado se consulta por identificador de usuario y no por filtros globales. El mismo patrón se aplica en el panel de administración, que consulta y muestra datos ya almacenados sin necesidad de replicarlos en el cliente.

Capítulo 6

Implementación

En este capítulo se describen los cambios implementados en la plataforma.

6.1. Cambios solicitados por las responsables académicas

El menú principal se ajustó para reflejar la nomenclatura completa indicada en los documentos docentes y se renombró el acceso a resultados como My Dashboard. La página de inicio se reorganizó siguiendo el recorrido en cuatro pasos, con textos alineados al diseño y llamadas a la acción coherentes. Además, los recuadros se convirtieron en áreas clicables completas, se unificó el fondo y se afinó la alineación de títulos, subtítulos y descripciones, de modo que la jerarquía visual queda estable y el itinerario educativo se percibe con claridad.

Las páginas informativas (About y FAQ) se actualizaron con los textos y recursos visuales entregados por la tutoría académica. Se reubicaron imágenes, se incorporaron ilustraciones coherentes con cada sección y se justificó el contenido para mejorar la lectura y la presentación editorial. La sección Concepts se mantiene como en el despliegue realizado en la máquina de GCP por Javier Hernández Contreras, sin cambios funcionales [1]. Con ello se elimina la divergencia entre la documentación académica y la plataforma pública y se facilita el mantenimiento de contenidos desde el equipo docente.

También se ajustaron enlaces y estados de herramientas externas. El botón de entrada y el acceso a ACAIA-FEEDBACK se alinearon con el material docente, dejando definido el enlace al vídeo introductorio y el estado Work in Progress mientras no esté disponible la herramienta final. Esto evita ambigüedad en el recorrido y prepara la activación posterior sin rediseños.

6.2. Cambios propuestos por el autor

Se trasladó la autenticación crítica al backend, siguiendo las recomendaciones de OWASP para evitar exponer lógica sensible en el cliente y centralizar el control de sesiones y credenciales, mejorando la seguridad [27], [28]. Antes, la verificación

de alta y la recuperación de contraseña dependían de lógica en el navegador y del envío con EmailJS, lo que exponía configuración sensible y limitaba el control de caducidad. Con el cambio, los códigos se generan y validan en servidor, se almacenan con expiración y se envían mediante Mailjet. El cliente solo solicita el envío y presenta el código.

La validación en frontend se mantiene únicamente como ayuda de experiencia de usuario (mensajes inmediatos, formato o campos vacíos), pero no como control de seguridad. El código del cliente es modificable y no constituye un entorno confiable para autenticación o autorización; por eso todas las comprobaciones relevantes se ejecutan en el servidor, tal y como recomienda OWASP [27].

En el enfoque anterior, la lógica de verificación vivía en el frontend y el envío de correos dependía de EmailJS, lo que implica que el navegador conocía parámetros de servicio y podía ejecutar envíos sin control centralizado. Esto abre riesgos claros: exposición de claves o public keys reutilizables, abuso del endpoint de correo para spam, dificultad para aplicar rate limiting, y posibilidad de manipular el flujo (por ejemplo, forzar validaciones o saltarlas si el cliente se modifica). Además, al no persistir la verificación en servidor, era más difícil auditar intentos, bloquear abusos o invalidar códigos al primer uso. En recuperación de contraseña, el riesgo era similar: un atacante podía automatizar solicitudes o intentar códigos sin un límite robusto de intentos gestionado en backend.

Con la nueva solución, el backend controla todo el ciclo: genera códigos con entropía suficiente, los guarda con caducidad, registra intentos y verifica que el correo está asociado a un usuario real antes de aceptar el alta o el cambio. Mailjet se usa desde servidor con credenciales que no se exponen al cliente, lo que reduce drásticamente el riesgo de uso indebido y mejora la entregabilidad al estar configurado con remitentes verificados. Este cambio también evita depender de CORS o de estado en el navegador y facilita aplicar métricas, alertas y futuras políticas de bloqueo si se detecta abuso.

En seguridad, el paso al backend elimina un punto de fallo crítico: el cliente ya no puede construir o validar el proceso por sí mismo. La verificación deja de ser un paso “decorativo” y se convierte en un requisito real controlado por el servidor. Además, al centralizarlo en backend se alinea con el resto de controles: JWT, sesiones de admin y auditoría de operaciones sensibles. Esto reduce riesgos de suplantación, evita reenvíos no autorizados y facilita la rotación de claves sin afectar al frontend.

A futuro, este cambio habilita mejoras difíciles de implementar en el cliente: limitación de intentos por IP, listas de bloqueo, integración con CAPTCHA, doble factor en acciones sensibles, rotación periódica de claves y análisis de patrones anómalos. También simplifica la migración a nuevos proveedores de correo y permite un control completo sobre plantillas, métricas y trazabilidad del proceso de autenticación.

6.2.1. Flujos de registro y recuperación

Las Figuras 6.1 y 6.2 resumen los flujos de verificación de alta y de recuperación de contraseña gestionados desde backend.

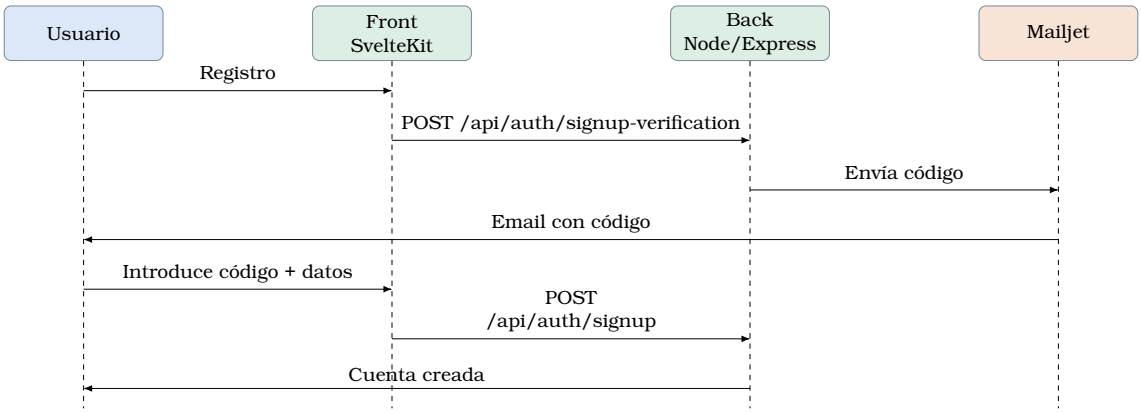

Figura 6.1: Secuencia de registro y verificación de cuenta.

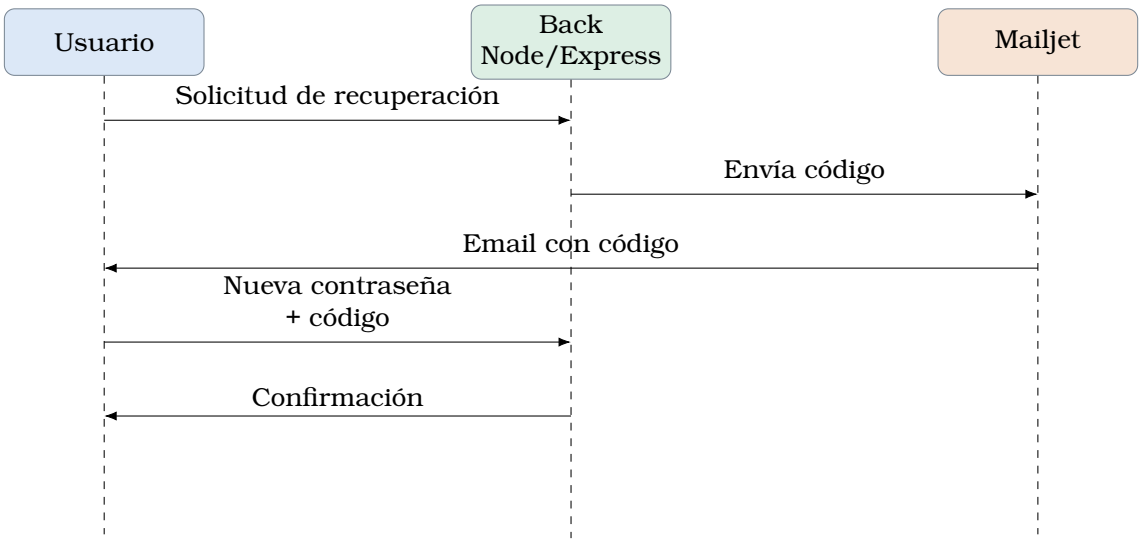

Figura 6.2: Secuencia de recuperación de contraseña.

A continuación se muestran capturas de los correos transaccionales y de las pantallas de verificación y recuperación para documentar el cambio de flujo (Figuras 6.3, 6.4, 6.5, 6.6 y 6.7).

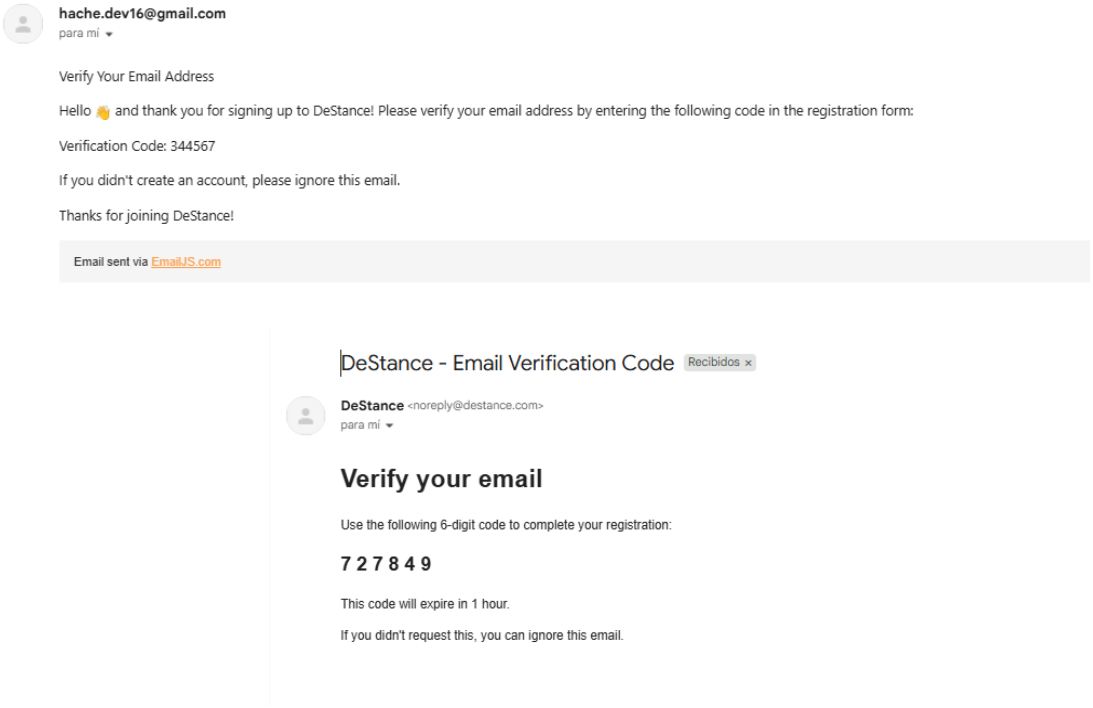

Figura 6.3: Correo de verificación de registro: antes y después.

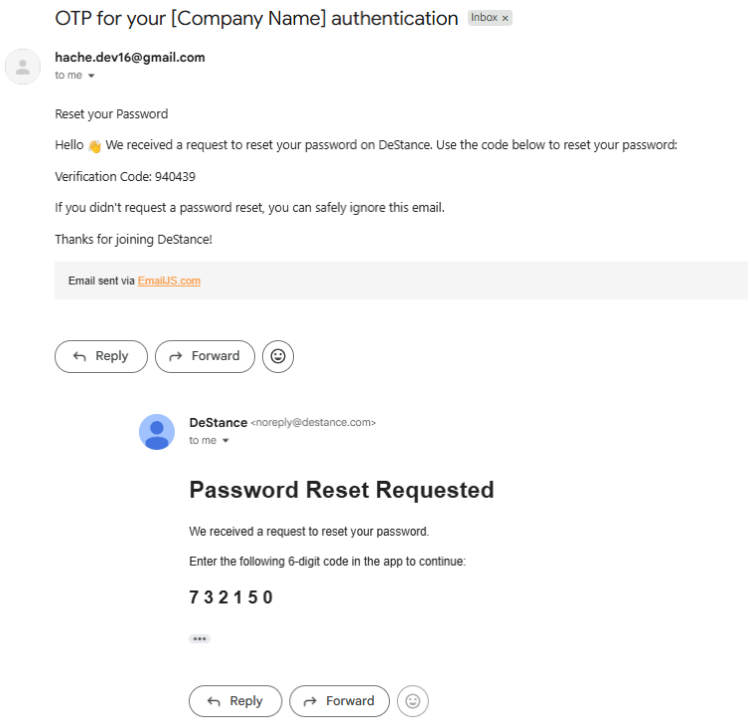

Figura 6.4: Correo de recuperación de contraseña: antes y después.

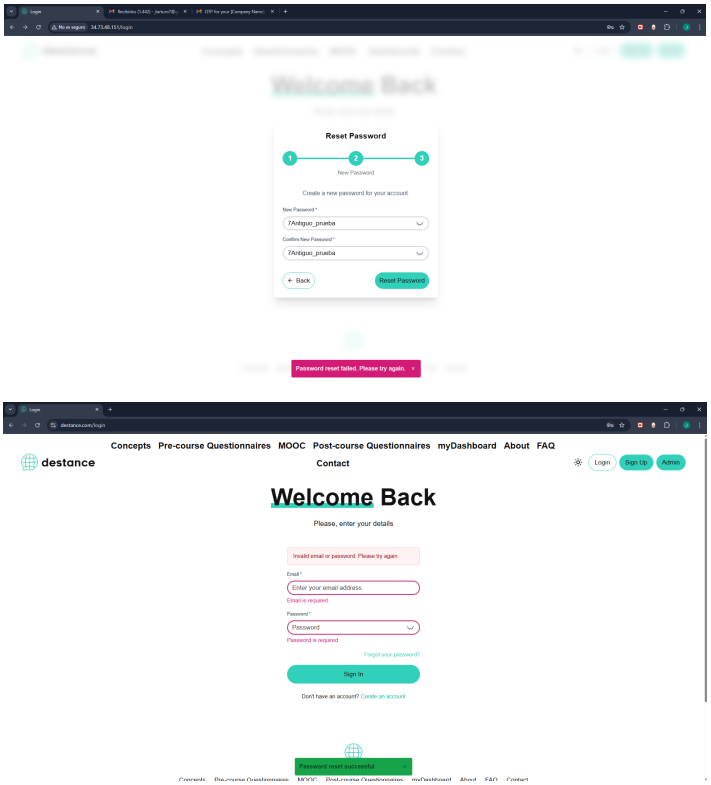

Figura 6.5: Pantalla de cambio de contraseña: antes y después.

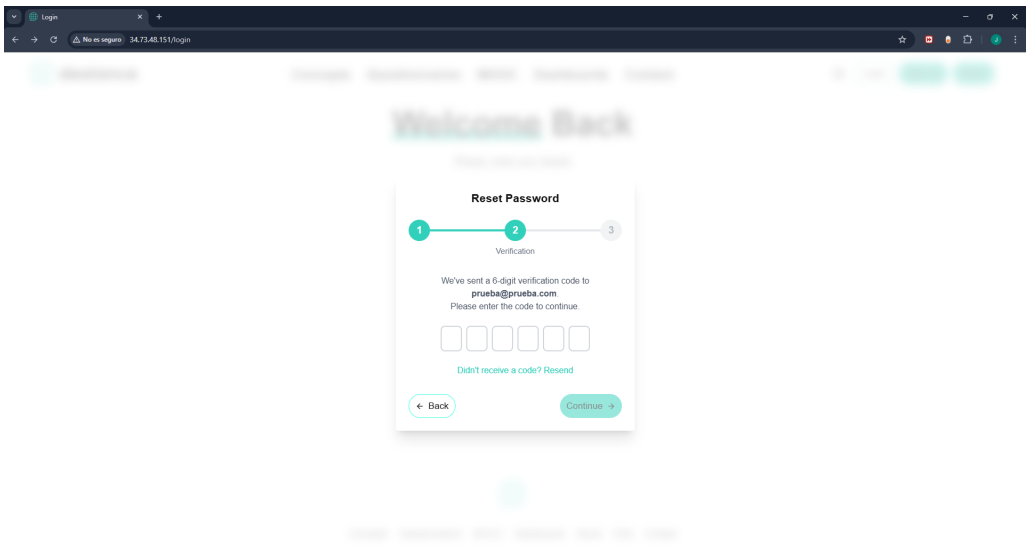

Figura 6.6: Solicitud de recuperación con correo no registrado (versión anterior).

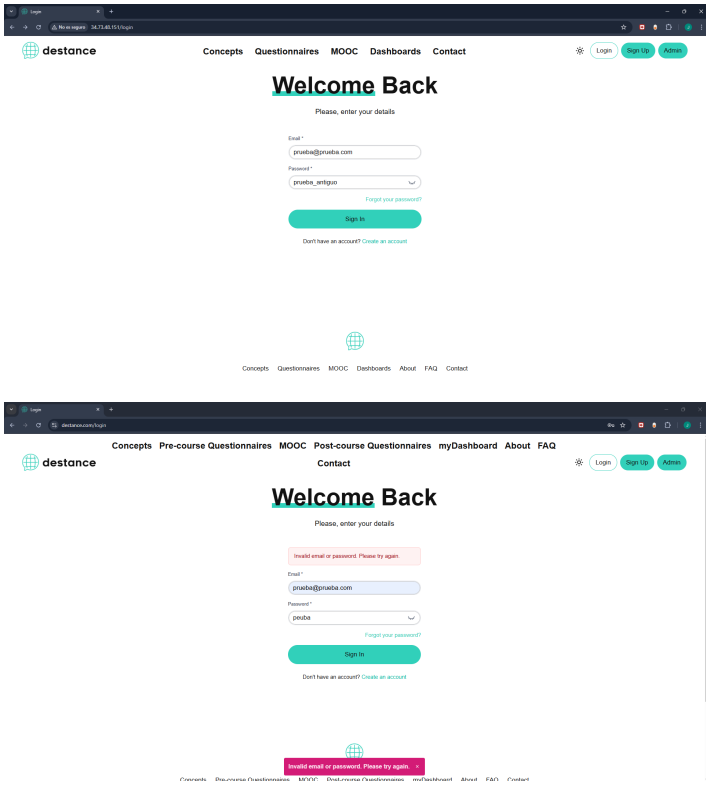

Figura 6.7: Login fallido: antes y después.

En los MOOC se priorizó el vídeo como primer recurso, se permitió la navegación libre entre secciones y se redujo el mínimo de palabras a 10 con un aviso de calidad para respuestas breves. Los textos se guardan al pulsar Continue y se rehidratan al volver a entrar; con ello se evita que el usuario pierda trabajo si abandona una sección, porque ya no se espera al final del MOOC para persistir resultados.

Se unificó la plantilla de cursos y el tipado de contenido de vídeo para admitir proveedores externos, lo que estabilizó Genially en los tres módulos y permitió corregir los enlaces por sesión.

Como línea de mejora, se propone almacenar en base de datos o en caché los resultados de análisis de Hugging Face asociados a escritos ya realizados, para evitar llamadas repetidas cuando el alumno entra y sale de los MOOC.

La parte emocional queda pendiente de completar porque todavía no se dispone de suficientes recursos para entrenar ese módulo, y se prevé terminarla en la fase final del proyecto. En esa mejora, la caché se podría definir por usuario y sección, con un tiempo de vida (TTL) razonable que permita reutilizar resultados recientes sin impedir recalcularlos cuando el texto cambie.

La consistencia visual se reforzó con tarjetas clicables en la home, alineación tipográfica homogénea y ajustes de contenido en páginas informativas. Se conectó la sección de conceptos al panel de administración para mostrar imágenes y textos

reales, y se incorporó un mensaje de error en login para clarificar fallos, lo que reduce barreras y deja la edición de contenidos en manos del equipo académico sin intervención técnica.

En los dashboards se eliminó el filtrado por lengua materna, nacionalidad o género y se dejó la vista únicamente con los resultados del usuario autenticado; el objetivo fue evitar que filtros sobre grupos pequeños pudieran comprometer la privacidad.

El análisis automático de textos se movió a un proxy en backend para evitar fallos de CORS y restricciones de red en Docker. Con este cambio, el frontend consume una ruta propia y el servidor controla errores y disponibilidad del servicio externo, lo que incrementa la estabilidad en despliegue y facilita futuros cambios de proveedor.

El diseño principal mantiene el backend como fuente de verdad del análisis automático, pero se incluyó un fallback en el frontend: si el backend no está disponible, se permite una llamada directa al servicio de Hugging Face para no bloquear al usuario. Esta decisión añade resiliencia y tolerancia a fallos sin desplazar la lógica centralizada.

En infraestructura se separó el build del despliegue, publicando imágenes en Docker Hub y consumiéndolas desde la VPS. Se diferenció un docker-compose local (HTTP 8080) de un docker-compose.deploy.yml para producción, que incorpora TLS, Certbot y los servicios de monitorización, en fase de implementación avanzada pero no terminada. Esta separación evita builds en la máquina académica y reduce el consumo de memoria.

Nginx se configuró como proxy inverso para /, /api y /admin, y en producción también para /grafana y /prometheus con autenticación básica. El dominio se resolvió con un CNAME hacia el FQDN de CeSvImA, requisito institucional ante IPs cambiantes, y el dominio raíz se redirige a https://www.destance.com desde el registrador. El dominio destance.com se adquirió en IONOS por 5 años y se contrató un correo profesional asociado para canalizar dudas, ideas y comunicaciones de los usuarios.

La transición a HTTPS se completó habilitando 80/443 y desplegando Certbot en contenedor con desafío webroot. Los certificados se almacenan en volúmenes persistentes y Nginx aplica redirección 301 de HTTP a HTTPS, garantizando cifrado para sesiones, credenciales y cookies.

Para mantener sesiones seguras tras el proxy se activaron SECURE_COOKIES, TRUST_PROXY y COOKIE_DOMAIN=.destance.com. El primer ajuste fuerza cookies seguras y SameSite=None; el segundo permite que Express confíe en cabeceras X-Forwarded-*; y el tercero evita pérdida de sesión al alternar entre destance.com y www.destance.com.

Se implementó un workflow de GitHub Actions para construir imágenes y desplegar por SSH en CeSvImA. El flujo utiliza secrets y variables del repositorio para evitar exponer credenciales y permite desplegar versiones trazables por commit.

Para habilitar este paso se está coordinando con Javier Hernández Contreras la creación de una organización de De-Stance en GitHub, lo que permitiría gestionar permisos y configurar secretos de forma centralizada. Este flujo puede configurarse para ejecutarse en cada push a una rama de despliegue o de forma manual, según el nivel de control que se requiera en cada fase del proyecto.

6.3. Implementación técnica y operación

En este apartado se describen mejoras que facilitan el mantenimiento y la administración de la plataforma, con una operación más clara y controlada.

6.3.1. Separación de servicios y responsabilidades

La implementación técnica se apoya en una separación clara de responsabilidades entre front, back y nginx. Esta organización permite mantener la interfaz, la lógica de negocio y el proxy inverso de forma independiente, lo que reduce el acoplamiento, mejora el mantenimiento y facilita la administración de la plataforma.

En paralelo, se definió un inventario de rutas de API que cubre autenticación, envío de correo, cuestionarios pre y post, cursos MOOC y análisis de textos, proporcionando un mapa operativo de peticiones y facilitando pruebas y documentación.

6.3.2. Ejecución local

La ejecución local se plantea de forma sencilla: el equipo necesita tener configurado Node.js, una base de datos MongoDB (local o en contenedor) y las variables de entorno mínimas del backend y el frontend. En el backend, deben definirse las claves de Mailjet para el envío de correos y la URL de la base de datos; en el frontend, el prefijo de API y la URL del backend. Con esto, el frontend puede levantarse y consumir la API, y las operaciones de registro, cuestionarios y MOOC quedan disponibles en un entorno de desarrollo controlado.

6.3.3. Ejecución con Docker

En entorno Docker, la ejecución se simplifica a la orquestación de contenedores con docker-compose. Se levantan los servicios de frontend, backend, base de datos y Nginx con una configuración única, y se aseguran volúmenes persistentes para no perder datos. En este esquema, la operación es más reproducible porque la versión de cada servicio queda definida por la imagen, y los cambios se aplican mediante actualización de imágenes y recreación de contenedores.

6.3.4. Despliegue en CeSvImA y proxy inverso

En CeSvImA, el despliegue se apoya en el mismo esquema de contenedores, pero evitando builds en la VPS para no consumir memoria. Esta línea permite

cumplir el objetivo del TFG: un despliegue seguro y automatizado en la nube. Las imágenes se publican en un registro y se descargan en la máquina, y Nginx actúa como puerta de entrada con rutas /, /api y /admin. La configuración requiere tener preparados los ficheros .env.production, el default.conf de Nginx y las credenciales de Mailjet, de modo que el sistema pueda enviar correos y guardar datos en MongoDB. Esta documentación queda como referencia para cualquier miembro del equipo que necesite reproducir el despliegue.

La elección de este enfoque se justifica en el apartado de arquitectura técnica, donde se analizan las ventajas de usar un proxy dedicado frente a uno integrado en el backend. La Figura 6.8 resume el enrutado principal del proxy en despliegue.

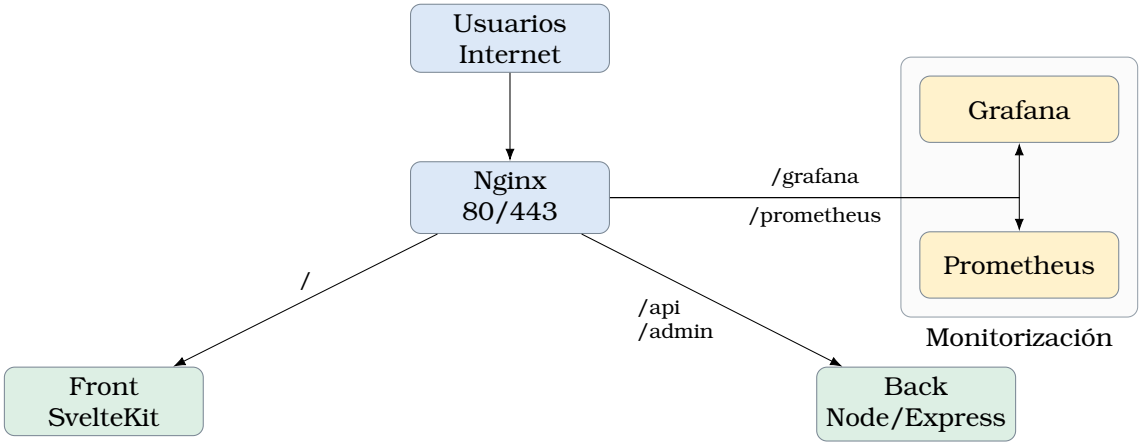

Figura 6.8: Enrutado principal del proxy Nginx en despliegue.

6.3.5. Inventario de endpoints y seguridad de acceso

Para lectores no técnicos, un endpoint es una dirección concreta a la que la aplicación envía datos o solicita información, como si fuera una ventanilla especializada. Al reunirlos y describirlos, se facilita entender qué partes del sistema intercambian información y qué tipo de protección requiere cada una.

La reorganización de endpoints simplificó la experiencia de usuario y la seguridad del sistema.

Antes, parte del flujo se resolvía en el cliente con llamadas directas a servicios externos (EmailJS y Hugging Face), lo que añadía capas de configuración en el navegador y duplicaba puntos de fallo. Ahora, esas rutas pasan por el backend, que valida, registra y responde con criterios uniformes.

Se eliminan dependencias directas del cliente y se centraliza el control de errores y credenciales. El inconveniente es que el servidor asume más carga y debe estar correctamente monitorizado; en el despliegue actual la monitorización está en fase de implementación avanzada pero no terminada. La ventaja es una arquitectura más limpia, predecible y fácil de mantener a futuro.

En la práctica, el usuario accede a un único punto de entrada y todas las llamadas

pasan por el prefijo /api detrás del proxy, mientras que antes coexistían rutas internas y conexiones directas a terceros desde el navegador. Esta simplificación elimina capas intermedias en el cliente, reduce los problemas de CORS y permite aplicar políticas de seguridad y registro en un solo lugar.

La contrapartida es que el backend concentra más responsabilidad y requiere una gestión cuidadosa de tiempos de respuesta y de los recursos de la VPS, pero a cambio se obtiene una base más sólida para añadir autenticación avanzada, control de cuota y auditoría de operaciones.

El panel de administración se expone bajo la ruta /admin y se publica a través de Nginx junto al resto de la aplicación. En despliegue fue necesario ajustar el origen del enlace en el frontend para que apunte al mismo host y puerto que utiliza el navegador, evitando discrepancias entre entornos locales y el proxy inverso. Con este ajuste, el acceso al panel queda integrado en el mismo dominio y evita errores de conexión cuando el backend se ejecuta detrás de Nginx.

Solo endpoints usados en producción. Existen rutas de test/legacy en el repositorio que no se exponen en despliegue. El inventario se recoge en el Anexo A.2 (Tabla A.1).

Para interpretar la tabla: Público indica que no hay middleware de autenticación en la ruta; Sesión web (JWT) se valida con cookie HttpOnly (y Authorization: Bearer en casos puntuales); Admin (JWT) exige token con rol admin; Sesión AdminJS usa cookie de AdminJS; y Auth básica corresponde a credenciales de Nginx.

6.3.6. Rutas del frontend y control de acceso

Las rutas del frontend se gobiernan en el layout principal, con redirección a /login cuando no hay sesión para /questionnaires, /mooc, /dashboards y /user-details. Además, /dashboards requiere rol admin.

El detalle completo se recoge en el Anexo A.3 (Tabla A.2).

6.3.7. Persistencia e integraciones externas

La persistencia por usuario se consolidó al unificar la relación entre token de sesión y usuario real, lo que evita pérdidas de escritos y permite rehidratar contenido al volver a entrar. Esta decisión tiene impacto directo en dashboards y en la continuidad del aprendizaje.

En cuanto a integraciones externas, se adoptó un enfoque de control desde servidor para reducir dependencia del navegador, manteniendo Genially a través de iframe y encapsulando el análisis automático en un proxy backend.

6.3.8. Gestión del correo transaccional y entregabilidad

El correo se trató como un componente crítico de seguridad, al intervenir en registro, verificación y recuperación. En la implementación, el backend genera los mensajes y delega el envío en el proveedor transaccional, manteniendo separada la lógica de negocio de la infraestructura de correo y facilitando controles de entregabilidad.

En esta fase se comparó el envío desde cliente con EmailJS frente a un servicio externo como Mailjet.

EmailJS simplifica la integración inicial al permitir envíos desde el frontend, pero expone credenciales y no permite aplicar de forma robusta políticas de autenticación de dominio ni controles de seguridad en servidor.

Mailjet se integra desde backend y facilita la aplicación de SPF, DKIM y DMARC, requisitos clave para mejorar la legitimidad del remitente y evitar clasificación como spam [29], [30]. Por ello, se tomó la decisión de utilizar Mailjet, apoyada también en la recomendación de Javier Hernández Contreras (TFG previo en De-Stance), priorizando seguridad, entregabilidad y escalabilidad sobre la simplicidad inicial [1].

Para proteger el dominio frente a suplantaciones se configuraron SPF, DKIM y DMARC, necesarios para que los proveedores validen la legitimidad del remitente y eviten clasificar los mensajes como spam [29], [30]. La política DMARC se aplicó de forma progresiva (modo de observación p=none y, después, p=quarantine) para reforzar la confianza del dominio sin afectar a usuarios reales durante el ajuste inicial; la monitorización de la plataforma está en fase de implementación avanzada pero no terminada.

Durante pruebas se observó un patrón relevante: los correos de verificación de cuenta tendían a ir a spam, mientras que los de recuperación de contraseña llegaban a bandeja principal. El análisis indicó tres causas principales: mayor riesgo percibido en correos de alta, dominio con reputación reciente y contenido con enlaces largos y tokens de verificación.

Para mitigar el problema se ajustó el contenido de los correos: se priorizó el uso de códigos numéricos frente a enlaces largos, se simplificó el lenguaje y se mantuvo una estructura coherente con el correo de recuperación.

Como mejora futura se plantea habilitar BIMI para asociar el logotipo de la organización a los correos salientes, publicando el logo y el registro TXT correspondiente en DNS [31]. Esta medida no está todavía implementada y debe validarse antes de considerarse parte del flujo de entregabilidad.

6.3.9. Despliegue operativo y automatización

El despliegue operativo se estructuró en fases: construcción de imágenes fuera de la VPS para ahorrar RAM, etiquetado por commit (back-<sha>, front-<sha>) y despliegue mediante docker compose con IMAGE_TAG. La VPS solo realiza pull

de imágenes y recrea contenedores, manteniendo volúmenes persistentes para no perder datos.

Se consiguió un despliegue seguro con la apertura de puertos 80/443 se habilitó HTTPS con Certbot y Nginx, incluyendo redirección 301 y renovación automática. La emisión inicial se realizó de forma manual y la renovación quedó automatizada con Certbot. Esta configuración mejora la seguridad frente al servicio anterior en HTTP 8080 y permite activar cookies seguras y sesiones con SameSite=None.

TLS (Transport Layer Security) es el protocolo que cifra las comunicaciones entre el navegador y el servidor. Su función principal es triple: confidencialidad (nadie puede leer el tráfico), integridad (los datos no pueden alterarse sin ser detectados) y autenticidad (el usuario verifica que el servidor es el dominio correcto). En la práctica, el navegador negocia una clave de sesión mediante un handshake criptográfico; a partir de ese momento, todas las peticiones y respuestas viajan cifradas. Esto es crítico cuando se transmiten credenciales, cookies de sesión o datos personales. Además, HTTPS ayuda a evitar que intrusos manipulen la comunicación entre el sitio y el navegador, y es un componente clave para habilitar permisos en funciones modernas del navegador [32].

Certbot es un cliente oficial de Let's Encrypt que automatiza la obtención y renovación de certificados TLS. Funciona mediante el protocolo ACME, que valida que el servidor controla el dominio. En este despliegue se usa el desafío webroot: Certbot crea un archivo temporal en /.well-known/acme-challenge/ y la autoridad certificadora lo solicita por HTTP; si la respuesta es correcta, se emite el certificado. Una vez emitido, Nginx carga el fullchain.pem y el privkey.pem desde los volúmenes persistentes y aplica HTTPS en el puerto 443, redirigiendo todo el tráfico HTTP a HTTPS con un 301 permanente.

Esto aporta ventajas directas: el navegador muestra un sitio seguro, las cookies pueden marcarse como secure y se evita exposición de sesiones ante ataques de intermediario. Además, la renovación automática evita caducidades silenciosas y reduce la probabilidad de interrupciones. El coste operativo es bajo porque Certbot se ejecuta en contenedor y los certificados se gestionan sin intervención manual.

La automatización se materializó con GitHub Actions: un workflow reutilizable (ci.yml) ejecuta pruebas y construcción, y deploy-cesvima.yml publica imágenes y despliega por SSH en CeSvImA. Se usan secrets para Docker Hub y SSH y una variable no sensible para el host, evitando credenciales en el repositorio y proporcionando trazabilidad y rollback por commit. El flujo completo se muestra en la Figura 6.9.

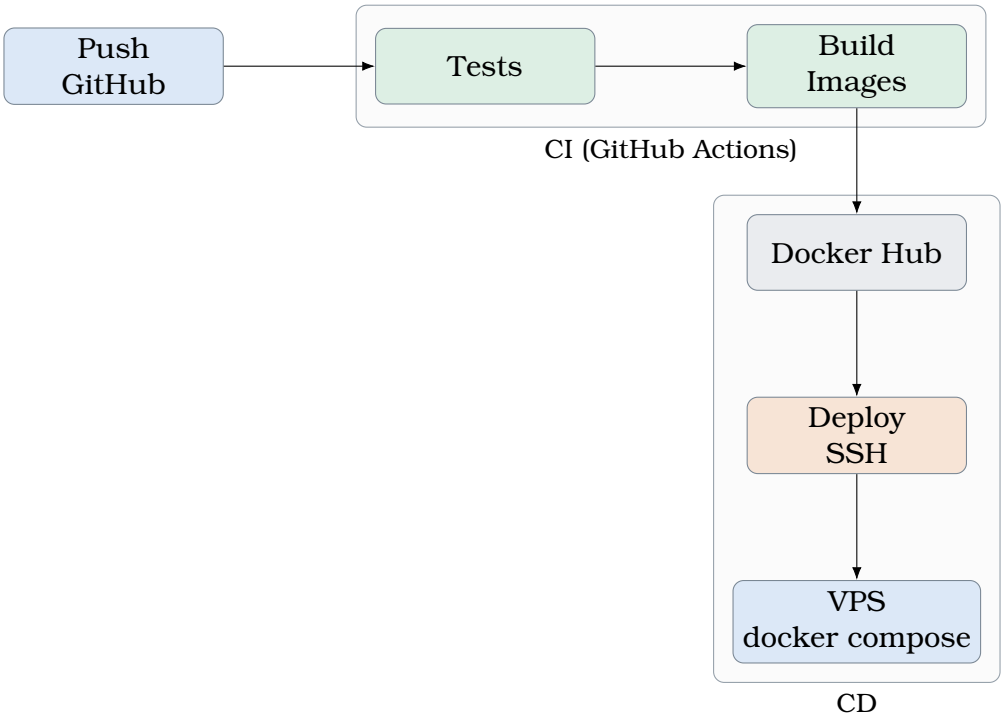

Figura 6.9: Flujo de despliegue CI/CD desde el push hasta el despliegue en CeSvImA.

6.3.10. Monitorización y observabilidad (fase de implementación avanzada pero no terminada)

Se incorporó una monitorización en fase de implementación avanzada pero no terminada mediante Prometheus y visualización en Grafana [33], herramientas ampliamente adoptadas para observabilidad en entornos cloud-native (en el despliegue actual, en fase de implementación avanzada pero no terminada) [34], [35], con métricas del host y de contenedores a través de node_exporter y cAdvisor para métricas del host y de contenedores [36], [37]. node_exporter expone CPU, memoria, disco y red del servidor; cAdvisor expone consumo por contenedor, reinicios y uso de recursos por servicio (backend, frontend, base de datos, Nginx). Prometheus realiza el scrape periódico de estas métricas y las conserva en su base de datos de series temporales.

Grafana se conecta a Prometheus como fuente de datos y permite construir paneles con indicadores clave: carga media, RAM disponible, almacenamiento, latencia en endpoints, estado de salud de contenedores y tendencias por hora/día. Esto aporta trazabilidad operativa, facilita el diagnóstico de cuellos de botella y permite anticipar problemas de capacidad (por ejemplo, saturación de RAM al ejecutar tareas de análisis o picos de usuarios).

Para evitar abrir nuevos puertos en CeSvImA, los paneles se publican bajo el mismo dominio mediante Nginx en /grafana y /prometheus, protegidos con autenticación básica.

Esta decisión mantiene el acceso unificado y reduce superficie de exposición, a costa de añadir una capa de administración adicional que debe mantenerse con credenciales seguras. También limita el acceso directo a Prometheus a la red interna del despliegue, mejorando el aislamiento.

Las ventajas principales son: detección temprana de fallos, visibilidad del consumo real, soporte a la planificación de recursos y base para alertas. Los inconvenientes son el consumo adicional de CPU/RAM, la necesidad de mantener volúmenes de datos y la complejidad operativa añadida (credenciales, actualización de imágenes y políticas de retención).

En esta fase no ha sido posible validar de forma completa la monitorización, en fase de implementación avanzada pero no terminada. El despliegue se realizó a última hora, con varias incidencias de implementación, y además fue necesario coordinar la apertura de los puertos 80/443 en CeSvImA, lo que retrasó la comprobación de Grafana y Prometheus. Se intentará dejar una implementación mínima antes de la entrega, pero aun así será necesario un estudio específico para validar la configuración, ya que no se podrán realizar pruebas completas de la monitorización en fase de implementación avanzada pero no terminada. Queda pendiente verificar la estabilidad del scrape, el acceso con autenticación básica y el correcto funcionamiento de los paneles.

Se eligió esta solución por su bajo coste, su integración sencilla con Docker y la compatibilidad con restricciones de CeSvImA (no abrir puertos extra). Más adelante, esta arquitectura permite añadir alertas, métricas de aplicación (tiempos de respuesta, errores 4xx/5xx) y una gestión de logs centralizada (por ejemplo, Loki), además de paneles específicos para el rendimiento de los módulos de análisis.

6.3.11. Gestión de incidentes y operación

De forma sencilla, la gestión de incidentes busca detectar cuando la plataforma deja de responder, entender si el problema es general o de un servicio concreto y restaurar el acceso lo antes posible. Para ello se recomienda un procedimiento simple: comprobar si la web carga, revisar los paneles de monitorización, en fase de implementación avanzada pero no terminada, y, si es necesario, reiniciar los servicios y comunicar el estado a las personas responsables.

Desde el punto de vista técnico, los pasos básicos ante una caída son: verificar /health y /healthz, comprobar el estado de los contenedores (docker ps), revisar logs de Nginx, backend y MongoDB, medir consumo de disco y memoria, y reiniciar el servicio afectado. Si el fallo persiste, se puede revertir a la última imagen estable y validar que los endpoints principales vuelven a responder.

Actualmente no hay alertas automáticas configuradas; como trabajo futuro se propone añadir alertas en Grafana o Prometheus y un uptime check externo para detectar caídas fuera del equipo.

6.3.12. Copias de seguridad y recuperación ante fallos

Actualmente no hay un sistema de copias de seguridad automatizado en la plataforma ni un plan operativo formal de recuperación. Por ello, queda como línea futura prioritaria: los backups son el mecanismo que permite recuperar servicio tras fallos, errores humanos o corrupción de datos, y reducen el tiempo de parada y la pérdida de información.

El activo crítico es la base de datos MongoDB (volumen mongo_data), donde se almacenan usuarios, respuestas a cuestionarios, escritos de MOOC y resultados. También conviene respaldar los ficheros de configuración (back/.env.production, front/.env.production, nginx/default.conf) y, si se desea conservar la monitorización, en fase de implementación avanzada pero no terminada, los volúmenes de Grafana y Prometheus.

Se propone un esquema sencillo: copia diaria de MongoDB con mongodump (archivo comprimido), copia semanal de configuración, y copia adicional antes de cada despliegue. Las copias deben almacenarse fuera de la VPS (almacenamiento externo o repositorio privado de backups) para evitar pérdida total.

La restauración consiste en recuperar los ficheros de configuración, levantar los contenedores y aplicar mongorestore sobre el último backup válido. En caso de fallo completo de la máquina, el entorno puede recrearse desde Docker y rehidratar los datos a partir de las copias.

6.3.13. Privacidad, GDPR y retención de datos

La plataforma almacena datos personales y educativos.

En el perfil del usuario se guardan correo, nombre, fecha de nacimiento (día/mes/año), género, nacionalidades, lengua materna, idioma y nivel, además de un avatar opcional. Las credenciales se almacenan como hash (no en claro) junto con el rol de administrador; los códigos de verificación y los tokens de recuperación son temporales. Los datos de actividad incluyen respuestas a cuestionarios, escritos de MOOC (pre y post) y resultados de análisis, así como registros de interacción en el módulo de noticias falsas. En el modelo actual no se almacenan IP ni user agent.

El acceso a funcionalidades sensibles se controla con JWT (perfil, cursos, dashboards personales) y el rol administrador habilita vistas agregadas y el panel AdminJS. El contenido editorial es público y existen rutas históricas de user-questionnaires sin autenticación, por lo que en un despliegue con datos reales se recomienda endurecerlas y limitar su acceso.

La base de datos se despliega como servicio interno en Docker y debe quedar protegida por red o firewall para evitar exposición pública. En particular, el puerto 27017 de MongoDB no se expone al exterior, ya que el acceso se limita al contenedor del backend dentro de la red interna de Docker, garantizando el aislamiento de la capa de datos frente a conexiones no autorizadas.

La retención de datos temporales está limitada por diseño: los códigos de verificación expiran a la hora y los tokens de recuperación caducan en aproximadamente una hora. No hay purgas automáticas para el resto de información, por lo que se conserva mientras exista la cuenta o hasta que se ejecute una limpieza manual; esto debe formalizarse con una política de retención.

Para alinearse con GDPR se recomienda mantener una política de privacidad visible, registrar la finalidad educativa de uso y aplicar principios de minimización, acceso restringido y supresión bajo solicitud.

6.3.14. CI/CD y gobernanza de secretos

La automatización de CI/CD mejora el mantenimiento de la plataforma al estandarizar builds, despliegues y verificación en cada cambio. Se preparó con GitHub Actions para ejecutar pruebas, construir imágenes con etiquetas fijas y desplegar por SSH en CeSvImA. El flujo está diseñado para bloquear el despliegue si los tests fallan y para trabajar con secrets en GitHub; la falta de permisos de organización obliga a coordinar la creación de una organización de De-Stance y la habilitación de secretos antes de activar el proceso completo. La Figura 6.10 resume el pipeline desde el push hasta el despliegue.

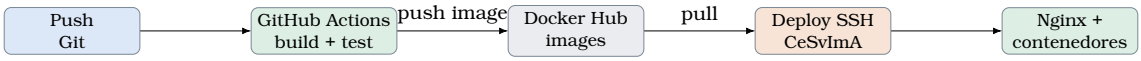

Figura 6.10: Pipeline CI/CD desde el push hasta el despliegue en CeSvImA.

Esta configuración aporta trazabilidad y consistencia, aunque requiere gobernanza de accesos y revisión de credenciales en producción.

Frente al despliegue manual anterior, la ventaja es la repetibilidad y la reducción de errores humanos, mientras que el inconveniente es la dependencia de una configuración inicial de permisos y de una disciplina de versionado más estricta.

6.3.15. Secrets en GitHub Actions

Para que el despliegue automatizado funcione sin exponer credenciales, los secretos se guardan en GitHub (Settings $\rightarrow$ Secrets) y se inyectan en el workflow como variables de entorno. GitHub cifra los secretos en reposo y en tránsito, los oculta en los logs y limita su acceso a los jobs autorizados; además, no se exponen a workflows de forks por defecto y pueden protegerse con environments y reglas de aprobación [38], [39]. Esto permite automatizar el despliegue sin escribir claves en el repositorio ni en el histórico de commits.

Si un secreto se filtra, el impacto puede ser crítico: acceso no autorizado a la VPS, envío de correo fraudulento, lectura de base de datos o falsificación de tokens de sesión. Por eso se almacenan en GitHub, se rotan periódicamente y nunca se imprimen en consola dentro del workflow.

El listado completo se incluye en el Anexo A.4 (Tabla A.3).

Las variables no sensibles (por ejemplo, CORS_ORIGIN, NODE_ENV, VITE_API _SLUG o VITE_BACK_URL) pueden almacenarse como Variables de GitHub o mantenerse en los .env del servidor. La clave es mantener los secretos fuera del repositorio y limitar su exposición a los jobs estrictamente necesarios.

6.3.16. Guía operativa y variables de entorno

Para mejorar el mantenimiento y la administración se ha documentado el despliegue y la operación para futuros integrantes del equipo.

El despliegue se basa en imágenes publicadas en un registro (Docker Hub), un proxy inverso con Nginx y un conjunto de variables de entorno por servicio. La puesta en marcha requiere preparar los ficheros .env.production en front y back, revisar el default.conf de Nginx y ejecutar el docker-compose de despliegue desde el directorio de la VPS.

Las actualizaciones se realizan con pull de nuevas imágenes y recreación de contenedores, manteniendo volúmenes para persistencia de datos, y los diagnósticos se apoyan en los logs de cada servicio.

Como norma operativa, no se deben subir ficheros .env al repositorio, se deben usar etiquetas fijas en las imágenes y evitar builds en la VPS para no agotar memoria.

Esta guía permite que el equipo pueda reproducir el despliegue, entender la arquitectura y aplicar cambios de forma controlada sin exponer credenciales.

Para que el sistema funcione correctamente en local y en despliegue es necesario definir un conjunto mínimo de variables de entorno. En el backend, además de la base de datos y el origen permitido para CORS, se requieren las claves de Mailjet y las direcciones de correo asociadas al dominio destance.com. En el frontend, la API se consume por un prefijo común y la URL interna del backend se ajusta según el entorno.

Las variables se resumen en la Tabla 6.1 y la Tabla 6.2.

Tabla 6.1: Variables de entorno (frontend).

Variable (front)	Propósito	Ejemplo local	Ejemplo despliegue
VITE_API_SLUG	Prefijo de rutas API desde el frontend.	/api	/api
VITE_BACK_URL	URL del backend para peticiones internas.	http://localhost:3000	http://back:3000
NODE_ENV	Modo de ejecución del frontend.	development	production

Tabla 6.2: Variables de entorno (backend).

Variable (back)	Propósito	Ejemplo local	Ejemplo despliegue
NODE_ENV	Modo de ejecución del servidor.	development	production
PORT	Puerto del backend.	3000	3000
MONGO_DB_URL	Conexión a base de datos.	mongodb://app_user:your_secure_password@localhost:27017/mydatabase	mongodb://mongo:27017/juanlo
CORS_ORIGIN	Origen permitido en API.	http://localhost:5173 (local) http://localhost:8080 (Docker local)	https://www.destance.com
SECURE_COOKIES	Cookies seguras y SameSite=None en HTTPS.	false	true
TRUST_PROXY	Confía en cabeceras del proxy inverso.	false	true
COOKIE_DOMAIN	Dominio común de cookies.	(no definido)	.destance.com
JWT_KEY	Firma de tokens de sesión.	<secreto>	<secreto>
ADMIN_COOKIE_SECRET	Firma de sesión del panel admin.	<secreto>	<secreto>
MAILJET_API_KEY	Clave API de Mailjet.	<secreto>	<secreto>
MAILJET_API_SECRET	Secreto API de Mailjet.	<secreto>	<secreto>
MAILJET_FROM_EMAIL	Remitente verificado.	noreply@destance.com	noreply@destance.com
MAILJET_CONTACT_TO	Destino del formulario de contacto.	support@destance.com	support@destance.com

6.3.17. Diagnóstico de incidencias comunes

En la operación diaria se han identificado errores típicos y su solución básica.

Cuando el frontend no puede llamar al backend suele ser un problema de CORS_ORIGIN o de rutas del proxy; se corrige alineando el origen con el dominio real y verificando las rutas en Nginx.

Si el envío de correo falla, normalmente falta una clave de Mailjet o el remitente no está validado en el dominio; se revisan las variables de entorno y la configuración del proveedor.

Si no se guardan datos, el origen suele estar en la conexión a MongoDB o en volúmenes que no están montados; se valida la cadena MONGO_DB_URL y se comprueba la persistencia.

Cuando AdminJS pierde sesión al cambiar entre destance.com y www.destance.com, la causa habitual es un COOKIE_DOMAIN mal definido o el acceso por HTTP en lugar de HTTPS.

Este conjunto de comprobaciones reduce tiempos de diagnóstico y permite que nuevos integrantes localicen fallos sin depender de conocimiento previo.

Capítulo 7

Pruebas y validación

Las validaciones se realizaron de forma manual: navegación entre secciones, carga de vídeos, funcionamiento de los cuestionarios y verificación de textos. Se comprobó que el aviso de palabras aparece en el rango esperado y que el contenido Genially se muestra correctamente.

No se han realizado pruebas unitarias ni pruebas de sistema completas por falta de tiempo y por la salida de compañeros del proyecto. Aunque se reconoce el riesgo de desplegar sin esa batería de pruebas, la plataforma se publicó siguiendo la narrativa de este TFG y las directrices de las responsables académicas. La automatización de pruebas queda preparada en el pipeline de CI, pero su ejecución continua depende de la configuración de secretos y permisos en GitHub.

7.1. Pruebas funcionales

Se verificó el flujo completo de autenticación (registro, verificación, login y recuperación) y el funcionamiento del formulario de contacto. Se comprobó la navegación en MOOC, el orden de secciones y la visualización del vídeo y los recursos descargables. Hasta el cierre del proyecto no ha sido posible realizar pruebas exhaustivas ni con usuarios reales ni con métricas cuantitativas de usabilidad; la validación se limitó a comprobaciones manuales en los flujos principales.

7.2. Pruebas de persistencia

Se validó que los escritos se guardan al pulsar Continue y que se recuperan tras cerrar sesión o recargar. Se revisaron resultados en paneles y su correspondencia con los registros almacenados.

7.3. Pruebas de despliegue

Se probó el despliegue local con Docker y el despliegue en CeSvImA a través de Nginx con HTTPS, confirmando el enrutado de /api y /admin y la redirección HTTP→HTTPS. Se verificó la apertura de firewalls y puertos 80/443, la obtención de certificados con Certbot y la respuesta correcta del dominio. Se documentaron incidencias habituales: propagación de DNS en CNAME, validación de certificado fallida cuando el puerto 80 estaba cerrado y problemas de permisos en volúmenes de monitorización, en fase de implementación avanzada pero no terminada. La Tabla 7.1 resume incidencias reales y su resolución durante el despliegue.

Tabla 7.1: Incidencias reales y soluciones aplicadas en despliegue.

Incidencia	Causa identificada	Solución aplicada
DNS/CNAME no resolvía a la VPS	El dominio debía apuntar al FQDN de CeSvImA mediante CNAME; el dominio raíz no admitía A y la propagación era lenta.	Configurar CNAME para www al FQDN institucional y redirigir el dominio raíz desde el registrador; verificar con dig.
Puertos 80/443 cerrados	Firewall institucional bloqueaba tráfico entrante, impidiendo servir HTTP/HTTPS y validar certificados.	Solicitar apertura de 80/443 en CeSvImA y comprobar escucha en Nginx antes de emitir el certificado.
Certificado TLS no encontrado	Certbot no pudo completar el desafío webroot o los volúmenes no estaban montados en Nginx.	Ejecutar Certbot con webroot, asegurar rutas /var/www/certbot y reiniciar Nginx con los volúmenes correctos.
Cookies no persistían en HTTPS	Faltaban SECURE_COOKIES, TRUST_PROXY y COOKIE_DOMAIN.	Activar cookies seguras, confiar en el proxy y fijar COOKIE_DOMAIN=.destance.com.

7.4. Plan de pruebas automatizadas y métricas de calidad (trabajo futuro)

En palabras simples, el objetivo es asegurar que cada cambio no rompe funcionalidades ya probadas y que la experiencia del alumnado se mantiene estable. Las pruebas automatizadas permiten detectar errores antes de publicar y aportan confianza al equipo, reduciendo tiempos de revisión manual.

Desde el punto de vista técnico se propone integrar un plan incremental en CI: pruebas unitarias en backend (controladores, validaciones y utilidades), pruebas de integración de la API con base de datos de test, y pruebas E2E en frontend (flujos de registro, cuestionarios y MOOC). Como métricas de calidad se plantean criterios mínimos: cobertura de código en backend con un umbral definido (p.ej., 70 %), lint sin errores críticos, compilación sin fallos y tiempos de respuesta estables en endpoints clave. Estos criterios no están activos todavía, pero se usarán como puerta de salida para despliegues automáticos cuando el pipeline esté habilitado.

Capítulo 8

Resultados

La plataforma ofrece ahora un recorrido más claro, con contenidos alineados a la tutoría académica y una navegación sin bloqueos en los MOOC. Los cambios mejoran la coherencia visual, reducen barreras y facilitan la comprensión de la ruta educativa.

Se ha reforzado la persistencia de escritos y resultados por usuario, el formulario de contacto opera con correo institucional y el análisis automático funciona de forma estable en Docker. Además, la configuración de despliegue con proxy inverso garantiza un acceso unificado a frontend, backend y admin.

Frente al servicio anterior (HTTP 8080 y despliegues manuales), el entorno actual añade HTTPS con certificados renovables, cookies seguras, redirección a un único dominio y monitorización integrada, en fase de implementación avanzada pero no terminada, bajo rutas protegidas. Esto reduce exposición de credenciales, mejora la fiabilidad operativa y deja una base preparada para automatizar despliegues y hacer rollback por versión.

El valor añadido del trabajo se concreta en mejoras de usabilidad y seguridad que no estaban resueltas en la versión previa. En usabilidad, se ordenó el recorrido para priorizar el vídeo, se redujeron bloqueos por mínimos de palabras y se aseguró la recuperación de textos al volver a entrar. En seguridad y operación, se trasladaron validaciones sensibles al backend, se unificó el envío de correos con un proveedor transaccional y se estabilizó la integración de análisis mediante un proxy controlado desde servidor. Esto reduce exposición de claves, evita flujos manipulables desde el cliente y aporta trazabilidad sobre el uso del sistema.

Además, se incorporó una evaluación de proveedores cloud y de alternativas serverless como referencia para un uso más extendido de la plataforma, comparando coste, complejidad y adecuación al contexto académico. Esta revisión deja definidos criterios de decisión y posibles líneas de evolución sin comprometer el despliegue actual en CeSvImA.

Junto a ello, se consolidaron decisiones arquitecturales clave, como el uso de proxy inverso dedicado, la separación de capas y el correo transaccional en

backend, y se documentó una guía operativa con un pipeline de CI/CD en GitHub Actions y gobernanza de secretos. Estas piezas mejoran el mantenimiento y la operación y reducen el riesgo de errores en despliegues y cambios.

8.1. Contribuciones y enfoque

Este trabajo parte de la base construida en los TFGs previos sobre De-Stance (módulos, administración y analítica) [1], [5]-[7], pero no busca rehacer esas piezas. El foco se centra en consolidar la experiencia de usuario y cerrar la línea de despliegue seguro: navegación y contenidos alineados con las guías docentes, flujos de autenticación/correo en backend y operación con proxy inverso y HTTPS.

Desde la perspectiva del usuario final, el cambio más visible es que el vídeo aparece como primer recurso de cada MOOC y que los escritos no se pierden al salir. El alumno puede avanzar sin bloqueos excesivos y encuentra siempre sus textos y resultados al volver a entrar. La navegación es más clara, las secciones están ordenadas y los contenidos informativos se corresponden con lo esperado por el equipo docente.

Desde la perspectiva de administración, los resultados se almacenan de manera consistente y se muestran por usuario, lo que facilita el seguimiento del progreso sin exponer datos de otros participantes. El panel de administración se accede desde la misma URL gracias al proxy inverso, y los conceptos, cuestionarios y MOOC se alimentan desde la base de datos con un flujo más estable. Esto reduce la necesidad de ajustes manuales y permite que el equipo académico gestione contenidos sin depender de cambios técnicos.

Antes de los cambios, el usuario encontraba un flujo disperso: el vídeo no era el primer recurso, los textos podían perderse al salir y el acceso a resultados no estaba claramente vinculado a su cuenta. Tras la implementación, el itinerario es lineal y comprensible (vídeo, pre-writing, post-writing), los escritos se recuperan al reentrar y el panel refleja el progreso de cada usuario.

En la capa técnica, se pasó de un envío de correo dependiente del cliente y con límites bajos a un servicio transaccional en backend con mayor trazabilidad y capacidad; y de integraciones externas frágiles en navegador a un proxy controlado desde servidor para estabilizar el análisis automático.

En autenticación, antes la verificación de alta y el cambio de contraseña dependían de lógica en el frontend, lo que limitaba el control de caducidad y la auditoría. Después de mover estos flujos al backend, el registro y la recuperación se validan con códigos generados y almacenados en servidor, mejorando la seguridad y la trazabilidad. El resultado es un proceso de registro y cambio de contraseña más robusto y consistente, con menor probabilidad de fallos tras recargas o cambios de sesión.

8.2. Cambios de interfaz y evidencias visuales

Para documentar el antes y el después de la plataforma se incluyen capturas comparativas organizadas por áreas. Las pantallas no incluidas en esta comparativa se mantienen sin cambios respecto a la implementación anterior. A partir de este punto se indica explícitamente en cada pie de figura qué capturas corresponden al estado anterior (antes) y cuáles al estado actual (después).

8.2.1. Inicio y navegación

Se reorganizó la home para reforzar la jerarquía visual, se añadieron bloques más claros y se ajustaron el menú superior y el pie para mejorar la usabilidad y el recorrido principal (Figuras 8.1 y 8.2).

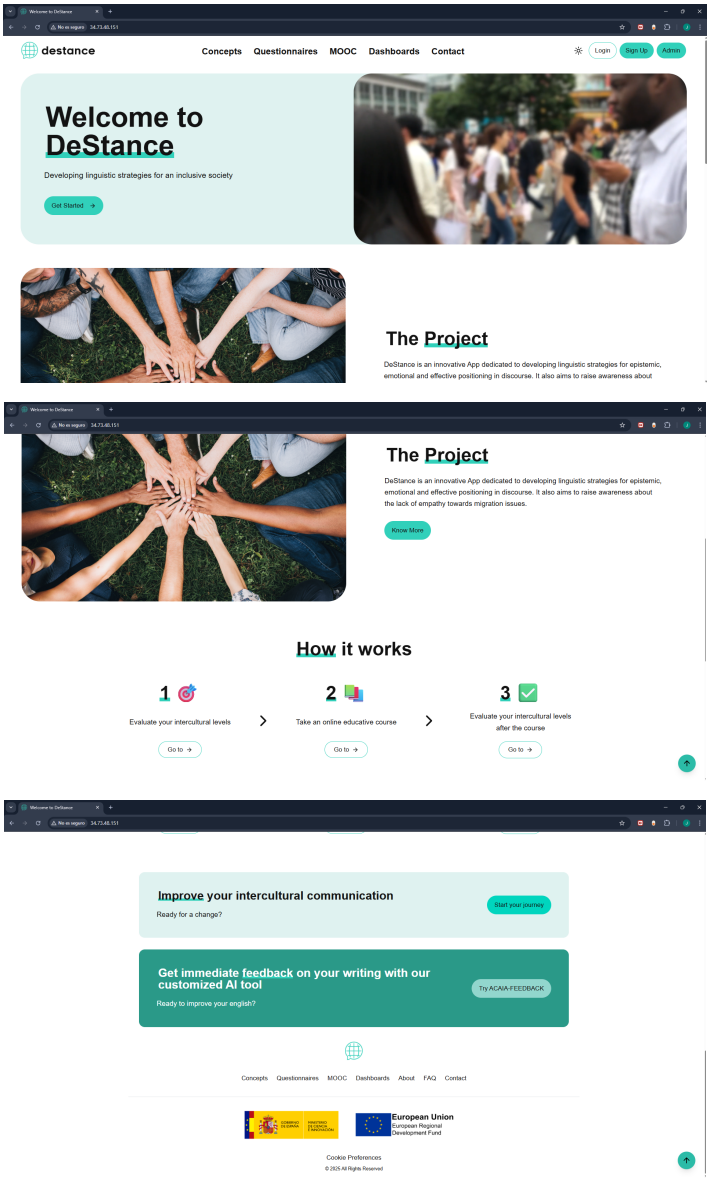

Figura 8.1: Página de inicio antes de los cambios (tres vistas).

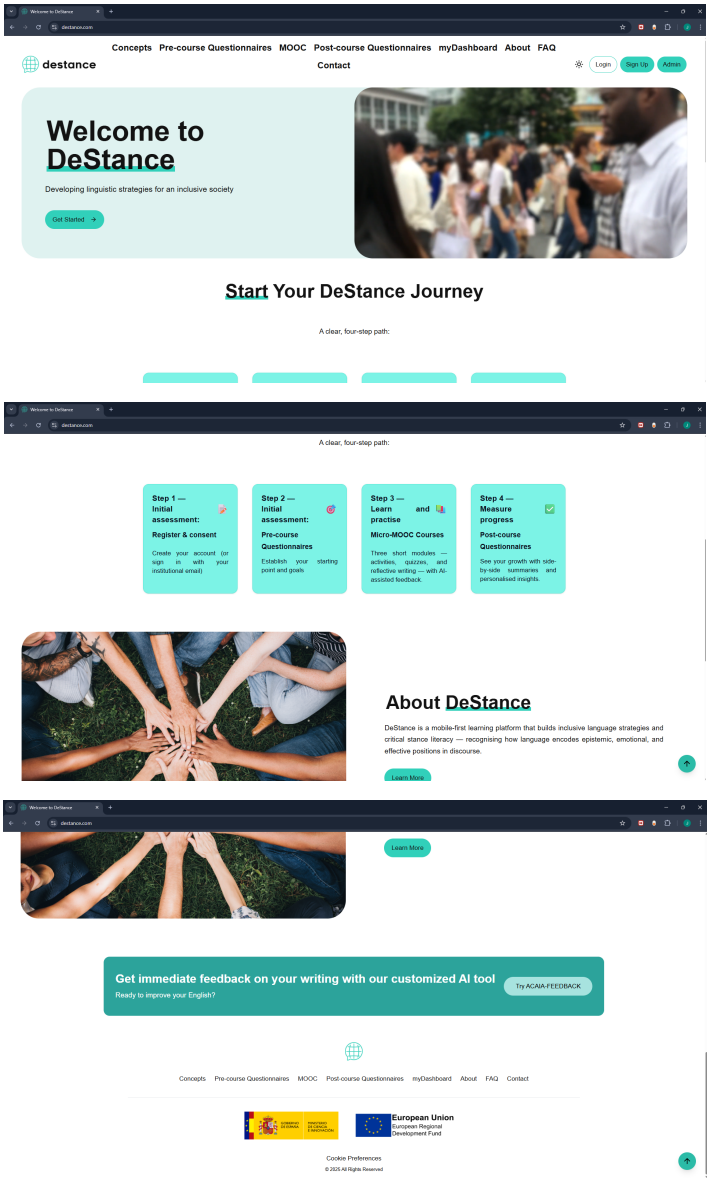

Figura 8.2: Página de inicio después de los cambios (tres vistas).

8.2.2. Páginas informativas y contacto

En About se añadieron imágenes generadas con IA, se ajustaron textos y se corrigió la sección The Project para evitar problemas de ancho (estilos tipo w-40). En FAQ solo se actualizaron las preguntas. Concepts se mantiene igual que en el despliegue de GCP realizado por Javier Hernández Contreras y se incluyen capturas como evidencia [1]. La sección de contacto se actualizó visualmente (Figuras 8.3, 8.4, 8.5, 8.6, 8.7, 8.8 y 8.9).

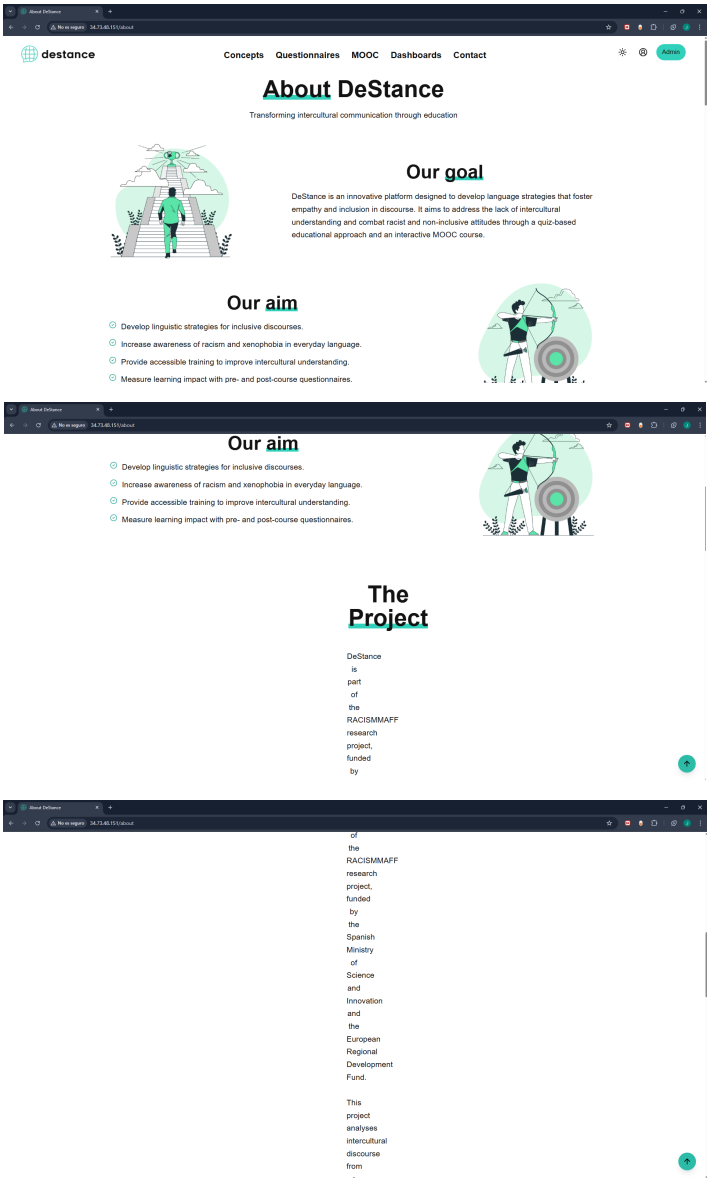

Figura 8.3: Sección About antes de los cambios (muestras 1).

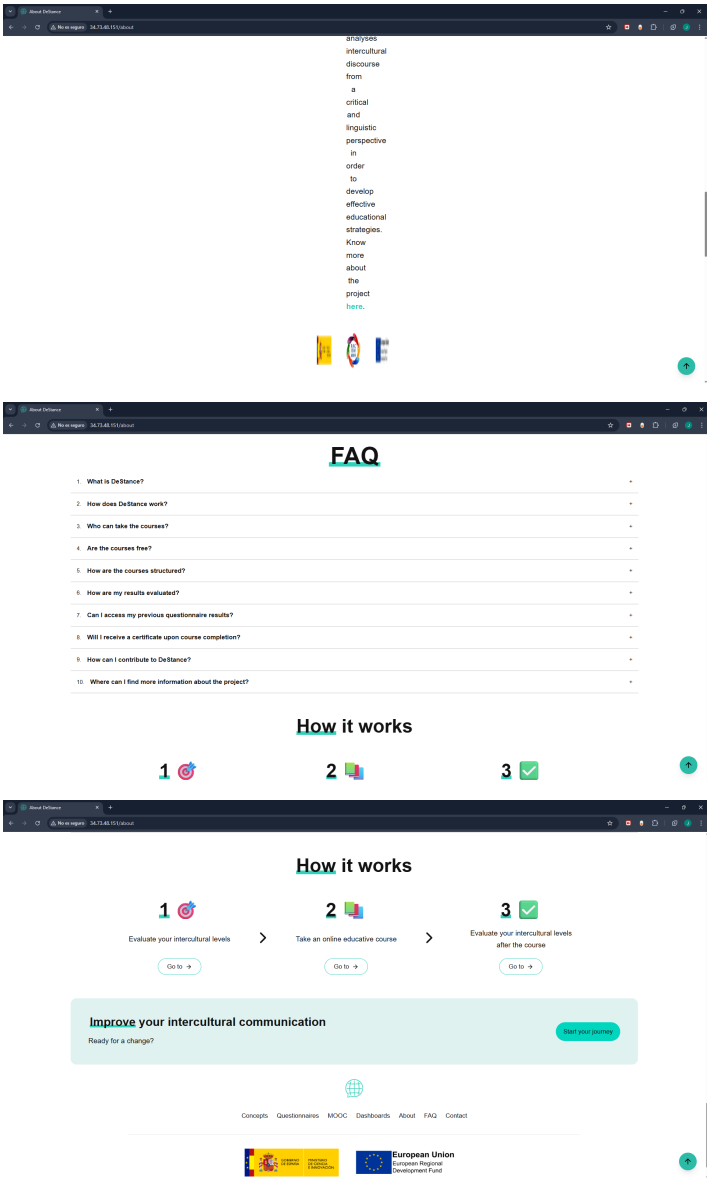

Figura 8.4: Sección About antes de los cambios (muestras 2).

Figura 8.5: Sección About después de los cambios (muestras).

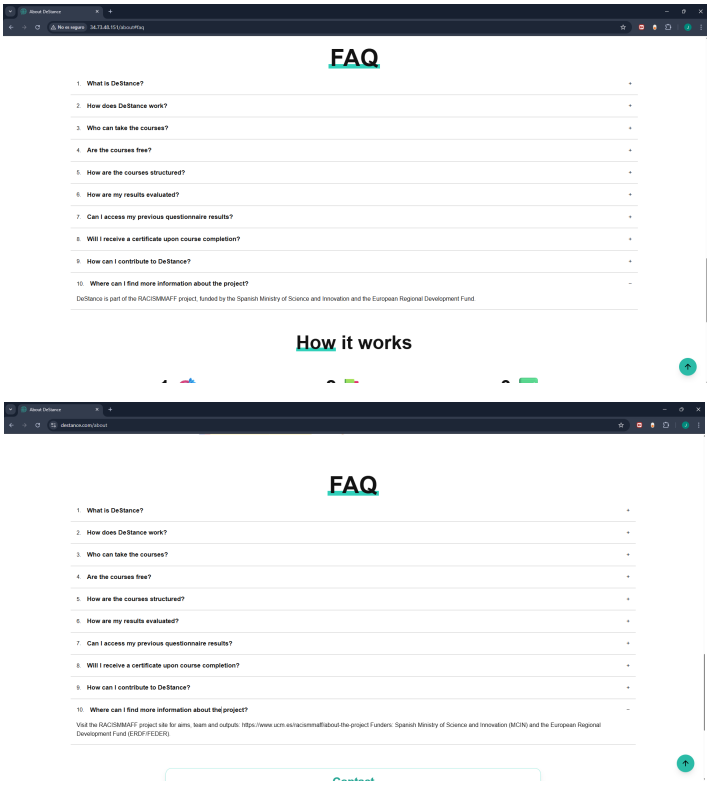

Figura 8.6: FAQ antes y después (actualización de preguntas).

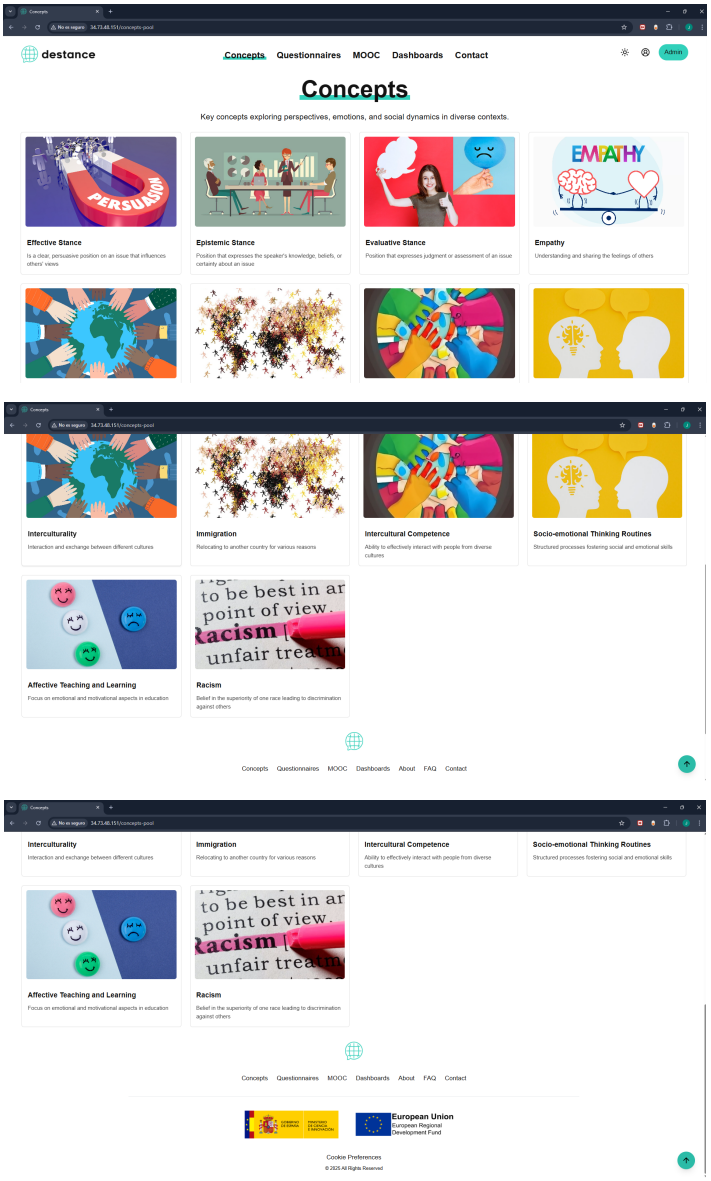

Figura 8.7: Concepts antes de los cambios.

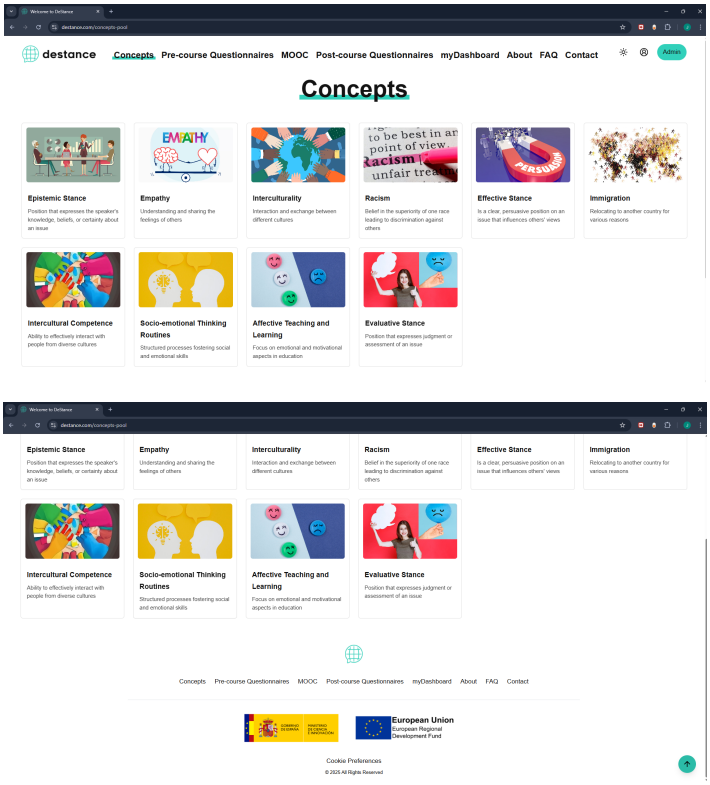

Figura 8.8: Concepts después de los cambios (sin cambios funcionales).

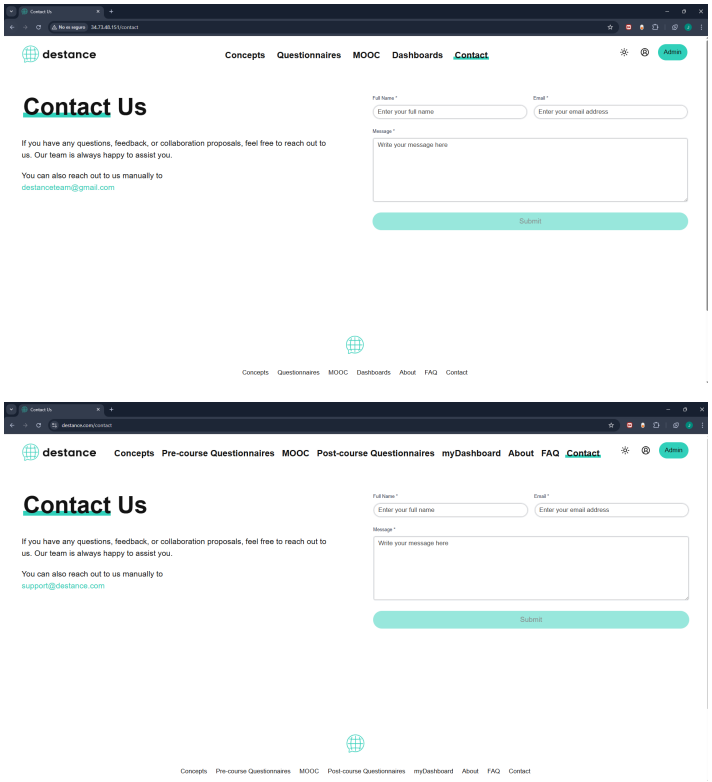

Figura 8.9: Contacto antes y después.

8.2.3. MOOC, vídeos y feedback

En los MOOC se incorporaron vídeos interactivos y el vídeo se convirtió en el primer recurso visible. Se redujo el mínimo de palabras en los writings a 10 para permitir avanzar sin bloqueos y se añadieron recursos descargables. También se mejoraron las gráficas de feedback con etiquetas integradas en el texto (trabajo realizado por Guillermo Palomero Bernal). Las Figuras 8.10, 8.11, 8.12, 8.13 y 8.14 recogen el antes y después.

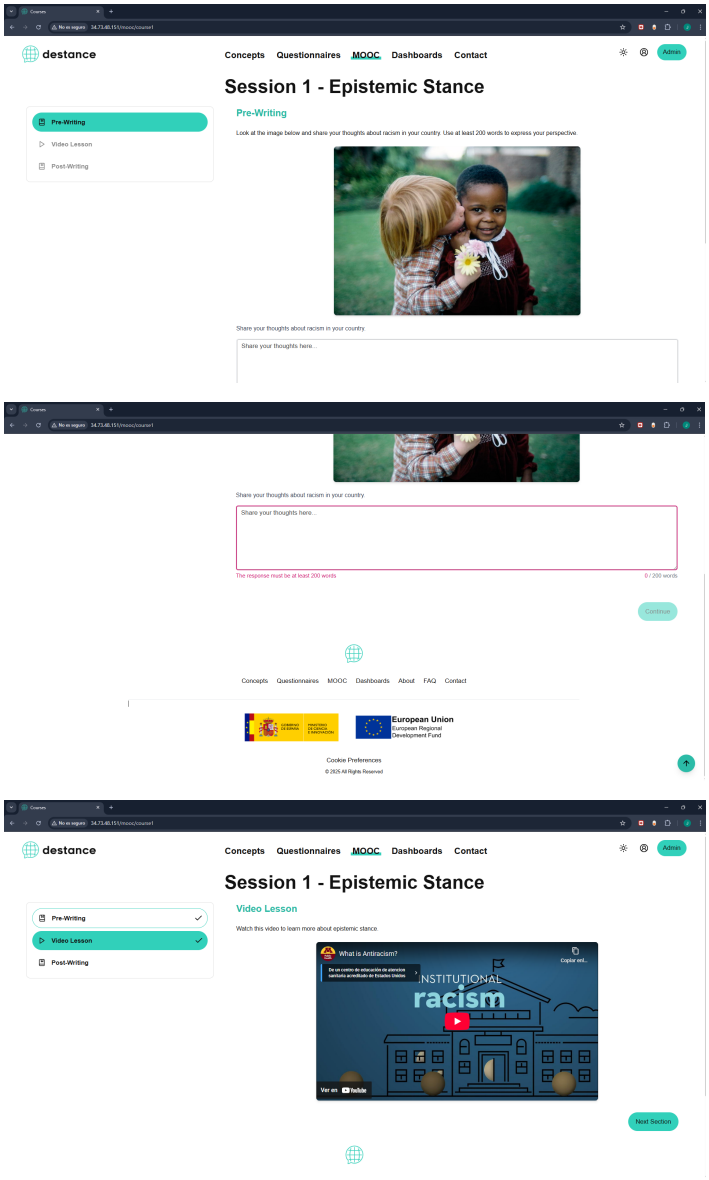

Figura 8.10: MOOC 1 antes de los cambios (vista general y vídeo).

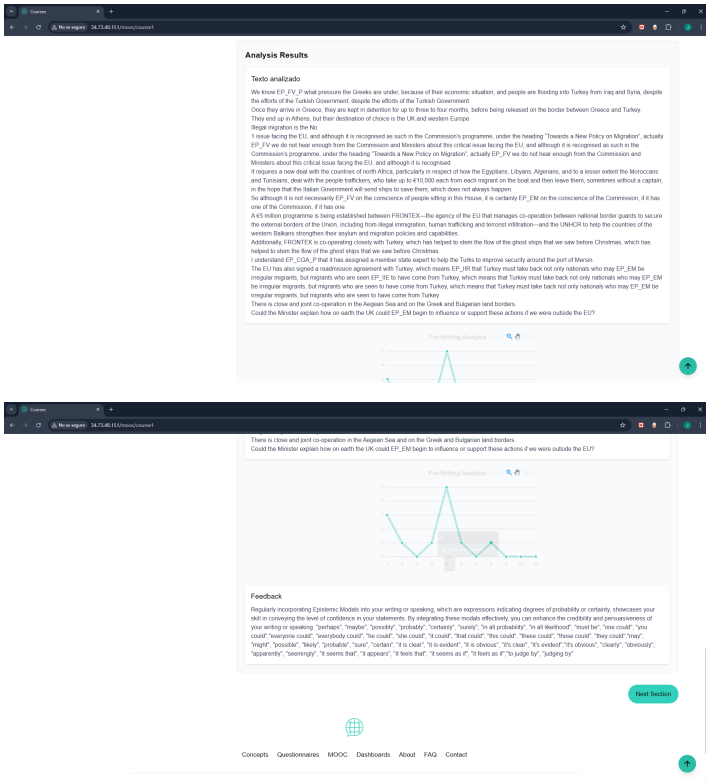

Figura 8.11: MOOC 1 antes: feedback y bloqueo por mínimo de palabras.

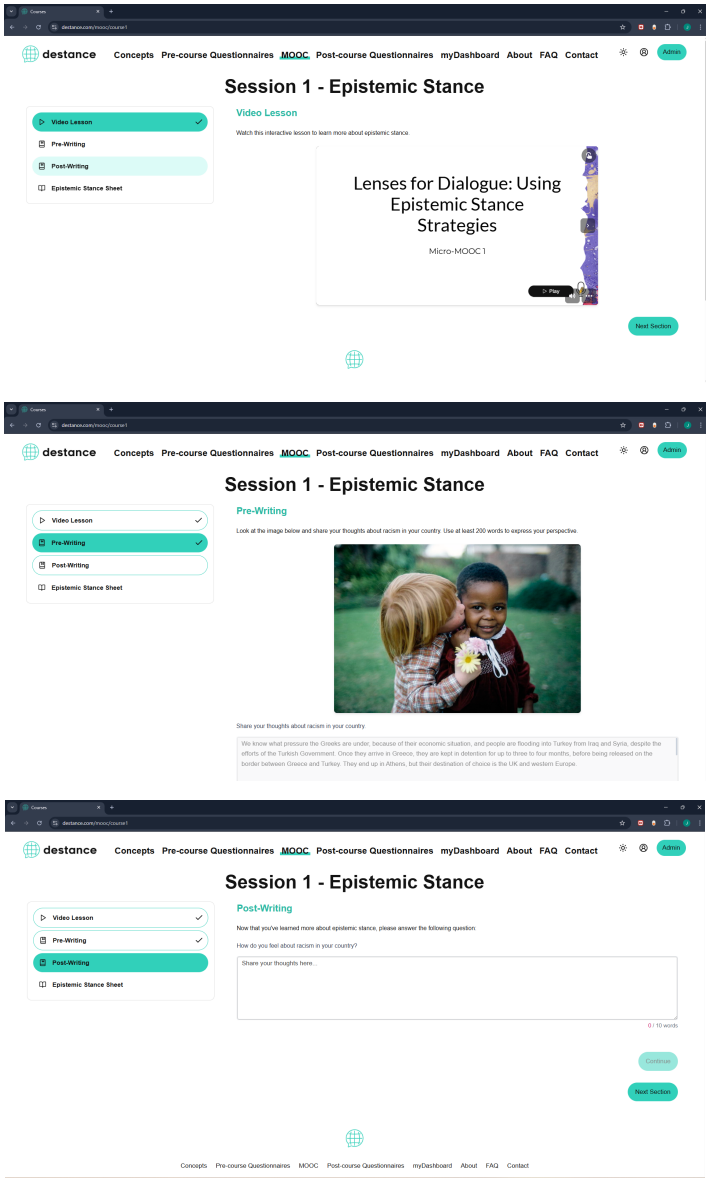

Figura 8.12: MOOC 1 después: vídeo primero y pasos pre/post-writing.

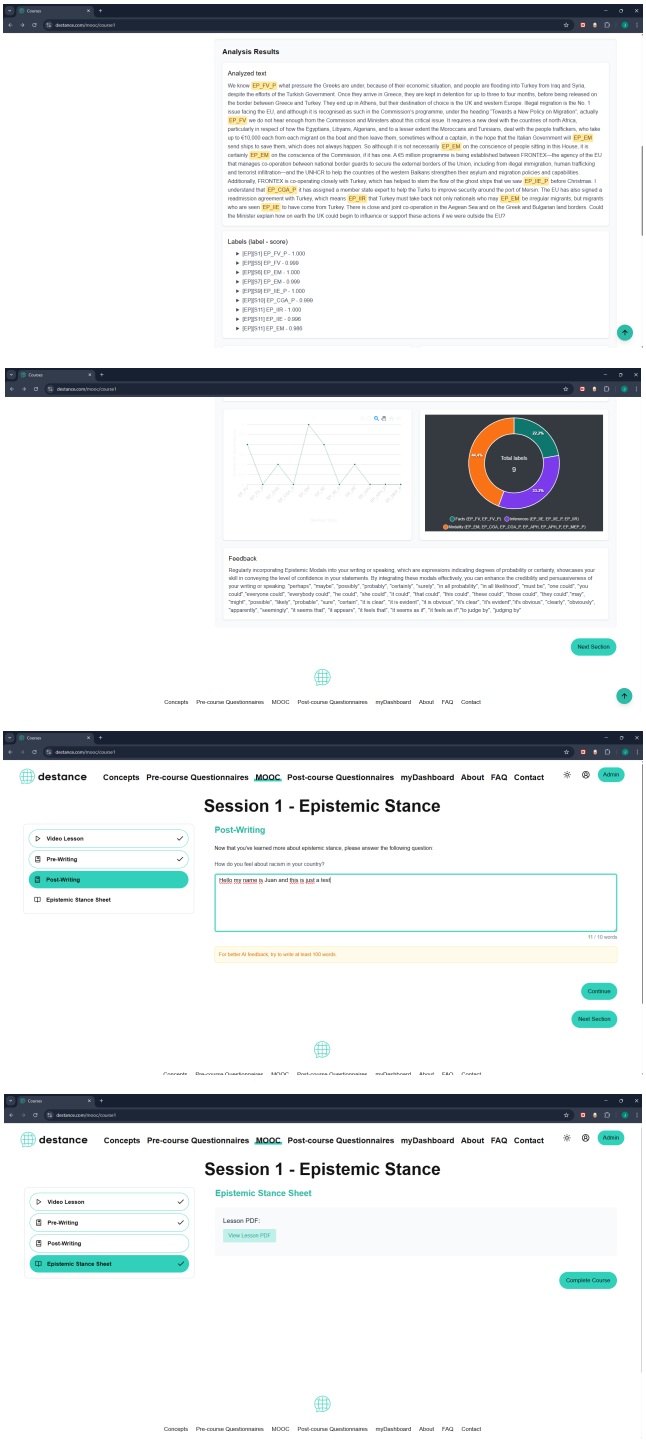

Figura 8.13: MOOC 1 después: feedback etiquetado, mínimo de 10 palabras para continuar y recomendación de 100, con recursos descargables.

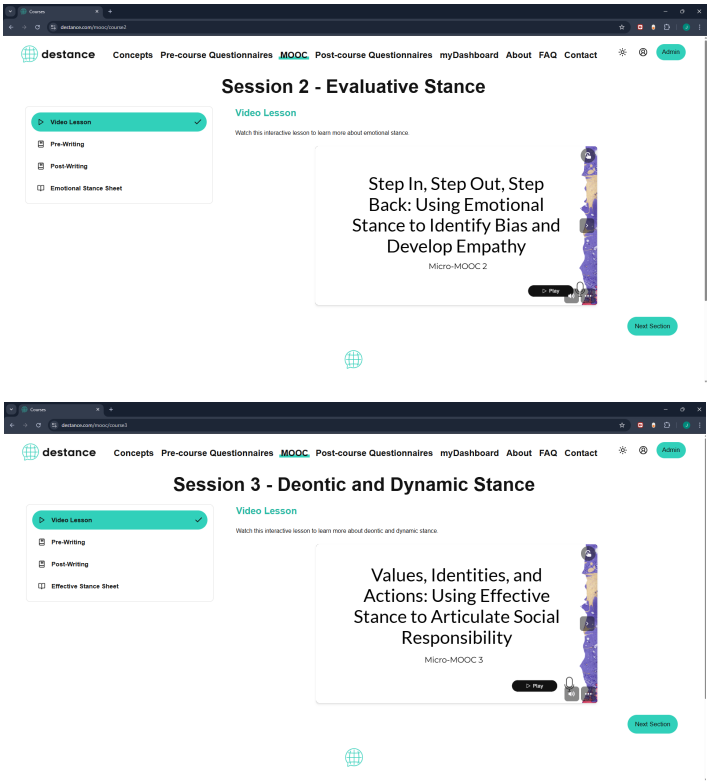

Figura 8.14: MOOC 2 y MOOC 3 después de los cambios.

8.2.4. Cuestionarios

Se simplificó la navegación para acceder directamente a pre y post, sin pasar por una pantalla intermedia. La Figura 8.15 muestra el cambio.

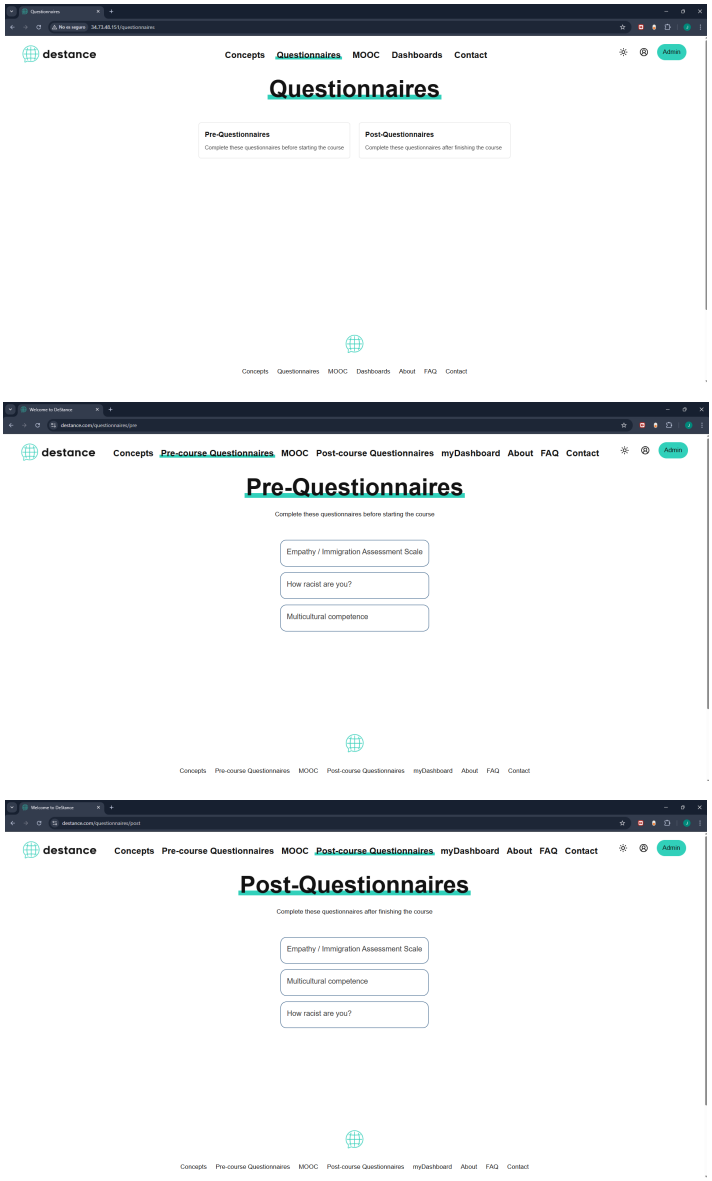

Figura 8.15: Cuestionarios: antes (listado general) y después (acceso directo a pre/post).

8.2.5. Dashboards y privacidad

Se eliminaron filtros para evitar comparar datos de otros usuarios y se reforzó la privacidad mostrando solo resultados del usuario autenticado. La Figura 8.16 recoge el estado antes y después; el dashboard MOOC se mantiene sin cambios y no se incluye.

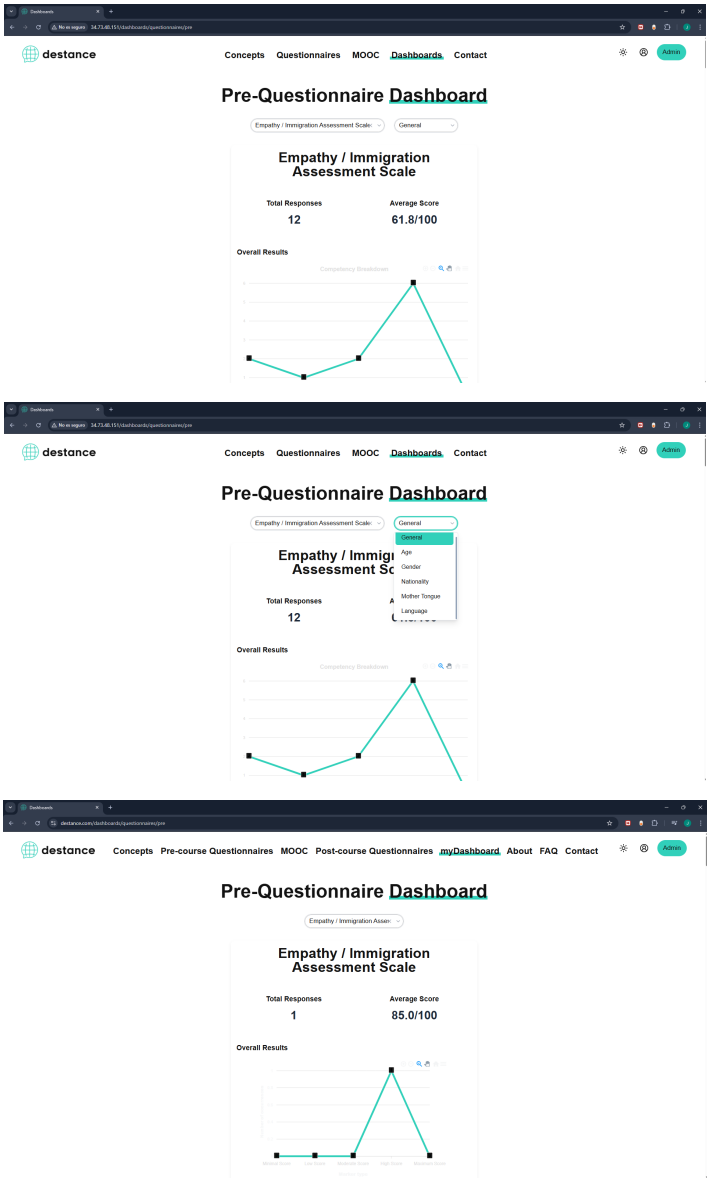

Figura 8.16: Dashboards de cuestionarios: antes (vista y filtro) y después (sin filtro y con resultados solo del usuario en su sesión).

8.2.6. Estados de trabajo en curso y control de acceso

En la versión anterior, el módulo de feedback ACAIA mostraba un error en pantalla; ahora se indica explícitamente como work in progress. Además, se añadieron estados work in progress para secciones aún no implementadas (por ejemplo, el vídeo de Get started) y se ocultó el acceso a Admin para usuarios sin rol administrador (Figuras 8.17, 8.18 y 8.19).

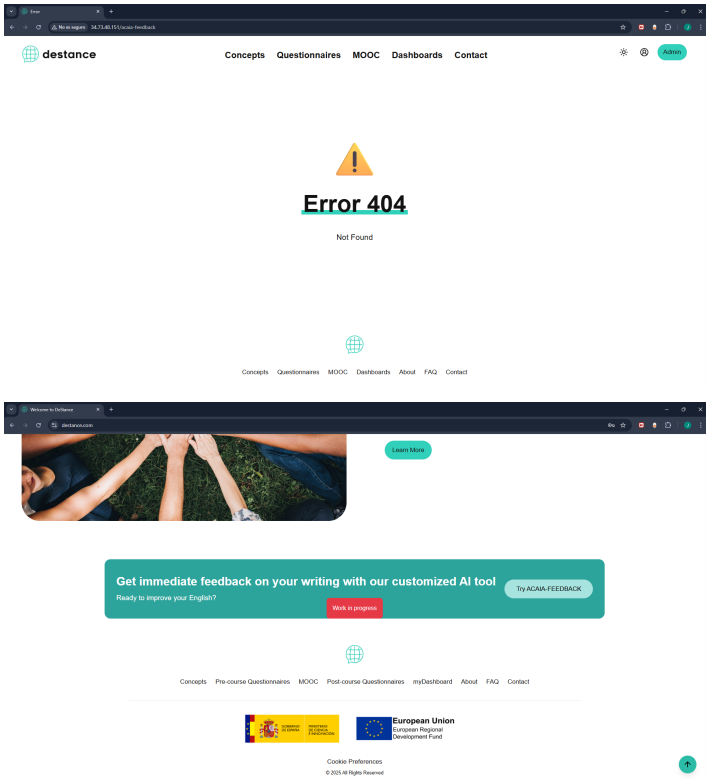

Figura 8.17: ACAIA antes (error) y después (aviso work in progress).

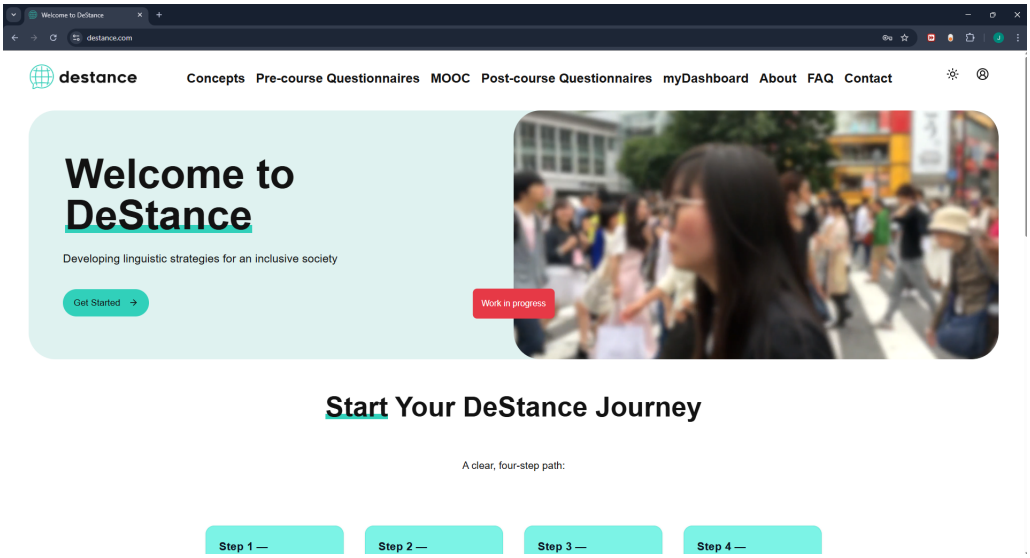

Figura 8.18: Estado actual (después): work in progress para Get started.

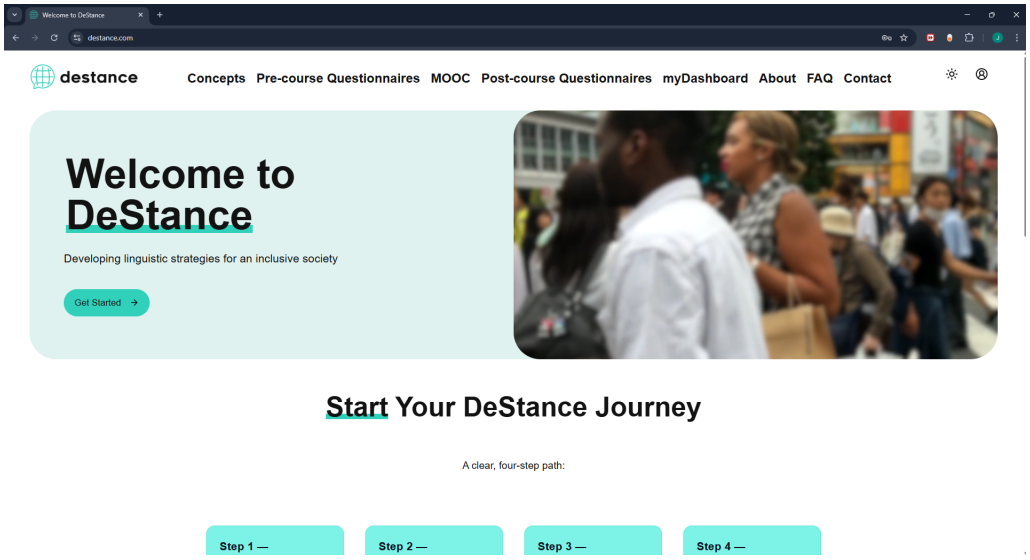

Figura 8.19: Estado actual (después): el acceso a Admin se oculta cuando el usuario no tiene rol administrador.

Capítulo 9

Análisis de impacto

9.1. Experiencia de usuario

Se mejora la comprensión del flujo (vídeo, pre, post), se reduce la barreras en escritura y se ofrece orientación explícita para mejorar la calidad del feedback. Como impacto directo, los estudiantes encuentran el recurso principal al inicio y pueden continuar sin bloqueos, y los textos se mantienen entre sesiones, evitando frustración y repeticiones innecesarias.

Como inconveniente, la mayor libertad de navegación puede requerir una guía adicional para usuarios menos autónomos, por lo que la comunicación de los pasos adquiere mayor importancia.

9.2. Mantenibilidad

La estructura modular de los MOOC permite actualizar vídeos, textos y PDFs sin reescribir la lógica central. La actualización de contenidos desde documentos docentes reduce deuda editorial y facilita su mantenimiento por parte del equipo académico.

El riesgo principal es la dependencia de un único repositorio de datos en administración: si el contenido no se valida o se sube con errores, se reflejará en la plataforma, por lo que se recomienda mantener revisiones editoriales.

9.3. Escalabilidad

El enfoque facilita incorporar nuevos módulos y recursos, así como migrar Genially a formatos autoalojados en el futuro. La arquitectura está preparada para migrar y escalar, pero en CeSvImA la escalabilidad está limitada por una única VPS.

El inconveniente potencial es que cada integración externa añade puntos de fallo y dependencias de terceros, por lo que conviene definir estrategias de

contingencia (versiones estáticas o backups) si los proveedores cambian su política o disponibilidad.

9.4. Seguridad y operación

La centralización del envío de correos en backend reduce exposición de claves, mejora la trazabilidad y facilita auditorías. El proxy de análisis en backend evita problemas de CORS y permite controlar el acceso al servicio externo.

Como contrapartida, se incrementa la responsabilidad del servidor en la gestión de secretos y en la observabilidad de errores, todavía en fase de implementación avanzada pero no terminada, por lo que la operación debe contemplar rotación de claves y una monitorización también en fase de implementación avanzada pero no terminada. El mantenimiento de la línea de despliegue en CeSvImA preserva la operatividad futura con criterios de seguridad y automatización.

Sobre los riesgos mitigados, el cambio a validaciones en servidor reduce la posibilidad de manipular el flujo de registro y recuperación desde el cliente, limita la reutilización de códigos y evita exponer claves en el navegador. El proxy de análisis reduce fallos de red y CORS y permite controlar la salida y entrada de datos hacia servicios externos. El proxy inverso unifica puntos de entrada y reduce configuraciones dispersas, aunque exige mantener una política clara de seguridad y actualizaciones del servidor.

9.5. Riesgos operativos

Los riesgos operativos principales se concentran en la dependencia del correo transaccional (fallos de proveedor o bloqueo de dominio), la configuración DNS (propagación lenta o errores de CNAME), la gestión de credenciales y secrets (exposición accidental, rotación insuficiente o permisos mal asignados), y la disponibilidad de la VPS (caídas de servicio, mantenimiento o saturación de recursos). En CeSvImA, la apertura y cierre de puertos debe seguir políticas institucionales; una apertura no autorizada puede provocar bloqueos, incumplimientos normativos o la exposición de servicios en un entorno compartido. También existen riesgos asociados a firewalls mal configurados (puertos críticos cerrados o abiertos sin necesidad), a certificados TLS caducados o mal instalados (errores de navegador y pérdida de confianza) y a almacenamiento insuficiente para bases de datos o logs. Estos riesgos se mitigan con control de cambios, copias de seguridad, revisión de reglas de red, rotación de claves y monitorización continua del sistema, en fase de implementación avanzada pero no terminada.

9.6. Impacto social y relación con los Objetivos de Desarrollo Sostenible

Este Trabajo Fin de Grado se alinea con varios Objetivos de Desarrollo Sostenible (ODS) de la Agenda 2030 de Naciones Unidas, tanto desde la perspectiva educativa

como tecnológica [40], [41].

ODS 4 – Educación de calidad: la plataforma De-Stance contribuye a una educación inclusiva, equitativa y de calidad, promoviendo la competencia intercultural y el pensamiento crítico en el alumnado universitario. El rediseño técnico mejora la accesibilidad, la claridad del recorrido educativo y la estabilidad del sistema, facilitando la participación de estudiantes de distintos contextos.

ODS 10 – Reducción de las desigualdades: el proyecto apoya la reducción de desigualdades al ofrecer un entorno digital que visibiliza barreras culturales y fomenta la empatía, la reflexión y el respeto en torno a cuestiones de racismo e inmigración.

ODS 16 – Paz, justicia e instituciones sólidas: De-Stance impulsa una educación orientada a valores de pluralidad, diálogo y respeto mutuo. Su implementación en un entorno académico abierto y colaborativo contribuye a fortalecer la cooperación entre instituciones universitarias.

ODS 9 – Industria, innovación e infraestructura: en el plano técnico, el proyecto aplica tecnologías innovadoras de despliegue y monitorización en la nube (Docker, Nginx, GitHub Actions, Prometheus y Grafana), consolidando una infraestructura segura, reproducible y sostenible para futuras ampliaciones.

Capítulo 10

Conclusiones

La actualización de la plataforma alinea el producto con los requisitos docentes y mejora la experiencia educativa. El trabajo consolida la navegación y el contenido, refuerza la autenticación y el correo transaccional en backend y estabiliza la línea de despliegue en CeSvImA con HTTPS, proxy inverso y monitorización en fase de implementación avanzada pero no terminada.

Con ello se cumplen los objetivos del TFG: diseñar un entorno de despliegue seguro, automatizado y en la nube, y mejorar la funcionalidad, la seguridad, el mantenimiento y la operación de la plataforma.

10.1. Futuras líneas de trabajo

Como trabajo futuro, se recomienda integrar el vídeo introductorio pendiente, formalizar el enlace a ACAIA y completar las revisiones pendientes en resultados de cuestionarios, además de consolidar la automatización del despliegue y la capa de alertas de monitorización, en fase de implementación avanzada pero no terminada.

También se propone estudiar las etiquetas devueltas por Hugging Face en el pre-writing y el post-writing para comparar la evolución del estudiante y ofrecer un feedback claro sobre su mejora o estancamiento entre ambos textos.

En usabilidad quedan mejoras concretas: un onboarding breve en el primer MOOC que contextualice objetivos, un indicador de progreso por sección y ajustes visuales en las gráficas de los dashboards para mejorar legibilidad de ejes, leyendas y escalas. Los mensajes de confirmación al guardar writings ya están presentes, pero pueden reforzarse con una presentación más consistente para que el usuario perciba claramente el guardado automático.

Como evolución funcional, se propone añadir un botón en los MOOC que solo se habilite al completar el curso y permita iniciar una nueva escritura; al activarlo se limpiaría el texto en pantalla y se comenzaría un nuevo intento, mientras que el texto anterior permanecería guardado y consultable desde Dashboards. En la

misma línea, otro botón permitiría abrir el historial de resultados del usuario (textos previos y posteriores) para revisar sus textos sin salir del MOOC. Por último, se recomienda persistir el feedback generado por Hugging Face en base de datos y exponerlo también en Admin, de forma que los resultados puedan consultarse sin recalcular ni repetir llamadas al servicio externo. Siguiendo en la línea de guardar en base de datos los resultados desde hugging face de los usuarios, estos mismos resultados se podrían mostrar en Dashboards en los MOOC, ya que ahora mismo, el feedback generado en el apartado de MOOC es mucho más extenso que el que se muestra en el apartado de Dashboards, igual se debería pensar en integrar los resultados de Huggingface en Dashboards junto con los ya existentes.

Desde el punto de vista de infraestructura, una línea futura útil es sustituir el esquema de CNAME y redirección del dominio raíz por un enfoque de DNS dinámico (DDNS) con IONOS. La idea es mantener un registro A para destance.com y actualizar automáticamente la IP pública de la VPS mediante un cliente DDNS (por ejemplo, compatible con Domain Connect), ejecutado de forma periódica. Esta opción implica usar IONOS como DNS y solo es viable si permite crear registros A/AAAA. Con ello, la petición del usuario llegaría directamente a la VPS sin intermediarios, se reducen puntos de fallo y se simplifica la gestión del certificado TLS al centralizar el dominio en el servidor. El coste es que, si la IP cambia, la web podría quedar inaccesible durante la ventana entre actualizaciones (por ejemplo, hasta 5 minutos). Aunque las IPs de CeSvImA no deberían variar con frecuencia, no existe una confirmación técnica definitiva y conviene asumir ese riesgo residual.

En la misma línea de mejorar la experiencia y la privacidad, conviene revisar si es necesario solicitar tantos datos en el registro. Actualmente se piden campos como nacionalidad o lengua materna, pero esos filtros ya no se usan en los dashboards. Simplificar el alta reduciría barreras, aceleraría el acceso y permitiría que los usuarios compartan solo lo imprescindible. Además, aunque los datos no sean especialmente sensibles, el criterio de minimización recomienda pedir únicamente lo necesario para el objetivo educativo. También queda pendiente habilitar BIMI para reforzar la confianza en el correo transaccional y su entregabilidad.

La Tabla 10.1 resume las principales líneas de trabajo y su urgencia estimada.

Tabla 10.1: Futuras líneas de trabajo y urgencia estimada.

Línea de trabajo	Motivación/beneficio	Urgencia
Vídeo introductorio y enlace ACAIA	Completar el recorrido docente y eliminar estados pendientes.	Media
Revisión de resultados de cuestionarios	Completar ajustes pendientes y asegurar consistencia en la presentación de resultados.	Media

Línea de trabajo	Motivación/beneficio	Urgencia
Alertas de monitorización (fase de implementación avanzada pero no terminada) y uptime checks	Detectar caídas y reducir tiempos de respuesta operativa.	Alta
Persistencia y análisis del feedback de Hugging Face	Evitar recomputaciones, comparar evolución pre/post y mostrar resultados completos en dashboards y Admin.	Alta
Onboarding y progreso por sección en MOOC	Reducir barreras y guiar al alumnado durante el recorrido.	Media
Ajustes visuales en dashboards y confirmaciones de guardado	Mejorar legibilidad y reforzar la percepción del guardado automático.	Media
Nuevo intento e historial de resultados en MOOC	Permitir nuevas escrituras y revisión de textos sin salir del curso.	Media
Validación de monitorización (Grafana/Prometheus, en fase de implementación avanzada pero no terminada)	Confirmar la estabilidad de la recolección de métricas y la configuración tras una implementación mínima.	Alta
Implementación de BIMI para correo transaccional	Asociar el logotipo al remitente y mejorar la confianza en los envíos.	Baja
Pruebas unitarias y de sistema	Asegurar calidad y cerrar tareas pendientes (trabajo faltante del compañero que dejó el proyecto).	Alta
DDNS con registro A en IONOS para destance.com	Acceso directo a la VPS, menos intermediarios y gestión TLS centralizada.	Media
Simplificación de datos en registro	Reducir barreras y aplicar minimización de datos personales.	Media

10.2. Limitaciones actuales

El despliegue en CeSvImA ya opera en HTTPS con certificados y redirección a www, pero el dominio raíz depende de una redirección gestionada por el registrador porque CeSvImA exige CNAME y no permite registros A. Actualmente se usa un CNAME al FQDN de CeSvImA por la política de IP variable. Como alternativa futura (si se acepta usar A+DDNS en IONOS y asumir el riesgo de cambio de IP), se podría apuntar al apex directamente con un registro A dinámico. Esta restricción condiciona el control del DNS y complica un acceso directo al dominio raíz; por ello, se propone como trabajo futuro un esquema DDNS con registro A actualizado de forma periódica.

La automatización completa del despliegue depende de la configuración de secretos en el repositorio y de la validación institucional. Algunas mejoras del

dashboard y visualización de resultados por usuario siguen en fase de ajuste.

En el planteamiento inicial se contemplaba Infraestructura como Código con Terraform, pero no aplica en esta fase porque la VPS ya existe y está gestionada por CeSvImA, por lo que no se dispone de control para crear o reprovisionar infraestructura desde código. Terraform se plantea únicamente como trabajo futuro si la plataforma se migra a un cloud propio donde sí se pueda definir y reproducir la infraestructura completa. En ese escenario, el despliegue se describiría con módulos de red, instancias, firewall y almacenamiento, permitiendo reproducibilidad, control de cambios y auditoría de configuraciones. Sus ventajas serían la reducción de errores manuales, la capacidad de replicar entornos (desarrollo, staging, producción) y la posibilidad de aplicar cambios de forma segura y versionada.

Por limitaciones de tiempo y por la salida de compañeros del proyecto, no se han realizado pruebas unitarias ni pruebas de sistema completas. Del mismo modo, la automatización de pruebas queda como trabajo futuro: se propone integrar pruebas unitarias y de integración en el pipeline de CI para validar cambios antes de desplegar.

En seguridad de sesiones se han aplicado medidas básicas (cookies seguras con Secure, HttpOnly y SameSite, autenticación centralizada en backend y HTTPS mediante proxy inverso). Dado el tiempo y el alcance del TFG, no se han incorporado mecanismos avanzados recomendados por OWASP, como rotación periódica de identificadores de sesión, invalidación ante comportamientos anómalos, protección explícita frente a CSRF, límites de sesiones concurrentes por usuario o rate limiting en procesos de autenticación. Estas medidas se dejan como trabajo futuro para una puesta en producción con requisitos más altos.

Respecto a los servicios de IA, con la infraestructura actual no es viable ejecutar localmente modelos de tamaño medio como RoBERTa junto al resto de servicios, debido al límite de memoria disponible y al tiempo de preparación necesario. En esta fase, el análisis se mantiene como servicio externo; esto reduce el impacto local y permite estabilizar el sistema con los recursos actuales. La ejecución local de modelos solo sería realista con modelos pequeños y con tráfico bajo, mientras que los modelos medianos quedarían muy ajustados en consumo de RAM y CPU. Por limitaciones de tiempo y de recursos, esta parte se mantiene como línea futura del proyecto y se apoya en trabajos previos sobre análisis automático y feedback en De-Stance [7].

10.3. Evolución futura de la plataforma y capacidad de infraestructura

CeSvImA (Centro de Supercomputación y Visualización de Madrid) proporciona infraestructura académica para proyectos de investigación y docencia. En este contexto se ha utilizado una máquina de servicio gratuito, con 2 vCPU (Intel Xeon), 4 GB de memoria RAM, 100 GB de disco y sistema operativo Linux Debian. Esta capacidad es suficiente para un despliegue académico de baja concurrencia,

validación funcional y uso docente controlado. A medio plazo, si se incrementa el número de usuarios simultáneos o se añaden procesos de análisis más costosos, será necesario evaluar opciones: ampliar recursos en CeSvImA, mover el análisis a servicios externos dedicados o incorporar caché y colas de tareas. También se recomienda habilitar monitorización y métricas de consumo, en fase de implementación avanzada pero no terminada, para ajustar la infraestructura según la demanda real.

Desde el punto de vista de coste y beneficio, mantener el despliegue en la VPS gratuita permite validar el sistema con bajo coste operativo, pero limita el margen para picos de carga y tareas intensivas. Ampliar recursos o migrar servicios específicos a proveedores gestionados incrementa el coste, pero aporta mayor estabilidad, redundancia y capacidad de crecimiento. Esta comparación debe valorarse según el número de usuarios reales y la criticidad del servicio en el calendario académico.

10.3.1. Estimación básica de costes y límites de escalado

En simple, la VPS actual es una opción de coste muy bajo (entorno académico) que funciona bien con pocos usuarios. A medida que crece la plataforma, las necesidades de memoria, CPU y almacenamiento aumentan y aparecen costes adicionales: un servidor comercial tiene un coste fijo mensual bajo y los servicios gestionados añaden coste variable según el uso.

Desde el punto de vista técnico, el límite principal del despliegue actual es la capacidad de una sola máquina (CPU, RAM y disco). No hay escalado automático ni réplicas, por lo que el crecimiento requiere ampliar la VPS o migrar servicios críticos (por ejemplo, la base de datos o el análisis) a infraestructuras gestionadas. En escenarios futuros se recomienda separar frontend estático en CDN, backend en PaaS con autoescalado y base de datos gestionada para aumentar disponibilidad y reducir mantenimiento.

La comparativa se resume en la Tabla 10.2.

Tabla 10.2: Comparativa de costes y límites de escalado.

Escenario	Coste (orden)	Límites y escalado
VPS CeSvImA	Muy bajo (académico)	2 vCPU/4 GB RAM; sin autoescalado; válido para baja concurrencia.
VPS comercial	Bajo y fijo	Escalado vertical manual; mantenimiento del sistema y de la base de datos.
PaaS + BD gestionada + CDN	Medio, variable	Escalado automático y mejor disponibilidad; coste depende del tráfico.
Kubernetes gestionado	Alto + operación	Máxima flexibilidad; mayor complejidad y coste operativo.

Para los modelos de IA, una línea futura razonable es mantener el análisis en un proveedor externo (por ejemplo, Hugging Face) mientras no se disponga

de infraestructura con más memoria. Si se quisiera ejecutar los modelos en contenedor, sería necesario dimensionar la VPS o migrar a un servicio gestionado con recursos suficientes, ya que el consumo de RAM y CPU crece con el tamaño del modelo y el número de peticiones. Esta decisión permite equilibrar coste y rendimiento: el proveedor externo minimiza el consumo local, mientras que la ejecución local ofrece más control pero exige mayor capacidad y mantenimiento.

10.4. Agradecimientos

Agradezco al proyecto RACISMMAFF la oportunidad de realizar este TFG dentro de su marco de investigación y al equipo de De-Stance por permitirme trabajar en una plataforma educativa tan interesante y formativa. Agradezco de forma especial a mis responsables académicas, Jelena Bobkina y Elena Domínguez, y a Guillermo Palomero Bernal por su acompañamiento durante el proyecto. También agradezco a Nicolás Sáenz Lechón, miembro de GAMMA, su ayuda para conseguir la VPS académica en CeSvImA. Y, sobre todo, quiero agradecer a Javier Hernández Contreras su ayuda constante y el apoyo que me ha brindado a lo largo de todo el trabajo.

Bibliografía

[1] J. Hernández Contreras, «Trabajo Fin de Grado: administración y dashboards en De-Stance», Tesis de Grado, Escuela Técnica Superior de Ingenieros Informáticos, Universidad Politécnica de Madrid, Madrid, España, 2025. [En línea]. Disponible: https://oa.upm.es/91177/1/JAVIER_HERNANDEZ_CONTRERAS_TFG.pdf. Accedido: 26-sept.-2025.

[2] CeSvImA. «Centro de Supercomputación y Visualización de Madrid». (2024), [En línea]. Disponible: https://www.cesvima.upm.es/. Accedido: 2-ene.-2026.

[3] RACISMMAFF Consortium. «RACISMMAFF Project». (2024), [En línea]. Disponible: https://www.ucm.es/racismmaff/. Accedido: 25-sept.-2025.

[4] CITSEM–UPM. «Grupo de Aplicaciones Multimedia y Acústica (GAMMA)». (2024), [En línea]. Disponible: https://www.citsem.upm.es/es/quienes-somos/grupos-de-investigacion/grupo-de-aplicaciones-multimedia-y-acustica-gamma. Accedido: 11-ene.-2026.

[5] G. Álvarez García, «Trabajo Fin de Grado sobre la plataforma De-Stance», Tesis de Grado, Escuela Técnica Superior de Ingenieros Informáticos, Universidad Politécnica de Madrid, Madrid, España, 2025. [En línea]. Disponible: https://oa.upm.es/90162/1/TFG_GONZALO_ALVAREZ_GARCIA.pdf. Accedido: 27-sept.-2025.

[6] D. Ramón Robertson, «Trabajo Fin de Grado: módulos interactivos en De-Stance», Tesis de Grado, Escuela Técnica Superior de Ingenieros Informáticos, Universidad Politécnica de Madrid, Madrid, España, 2025. [En línea]. Disponible: https://oa.upm.es/89550/3/TFG_Daniel_Ramon_Robertson_1.pdf. Accedido: 27-sept.-2025.

[7] A. C. Murillo García, «Trabajo Fin de Grado: análisis automático de textos en De-Stance», Tesis de Grado, Escuela Técnica Superior de Ingenieros Informáticos, Universidad Politécnica de Madrid, Madrid, España, 2025. [En línea]. Disponible: https://oa.upm.es/91053/1/TFG_ANA_CLARA_MURILLO_GARCIA.pdf. Accedido: 28-sept.-2025.

[8] S. D. Montero Cabezas, «Diseño y Despliegue de una Solución con CI/CD en Cloud», Tesis de Grado, Escuela Técnica Superior de Ingenieros Informáticos, Universidad Politécnica de Madrid, Madrid, España, 2024. [En línea]. Disponible: https://oa.upm.es/82634/1/TFG_SANTIAGO_DARIO_MONTERO_CABEZAS.pdf. Accedido: 3-ene.-2026.

[9] J. A. Pérez López, «Integración Continua y Despliegue Continuo de una Aplicación Web», Tesis de Grado, Universidad de Alicante, Alicante, España, 2023. [En línea]. Disponible: https://rua.ua.es/bitstream/10045/136214/1/Integracion_Continua_y_Despliegue_Continuo_de_una_ap_Perez_Lopez_Jose_Andres.pdf. Accedido: 3-ene.-2026.

[10] OWASP Foundation. «Application Security Verification Standard (ASVS)». (2024), [En línea]. Disponible: https://owasp.org/www-project-application-security-verification-standard/. Accedido: 5-ene.-2026.

[11] Docker Inc. «Docker overview». (2024), [En línea]. Disponible: https://docs.docker.com/get-started/docker-overview/. Accedido: 6-ene.-2026.

[12] IBM Corp. «Hugging Face». (2024), [En línea]. Disponible: https://www.ibm.com/es-es/think/topics/hugging-face. Accedido: 6-ene.-2026.

[13] DataCamp. «AWS vs Azure vs GCP: Una comparación completa». (2025), [En línea]. Disponible: https://www.datacamp.com/es/blog/aws-vs-azure-vs-gcp. Accedido: 11-ene.-2026.

[14] Amazon Web Services. «Decision Guides: Containers on AWS (incluye App Runner)». (2024), [En línea]. Disponible: https://aws.amazon.com/es/getting-started/decision-guides/containers-on-aws-how-to-choose/. Accedido: 11-ene.-2026.

[15] Microsoft Azure. «Azure vs AWS – Por qué elegir Azure». (2024), [En línea]. Disponible: https://azure.microsoft.com/es-es/pricing/azure-vs-aws/. Accedido: 11-ene.-2026.

[16] Microsoft Azure. «Azure Front Door and Azure CDN Documentation». (2024), [En línea]. Disponible: https://learn.microsoft.com/en-us/azure/frontdoor/. Accedido: 11-ene.-2026.

[17] Google Cloud. «Nivel gratuito de Compute Engine». (2024), [En línea]. Disponible: https://cloud.google.com/free/docs/free-cloud-features. Accedido: 7-ene.-2026.

[18] Amazon Web Services. «Lightsail pricing». (2024), [En línea]. Disponible: https://aws.amazon.com/es/lightsail/pricing/. Accedido: 7-ene.-2026.

[19] Google Cloud. «Google Cloud Pricing Calculator (Compute Engine)». (2024), [En línea]. Disponible: https://cloud.google.com/products/calculator. Accedido: 7-ene.-2026.

[20] Microsoft Azure. «Calculadora de precios de Azure». (2024), [En línea]. Disponible: https://azure.microsoft.com/es-es/pricing/calculator/. Accedido: 7-ene.-2026.

[21] Microsoft Corp. «Azure Machine Learning: Despliegue de contenedores personalizados». (2024), [En línea]. Disponible: https://learn.microsoft.com/es-es/azure/machine-learning/how-to-deploy-custom-container. Accedido: 8-ene.-2026.

[22] Amazon Web Services. «SageMaker Serverless Inference». (2024), [En línea]. Disponible: https://docs.aws.amazon.com/es_es/sagemaker/latest/dg/serverless-endpoints.html. Accedido: 8-ene.-2026.

[23] Microsoft Azure. «Hugging Face on Azure». (2024), [En línea]. Disponible: https://azure.microsoft.com/es-es/solutions/hugging-face-on-azure/. Accedido: 8-ene.-2026.

[24] Amazon Web Services. «AWS Pricing Calculator: SageMaker». (2024), [En línea]. Disponible: https://calculator.aws/#/createCalculator/SageMaker. Accedido: 8-ene.-2026.

[25] NGINX Inc. «Reverse Proxy Guide». (2024), [En línea]. Disponible: https://docs.nginx.com/nginx/admin-guide/web-server/reverse-proxy/. Accedido: 9-ene.-2026.

[26] Cloudflare Inc. «What is a reverse proxy?» (2024), [En línea]. Disponible: https://www.cloudflare.com/learning/cdn/glossary/reverse-proxy/. Accedido: 9-ene.-2026.

[27] OWASP Foundation. «Authentication Cheat Sheet». (2024), [En línea]. Disponible: https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html. Accedido: 5-ene.-2026.

[28] OWASP Foundation. «Session Management Cheat Sheet». (2024), [En línea]. Disponible: https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html. Accedido: 5-ene.-2026.

[29] Google LLC. «Autenticación del correo electrónico». (2024), [En línea]. Disponible: https://support.google.com/mail/answer/81126. Accedido: 9-ene.-2026.

[30] Cloudflare Inc. «¿Qué es DMARC?» (2024), [En línea]. Disponible: https://www.cloudflare.com/es-es/learning/email-security/dmarc-dkim-spf/. Accedido: 9-ene.-2026.

[31] Google LLC. «Configurar BIMI». (2024), [En línea]. Disponible: https://support.google.com/a/answer/10911320. Accedido: 9-ene.-2026.

[32] Google Developers. «Security best practices: Why HTTPS matters». (2024), [En línea]. Disponible: https://web.dev/articles/why-https-matters. Accedido: 9-ene.-2026.

[33] Grafana Labs. «Prometheus data source documentation». (2024), [En línea]. Disponible: https://grafana.com/docs/grafana/latest/datasources/prometheus/. Accedido: 9-ene.-2026.

[34] Prometheus Authors. «Prometheus Overview». (2024), [En línea]. Disponible: https://prometheus.io/docs/introduction/overview/. Accedido: 9-ene.-2026.

[35] Grafana Labs. «Grafana documentation». (2024), [En línea]. Disponible: https://grafana.com/docs/grafana/latest/. Accedido: 9-ene.-2026.

[36] Prometheus Authors. «node_exporter». (2024), [En línea]. Disponible: https://github.com/prometheus/node_exporter. Accedido: 9-ene.-2026.

[37] Google LLC. «cAdvisor: Container Resource Usage and Performance». (2024), [En línea]. Disponible: https://github.com/google/cadvisor. Accedido: 9-ene.-2026.

[38] GitHub Docs. «Using secrets in GitHub Actions». (2024), [En línea]. Disponible: https://docs.github.com/en/actions/security-guides/using-secrets-in-github-actions. Accedido: 9-ene.-2026.

[39] GitHub Docs. «Workflow syntax for GitHub Actions». (2024), [En línea]. Disponible: https://docs.github.com/en/actions/writing-workflows/workflow-syntax-for-github-actions. Accedido: 9-ene.-2026.

[40] Naciones Unidas. «Objetivos de Desarrollo Sostenible». (2024), [En línea]. Disponible: https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/. Accedido: 15-ene.-2026.

[41] Gobierno de España. «Conoce la Agenda 2030». (2024), [En línea]. Disponible: https://www.dsca.gob.es/es/agenda-2030/conoce-la-agenda. Accedido: 15-ene.-2026.

Apéndice A

Anexos

A.1. Informe de originalidad mediante Turnitin

Siguiendo las recomendaciones para la entrega del TFG, se incluye como anexo el informe de originalidad generado con Turnitin. El documento completo se sometió a un análisis de similitud para verificar coincidencias textuales y reforzar la transparencia académica del trabajo.

El informe indica un 5 % de similitud global, un porcentaje reducido, con coincidencias en fuentes de Internet y también en publicaciones y trabajos previos, especialmente en elementos comunes como la portada, la bibliografía y referencias institucionales. Las fuentes con mayor coincidencia corresponden a repositorios académicos y páginas institucionales (por ejemplo, oa.upm.es, upcommons.upc.edu, repositorio.unbosque.edu.co, docta.ucm.es y otras fuentes con porcentajes individuales bajos).

Estas coincidencias se explican por el uso de terminología técnica estándar, referencias a tecnologías ampliamente descritas y enunciados comunes en el contexto académico, además de la presencia de nombres de instituciones y conceptos propios del área.

No se identifican bloques extensos de texto no original, por lo que el resultado se interpreta como compatible con una memoria técnica con citas y descripciones de referencia.

El informe puede consultarse mediante un enlace al drive personal del autor de este TFG: Memoria TFG ETSIINF UPM - Juan Loscos Ortega - Turnitin.

A.2. Inventario de endpoints usados en producción

Tabla A.1: Inventario de endpoints usados en producción (web, admin y monitorización en fase de implementación avanzada pero no terminada).

Endpoint	Acceso	Origen	Descripción
GET /	Público	Frontend web	Acceso a la web pública (SvelteKit).
GET /admin/	Sesión AdminJS	Panel AdminJS	Interfaz de administración de datos.
GET/POST /admin/login	Sesión AdminJS	Panel AdminJS	Inicio de sesión del panel de administración.
GET /admin/logout	Sesión AdminJS	Panel AdminJS	Cierre de sesión del panel de administración.
GET/POST /admin/api/*	Sesión AdminJS	Panel AdminJS	API interna de AdminJS (recursos y acciones).
GET /grafana/	Auth básica	Monitorización (fase de implementación avanzada pero no terminada)	Panel Grafana publicado por Nginx.
GET /prometheus/	Auth básica	Monitorización (fase de implementación avanzada pero no terminada)	Panel Prometheus publicado por Nginx.
GET /healthz	Público	Proxy Nginx	Healthcheck del proxy.
POST /api/auth/signup	Público	Frontend web	Alta de usuario con código de verificación.
POST /api/auth/signup-verification	Público	Frontend web	Envío de código de verificación por correo.
POST /api/auth/login	Público	Frontend web	Login y guardado del JWT en cookie HttpOnly.
GET /api/auth/session	Sesión web (JWT)	Frontend web	Comprueba si hay sesión activa (usuario, email/ID y rol admin).
POST /api/auth/logout	Sesión web (JWT)	Frontend web	Cierra sesión y borra la cookie del token.
PUT /api/auth/user	Sesión web (JWT)	Frontend web	Actualización de credenciales del usuario.
POST /api/auth/reset-request	Público	Frontend web	Solicitud de recuperación de contraseña.
PUT /api/auth/reset-password	Público	Frontend web	Cambio de contraseña con código.
GET /api/about-us	Público	Frontend web	Obtiene contenido de la sección About.
GET /api/concept-pool	Público	Frontend web	Lista conceptos y contenidos asociados.

Endpoint	Acceso	Origen	Descripción
POST /api/contact	Público	Frontend web	Envía el formulario de contacto.
GET /api/course1	Sesión web (JWT)	Frontend web	Recupera respuestas del MOOC 1.
POST /api/course1	Sesión web (JWT)	Frontend web	Guarda respuestas del MOOC 1.
GET /api/course2	Sesión web (JWT)	Frontend web	Recupera respuestas del MOOC 2.
POST /api/course2	Sesión web (JWT)	Frontend web	Guarda respuestas del MOOC 2.
GET /api/course3	Sesión web (JWT)	Frontend web	Recupera respuestas del MOOC 3.
POST /api/course3	Sesión web (JWT)	Frontend web	Guarda respuestas del MOOC 3.
GET /api/dashboards/my/:type	Sesión web (JWT)	Frontend web	Dashboards del usuario autenticado.
GET /api/dashboards/:type	Admin (JWT)	Frontend web	Dashboards agregados por tipo.
GET /api/fake-news/snippet/:userId	Sesión web (JWT)	Frontend web	Obtiene snippet de análisis para usuario.
POST /api/fake-news/response	Sesión web (JWT)	Frontend web	Envía respuesta del análisis de fake news.
GET /api/fake-news/history/:userId	Sesión web (JWT)	Frontend web	Historial del análisis por usuario.
POST /api/mooc-analysis	Sesión web (JWT)	Frontend web	Proxy para análisis automático de texto.
GET /api/questionnaires/pre-q	Sesión web (JWT)	Frontend web	Lista de cuestionarios pre.
GET /api/questionnaires/post-q	Sesión web (JWT)	Frontend web	Lista de cuestionarios post.
GET /api/users/:email	Sesión web (JWT)	Frontend web	Obtiene un usuario por email.
PUT /api/users/:id	Sesión web (JWT)	Frontend web	Actualiza un usuario por id.
GET /api/users/:idUser/questionnaires	Sesión web (JWT)	Frontend web	Lista cuestionarios respondidos por usuario.
GET /api/users/:idUser/questionnaires/:idQuestionnaire	Sesión web (JWT)	Frontend web	Recupera un cuestionario de usuario.
POST /api/users/:idUser/questionnaires/:idQuestionnaire	Sesión web (JWT)	Frontend web	Guarda respuestas de cuestionario de usuario.

Endpoint	Acceso	Origen	Descripción

A.3. Rutas del frontend

Tabla A.2: Rutas principales del frontend y control de acceso.

Ruta	Acceso	Descripción
/	Público	Página principal y resumen del recorrido.
/about	Público	Página informativa About.
/concepts	Público	Conceptos básicos (contenido editorial).
/concepts-pool	Sesión web	Conceptos dinámicos desde base de datos.
/contact	Público	Formulario de contacto.
/login	Público	Acceso de usuario y recuperación básica.
/signup	Público	Registro de usuario con verificación.
/user-details	Sesión web	Perfil y datos del usuario.
/questionnaires	Sesión web	Entrada al bloque de cuestionarios.
/questionnaires/pre	Sesión web	Listado de pre-questionnaires.
/questionnaires/pre/[id]	Sesión web	Cuestionario pre por id.
/questionnaires/post	Sesión web	Listado de post-questionnaires.
/questionnaires/post/[id]	Sesión web	Cuestionario post por id.
/mooc	Sesión web	Índice de módulos MOOC.
/mooc/course1	Sesión web	MOOC 1 (contenido y tareas).
/mooc/course2	Sesión web	MOOC 2 (contenido y tareas).
/mooc/course3	Sesión web	MOOC 3 (contenido y tareas).
/mooc/course4	Sesión web	MOOC 4 (contenido y tareas).
/dashboards	Admin (sesión web)	Acceso a dashboards.
/dashboards/questionnaires	Admin (sesión web)	Selección de dashboards de cuestionarios.
/dashboards/questionnaires/pre	Admin (sesión web)	Dashboards pre-questionnaire.
/dashboards/questionnaires/post	Admin (sesión web)	Dashboards post-questionnaire.
/dashboards/mooc	Admin (sesión web)	Dashboards de MOOC.

A.4. Secrets para GitHub Actions

Tabla A.3: Secrets necesarios para el despliegue automatizado.

Secret en GitHub	Uso en el workflow	Riesgo si se filtra
SSH_KEY	Clave privada para abrir sesión SSH en la VPS y ejecutar despliegues.	Acceso remoto no autorizado a la infraestructura.
SSH_USER	Usuario SSH para conexión a la VPS.	Facilita ataques dirigidos a la VPS.
SSH_HOST	Host/FQDN de la VPS.	Descubrimiento del destino de despliegue.
SSH_PORT	Puerto SSH.	Ayuda a escaneo y enumeración.
DOCKERHUB_USERNAME	Autenticación para publicar imágenes en Docker Hub.	Publicación de imágenes maliciosas en el repositorio.
DOCKERHUB_TOKEN	Token de acceso a Docker Hub con permisos de push.	Robo de imágenes, subida de versiones falsas.

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=ETS Ingenieros Informaticos - UPM, C=ES
	Fecha/Hora	Thu Jan 15 18:32:58 CET 2026
	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)